Lot Streaming in a Two-stage Assembly System and a Hybrid Flow Shop

Ming Cheng

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Subhash C. Sarin, Chair
Douglas R. Bish
Deborah F. Cook
John P. Shewchuk

September 7, 2012
Blacksburg, Virginia

Keywords: Lot streaming, hybrid flow shop, assembly system, column generation, scheduling

Lot Streaming in a Two-stage Assembly System and a Hybrid Flow Shop

Ming Cheng

(ABSTRACT)

In this dissertation, we investigate the use of lot streaming in a two-stage assembly system and a two-stage hybrid flow shop in order to improve system performance. Lot streaming accelerates the flow of a production lot through a production process by splitting it into sublots, and then, processing these sublots in an overlapping fashion over the machines, thereby reducing work-in-process and cycle-time. Traditionally, lot streaming has been applied to problems in various flow shop machine configurations. It has also been applied to machine environments of job shop, open shop, and parallel machines. Its application to assembly system is relatively new.

The two-stage assembly system that we consider consists of multiple suppliers at Stage 1 with each supplier producing one type of a subassembly (or a component), and one or more assembly locations at Stage 2, where the subassemblies are then put together. Lot-attached and sublot-attached setup time and cost are encountered on the machines at both the stages, and sublot-attached time and cost are encountered for the transfer of sublots from Stage 1 to Stage 2. Mass customization is an example of such a system in which the final assembly of a product is postponed to capture specific customer demands. Dell Computer constitutes a real-life example of this system. A customer picks his/her computer processor, memory, storage, and other equipment, on Dell's web site. Dell's supply chain is configured to obtain subassemblies from suppliers (stage 1), and then, to assemble the requisite systems in different market areas (stage 2). This enables a reduction in operating cost while improving responsiveness to customers. The problem that we address is as follows: Given a maximum number of sublots of each lot, determine the number of sublots to use (assuming equal sublot

sizes), and also, the sequence in which to process the lots, in order to minimize two criteria, namely, makespan, total cost. We propose two column generation-based methods that rely on different decomposition schemes. The results of our computational investigation conducted by using randomly generated data sets reveal that the proposed column generation methods obtain solutions in a few seconds of CPU time while the direct solution by CPLEX of a mixed integer programming model of the problem requires much larger CPU times.

For the hybrid flow shop lot streaming problem, the machine configuration that we consider consists of one machine at Stage 1 and two machines at Stage 2 (designated as 1+2 system). A single lot is to be processed in the system, and the objective is to minimize the makespan. A removal time is associated with each sublot at Stage 1. We present a mixed integer programming model for this problem to determine optimal number of sublots and sublot sizes. First, we consider the case of a given number of sublots for which we develop closed-form expressions to obtain optimal, continuous sublot sizes. Then, we consider determination of optimal number of sublots in addition to their sizes. We develop an upper bound on optimal number of sublots, and use a simple search procedure in conjunction with the closed-form expressions for sublot sizes to obtain an optimal solution. We also consider the problem of determining integer sublot sizes, and propose a heuristic method that directly solves the mixed integer programming model after having fixed values of appropriate variables. The results of our numerical experimentation reveal the efficacy of the proposed method to obtain optimal, continuous sublot sizes, and also, that of the proposed heuristic method to obtain integer sublot sizes, which are within 0.2% of optimal solutions for the testbed of data used, each obtained within a few seconds of CPU time.

The last problem that we address is an extension of the single-lot lot streaming problem for a $1 + 2$ hybrid flow shop considered above to the case of multiple lots, where each lot contains items of a unique product type. We consider two objectives: minimize makespan, and minimize the sum of the completion times for all the lots. The consideration of multiple lots introduces a complicating issue of sequencing the lots. We use the results derived for the single-lot problem and develop effective heuristic methods for this problem. The results of

our computational investigation on the use of different heuristic methods reveal their efficacy in solving this problem.

Dedication

This dissertation is dedicated to my wife, Wei Xu, and to my parents Xingbo Cheng and Weimin Gu.

Acknowledgements

First of all, I offer my sincerest gratitude to my advisor, Dr. Subhash C. Sarin, for his guidance, support, patience, and inspiration throughout the entire progress of my research in Virginia Tech. He is one of the most hard-working professors I have ever known. He has spent tremendous amount of valuable time discussing ideas for every step of the progress in my research. This dissertation, as well as all my research achievements, would not have been possible without his help and encouragement. I would like to sincerely thank the other members of my advisory committee: Dr Douglas R. Bish, Dr. Deborah F. Cook, and Dr. John P. Shewchuk. I really appreciate the opportunity of taking their courses, and the time and effect they have put in reading and commenting on my work.

I would also like to thank all the faculty and staff in the ISE department. In particular, my special thanks go to Mr. Phillip Ratcliff for his assistance and guidance of my GTA course for the past four years, and Dr. Hanif D. Sherali for offering many unforgettable courses and optimization concepts.

I am extremely grateful to all my friends in Blacksburg to make my life so beautiful. I am lucky and enjoyable to be a roommate with Zhuang Wang, Dong Wang, Chengbin Zhu and Rui Chen. I would like to thank Lingrui Liao, Tao Jia, Lixin Wang, Liming Yao, Yuqiang Wang, Yazhe Feng, Cheng Guo, Zhen Fu, Shiguang Xie, Shengzhi Shao, Yu Chen, Juqi Liu, Xiaomeng Chang, Yanfeng Li, Xiaoming He, Jason Judd, Niloy Mukherjee, Daniel Steeneck, and more, for all the good times that we shared. I would especially like to thank Lingrui Liao for all his help in my research. He spent many hours helping me find out bugs in my coding, and also, sparked my idea on my research.

I would like to express my earnest thanks to my parents Xingbo Cheng and Weimin Gu, for their love and support in my entire life.

Finally, I am so deeply grateful to my wife, Wei Xu, for her understanding and encouragement during these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Scope of Research

This dissertation is motivated by the desire to improve effectiveness of a two-stage assembly system and a hybrid flow shop that are devoted to the production of one or multiple batches (lots) of products. Our objective is to study the use of lot streaming in such environments in order to reduce the cycle time of the lots. Lot streaming is the process of effectively splitting production lots into sublots, and then, processing the sublots in an overlapping fashion over the machines, a process with a great potential for cycle time reduction. Investigation into the use of lot streaming in assembly systems and hybrid flow shops is an important undertaking because of its general appeal as well as its applicability to the operational control of a variety of batch production environments. Currently, both reduction in production lead times and meeting customer requirements are critical issues for companies to stay competitive.

First, we investigate the use of lot streaming in a two-stage assembly system, which is as follows: Stage 1 consists of suppliers, each of which produces one type of a subassembly (or a component). These subassemblies are then put together on one or more assembly locations at Stage 2. Such a system is similar to a mass customization system that promotes the idea of postponing the final assembly step in order to differentiate one product from another. A successful implementation of this concept requires an appropriate redesign of the product, now consisting of separate and standard modules. These modules, once produced in individual facilities, are shipped to regional distribution/assembly facilities where they are assembled in response to customer demand. A practical example of mass customization is the system followed by Dell Computer to serve customer demand. A customer picks his/her computer processor, memory, storage, and other equipment, on Dell's web site. Dell's supply chain is configured to obtain subassemblies from suppliers (Stage 1), and then, assemble the requisite systems in different market areas (Stage 2) to reduce operating costs while improving responsiveness to customer demand. Given the demands of finished products at various locations as lots, our problem is to determine both the number of sublots and their sizes as well as the sequence in which to process the lots. We consider two objective functions: minimize makespan, and minimize the total cost due to makespan, sublot-attached setups incurred on subassembly and assembly machines, lot-attached setups incurred on subassembly and assembly machines, transfer of sublots from Stage 1 to Stage 2, tardiness of products, and inventory of subassemblies in the system. We propose two column generation-based methods for the solution of this problem, and present results of our computational investigation on the use of these methods.

The next problem that we address is a single-lot lot streaming problem in a $1 + 2$ two-stage hybrid flow shop, for the objective of minimizing the makespan. The two-stage hybrid flow shop consists of one single flexible machine at the first stage and m identical parallel machines at the second stage. We show that $1 + m$ and $m + 1$ two-stage hybrid flow shop problems are equivalent. The lot consists of U items (products), each of which, after being processed on the single machine at Stage 1, is processed on one of the parallel machines at Stage 2. A practical example of this problem occurs while executing a program on a parallel task computer. Running each program consists of two consecutive parts: loading the data from an external memory, and executing the program. Generally, in a parallel task computer, loading of data from an external memory could only be done in sequence, but the CPU may execute the program concurrently on the available processors. For the $1 + 2$ situation, we develop closed-form solutions for the optimal continuous sublot sizes, with or without sublot-attached setup times, if the number of sublots is given. Then, we extend the problem to include determination of optimal number of sublots (with the maximum number of sublots given), and integer sublot sizes. We also present a method to determine a lower bound, and a heuristic method to obtain integer sublot sizes.

Lastly, we extend the single-lot, two-stage $1 + 2$ hybrid flow shop lot streaming problem to the case of multiple lots and the objectives of minimizing the makespan and the sum of the completion times of all the lots. The consideration of multiple lots introduces another issue pertaining to the sequencing of the lots. We develop effective heuristic methods for the solution of the problem that rely on the results for the single-lot problem. The results of a computational experimentation on the use of these methods reveal their efficacy in solving the problem.

1.2 Organization of Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, we present a literature review on the use of lot streaming in different machine configurations for both time-based and cost-based objective functions. In Chapter 3, we introduce and develop the use of lot streaming in a two-stage assembly system that can also be viewed as a mass customization system. We develop comprehensive models, and present two decomposition-based (column generation) approaches for the solution of this problem, and also, present results of our computational investigation on the use of these methods. In Chapter 4, we address a single-lot, lot streaming problem in a $1 + 2$ hybrid flow shop for the objective of minimizing the makespan. We develop closed-form expressions for the determination of continuous sublot sizes, when the number of sublots is given. Then, we extend the problem to include determination of optimal number of sublots and integer sublot sizes. Results of our computational investigation on the use of the proposed methods are also presented. In Chapter 5, we, then, extend the work presented in Chapter 4 to the case of multiple-lots, and develop heuristic methods based on different dispatching rules. Results of a computational investigation are presented that depict the efficacy of our proposed solution methods for this problem. Finally, in Chapter 6, we conclude the dissertation with a summary of the work accomplished, and also, present the ideas for future research.

Chapter 2

A Literature Review on Lot Streaming

In this chapter, we present a review of literature on lot streaming as it has been developed over the last two decades. Lot streaming has been shown to offer advantages in the form of reduced cycle time, inventory, and makespan. We have divided the work presented in the literature into two categories, based on the type of objective function used, namely, performance based and cost based.

2.1 Literature on Lot Streaming for Performance-based Objective Function

Sarin and Jaipraskash (2007) [74] have presented a seven-field classification scheme for flow shop lot streaming problems, involving performance based objective. We extend this scheme to include machine configuration, and it is as follows:

- *Machine Configuration.* Machine configuration refers to the arrangement of the machines. Most common machine configurations include flow shop (F), job shop (J), open shop (O), parallel machines (P), hybrid flow shop (H), and assembly system (A).

- *Number of product types.* Single product (1) or multiple product types (n).

- *Sublot Sizes. Consistent sublots* (C) refer to the situation in which the size of a sublot remains the same over the machines. *Variable sublots* (V) permit the size of a sublot to vary among the machines. *Equal sublots* (E) is a special case of consistent sublots in which the sublots are of the same size.

- *Idling.* Intermittent idling (II) refers to the situation in which an idle time is permissible between two sublots, while the no idling (NI) situation does not permit such an idle time, i.e., a sublot has to be processed right after the completion of its predecessor.

- *Number of sublots.* Number of sublots may be known a priori, designated by *FixN*, or is to be determined, designated as *FlexN*.

- *Continuous (CV) or Discrete (DV) Sublot Sizes.* Continuous sublot sizes are real-valued, while discrete sublot sizes permit only integer values.

- *Setups.* Lot-attached setup (L(a)) refers to the case in which a setup required for a lot can be started only after the lot is available on the machine. However, in lot-detached setup (L(d)), a setup can be performed on a machine even before the arrival of the lot on that machine. Similarly, we can have sublot-attached setup (S(a)) and sublot-detached setup (S(d)).

- *Transfer Times (T).* This is the time required to transfer a lot or a sublot from one machine to another.

- *Objective Function.* The performance-based objective functions, generally considered are makespan ($C_{\max}$), mean flow time ($\bar{F}$), total flow time ($\sum F$), total tardiness, among others.

We use the following classification scheme to represent various lot streaming problems:

{machine configuration}{number of machines}/{number of product types}/{sublot type}/

{idling}/{Number of sublots}/{sublot sizes}/{setups}/{transfer}/{objective function}.

2.1.1 Flow Shop

A vast majority of research on lot streaming has been devoted to flow shops. In a flow shop, each job follows the same order of processing over the machines.

The two-machine, single-lot, flow shop lot streaming problems addressed in the literature are depicted in Table 2.1. Trietsch (1987) [93] and Potts and Baker (1989) [69] have provided some basic properties for continuous optimal sublot sizes when the number of sublots is given. In particular, they showed that the optimal sublot sizes are geometric in nature with a ratio $\frac{p_2}{p_1}$, where p_j is the processing time per item on machine j, and that there is no idle time between the processing of sublots on the second machine. Sen et al. (1998) [80] have considered the same problem for the objective of minimizing weighted completion time when sublot sizes are equal, consistent, or variable.

Vickson and Alfredson (1992) [100] and Trietsch and Baker (1993) [94] have considered the single-lot problem for both two-machine and three-machine flow shops. Vickson and Alfredson (1992) [100] considered equal-sized sublots, and they developed a modification of Johnson's rule [45] to obtain optimal number of sublots in order to minimize the makespan. Finally, they presented an empirical study to show the effect of transfer batches on makespan and total flow time values. Trietsch and Baker (1993) [94] developed models for the problem of determining continuous and discrete sublot sizes, with and without intermittent idling of machines, and equal, consistent and variable sublots, and minimize the makespan.

When setups are considered in the two-machine flow shop, Sriskandarajah and Wagneur (1999) [84] have applied lot streaming to schedule both single lot and multiple lots in no-wait flow shops with lot-detached setups when number of sublots is given. They provided optimal continuous-sized sublots for processing the jobs in a single lot, and they showed that these sublot sizes are optimal for the case of multiple lots as well. They proposed a heuristic procedure to sequence the lots. They have also addressed the problem of determining integer-sized sublots, and determination of optimal number of sublots. Bukchin et al. (2002) [7] have investigated the lot splitting method in a two-machine flow shop with sublot-attached setups and batch availability to minimize average flow time. Their solution technique relies on the determination of optimal solution of the bottlenect machine. They have reported that the optimal solution for minimizing average flow time is also good for the objective of minimizing the makespan, even though it is not optimal. Cetinkaya (2006) [11] presented the unit-sized transfer batch scheduling problem in an automated two-machine flow shop with only one transport agent. They considered both sublot-detached and sublot-attached setups and obtained the optimal makespan value using a modified Johnson's rule [45].

The two-machine, multiple lot, flow shop lot streaming problems addressed in the literature are shown in Table 2.2. Cetinkaya and Kayaligil (1992) [12] have employed a unit-sized sublot streaming policy, and have solved the problem with lot-attached setup on the first machine and a lot-detached setup on the second machine, using a modification of Johnson's algorithm [45]. Cetinkaya (1994) [10] present the lot streaming problem that includes lot-detached setup, lot-attached transfer, and lot-attached removal times on both the machines, and for the objective of minimizing the makespan for a given number of sublots. An optimal polynomial-time algorithm is presented for its solution. Vickson (1995) [99] has considered the same problem with lot-detached or lot-attached setup and sublot-attached transfer times. A closed-form expression is provided for the determination of optimal, continuous sublot sizes, while a fast algorithm is presented to determine integer sublot sizes. Baker (1995) [1] have presented a modified version of the Johnson's algorithm [45] for solving the sequencing problem of multiple lots taking into account lot-detached and -attached setups. They have considered flexible number of sublots and equal sublot sizes for two-machine and three-machine cases.

Table 2.1: The two-machine, single-lot, flow shop lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
F2/1/E,C/II,NI/FixN/CV/-/-/$C_{\max}$	Potts and Baker (1989) [69]	Closed-form expression for consistent sublot sizes, heuristic for equal sublot sizes
F2/1/C/NI/FixN/CV,DV/-/T/$C_{\max}$	Trietsch and Baker (1993) [94]	Polynomial time algorithm and limited transporter capacity
F2/1/E,C,V/II/FixN/CV/-/-/$C_{\max}$,$\bar{F}$	Sen et al. (1998) [80]	Equal, consistent, and variable sublot sizes
F2/1/C/NI/FixN/CV,DV/L(d)/-/$C_{\max}$	Sriskandarajah and Wagneur (1999)[84]	Closed-form expression and heuristic metohd for integer sublot sizes in no-wait flow shop
F2/1/C/II/FlexN/CV/S(a)/-/$\bar{F}$	Bukchin and Mason (2002) [7]	Frontier approach and tradeoff between alternative objective functions
F2/1/C/II/FixN/DV/S(a),S(d)/T/$C_{\max}$	Cetinkaya (2006) [11]	Dependent and independent sublot-attached setup and optimal algorithm

Table 2.2: The two-machine, multiple-lot, flow shop lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
F2/n/E/II/FlexN/CV/-/-/$C_{\max}$, $\sum F$	Vickson and Alfredsson (1992) [100]	Branch and bound-based and local neighborhood search-based heuristic methods
F2/n/E/II/FlexN/DV/L(a),L(d)/-/$C_{\max}$	Cetinkaya and Kayaligil (1992) [12]	Optimal solution procedure similar to Johnson's rule for unit sized transfer batch
F2/n/C/II/FixN/CV,DV/L(d)/T/$C_{\max}$	Cetinkaya (1994) [10]	Optimal algorithm in the presence of separable setup and removal times
F2/n/C/II/FixN/CV,DV/L(a),L(d)/T/$C_{\max}$	Vickson (1995)[99]	Closed-form expression for continuous sublot sizes and polynomial time algorithm for integer sublot sizes
F2/n/E/II/FlexN/CV/L(a),L(d)/-/$C_{\max}$	Baker (1995) [1]	Optimal algorithm for two-machine case and extension to m-machine case
F2/n/C/II/FixN/CV,DV/L(d)/-/$C_{\max}$	Sriskandarajah and Wagneur (1999) [84]	Closed-form expression and heuristic method for integer sublot sizes
F2/n/E/II/FlexN/CV,DV/S(a)/-/$C_{\max}$	Kalir and Sarin (2001b) [47]	Heuristic method (polynomial time method)

The three-machine, flow shop lot streaming problems addressed in the literature are shown in Table 2.3. Baker and Jia (1993) [2] have addressed the cases of consistent and equal sublot sizes but with no setups and idling among the sublots, for the objective of minimizing the makespan. They report that, when the second machine is dominant, the no-idling constraint becomes redundant, and when the second machine is dominated, the consistent sublot constraint becomes redundant. They have presented a method to determine optimal solutions, but their results cannot be extended to the m-machine case. Glass et al. (1994) [28] have introduced a network representation for this problem, and have presented an O(log n) algorithm to determine minimum makespan in the three-machine flow shop environment, and also, some results that are applicable for the case when the number of machines exceeds three.

In the presence of lot-detached setups, Chen and Steiner (1997a) [15] have presented structural properties of the optimal solution using a network representation similar to that of Glass et al. (1994) [28], and have developed expressions for the consistent and continuous sublot sizes. Chen and Steiner (1997b) [14] have presented similar results for discrete sublot sizes.

The m-machine, single-lot, flow shop lot streaming problems addressed in the literature are shown in Table 2.4. For equal sublot sizes, Truscott (1985, 1986) [95, 96] introduced the lot streaming problem with sublot-attached setup time, transfer time, and capacity constraints. They developed algorithms that generate schedules for minimizing makespan. Steiner and Truscott (1993) [86] have addressed the problem with transfer time for the objective of minimizing cycle time, flow time, and processing cost.

Kalir and Sarin (2001a)[46] have presented analytical expressions for the potential benefits of lot streaming in flowshops for a number of common performance measures, namely, makespan, mean flow time, and average work-in-process (WIP). Kalir and Sarin (2001b) [47] have presented a method to determine optimal number of sublots in the presence of sublot transfer time and sublot-attached setup times.

Table 2.3: The three-machine, flow shop lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
F3/1/E,C,V/NI,II/FixN/CV,DV/-/-/$C_{\max}$	Glass et al. 1994 [28]	Network representation and O(log n) algorithm
F3/1/E,C/NI/FixN/CV/-/-/$C_{\max}$	Baker and Jia (1993) [2]	Optimal algorithm for equal and consistent sublot sizes but no idling
F3/n/E/II/FlexN/CV/L(a),L(d)/-/$C_{\max}$	Baker (1995)[1]	Optimal algorithm for three-machine case and extension to m-machine case
F3/1/C/II/FixN/CV/L(d)/-/$C_{\max}$	Chen and Steiner (1997a)[15]	Network representation and optimal O(logn) algorithm
F3/1/C/II/FixN/DV/L(a)/-/$C_{\max}$	Chen and Steiner (1997b) [14]	Network representation and two approximation methods
F3/n/E/II/FlexN/CV/-/-/$C_{\max}$, $\sum F$	Vickson and Alfredsson (1992) [100]	Branch and bound-based and local neighborhood search-based heuristic methods

Table 2.4: The m-machine, single-lot, flow shop lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
Fm/1/E/II/FixN/DV/S(a)/T/$C_{\max}$	Truscott (1985, 1986) [95, 96]	Mathematical programming models
Fm/1/C/II/FixN/DV/-/-/$C_{\max}$	Glass et al. (1994) [28]	Network representation and O(log n) algorithm
	Chen and Steiner (1997b) [14]	Network representation and two approximation methods
Fm/1/C/NI/FixN/CV,DV/-/-/$C_{\max}$	Chen and Steiner (2003) [16]	Network representation and approximation method in no-wait flow shop
Fm/1/C/II/FixN/CV/-/-/$C_{\max}$	Baker and Pyke (1990) [3]	Heuristic method for more than two sublots
	Glass et al. (1994) [28]	Network representation and O(log n) algorithm
	Williams et al. (1997) [102]	Network representation and heuristic method (feasible index search method)
	Glass and Potts (1998) [29]	Network representation and optimal algorithm
Fm/1/C/II/FixN/CV/L(a)/-/$\bar{F}$	Kropp and Smunt (1990) [55]	Optimal algorithm and heuristic method, deterministic and stochastic environment
Fm/1/E/NI/FixN/CV/-/T/$C_{\max}$,$\sum F$	Steiner and Truscott (1993) [86]	Optimal algorithm for different measures of performance
Fm/1/E/II/FlexN/CV/S(a)/T/$C_{\max}$	Kalir and Sarin (2001a) [49]	Optimal algorithm in the presence of sublot-attached setup and transfer time
Fm/1/E/II/FlexN/CV,DV/S(a)/-/$C_{\max}$	Kalir and Sarin (2001b) [47]	Optimal algorithm (O(nlogn))
Fm/1/C/NI/FixN/CV,DV/L(d)/-/$C_{\max}$	Kumar et al. (2000) [56]	Genetic algorithm-based heuristic method
Fm/1/C/II/FlexN/CV/S(a)/-/$C_{\max}$, $\bar{F}$	Bukchin and Mason (2002) [7]	Frontier approach and tradeoff between alternative objective functions

For consistent sublot sizes, Baker and Pyke [3] have presented a method based on the concept of identifying a bottleneck machine for the 2-sublot version of the problem, which is then generalized to a heuristic procedure for the m-machine, n-sublot problem. Williams et al. (1997) [102] and Glass and Potts (1998) [29] have addressed the same problem for consistent sublot sizes and a given number of sublots, for the objective of minimizing the makespan. Williams et al. (1997) [102] have developed an $O(n^2)$ algorithm by using a network representation of the problem with no setup times and presence of two or three sublots, and a heuristic method is presented for the m-machine, n-sublot problem. Glass and Potts (1998) [29] also presented a network representation and developed an algorithm based on dominance machines.

Kumar, et al (2000) [56] have dealt with the lot streaming and sequencing problem for the m-machine, no-wait, flow shop with multiple lots and detached setup times. For continuous-sized sublots, they proved that the optimal sequence can be obtained by solving a TSP problem, whereas for integer-sized sublots they developed a heuristic procedure. They also used a genetic algorithm to solve the problem, which produced results comparable to those for the heuristic procedure, and it also enabled determination of the number of sublots to use. Chen and Steiner (2003) [16] have addressed a discrete lot streaming problem to obtain a single lot in no-wait m-machine flow shop. They have presented a polynomial-time method for continuous-sized sublots when the number of sublots is equal to two. They have used an approximation method to obtain good solution for discrete sublot sizes Bukchin and Masin (2004) [6] have considered the problem with sublot-attached setup time as well and have used a dynamic programming-based method to obtain optimal solutions for the objective of minimizing the makespan and mean flow time.

For the objective of minimizing mean flow time, Kropp and Smunt (1990) [55] have presented a quadratic programming formulation. Smunt et al. (1996) [82] have extended the lot streaming policies to stochastic job shop and flow shop environments for the performance measures of mean flow time and standard deviation of flow time. Their numerical experimentation reveals that lot splitting substantially improves both mean flow time and standard deviation of flow time in almost all the instances that they tested. They used both equal and consistent sublot sizes, and they emphasize the importance of determining optimal number of sublots.

The m-machine, multiple-lot, flow shop lot streaming problems addressed in the literature are presented in Table 2.5. For equal sublot sizes, Kalir and Sarin (2001c) [48] have developed a heuristic procedure, called the bottleneck minimal idleness heuristic, to minimize the makespan. Tseng and Liao (2008) [97] have examined the same lot streaming problem for the objective of minimizing the total weighted earliness and tardiness. They propose a new discrete particle swarm optimization (DPSO) algorithm that incorporates the net benefit of movement (NBM) algorithm to search for the sequence of jobs.

For the case of continuous sublot sizes, Yoon and Ventura (2002a, 2002b) [106, 107] have addressed the n-job, m-machine flow shop problem to minimize the mean weighted absolute deviation from due dates. A genetic algorithm-based heuristic method is presented in [106], while pairwise interchange method and neighborhood search mechanisms are developed in [107]. Kim and Jeong (2008) [52] have investigated the problem of minimizing the makespan using genetic algorithm (GA). They develop an adaptive genetic algorithm (aGA), which outperforms other traditional GAs for this problem. Martin (2009) [64] has considered sequence dependent setups. The problem is solved by a genetic algorithm-based heuristic method.

Table 2.5: The m-machine, multiple-lot, flow shop lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
Fm/n/E,C/II/FixN/CV/L(a)/-/$\bar{F}$	Smunt et al. (1996) [82]	Different splitting policy in stochastic flow shop environment
Fm/n/E/II/FlexN/CV,DV/S(a)/-/$C_{\max}$	Kalir and Sarin (2001b) [47]	Heuristic method (polynomial time method)
Fm/n/E/II/FlexN/CV/-/-/$C_{\max}$	Kalir and Sarin (2001c) [48]	Heuristic method (bottleneck minimal idleness heuristic)
Fm/n/C/NI/FixN/CV,DV/L(d)/-/$C_{\max}$	Kumar et al. (2000) [56]	Genetic algorithm-based heuristic methods
Fm/n/E/II/FixN/CV/L(a)/-/Mean weighted absolute deviation from due date	Yoon and Ventura (2002a) [106]	Heuristic method (genetic algorithm)
Fm/n/E/II,NI/FixN/CV/L(a)/-/Mean weighted absolute deviation from due date	Yoon and Ventura (2002b) [107]	Heuristic method (pairwise interchange methods and neighborhood search mechanisms)
Fm/n/E/II/FixN/CV/-/-/Total weighted tardiness and earliness	Tseng and Liao (2008) [97]	Optimal algorithm (discrete particle swarm optimization)
Fm/n/C/NI/FixN/DV/L(a)/-/$C_{\max}$	Kim and Jeong (2008) [52]	Genetic algorithm-based heuristic method
Fm/n/C/II/FixN/DV/L(a),S(a)/-/$C_{\max}$	Martin (2009) [64]	Genetic algorithm-based heuristic method

Table 2.6: The job shop, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
Jm/n/C/II/FixN/DV/-/-/$C_{\max}$	Dauzere-Peres and Lasserre (1993) [21]	Optimal algorithm (modified shifting bottleneck procedure)
Jm/n/C/II/FixN/DV/-/-/$C_{\max}$	Dauzere-Peres and Lasserre (1997) [22]	Branch and bound-based algorithm
Jm/n/E,C,V/II/FixN/CV/L(a)/T/$C_{\max}$	Jeong et al. (1999) [43]	Heuristic method (modified shifting bottleneck procedure)
Jm/n/C/II/FixN/CV/-/-/$C_{\max}$	Buscher and Shen (2009) [8]	Heuristic method (tabu search)
Jm/n/C/II/FixN/CV/L(a)/-/$\bar{F}$,Tardiness	Kannan and Lyman (1994) [50]	Simulation for different family scheduling rules and transfer batch sizes
Jm/n/E,C/II/FixN/CV/L(a)/-/$\bar{F}$	Smunt et al. (1996) [82]	Different splitting policy in stochastic job shop environment
Jm/n/E/II/FixN/CV/L(a)/-/$\bar{F}$,Tardiness	Jacobs and Bragg (1988) [42]	Simulation model to test repetitive lots (RL) concept
Jm/n/C/II/FixN/CV/L(a)/-/Tardiness	Wagner and Ragatz (1994) [101]	Repetitive lots (RL) on due date performance
Jm/n/E,C,V/II/FixN/CV/L(a)/-/Multiple Objectives	Jin et al. (1999) [44]	Heuristic method (combination of Lagrangian relaxation and backward dynamic programming)

2.1.2 Job Shop

For the job shop scheduling problem. We assume n jobs to be processed on m machines, where each job has a given routing on the machines.

The job shop, lot streaming problem addressed in the literature are shown in Table 2.6. For the objective of minimizing makespan, Dauzere-Peres and Lasserre (1993, 1997) [21, 22] have presented an iterative procedure for the use of lot streaming in a general job shop with and without capacitated sublot sizes. Optimal sublot-sizes are computed in the first step given a sequence of sublots on the machines, while in the second step, a better sequence is computed by solving a standard job shop scheduling problem with fixed sublot sizes. Results of numerical experimentation are presented to show an improvement in makespan obtained because of lot streaming. Jeong et al. (1999) [43] present a procedure for batch splitting for the objective of minimizing the makespan. In order to meet the due date requirement and reduce the number of batches, a "modified shifting bottleneck" procedure is developed. It is shown to reduce the makespan by up to 80% from its value obtain without batch splitting. Buscher and Shen (2009) [8] present a three-phase algorithm to solve lot streaming problem in a job shop environment for minimizing makespan. The three-phase algorithm consists of a pre-determination of sublot sizes, determination of schedules (by a tabu search method), and variation of sublot sizes. Performance of the basic tabu search method is confirmed by experimentation, and near-optimal solutions are generated in reasonable computing times.

Jacobs and Bragg (1988) [42] and Kannan and Lyman (1994) [50] have considered the objective of minimizing the mean flow time. Jacobs and Bragg (1988) [42] have presented use of the lot-sizing concept without considering setup and transfer time/cost. They develop the concept of "repetitive lots (RL)" , which is similar to lot streaming, in order to reduce the flow time. Simulation experimentation is used to evaluate the performance of the RL concept, and it is shown to significantly reduce flow times and their variability. Kannan and Lyman (1994) [50] have presented the results of their experimentation obtained by using two factors (namely, setup times and dispatching rules), and three family selection rules, which are: WORK (select jobs with the highest total work content), FCFAM (select the first transfer batch to arrive at the queue), and SKFAM (transfer batch with the lowest job slack).

For other objectives, Wagner and Ragatz (1994) [101] have presented the impact of lot streaming on due date-based objectives. They address two issues: the impact of setup times on lot streaming and determination of the sizes of transfer batches. They show that mean tardiness could be reduced (up to 39%) and number of tardy jobs could be reduced as well because of the use of lot streaming. Jin et al. (1999) [44] use multiple objectives with fixed-size transfer lots, for which a synergistic combination of Lagrangian relaxation, backward dynamic programming (BDP), and heuristic methods are used for its solution. Numerical results are presented to show that the method can generate on-time delivery schedules with lower work-in-process (WIP) level in reasonable computational time.

2.1.3 Open Shop

For the open shop single-lot problem, the aim is to simultaneously deal with the routing of the jobs of the lot and sublot sizes.

The open shop, lot streaming problems addressed in the literature are shown in Table 2.7. For a single lot case, Sen and Benli (1998) [79] have determined properties for optimal routing, while for multiple lots in two machine open shop, they have shown that lot streaming will reduce the makespan only if there is a job with large processing times. They also state that one of the important characteristics of open shop is the one-to-one correspondence between an m-machine, n-product problem and an n-machine, m-product problem, where machines and products could be switched. Glass et al. (1994) [28] have discussed the use of lot streaming in three-stage production processes including flow shop, job shop, and open shop. For open shop, they have shown that a methodology could be used to find optimal sublot sizes and optimal schedule in order to minimize makespan in constant time O(n).

Table 2.7: The open shop, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
Om/1/C/II/FixN/CV/-/-/$C_{\max}$	Glass et al. (1994) [28]	Network representation and O(log n) algorithm
Om/1/C/II/FixN/CV/L(d)/-/$C_{\max}$	Sen and Benli (1999) [79]	Optimal algorithm to determine routing
O2/n/C/II/FixN/CV/L(d)/-/$C_{\max}$	Sen and Benli (1999) [79]	Optimal algorithm for non-preemptive and preemptive cases

Table 2.8: The parallel machines, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
P/n/E/II/FixN/CV/L(a)/-/$\sum F$	Yalaoui and Chu (2006) [105]	Branch and bound-based algorithm
P/n/E/II/FixN/CV/L(a)/-/Tardiness	Kim et al. (2004) [54]	Two-phase heuristic method (1. Initial Sequence; 2. Split each lot into sublots)
	Logendran and Subur (2004) [62]	Tabu search-based heuristic method
P/n/C/II/FixN/CV/L(a)/-/# of Tardy Jobs	Suer et al. (1997)[87]	Mathematical programming
P/n/E/II/FixN/CV/L(a)/-/Lateness and # of Tardy Jobs	Lin and Jeng (2004) [59]	Dynamic programming-based algorithm and two heuristic methods

2.1.4 Parallel Machines

Lot streaming has also been used in the parallel machine environment in which a lot is split into sublots and the sublots are processed simultaneously on multiple machines.

Yalaoui and Chu (2006) [105] have addressed the parallel-machine scheduling problem for the objective of minimizing the sum of completion times. They have presented potential applications of the problem in real life, and have developed a branch-and-bound method that rely on the use of some theoretical properties.

For the total tardiness objective, Kim et al. (2004) [54] focus on the m-parallel-machine scheduling problem for the objective of minimizing total tardiness They have presented a two-phase heuristic method, in which an initial sequence is constructed in the first phase, and lot (job) splitting is implemented in the second phase, and the sublots (subjobs) are rescheduled. Logendran and Subur (2004) [62] have presented the job-splitting method for the unrelated parallel machines for minimizing the total weighted tardiness. A mixed integer linear programming model is developed for the problem, and a tabu search-based heuristic method is developed, and it is shown to be effective.

Suer et al. (1997) [87] have considered the objective of minimizing the number of tardy jobs. Four mathematical models are presented with and without the consideration of setup times and lot-splitting. The results show that lot-splitting-based method performs better in the majority of cases. Lin and Jeng (2004) [59] have addressed batch scheduling in the parallel-machine environment, where sequence-independent setup times is incurred and due date of each job is specified, in order to minimize the maximum lateness and the number of tardy jobs. First, two dynamic programming-based algorithms are developed to obtain optimal solutions, but then solution times grow exponentially. Then, several heuristic methods are proposed to find near-optimal solution in a reasonable amount of time.

2.1.5 Hybrid Flow Shop

The hybrid flow shop, also known as a flow shop with multiple processors at each stage, consists of two or more stages, each of which comprises one or more parallel machines, but at least one stage contains multiple machines. Each lot (job) is processed on each stage sequentially but may skip any number of stages provided it is processed on at least one stage.

The hybrid flow shop, lot streaming problems addressed in the literature are shown in Table 2.9. These problems have been studied by Kim et al. (1997) [51], Tsubone et al. (1996) [98], Zhang et al. (2003, 2005) [109, 110], and Liu (2008) [61].

Kim et al. (1997) [51] have discussed the two-stage hybrid flow shop, which they called "flexible flow shop", with identical parallel machines at each stage when transfer batches are applied to the lots, for the objective of minimizing the makespan. They consider equal sublot sizes, lot-attached and lot-detached setup times, and the optimal solution is similar to that obtained using Johnson's rule [45]. Numerical examples are also presented to illustrate the proposed method. Tsubone et al. (1996) [98] have considered the $1 + m$ problem and have used a simulation model to study the impact of lot sizing for the objectives of optimizing total flow time, makespan, capacity utilization, and maximum work-in-process level. Zhang et al. (2003, 2005) [109, 110] and Liu (2008) [61] have considered the $m + 1$ problem. Zhang et al. (2003) [109] assume one of the stages to be a bottleneck and sublot sizes to be integer. The problem, is then, formulated as a mixed integer linear programming model, and two heuristic methods are proposed that allocate the sublots as evenly as possible to the machines at Stage 1, and they are shown to produce near-optimal solutions. Zhang et al. (2005) [110] assume equal sublot sizes and the objective of minimizing mean completion time. A lower bound and two heuristic methods are presented. Liu (2008) [61] assumes a given number of sublots and the objective of minimizing the makespan. They prove a property for this case by which an optimal solution can be obtained by allocating the sublots in "rotation" over the machines at Stage 1. The optimal sublot sizes are obtained by using a linear program.

Table 2.9: The hybrid flow shop, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
H(m+m)/n/E/II/FixN/CV/L(a),L(d)/-/$C_{\max}$	Kim et al. (1997) [51]	Optimal algorithm similar to Johnson's rule
H(1+m)/n/-/II/-/S(a)/-/$C_{\max}$,$\sum F$,etc.,	Tsubone et al. (1996) [98]	Simulation models to test different performance measures
H(m+1)/1/E/II/FlexN/CV/-/-/$\bar{F}$	Zhang et al. (2003)[109]	Heuristic methods
H(m+1)/n/E/II/FlexN/CV/S(a)/-/$\bar{F}$	Zhang et al. (2005) [110]	Mathematical programming model and heuristic method
H(m+1)/1/E/II/FlexN/CV/S(a)/-/$C_{\max}$	Liu (2008) [61]	Heuristic method to test worst-case performance

Table 2.10: The assembly system, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
A(m+1)/1/C/II/FixN/CV,DV/L(d)/-/$C_{\max}$	Sarin et al. (2011a) [78]	Polynomial time optimal algorithm
A(m+1)/n/C/II/FixN/CV,DV/L(d)/-/$C_{\max}$	Sarin and Yao (2011b) [77]	Branch and bound-based algorithm

They also present a heuristic method to determine number of sublots when all the sublots are of the same size.

2.1.6 Two-stage Assembly System

A two-stage assembly system can be defined as follows: The first stage of this system consists of a set of subassembly stations, each of which produces a component type. These components are then transferred in sublots to Stage 2, where they are assembled into finished products. Stage 2 consists of multiple parallel machines. Each of these machines represents an assembly facility devoted to a product type.

The two-stage assembly system, lot streaming problems addressed in the literature are shown in Table 2.10. The single-lot problem has been addressed by Sarin et al. (2011a) [78]. They define a two-stage assembly system in which the first stage consists of parallel subassembly machines each of which produces a component type, and the second stage consists of only one assembly machine that assembles a final product after all requisite components are ready at Stage 1. Lot-detached setups are incurred on all the machines at Stage 1 and Stage 2. They have presented a polynomial-time algorithm to obtain optimal sublot sizes for the objective of minimizing the makespan, given a fixed maximal number of sublots, and also for integer sublot sizes. Sarin and Yao (2011b) [77] have extended the single-lot problem to multiple-lots. They use the results derived from the single-lot problem and propose a branch-and-bound-based method for its solution. Their proposed solution method outperforms the direct solution of a mathematical model of the problem by CPLEX for both small and large-sized problem instances.

2.2 Cost-based Lot Streaming Problems

The use of lot streaming for the objective of minimizing the total cost, including inventory cost, production cost, setup cost, transfer cost, etc., when a constant or linearly varying demand needs to be satisfied, has been addressed in the literature. It requires determination of both lot sizes and the number of sublots. Different stage configurations are considered as follows:

- One to One (V1+1): A single vendor supplies goods to a single buyer (see Figure 2.1) (Ertogral et al (2007) [25]). Note that Q is the production lot size, D is the demand rate, and q is the shipment lot size. There are 5 shipments for each production lot size.
- One to Many (V1+m): A single vendor supplies goods to multiple buyers.
- m-machine Flow Shop (Fm): A m-machine flow shop that represents a m-stage supply chain.
- m-machine Assembly Job Shop (AJm) : In this configuration, a m-machine assembly job shop is appended to the regular job shop (See Figure 2.2).

We use L for the setup cost that is based on the lots, and S for the setup cost that is based on the sublots, and TC for the total cost to be minimized. The other terms are the same as that for the performance-based objectives.

We use the following classification scheme to represent various lot streaming problems for the cost-based objectives:

{Stage configuration}/{number of product types}/{sublot type}/

/{Number of sublots}/{sublot sizes}/{setup costs}/{transfer}/{objective function}.

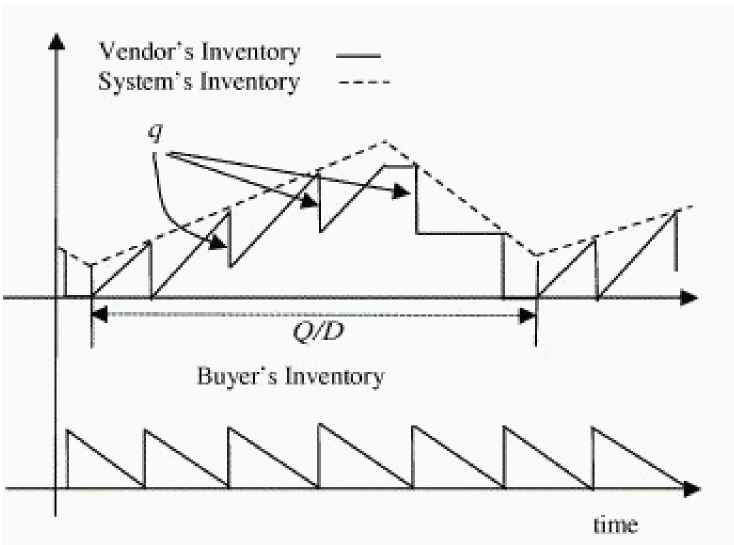

Figure 2.1: Inventory level for one-supplier, one-buyer example

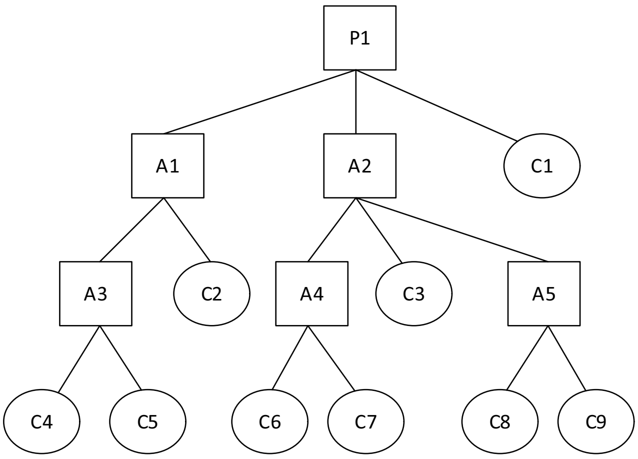

Figure 2.2: A product structure in assembly job shop system

2.2.1 Single-lot Lot Streaming Problems

The cost-based, single-lot, flow shop, lot streaming problems addressed in the literature are shown in Table 2.11.

Goyal (1977) [31] has presented a mathematical model in a two-machine flow shop to minimize the sum of total annual variable cost, cost of carrying finished product inventory, and the annual cost of (production) setups and the cost of moving sublots in a two-stage production system. The optimal sublot sizes are found to be geometrical in nature. Graves and Kostreva (1986) [35] have studied the overlapping of operations (lot streaming) in the material requirements planning system (MRP) in order to reduce lead time and total cost. They have presented a method for two-machine flow shop with equal processing speed of both the machines. Li and Xiao (2004) [58] have developed mechanisms to coordinate lot streaming decisions in order to achieve a system-wide optimal solution.

For multi-stage production system, Szendrovits (1975) [88] has studied the use of lot streaming idea, which the author calls as lot sizing, to minimize the sum of the fixed cost per lot and the inventory holding cost. The economic production quantity (EPQ) model with equal sublot sizes and the cost sensitivity analysis used, show the benefits of using lot streaming. Moily (1986) [65] considers the use of the lot streaming method for arborescent flow, and a heuristic method is proposed for large-sized problems. Szendrovits (1976) [89] and Goyal (1976) [30] and Drezner et al. (1984) [23] extend the problem to include the transportation cost in the model. Goyal (1978) [32] has presented the problem to determine optimal number of sublots in a multi-stage production system. The multi-stage problem is then further discussed by Goyal (1988) [33]. Szendrovits (1987) has presented an equal sublot size model when interruption between production is allowed. He has developed a heuristic method based on intuitive observations to determine number of sublots and sublot sizes. Ramasesh, et al. (2000) [70] have developed a cost-based model to minimize the total cost consisting of setup cost, transportation cost, processing cost and holding cost for WIP. They also considered the case of unequal production rates at different stages.

Goyal and Szendrovits (1986) [34] have considered the case of consistent (but not equal) sublot sizes. The problem is solved by a heuristic procedure. Bogaschewsky et al. (2001)[5] have presented the same problem with consistent sublot sizes with a different heuristic method.

Szendrovits and Drezner (1980) [91] have addressed the multi-stage problem with equal sublot sizes but different number of sublots at each stage. An optimum seeking method for the deterministic case is developed when considering setup, inventory, and transportation costs. Szendrovits and Golden (1984) [92] have compared the total costs of the two models they developed by using simulated problems, and they conclude that the equal lot size model with batch shipment generates better results.

Van Nieuwenhuyse and Vandaele (2004) [66] have described a model for minimizing total cost consisting of inventory holding cost, transportation cost and intermittent idle cost. They provide expressions to determine intermittent idle time, and study its impact on system performance. Chiu et al. (2004) [20] have presented a model to minimize total cost, consisting of makespan cost and transfer cost, with a limited number of capacitated transporters for multistage batch production system, where both attached and detached setups are considered. Chiu and Chang (2005) [19] have proposed two cost models for solving flow shop problems where the optimal processing batch size and the optimal number of transfer batches are determined to minimize the total annual cost.

Sarin et al. (2008) [75] present a polynomial-time procedure for determining the number of sublots of a single-lot, multiple-machine flow shop lot-streaming problem in order to minimize a unified cost-based objective function that comprises criteria pertaining to makespan, mean flow time, work-in-process, sublot-attached setup and transfer times.

The cost-based, single-lot, one-vendor one-buyer, and one-vendor multiple-buyers, lot streaming problems addressed in the literature are shown in Table 2.12. For a one vendor, one buyer supply chian problem, Hill (1996) [39] has considered the two-stage problem with finite production rate (nearly increasing demand). Rau et al. (2003) [72] consider the integrated inventory model for the case of deteriorating items. Kim and Ha (2003) [53] have provided a lot-splitting strategy for a JIT system for a single-supplier and single-buyer. They report that optimal delivery policy can be economically beneficial to both parties and the delivery size converges to a unique optimal size when the number of deliveries and the order quantity can be changed. Huang (2004) [41] present a similar problem in the presence of process unreliability. The production process is assumed to deteriorate during processing, thereby producing defective items. The objective is to minimize the total (joint) annual cost incurred by vendor and buyer. A similar problem is presented by Ertogral et al. (2007) [25] with two stages and transfer cost, which is dependent of sublot sizes. They assume sublot sizes to be equal, and develop optimum seeking solution procedures.

Chiang (2001) [18] present order splitting of the multiple-delivery arrangement in a stochastic demand environment. They show that lot streaming can significantly reduce the total cost if the cost of dispatching an order for an item is not small. Eynan and Li (1997) [26] have considered the maximization of the net present value of the total revenue collected. Rau and Ouyang (2008) [71] have extended the problem to a general varying demand case (both increasing and decreasing). They develop an optimum seeking methodology for the objective of minimizing the joint total costs.

For a single vendor but multiple buyers, Zhang et al. (2007) [108] have presented the problem to simultaneously reducing inventory levels of raw materials, work-in-process, and finished goods, in order to minimize total cost. They have proposed an optimum seeking method, and numerical examples are presented to provide insights.

Table 2.11: The cost-based, single-lot, m-machine flow shop, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
F2/1/C/FlexN/CV/L/T/TC	Goyal (1977) [31]	Mathematical model and closed-form expression
F2/1/E/FlexN/CV/L/T/TC	Graves and Kosreva (1986) [35]	Closed-form expression
F2/1/E,C/FlexN/CV/-/T/TC	Li and Xiao (2004) [58]	Optimal algorithm and extension to multistage case
Fm/1/E/FlexN/CV/L/-/TC	Szendrovits (1975) [88]	Closed-form expression to obtain economic production quantity (EPQ)
	Moily (1986) [65]	Optimal algorithm and heuristic method
Fm/1/C/FlexN/CV/L/T/TC	Goyal and Szendrovits (1986) [34]	Heuristic method
	Bogaschewsky et al. (2001) [5]	Mathematical programming model and heuristic method
Fm/1/E/FlexN/CV/L/T/TC	Goyal (1976) [30]	Note on Szendrovits (1975)
	Szendrovits (1976) [89]	Closed-form expression
	Goyal (1978) [32]	Mathematical programming model
	Szendrovits and Drezner (1980) [91]	Optimal algorithm and heuristic method
	Drezner et al. (1984) [23]	Branch and bound-based heuristic method
	Szendrovits and Golden (1984) [92]	Optimal algorithm
	Szendrovits (1987) [90]	Optimal algorithm and heuristic method
	Ramasesh et al. (2000) [70]	Closed-form expression
Fm/1/E/FixN/CV/-/T/TC	Steiner and Truscott (1993) [86]	Optimal algorithm for different measures of performance
Fm/1/E,C/FlexN/CV/L/T/TC	Van Nieuwenhuyse and Vandaele (2004)[66]	Optimal algorithm in the presence of ”gap costs” which result from the intermittent idling
Fm/1/V/FlexN/DV/L/T/TC	Chiu et al. (2004) [20]	Mathematical programming model and two heuristic methods
Fm/1/E/FlexN/CV/L/T/TC	Chiu and Chang (2005)[19]	Mathematical programming models
Fm/1/E,C/FlexN/CV/S/T/TC	Sarin et al. (2008) [75]	Polynomial time algorithm

Table 2.12: The cost-based, single-lot, one-vendor one-buyer, and one-vendor multiple-buyer, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
V1+1/1/C/FlexN/CV/L/T/TC	Hill (1996) [39]	Optimal algorithm for linearly increasing demand
	Rau et al. (2003) [72]	Mathematical programming model and optimal algorithm
	Kim and Ha (2003) [53]	Optimal algorithm (single setup multiple delivery (SSMD) model)
	Huang (2004) [41]	Optimal algorithm in the presence of defective items
	Ertogral et al. (2007) [25]	Optimal algorithm
V1+1/1/E,C/FlexN/CV/L/T/TC	Chiang (2001) [18]	Closed-form expression
V1+1/1/E/FlexN/CV/L/-/TC	Rau and Ouyang (2008) [71]	Optimal algorithm for both increasing and deceasing demands
V1+1/1/E,C/FlexN/CV/-/-/Max Total Revenue	Eynan and Li (1997) [26]	Optimal algorithm and heuristic method in the presence of learning effects
V1+m/1/E/FlexN/DV/L/-/TC	Zhang et al. (2007)	Optimal algorithm (integrated vendor-managed inventory (VMI) model)

Table 2.13: The single-lot, assembly job shop, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
AJm/1/E,C/FlexN/CV/L/T/TC	Chan et al. (2008) [13]	Mathematical programming model and genetic algorithm-based heuristic method
	Wong et al. (2009) [104]	Genetic algorithm-based heuristic method

The single-lot, assembly job shop, lot streaming problems addressed in the literature are shown in Table 2.13. Chan et al. (2008) [13] have presented such a system in which an assembly stage is appended to the regular job shop. The assembly stage can start immediately after all jobs of the same bill of material have been completed in the regular job shop part of the system. A product structure with 4 assembly levels is presented in Figure 2.2, in which the final product (P1) and Assemblies (A1 - A5) require assembly stages. They have presented an efficient method based on genetic algorithm and dispatching rules, for the objective of minimizing total cost. Wong et al. (2009) [104] have addressed the same problem to minimize total lateness cost. They have extended the use of lot streaming technique to a Resource-Constrained Assembly Job Shop Scheduling Problem (RC_AJSSP), and they use genetic algorithm (GA) and particle swarm optimization (PSO) method for its solution.

2.2.2 Multiple-lot Lot Streaming Problems

The cost-based, multiple-lot, lot streaming problems addressed in the literature are shown in Table 2.14.

Blumenfeld et al. (1991) [4] have considered the problem of minimizing the total cost in an integrated production-transportation network, which consists of one orgin and many destinations. Manoj et al. (2008) [63] have addressed the multiple-lot problem for a supply chain consisting of a manufacturer, a distributor, and several retailers. Stecke and Zhao (2007) [85] have presented an MIP model to solve a commit-to-delivery business mode problem for minimizing cost flow. An efficient polynomial-time heuristic method is then developed to get a near optimal solution. Var Nieuwenhuyse and Vandaele (2006) [67] present the deterministic and stochastic cases of lot streaming in a two-stage supply chain. They prove analytically that lot streaming methodology could reduce total cost and improve delivery reliability of the manufacturing system.

Table 2.14: The multiple-lot, cost-based, lot streaming problems addressed in the literature

Problem	Addressed by	Approach, Feature, and Contribution
V1+m/n/C/FlexN/CV/L/T/TC	Blumenfeld et al. (1991) [4]	Closed-form expression
	Manoj et al. (2008) [63]	Mathematical model in just-in-time (JIT) environment
V1+1/n/C/FlexN/CV/-/T/TC	Stecke and Zhao (2007) [85]	Polynomial time heuristic method
F2/n/C/FlexN/CV/L/T/TC	Var Nieuwenhuyse and Vandaele (2006) [67]	Mathematical model and approximation method

Chapter 3

Lot Streaming in a Two-stage Assembly System

In this chapter, we address a scheduling problem in a two-stage assembly system. The first stage of this system consists of a set of subassembly stations, each of which produces a component type. These components are then transferred in sublots to Stage 2, where they are assembled into finished products. Stage 2 consists of multiple parallel machines. Each of these machines represents an assembly facility devoted to a product type. We represent the configuration as a $m+n$ system, where there are m parallel machines at Stage 1 and n parallel machines at Stage 2. Lot-attached and sublot-attached setup time and cost are encountered on the machines at both the stages, and sublot-attached time and cost are encountered for the transfer of a sublot from Stage 1 to Stage 2. We assume that the subassemblies are transferred in equal-sized sublots to Stage 2. Given a number of lots (jobs), the problem is to determine the number of sublots to use for each lot and the sequence in which to process the lots. We consider two different objective functions, namely, minimize makespan and minimize the total cost incurred. First, we present mixed integer programming models, which can be used to solve small-sized instances. For large-sized problem instances, we develop two column generation-based heuristic methods. Results of our computational investigation on the use of these methods reveal their efficacy in obtaining almost optimal solutions within a few seconds of CPU time.

3.1 Introduction

The two-stage assembly system that we consider in this chapter can be viewed as a mass customization system, which has emerged as an important production mode for industry to achieve lower cost, shorter cycle time, and greater product variety (See Pine II, etc (1993) [68]). The concept of mass customization originated in late 1980s and has been effectively used in apparel and computer industries, among others. Mass customization promotes the idea of postponing the final assembly step, which differentiates one product type from another until the very last stage of production. The best-known practice of mass customization is by Dell Computer [27], where a customer picks his/her computer processor, memory, storage, and other equipment, on Dell's web site. Dell's supply chain is configured to obtain subassemblies from suppliers (Stage 1), and then, to assemble the requisite systems in different market areas (Stage 2) to reduce operating costs while improving responsiveness to customers. A literature review on mass customization and a framework of future research direction can be found in Da Silveira et al. (2001)[81].

Lot streaming is a technique that can be used to meet the objective of a mass customization system because it accelerates the flow of a product through a production system by splitting a production lot into sublots, and then, processing the sublots in an overlapping fashion over the machines, thereby reducing work-in-process and cycle-time. In this chapter, we explore the use of lot streaming to approach the two-stage assembly system. Our work will entail development of new insights, results, and algorithms for the use of lot streaming in a new setting, thereby contributing to its knowledge base. It would also lead to the development of effective solution methodologies for an important class of problems. (for a literature review on lot streaming, please see Chapter 2)

The use of lot streaming in a two-stage assembly system has been addressed by Sarin et al. (2011) [78]. They define a two-stage assembly system in which the first stage consists of parallel subassembly machines each of which produces a component type, and the second stage consists of only one assembly machine that assembles a final product after all requisite components are ready at Stage 1. Lot-detached setups are incurred on all the machines at Stage 1 and Stage 2. They have presented a polynomial-time algorithm to obtain optimal sublot sizes for the objective of minimizing the makespan, given a fixed maximal number of sublots, and also for integer sublot sizes. Here, we generalize that problem to include lot and sublot-attached setups, multiple lots, multiple machines at Stage 2, and also, consideration of cost-based objective function. However, we consider equal-sized sublots.

Another machine configuration similar to the assembly system considered here to which lot streaming has been applied is hybrid flow shop (Tsubone et al. (1996) [98], Zhang et al. (2003, 2005) [109, 110], and Liu (2008) [61]). Tsubone et al. (1996) [98] have considered the $1 + m$ problem and have used a simulation model to study the impact of lot sizing from the viewpoint of optimizing the total flow time, makespan, capacity utilization, and the maximum work-in-process level. Zhang et al. (2003, 2005) [109, 110] and Liu (2008) [61] have considered the $m + 1$ problem. Zhang et al. (2003) [109] assume one of the stages to be a bottleneck and sublot sizes to be integer. The problem is, then, formulated as a mixed integer linear programming model, and two heuristic methods are proposed that allocate the sublots as evenly as possible to the machines at Stage 1, and they are shown to produce near-optimal solutions. Zhang et al. (2005) [110] assume equal sublot sizes and the objective of minimizing mean completion time. A good lower bound and two heuristic methods are presented. Liu (2008) [61] assume a given number of sublots, equal sublot sizes, and the objective of minimizing the makespan.

3.1.1 Difference between Two-stage Assembly System and Hybrid Flow Shop

In a hybrid flow shop, jobs are processed on the machines sequentially as shown in Figure 3.1a, while in a two-stage assembly system, Stage 1 consists of suppliers, each of which produces a type of a subassembly (or a component). These subassemblies are then put together on an assembly machine at Stage 2 (see Figure 3.1b). The real-life instances of such a system include dressing of engines and transmissions by suppliers (Stage 1) for their assembly at an assembly plant (Stage 2), and fabrication of integrated circuits and manufacturing of power supplies, among other components, by suppliers (at Stage 1) for their assembly on printed circuit boards (at Stage 2). A successful implementation of this concept requires an appropriate redesign of the product, now comprising of separate and standard modules. These modules, once produced in central facilities (Stage 1), are shipped to regional distribution/assembly facilities (Stage 2) where they are assembled in response to customer demand.

3.1.2 A Numerical Example of Lot Streaming in a Two-stage Assembly System

In a two-stage assembly system, Stage 1 consists of suppliers, each of which produces a type of a subassembly (or a component), and these subassemblies are then assembled at a location at Stage 2. Figures 3.2 and 3.3 depict a numerical example of a two-stage mass assembly system without and with the use of lot streaming, respectively. The first stage consists of five machines, each of which produces subassemblies, namely, CPU, memory, storage, camera and accessories. They are then transferred to one of the assembly locations (United States/Asia) at Stage 2. An attached setup time before the processing of each lot and its sublots on the machines at both the stages, and a lot-attached transfer time from Stage 1 to Stage 2, are incurred. Three different lots of product types with each product type requiring different

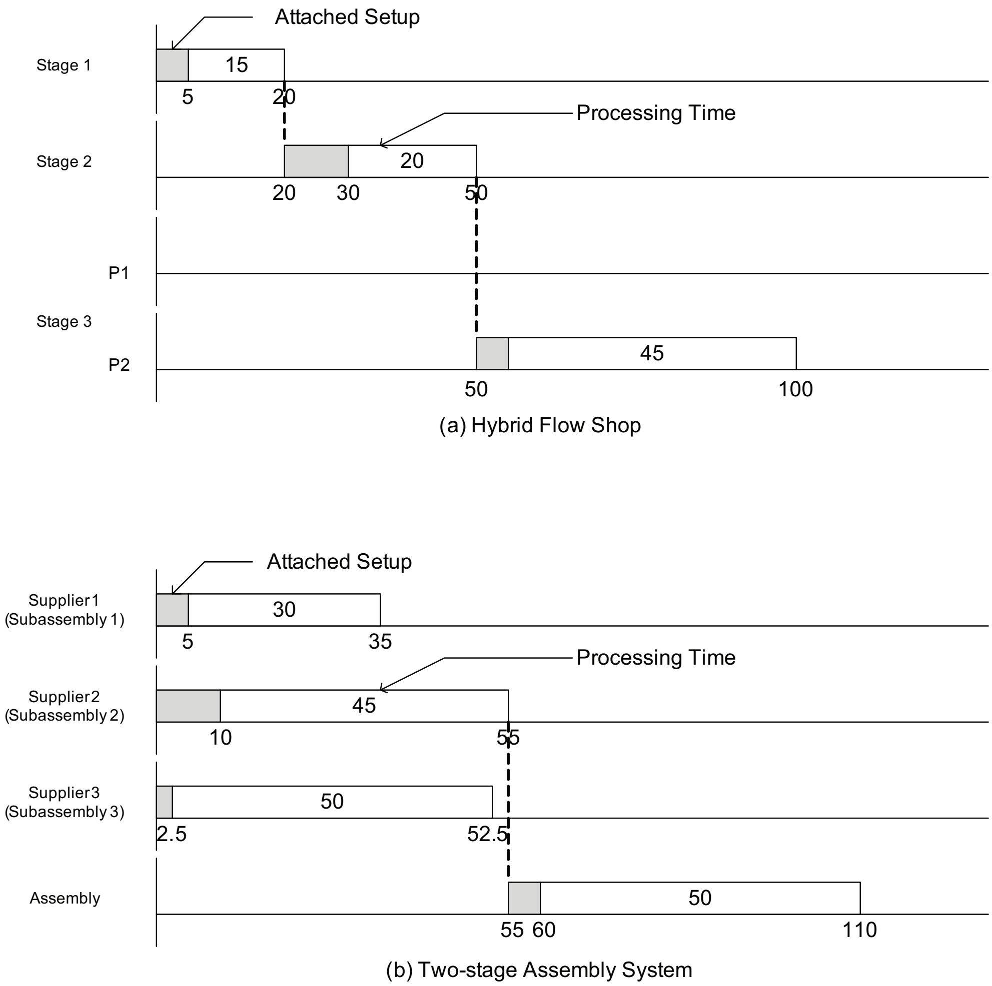

Figure 3.1: Hybrid flow shop and two-stage assembly system

types of subassemblies, are to be assembled on the locations at Stage 2. The products in lots 1, 2, and 3 require CPU, memory, camera, and accessories; CPU, memory, storage, and camera, and CPU, memory, storage, and accessories, respectively. Note that the makespan value reduces to 187.5 units when lot streaming is used (it splits lot 1 into 3 equal-sized sublots and lot 2 and 3 into 2 equal-sized sublots) from its value of 197.5 units when lot streaming is not used.

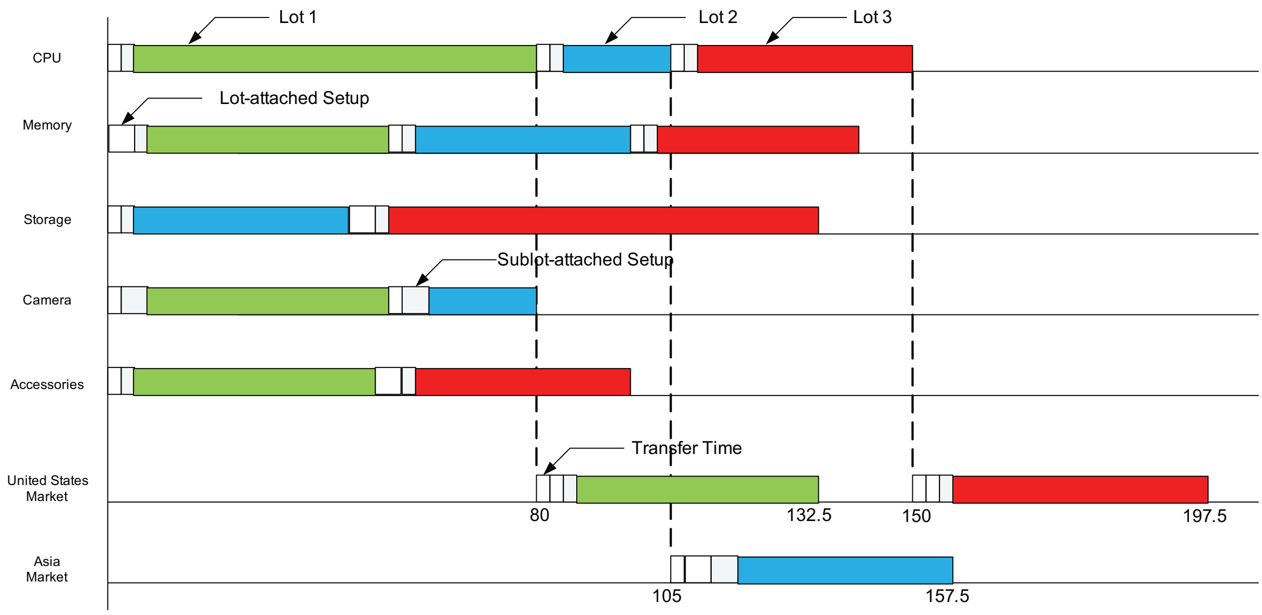

Figure 3.2: Processing of multiple lots without lot streaming in a two-stage assembly system

The problem that we address can be defined as follows: *Given a set of lots, each consisting of a known number of products, and a maximum number of sublots, split each lot into equal-sized sublots, sequence the lots for processing in a two-stage assembly system, that incurs an attached setup time before the processing of each lot and its sublots on the machines at both the stages and a lot-attached transfer time from Stage 1 to Stage 2, so as to optimize a performance measure. We consider both minimization of makespan and total cost.* Accordingly, we designate this problem as TSAS-MP (two-stage, assembly system, makespan problem) and TSAS-CP (two-stage, assembly system, cost-based problem).

The rest of this chapter is organized as follows. In Section 3.2, we present the problem

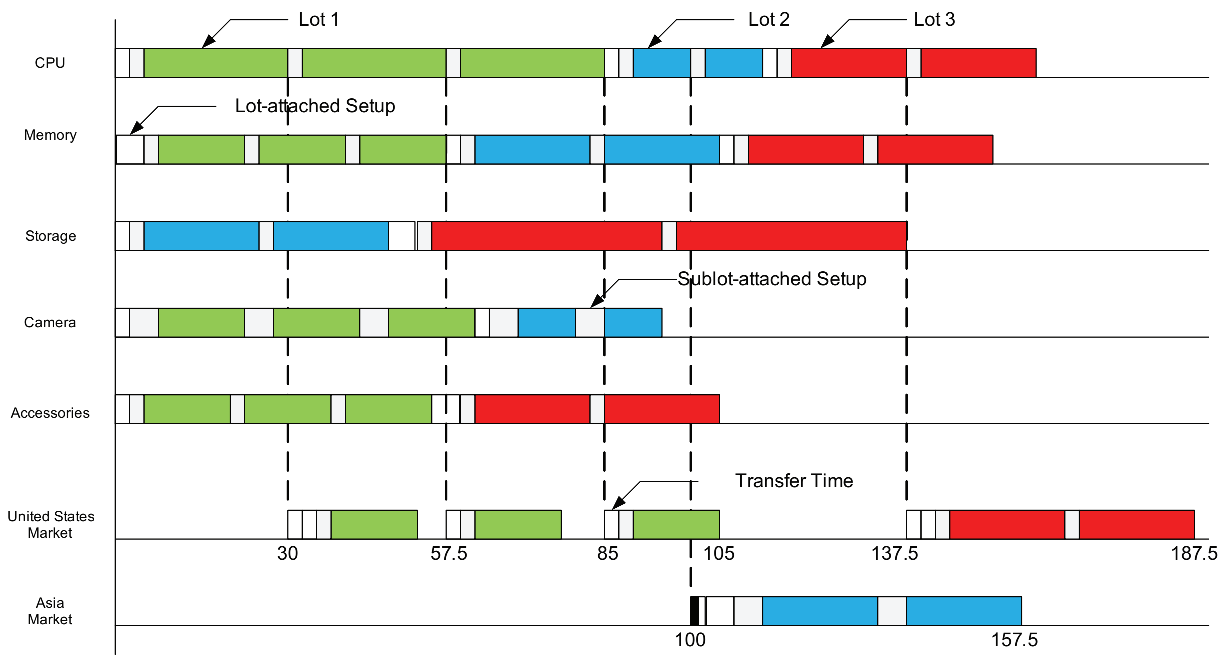

Figure 3.3: Processing of multiple lots with lot streaming in a two-stage assembly system

in detail and introduce model formulations for TSAS-MP and TSAS-CP, respectively. In Section 3.3, two column generation methods are presented. Results of our computational investigation are presented in Section 3.4. Finally, concluding remarks are made in Section 3.5.

3.2 Problem Description and Formulation

3.2.1 Problem Description

The TSAS-MP and TSAS-CP that we consider consist of a set of lots J that is to be processed in a two-stage system. Each lot $j \in J$ consists of Q_j products, and each product of lot j requires S_j subassemblies that are produced by S_j vendors (each vendor represented as a machine). Each vendor supplies one subassembly type. Each lot is split into equal-sized sublots, and requisite number of subassemblies for the products of each sublot of lot j are then transferred to assembly machine a_j at Stage 2 for the formation of their assemblies. Attached setup times are incurred before the processing of each lot and its sublots and for the transfer of sublots from Stage 1 to Stage 2. We consider the objectives of minimizing the makespan and minimizing total cost consisting of makespan cost, setup cost, transfer cost, tardiness cost, and inventory cost. The problem features are summarized below:

Production Configuration

1. There are multiple subassembly machines operating at the first stage.

2. There are one or more assembly machines at the second stage.

3. Each lot can be split into multiple equal-sized sublots, which do not exceed the maximum number of sublots allowed.

4. Attached setup times are incurred before the processing of a lot, sublot, and transfer of the sublots from Stage 1 to Stage 2. (See Figure 3.3).

5. Products of each lot are pre-assigned to one machine at Stage 2 for the assembly operation.

Objective

1. Minimize the makespan;

2. Minimize total cost, which comprises makespan cost, setup cost, transfer cost, tardiness cost, and inventory cost.

Decisions

1. Sequence in which to process the lots;

2. Number of sublots for each lot;

3. Size of each sublot.

3.2.2 Notation and Assumptions

In this section, we present the notation that we use in this chapter. Also, we list the assumptions that we have made.

Sets:

- J: Set of lots to be scheduled;
- J^k: Set of lots that require subassembly machine k, $k \in S$;
- S_j: Set of subassembly machines required by lot j, $\forall j \in J$, i.e., the set of subassemblies required by lot j;
- S: Set of subassembly machines at Stage 1;
- A: Set of assembly machines at Stage 2;
- A^k: Lots that are to be assembled on machine k, $k \in A$.

Parameters:

- a_j: Machine where products of lot j are assembled, $\forall j \in J$;
- Q_j : Number of products in lot j, $\forall j \in J$;
- n_j : Maximum number of sublots of lot j, $\forall j \in J$;
- ls_j^k : Lot-attached setup time for lot j on subassembly machine k, $\forall k \in S_j$, $j \in J$;

- la_j : Lot-attached setup time for lot j on assembly machine A_j, $\forall j \in J$;
- ps_j^k : Processing time per subassembly for each product of lot j on subassembly machine k at the first stage, $\forall k \in S_j,\ j \in J$;
- pa_j : Processing time per product of lot j on assembly machine A_j at the second stage, $\forall j \in J$;
- ts_j^k : Sublot-attached setup time prior to the processing of subassemblies for a sublot of lot j on subassembly machine k at the first stage, $\forall k \in S_j,\ j \in J$;
- ta_j : Sublot-attached setup time prior to the assembling of the sublots of lot j on assembly machine A_j at the second stage, $\forall j \in J$;
- t_j^k : Sublot-attached transfer time for the sublots of lot j from the subassembly machine k, $\forall k \in S_j$, to the assembly machine a_j, $j \in J$;
- M: A large positive number;
- ε: A small positive number used as the tolerance for constraint violation

Variables:

- $x_{ij} = \begin{cases} 1, & \text{if lot } i \text{ is processed before lot } j, \\ 0, & \text{otherwise, } \forall i \in J,\ j \in J,\ i \neq j; \end{cases}$
- $z_{ju} = \begin{cases} 1, & \text{if sublot } u \text{ of lot } j \text{ is used,} \\ 0, & \text{otherwise, } \forall u \in \{1, \ldots, n_j\},\ j \in J; \end{cases}$
- s_{ju} : Size of sublot u of lot j, $\forall u \in \{1, \ldots, n_j\},\ j \in J$;
- CS_{ju}^k : Completion time of sublot u of lot j on subassembly machine k, $u = 0, \ldots, n_j, k \in S_j,\ j \in J$; a dummy sublot $u = 0$ is used to capture the start of a lot on a machine at Stage 1 and Stage 2;

- $CA_{ju}^{a_j}$: Completion time of sublot u of lot j on assembly machine a_j, $u = 0, \ldots, n_j$, $j \in J$;

- $C_{\max}$: Makespan.

Note that CS_{j0}^{k} and $CA_{j0}^{a_j}$ are the start times of lot j on subassembly machine k and assembly machine a_j, $\forall k \in S_j$, $j \in J$, respectively.

In addition, for the cost-based model, we use the following parameters.

- d_j : Due date for lot j, $\forall j \in J$;

- c_1 : Makespan cost per unit time;

- c_2 : Sublot-attached setup cost per unit time for the subassembly machines at Stage 1;

- c_3 : Sublot-attached setup cost per unit time for the assembly machines at Stage 2;

- c_4 : Lot-attached setup cost per unit time for the subassembly machines at Stage 1;

- c_5 : Lot-attached setup cost per unit time for the assembly machines at Stage 2;

- c_6 : Transfer cost per unit time between the machines at Stage 1 and Stage 2;

- c_7 : Tardiness cost per unit time;

- c_8 : Inventory cost per unit time.

Assumptions:

- The products contained in a lot are identical.

- The machines at the first and second stages are available at time zero and remain continuously available.

- A machine can process only one sublot at any time.

- The setup times, transfer times and unit processing times are known in advance.
- The processing of a sublot i on assembly machine can begin only after all the necessary subassemblies have finished processing at the first stage.
- Unlimited buffers are available on the machines at the first and second stages.

3.2.3 Mathematical Formulation

Model TSAS-MP-MIP

Objective Function

$$\text{Minimize} \quad C_{\max}, \tag{3.1}$$

Constraints

Makespan Constraint:

$$C_{\max} \geq CA^{a_j}_{jn_j}, \qquad \forall j \in J \tag{3.2}$$

Stage 1 Subassembly Machine Constraints:

$$CS^k_{j0} \geq CS^k_{in_i} + (x_{ij} - 1) \cdot M, \qquad \forall i \neq j,\ i, j \in J^k,\ k \in S \tag{3.3}$$

$$CS^k_{j1} \geq CS^k_{j0} + ls^k_j + ts^k_j + ps^k_j \cdot s_{j1}, \qquad \forall k \in S_j,\ j \in J \tag{3.4}$$

$$CS^k_{ju} \geq CS^k_{j,u-1} + ts^k_j \cdot z_{ju} + ps^k_j \cdot s_{ju}, \qquad \forall k \in S_j,\ u = 2, \ldots, n_j,\ j \in J \tag{3.5}$$

Stage 2 Assembly Machine Constraints:

$$CA_{j0}^{a_j} \geq CA_{in_i}^{a_i} + (x_{ij} - 1) \cdot M, \qquad \forall i \neq j,\ i,j \in A^k,\ k \in A \tag{3.6}$$

$$CA_{j1}^{a_j} = CA_{j0}^{a_j} + la_j + ta_j + pa_j \cdot s_{j1}, \qquad \forall j \in J \tag{3.7}$$

$$CA_{ju}^{a_j} = CA_{j,u-1}^{a_j} + ta_j \cdot z_{ju} + pa_j \cdot s_{ju}, \qquad \forall u = 2, \ldots, n_j,\ j \in J \tag{3.8}$$

Transfer Constraints:

$$CA_{j1}^{a_j} \geq CS_{j1}^{k} + t_j^k + la_j + ta_j + pa_j \cdot s_{j1}, \qquad \forall k \in S_j,\ j \in J \tag{3.9}$$

$$CA_{ju}^{a_j} \geq CS_{ju}^{k} + (ta_j + t_j^k) \cdot z_{ju} + pa_j \cdot s_{ju}, \qquad \forall k \in S_j,\ u = 2, \ldots, n_j,\ j \in J \tag{3.10}$$

Sequence Constraints:

$$x_{ij} + x_{ji} = 1, \qquad \forall i,j \in J,\ i \neq j \tag{3.11}$$

$$x_{ij} + x_{jl} + x_{li} \leq 2, \qquad \forall i,j,l \in J,\ i \neq j \neq l \tag{3.12}$$

Sublot Formation Constraints:

$$\sum_{u=1}^{n_j} s_{ju} = Q_j, \qquad \forall j \in J \tag{3.13}$$

$$s_{ju} \leq z_{ju} \cdot Q_j, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.14}$$

$$s_{ju} \geq s_{j,u+1}, \qquad \forall u = 1, \ldots, n_j - 1,\ j \in J \tag{3.15}$$

Sublot Size Constraints:

$$s_{ju} - s_{j1} \geq Q_j \left(z_{ju} - 1 \right), \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.16}$$

$$s_{ju} - s_{j1} \leq Q_j \left(1 - z_{ju} \right), \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.17}$$

$$s_{ju} \leq Q_j \cdot z_{ju}, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.18}$$

$$s_{ju} - s_{j1} \leq z_{ju} - \varepsilon, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.19}$$

$$s_{ju} \geq \varepsilon - \varepsilon \left(1 - z_{ju} \right), \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.20}$$

Range of Variables:

$$C_{\max} \geq 0,\ CS_{ju}^{k} \geq 0,\ CA_{ju}^{a_j} \geq 0 \qquad \forall k \in S_j,\ u = 0, \ldots, n_j,\ j \in J,$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in J,\ i \neq j$$

$$z_{ju} \in \{0, 1\}, \qquad \forall u \in \{1, \ldots, n_j\}, j \in J$$

$$s_{ju} \geq 0. \qquad \forall u \in \{1, \ldots, n_j\}, j \in J$$

Explanation of Constraints:

1. Makespan constraint: The makespan of the system is greater than or equal to the completion time of the last lot.

2. Stage 1 (Subassembly) and Stage 2 (Assembly) Machine Constraints: Constraint sets (3.3) and (3.6) guarantee that a lot j can start on a subassembly machine and assembly machine, respectively, only after the previous lots have finished processing on those machines. We specify later the value of M to be used in these constraints. Constraints (3.4), (3.5), (3.7), and (3.8) ensure that a sublot u of lot j can start only after sublot $u-1$ has finished processing. Note that Constraints (3.4) and (3.7) determine the start times of the first sublot at Stage 1 and Stage 2, respectively, and are different from Constraints (3.5) and (3.8) because of the presence of lot-attached setup times.

3. Transfer Constraints: Constraint sets (3.9) and (3.10) assert that sublot 1 and sublot $u \in \{2, \ldots, n_j\}$ of lot j can start processing only after all needed subassemblies have been completed and transferred to assembly machine a_j. Note that Constraint (3.9) is different from (3.10) because of the presence of lot-attached setup time.

4. Sequence Constraints: Constraint sets (3.11) capture the fact that either lot i is processed before lot j or lot j is processed before lot i. Constraint sets (3.12) are the subtour elimination constraints that enforce that if lot i precedes lot j, lot j precedes lot l, then lot l cannot precede lot i. (see Sarin et al. (2005)[76])

5. Sublot Formation Constraints: Constraint set (3.13) ensures that the summation of all the sublot sizes of a lot is equal to the total items of the lot, while Constraint set (3.14)

enforces the size of a lot to be positive only if it is formed. In case not all the sublots of a lot are formed, without loss of generality, we enforce highest numbered sublots to take a value of zero. This is assured by Constraint set (3.15).

6. Sublot Size Constraints: Constraint sets (3.16) - (3.20) capture the relationships between the sizes of sublots, s_{ju} $\forall u = 1, \ldots, n_j$, $j \in J$, and their indicator variables, $z_{ju}, \forall u = 1, \ldots, n_j$, $j \in J$. These constraints guarantee that if $s_{ju} \geq 0$, then $z_{ju} = 1$, and if $z_{ju} = 1$, then $s_{ju} > 0$. Note that we assume equal sublot sizes at the beginning, and we use s_{j1} as the size of the first sublot, and let the size of a sublot to be less than or equal to the previous sublot. Therefore, by minimizing the size of the first sublot and using indicator variables z_{ju}, the above formulation captures the equality of all the sublots that are used. We have the following four instances:
   (a) If $z_{ju} = 1$, the constraint sets (3.16) and (3.17) enforce $s_{ju} = s_{j1}$, and the constraint sets (3.18), (3.19), and (3.20) hold.
   (b) If $z_{ju} = 0$, the constraint sets (3.18) and $s_{ju} \geq 0$ guarantee $s_{ju} = 0$, and the constraint sets (3.16), (3.17), (3.19), and (3.20) hold.
   (c) If $s_{ju} = s_{j1}$, the constraint sets (3.19) assert $z_{ju} = 1$, and constraint sets (3.16), (3.17), (3.18), and (3.20) hold.
   (d) If $s_{ju} = 0$, the constraint set (3.20) enforces $z_{ju} = 0$, and constraint sets (3.16), (3.17), (3.18), and (3.19) hold.

The TSAS-CP-MIP model is different because of the objective function, which now consists of makespan cost, lot-attached and sublot-attached setup cost, transfer cost, tardiness cost, and inventory cost, and it is as follows:

Model TSAS-CP-MIP

Minimize

$$c_1 \cdot C_{\max} + c_2 \cdot \sum_{j\in J}\sum_{k\in S_j}\sum_{u=1}^{n_j} ts_j^k z_{ju} + c_3 \cdot \sum_{j\in J}\sum_{u=1}^{n_j} ta_j z_{ju} + c_4 \cdot \sum_{j\in J}\sum_{k\in S_j} ls_j^k$$

$$+c_5 \cdot \sum_{j\in J} la_j + c_6 \cdot \sum_{j\in J}\sum_{k\in S_j}\sum_{u=1}^{n_j} t_j^k z_{ju} + c_7 \cdot \sum_{j\in J} \max\left\{0, CA_{jn_j}^{a_j} - d_j\right\}$$

$$+c_8 \cdot \sum_{j\in J}\sum_{k\in S_j} Q_j \cdot \left(CA_{jn_j}^{a_j} - CS_{j0}^k - ls_j^k - (\sum_{u=1}^{n_j} z_{ju} + 1) ts_j^k/2 - ps_j^k (Q_j + s_{j1})/2 \right) \quad (3.21)$$

Subject to Constraints (3.2) - (3.20).

The first seven terms are straight-forward and they represent makespan cost, sublot-attached setup cost incurred on subassembly and assembly machines, lot-attached setup cost incurred on subassembly and assembly machines, transfer cost from Stage 1 to Stage 2, and tardiness cost, respectively. The last term captures the inventory cost accrued because of subassemblies in the system and is developed as follows:

$$\begin{aligned}
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j} \Big[s_{j1} \cdot \left(CA_{jn_j}^{a_j} - CS_{j1}^k \right) \cdot z_{j1} + s_{j1} \cdot \left(CA_{jn_j}^{a_j} - CS_{j2}^k \right) \cdot z_{j2} + \cdots + \\
&\qquad s_{j1} \cdot \left(CA_{jn_j}^{a_j} - CS_{jn_j}^k \right) \cdot z_{jn_j} \Big] \\
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j} \left[\sum_{n=1}^{n_j} z_{ju} \cdot s_{j1} \cdot CA_{jn_j}^{a_j} - s_{j1} \cdot \left(CS_{j1}^k + CS_{j2}^k + \cdots + CS_{j\sum_{u=1}^{n_j} z_{ju}}^k \right) \right]
\end{aligned}$$

$$
\begin{aligned}
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j}\Bigg[Q_j \cdot CA^{a_j}_{jn_j} - s_{j1} \cdot \Bigg[\left(CS^k_{j0} + ls^k_j + ts^k_j + ps^k_j \cdot s_{j1}\right) + \\
&\quad \left(CS^k_{j0} + ls^k_j + 2ts^k_j + 2ps^k_j \cdot s_{j1}\right) + \cdots + \left(CS^k_{j0} + ls^k_j + ts^k_j \cdot \sum_{u=1}^{n_j} z_{ju} + ps^k_j \cdot s_{j1} \cdot \sum_{u=1}^{n_j} z_{ju}\right)\Bigg]\Bigg] \\
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j}\Bigg[Q_j \cdot CA^{a_j}_{jn_j} - \Bigg[s_{j1} \cdot CS^k_{j0} \cdot \sum_{u=1}^{n_j} z_{ju} + s_{j1} \cdot ls^k_j \cdot \sum_{u=1}^{n_j} z_{ju} + \\
&\quad \frac{\left(\sum_{u=1}^{n_j} z_{ju} + 1\right) \cdot \sum_{u=1}^{n_j} z_{ju} \cdot s_{j1}}{2} \cdot ts^k_j + \frac{\left(\sum_{u=1}^{n_j} z_{ju} + 1\right) \cdot \sum_{u=1}^{n_j} z_{ju} \cdot s^2_{j1}}{2} \cdot ps^k_j\Bigg]\Bigg] \\
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j}\Bigg[Q_j \cdot CA^{a_j}_{jn_j} - \Bigg[Q_j \cdot CS^k_{j0} + Q_j \cdot ls^k_j + \frac{Q_j}{2} \cdot ts^k_j \cdot \left(\sum_{u=1}^{n_j} z_{ju} + 1\right) + \\
&\quad ps^k_j \cdot \frac{Q_j\left(Q_j + s_{j1}\right)}{2}\Bigg]\Bigg] \\
&= c_8 \cdot \sum_{j\in J}\sum_{k\in S_j} Q_j \cdot \left(CA^{a_j}_{jn_j} - CS^k_{j0} - ls^k_j - \left(\sum_{u=1}^{n_j} z_{ju} + 1\right) ts^k_j/2 - ps^k_j\left(Q_j + s_{j1}\right)/2\right)
\end{aligned}
$$

Note that both models TSAS-MP-MIP and TSAS-CP-MIP contain a large positive number M. We determine a reasonable value of M as follows: First, we define temporary parameter $M^{'}_k, k \in S$, which is the completion time of subassembly station k.

$$
M^{'}_k = \sum_{j\in J^k}\left(ps^k_j \cdot Q_j + ts^k_j \cdot n_j + ls^k_j\right), \qquad \forall k \in S \tag{3.22}
$$

Then, an upper bound of M,

$$
\hat{M} = \max_{k\in S} M^{'}_k + \max_{m\in A}\left[\sum_{j\in A^m}\left[Q_j \cdot pa_j + la_j + n_j \cdot \left(ta_j + \sum_{l\in S_j} t^l_j\right)\right]\right], \tag{3.23}
$$

Note that $\hat{M}$ consists of the maximum of the completion times for all the lots at Stage 1 plus the maximum of the transfer time, processing time, and setup times for all the lots over all the assembly stations.

3.2.4 Basic Properties

Property 3.1. *Both the TSAS-MP and TSAS-CP are NP-hard.*

Gupta (1988) [36] has shown that if one of the two stages of a two-stage hybrid flow shop has more than one machine, then the problem is NP-complete. This problem is a special case of our two-stage assembly problem. Hence, our problem is NP-hard.

Property 3.2. *There exists an optimal schedule in which the sequence between any pair of lots is identical on the subassembly and assembly machines for the TSAS-MP.*

It is easy to see that if the lots are not processed in the same sequence on the subassembly and assembly machines, then we can modify the sequences and make them identical without worsening the makespan. We also consider such sequences for the TSAS-CP problem.

3.3 Column Generation-Based Method

In this section, we introduce two column generation-based methods to solve models TSAS-MP-MIP and TSAS-CP-MIP that rely on a two-stage decomposition scheme. At Stage I (master problem), we add a column and solve the problem as a linear program to generate dual variables for use at Stage II, which generates columns to pass on to Stage I. The subproblem generates columns until a specified termination criterion is met.

3.3.1 Column Generation Method 1 (TSAS-MP-CG1 and TSAS-CP-CG1)

In this decomposition scheme, a sequence among the lots is determined in a subproblem and is passed on to the master problem, which then determines sublot sizes for each lot. Consider the following notation.

Definitions:

- ξ: A set of all generated columns.
- x_{ij}^{p} : p^{th} column of variable x_{ij}, $\forall i \in J,\ j \in J,\ i \neq j,\ p \in \xi$.
- $\lambda_p = \begin{cases} 1, & \text{if column } p \text{ is selected; } \forall p \in \xi \\ 0, & \text{otherwise.} \end{cases}$

Master Problem:
At start, we add artificial variables to make the master problem feasible. The master problem (MP-TSAS-MP-CG1) is as follows:

MP-TSAS-MP-CG1
Minimize

$$C_{\max}$$

subject to:

$$\sum_{p \in \xi} \lambda_p = 1 \tag{3.24}$$

$$C_{\max} \geq CA^{a_j}_{jn_j}, \qquad \forall j \in J \tag{3.25}$$

$$CS^k_{j0} \geq CS^k_{in_i} + (\sum_{p \in \xi} \lambda_p x^p_{ij} - 1) \cdot \hat{M} \qquad \forall i \neq j,\ i, j \in J^k,\ k \in S \tag{3.26}$$

$$CS^k_{j1} \geq CS^k_{j0} + ls^k_j + ts^k_j + ps^k_j \cdot s_{j1}, \qquad \forall k \in S_j,\ j \in J \tag{3.27}$$

$$CS^k_{ju} \geq CS^k_{j,u-1} + ts^k_j \cdot z_{ju} + ps^k_j \cdot s_{ju}, \qquad \forall k \in S_j,\ u = 2, \ldots, n_j,\ j \in J \tag{3.28}$$

$$CA^{a_j}_{j0} \geq CA^{a_i}_{in_i} + (\sum_{p \in \xi} \lambda_p x^p_{ij} - 1) \cdot \hat{M}, \qquad \forall i \neq j,\ i, j \in A^k,\ k \in A \tag{3.29}$$

$$CA^{a_j}_{j1} = CA^{a_j}_{j0} + la_j + ta_j + pa_j \cdot s_{j1}, \qquad \forall j \in J \tag{3.30}$$

$$CA^{a_j}_{ju} = CA^{a_j}_{j,u-1} + ta_j \cdot z_{ju} + pa_j \cdot s_{ju}, \qquad \forall u = 2, \ldots, n_j,\ j \in J \tag{3.31}$$

$$CA^{a_j}_{j1} \geq CS^k_{j1} + t^k_j + la_j + ta_j + pa_j \cdot s_{j1}, \ \forall k \in S_j,\ j \in J \tag{3.32}$$

$$CA^{a_j}_{ju} \geq CS^k_{ju} + (ta_j + t^k_j) \cdot z_{ju} + pa_j \cdot s_{ju}, \forall k \in S_j,\ u = 2, \ldots, n_j,\ j \in J \tag{3.33}$$

$$\sum_{u=1}^{n_j} s_{ju} = Q_j, \qquad \forall j \in J \tag{3.34}$$

$$s_{ju} \leq z_{ju} \cdot Q_j, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.35}$$

$$s_{ju} \geq s_{j,u+1}, \qquad \forall u = 1, \ldots, n_j - 1,\ j \in J \tag{3.36}$$

$$s_{ju} - s_{j1} \geq Q_j \left(z_{ju} - 1 \right), \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.37}$$

$$s_{ju} - s_{j1} \leq Q_j \left(1 - z_{ju} \right), \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.38}$$

$$s_{ju} \leq Q_j \cdot z_{ju}, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.39}$$

$$s_{ju} - s_{j1} \leq z_{ju} - \varepsilon, \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.40}$$

$$s_{ju} \geq \varepsilon - \varepsilon \left(1 - z_{ju} \right). \qquad \forall u = 1, \ldots, n_j,\ j \in J \tag{3.41}$$

Subproblem:

Let π, ϕ^k_{ij}, and φ_{ij} be the dual variables associated with Constraint sets (3.24), (3.26), and (3.29). We have the following subproblem. The objective function is the reduced cost of λ

variable.

SP-TSAS-CP-CG1

Minimize

$$\sum_{i \in J} \sum_{j \in J, j \neq i} \sum_{k \in S_j} \hat{M} \cdot \left(\phi_{ij}^{k} + \varphi_{ij}\right) \cdot x_{ij} - \pi$$

Subject to:

$$\begin{aligned} & x_{ij} + x_{ji} = 1, && \forall i \neq j,\ i, j \in J \\ & x_{ij} + x_{jl} + x_{li} \leq 2, && \forall i \neq j \neq l,\ i, j, l \in J \\ & x_{ij} \in \{0, 1\}. && \forall i \neq j,\ i, j \in J \end{aligned}$$

If the reduced cost obtained from the SP-TSAS-MP-CG1 is negative, we add the corresponding column, designated x_{ij}^{p} to the master problem (MP-TSAS-MP-CG1), and then, re-solve MP-TSAS-MP-CG1 to generate a dual solution. This process is continued until the reduced cost becomes non-negative or the maximum allowable number of iterations is reached.

TSAS-CP-CG1 is the same as the TSAS-MP-CG1 except that the objective function of the master problem is different. The model for MP-TSAS-CP-CG1 is as follows:

Minimize

$$c_1 \cdot C_{\max} + c_2 \cdot \sum_{j \in J} \sum_{k \in S_j} \sum_{u=1}^{n_j} ts_j^k z_{ju} + c_3 \cdot \sum_{j \in J} \sum_{u=1}^{n_j} ta_j z_{ju} + c_4 \cdot \sum_{j \in J} \sum_{k \in S_j} ls_j^k$$

$$+c_5 \cdot \sum_{j \in J} la_j + c_6 \cdot \sum_{j \in J} \sum_{k \in S_j} \sum_{u=1}^{n_j} t_j^k z_{ju} + c_7 \cdot \sum_{j \in J} \max \left\{ 0, CA_{jn_j}^{a_j} - d_j \right\}$$

$$+c_8 \cdot \sum_{j \in J} \sum_{k \in S_j} Q_j \cdot \left(CA_{jn_j}^{a_j} - CS_{j0}^k - ls_j^k - (\sum_{u=1}^{n_j} z_{ju} + 1) ts_j^k / 2 - ps_j^k (Q_j + s_{j1})/2 \right).$$

Subject to Constraints (3.24) - (3.41).

3.3.2 Column Generation Method 2 (TSAS-MP-CG2 and TSAS-CP-CG2)

In this section, we propose another column generation-based method for both the problems and designate them as TSAS-MP-CG2 and TSAS-CP-CG2. The problem is decomposed around the formation of sublots for each lot, which thus constitutes the subproblem. The master problem, then, determines a sequence in which to process the lots.

Definitions:

- δ: A set of all generated columns.
- z_{ju}^p : p^{th} column of variable z_{ju}, $\forall j \in J,\ u = 1, \ldots, n_j,\ p \in \delta$.
- $\lambda_p = \begin{cases} 1, & \text{if column } p \text{ is selected,} \\ 0, & \text{otherwise, } \forall p \in \delta. \end{cases}$

Master Problem:

MP-TSAS-MP-CG2

Minimize

$$C_{\max}$$

subject to:

$$\sum_{p\in\delta}\lambda_p = 1 \tag{3.42}$$

$$C_{\max} \geq CA^{a_j}_{jn_j}, \qquad \forall j \in J \tag{3.43}$$

$$CS^k_{j0} \geq CS^k_{in_i} + (x_{ij}-1)\cdot\hat{M} \qquad \forall i \neq j,\ i,j \in J^k,\ k \in S \tag{3.44}$$

$$CS^k_{j1} \geq CS^k_{j0} + ls^k_j + ts^k_j + ps^k_j \cdot s_{j1}, \qquad \forall k \in S_j,\ j \in J \tag{3.45}$$

$$CS^k_{ju} \geq CS^k_{j,u-1} + \sum_{p\in\delta}\lambda_p \cdot ts^k_j \cdot z^p_{ju} + ps^k_j \cdot s_{ju}, \qquad \forall k \in S_j,\ u = 2,\ldots,n_j,\ j \in J \tag{3.46}$$

$$CA^{a_j}_{j0} \geq CA^{a_i}_{in_i} + (x_{ij}-1)\cdot\hat{M}, \qquad \forall i \neq j,\ i,j \in A^k,\ k \in A \tag{3.47}$$

$$CA^{a_j}_{j1} = CA^{a_j}_{j0} + la_j + ta_j + pa_j \cdot s_{j1}, \qquad \forall j \in J \tag{3.48}$$

$$CA^{a_j}_{ju} = CA^{a_j}_{j,u-1} + \sum_{p\in\delta}\lambda_p \cdot ta_j \cdot z^p_{ju} + pa_j \cdot s_{ju}, \qquad \forall u = 2,\ldots,n_j,\ j \in J \tag{3.49}$$

$$CA^{a_j}_{j1} \geq CS^k_{j1} + t^k_j + la_j + ta_j + pa_j \cdot s_{j1}, \qquad \forall k \in S_j,\ j \in J \tag{3.50}$$

$$CA^{a_j}_{ju} \geq CS^k_{ju} + \sum_{p\in\delta}\lambda_p \cdot (ta_j + t^k_j)\cdot z^p_{ju} + pa_j \cdot s_{ju}, \forall k \in S_j,\ u = 2,\ldots,n_j,\ j \in J \tag{3.51}$$

$$x_{ij} + x_{ji} = 1, \qquad \forall i \neq j,\ i,j \in J \tag{3.52}$$

$$x_{ij} + x_{jl} + x_{li} \leq 2, \qquad \forall i \neq j \neq l,\ i,j,l \in J \tag{3.53}$$

Subproblem:

Let π, ϕ^k_{ju}, φ_{ju}, and ψ^k_{ju} be the dual variables associated with Constraint sets (3.42), (3.46), (3.49), and (3.51). The objective function is the reduced cost of the λ variable.

SP-TSAS-MP-CG2

Minimize

$$\sum_{j\in J}\sum_{u\in 1,\ldots,n_j}\sum_{k\in S_j}\left[ts_j^k\cdot\phi_{ju}^k+ta_j\cdot\varphi_{ju}+(ta_j+t_j^k)\cdot\psi_{ju}^k\right]z_{ju}-\pi$$

Subject to:

$$
\begin{aligned}
&\sum_{u=1}^{n_j}s_{ju}=Q_j, && \forall j\in J\\
&s_{ju}\le z_{ju}\cdot Q_j, && \forall u=1,\ldots,n_j,\ j\in J\\
&s_{ju}\ge s_{j,u+1}, && \forall u=1,\ldots,n_j-1,\ j\in J\\
&s_{ju}-s_{j1}\ge Q_j\left(z_{ju}-1\right), && \forall u=1,\ldots,n_j,\ j\in J\\
&s_{ju}-s_{j1}\le Q_j\left(1-z_{ju}\right), && \forall u=1,\ldots,n_j,\ j\in J\\
&s_{ju}\le Q_j\cdot z_{ju}, && \forall u=1,\ldots,n_j,\ j\in J\\
&s_{ju}-s_{j1}\le z_{ju}-\varepsilon, && \forall u=1,\ldots,n_j,\ j\in J\\
&s_{ju}\ge \varepsilon-\varepsilon\left(1-z_{ju}\right). && \forall u=1,\ldots,n_j,\ j\in J\\
&z_{ju}\in\{0,1\} && \forall u=1,\ldots,n_j,\ j\in J
\end{aligned}
$$

TSAS-CP-CG2 is the same as the TSAS-MP-CG2 except that the objective function of the master problem is different. The model for MP-TSAS-CP-CG2 is as follows:

Minimize

$$c_1\cdot C_{\max}+c_2\cdot\sum_{p\in\delta}\sum_{j\in J}\sum_{k\in S_j}\sum_{u=1}^{n_j}\lambda_p ts_j^k z_{ju}^p+c_3\cdot\sum_{p\in\delta}\sum_{j\in J}\sum_{u=1}^{n_j}\lambda_p ta_j z_{ju}^p+c_4\cdot\sum_{j\in J}\sum_{k\in S_j}ls_j^k$$

$$+c_5 \cdot \sum_{j\in J} la_j + c_6 \cdot \sum_{p\in\delta}\sum_{j\in J}\sum_{k\in S_j}\sum_{u=1}^{n_j} \lambda_p t_j^k z_{ju}^p + c_7 \cdot \sum_{j\in J} \max\left\{0, CA_{jn_j}^{a_j} - d_j\right\}$$

$$+c_8 \cdot \sum_{j\in J}\sum_{k\in S_j} Q_j \cdot \left(CA_{jn_j}^{a_j} - CS_{j0}^k - ls_j^k - (\sum_{p\in\delta}\sum_{u=1}^{n_j} z_{ju}^p + 1)ts_j^k/2 - ps_j^k(Q_j + s_{j1})/2 \right).$$

Subject to Constraints (3.42) - (3.53)

3.3.3 Outline of the Column Generation Approach

- Step 1: Generate a set δ of initial columns for TSAS-MP-CG1 or TSAS-MP-CG2, respectively.
- Step 2. Solve LP relaxation of the master problem, and determine the values of dual variables.
- Step 3. Solve the subproblem by using the values of dual variables from Step 2 to obtain the minimum reduced cost. If the reduced cost is non-negative or the number of iterations is greater than or equal to the maximum number of iterations allowed, then go to Step 4; else, append the column generated from subproblem to the master problem, and go to Step 2.
- Step 4. Solve the master problem as an integer program to optimality, and terminate.

The steps for TSAS-CP-CG1 and TSAS-CP-CG2 are exactly the same except for the use of these formulations at Step 1.

3.4 Computational Investigation

In this section, we present the results of our computational investigation to study the CPU time required by direct solution of TSAS-MP-MIP and TSAS-CP-MIP by the OPL CPLEX Solver (version 12.4). We also study the performances of the proposed column generation methods. For the column generation methods, we imposed a maximum of 10 iterations. These methods were programmed using Visual C++ (Version 2008) and OPL CPLEX Solver (version 12.4). Besides the average CPU time, we also present the gap between the column generation solution and optimal solution values. The test data used is presented in Table 3.1.

We used four instances consisting of 7, 8, 9, and 10 lots. For the instance of 7 lots, we consider 5 subassembly machines at the first stage and 2 assembly machines at the second stage; while for the instance of 8 lots, we consider 6 subassembly machines at the first stage and 3 assembly machines at the second stage. For the bigger-sized problems, which consist of 9 and 10 lots, we consider 7 subassembly machines at Stage 1 and 3 assembly machines at Stage 2.

For each instance of number of lots, three different number of maximum sublots were used, namely, small, medium, and large. These values are 10, 15 and 20. Thus, there are a total of $4 \times 3 = 12$ datasets for both makespan and cost-based models. For each dataset, we used five randomly generated combinations of lot sizes, processing times, setup times, and transfer times, thereby, leading to 60 instances. All numerical tests were executed on a Sony computer with Intel i7 Q740 CPU and 8GB DDR3 memory.

Table 3.1: Data used for computational investigation

Number of jobs (J)		7, 8, 9, and 10
Maximum number of sublots (n_j)	High	20
	Medium	15
	Low	10
Number of products in a lot (Q_j)		Uniform distribution [1000,2000]
Lot-attached setup time $\left(ls_j^k, la_j\right)$		Uniform distribution [5.0,10.0]
Sublot-attached setup time $\left(ts_j^k, ta_j\right)$		Uniform distribution [0.1,2.0]
Processing time per item $\left(ps_j^k, pa_j\right)$		Uniform distribution [0.1,2.0]
Transfer time per sublot $\left(t_j^k\right)$		Uniform distribution [5.0,10.0]
Cost coefficient $(c_1, c_2, c_3, c_4, c_5, c_6, c_7)$		Uniform distribution [0.1,1.0]
Cost coefficient (c_8)		Uniform distribution [0.001,0.01]

3.4.1 Results for TSAS-MP-MIP, TSAS-MP-CG1 and TSAS-MP-CG2

Comparison of CPU times required by direct solution of MIP formulation by CPLEX and column generation methods

In this section, we compare the performances of the TSAS-MP-MIP, TSAS-MP-CG1, and TSAS-MP-CG2 formulations for the objective of minimizing the makespan. Tables 3.2, 3.3, 3.4, and 3.5 depict the CPU times for 60 different datasets grouped by different number of jobs and maximum number of sublots. The smaller CPU times are highlighted. Note that in 49 of the 60 instances, TSAS-MP-CG1 requires a smaller CPU time while TSAS-MP-CG2 requires a smaller time in 11 instances. The column generation methods dominate the original MIP model in view of CPU times for all 60 instances. For example, the instances 24, 35, 45, 46, 48, and 57 cannot be solved directly by CPLEX in 3600 seconds. However, both TSAS-MP-CG1 and TSAS-MP-CG2 could obtain solutions within 150 seconds. On average, TSAS-MP-MIP requires 1042.8 seconds of CPU time, while TSAS-MP-CG1 and TSAS-MP-CG2 require 20.3 and 157.8 seconds of CPU time, respectively.

Performance comparison between TSAS-MP-CG1 and TSAS-MP-CG2

Since the column generation methods may not obtain optimal solutions for all instances, we present the values of optimality gap between optimal solutions and those obtained by TSAS-MP-CG1 and TSAS-MP-CG2. These are shown in Tables 3.6, 3.7, 3.8, and 3.9. The optimality gap is obtained as follows:

$$\text{Optimality Gap} = \frac{\text{Solution Value - Optimal Value}}{\text{Optimal Value}}$$

The smallest optimality gap values are highlighted in Tables 3.6, 3.7, 3.8, and 3.9. Note that for instances 1, 11, 29, 32, and 52, TSAS-MP-CG1 obtains optimal solutions, while for instances 11, 18, 23, 33, and 34, TSAS-MP-CG2 obtains optimal solutions. The gap values between optimal solutions and those obtained by our two column generation methods are within 8.3% for TSAS-MP-CG1 and within 1.9% for TSAS-MP-CG2. The results of CPU times and the optimality gap values are also depicted graphically in Figures 3.4 and 3.5. (Note that instance 24, 35, 45, 46, 48, 50, 54, 56, 57, 58, 59, and 60 are omitted from Figure 3.5 since optimal solutions are not available.) Both TSAS-MP-CG1 and TSAS-MP-CG2 were found to perform quite well. Better optimality gap values are obtained for TSAS-MP-CG2, while TSAS-MP-CG1 requires smaller CPU times, in particular, 20.3 seconds on average, against 157.8 seconds for TSAS-MP-CG2.

Table 3.2: CPU times (in seconds) required for direct solution of TSAS-MP-MIP by CPLEX and by TSAS-MP-CG1 and TSAS-MP-CG2

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times (in seconds)		
					TSAS-MP-MIP	TSAS-MP-CG1	TSAS-MP-CG2
1	7	5	2	10	6.97	3.184	**3.06**
2	7	5	2	10	9.22	**0.791**	2.373
3	7	5	2	10	6.6	0.854	**0.843**
4	7	5	2	10	5.8	1.234	**1.118**
5	7	5	2	10	11.9	**1.417**	2.769
6	7	5	2	15	7.04	**1.469**	1.667
7	7	5	2	15	12.54	**3.837**	4.05
8	7	5	2	15	12.28	**2.821**	3.487
9	7	5	2	15	32.54	**1.836**	3.281
10	7	5	2	15	25.43	**1.792**	3.029
11	7	5	2	20	7.89	3.527	**1.873**
12	7	5	2	20	3.71	3.389	**1.79**
13	7	5	2	20	2.51	**1.608**	1.617
14	7	5	2	20	20.62	**4.075**	4.098
15	7	5	2	20	6.94	2.219	**1.565**

Table 3.3: CPU times (in seconds) required for direct solution of TSAS-MP-MIP by CPLEX and by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times (in seconds)		
					TSAS-MP-MIP	TSAS-MP-CG1	TSAS-MP-CG2
16	8	6	3	10	84.65	**5.125**	7.958
17	8	6	3	10	154.44	**3.57**	49.511
18	8	6	3	10	48.47	**2.21**	15.52
19	8	6	3	10	103.21	**6.716**	48.331
20	8	6	3	10	38.92	**4.137**	10.575
21	8	6	3	15	12.67	**1.939**	2.982
22	8	6	3	15	43.77	**2.339**	8.597
23	8	6	3	15	11.28	**1.483**	2.532
24	8	6	3	15	$3600^{\dagger}$	**8.393**	45.98
25	8	6	3	15	41.18	7.122	**2.601**
26	8	6	3	20	17.63	6.106	**3.131**
27	8	6	3	20	785.64	**5.596**	6.379
28	8	6	3	20	534.3	**53.999**	60.302
29	8	6	3	20	40.22	20.93	**9.33**
30	8	6	3	20	41.18	7.122	**2.601**

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.4: CPU times (in seconds) required for direct solution of TSAS-MP-MIP by CPLEX and by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times (in seconds)		
					TSAS-MP-MIP	TSAS-MP-CG1	TSAS-MP-CG2
31	9	7	3	10	50.61	**3.022**	8.462
32	9	7	3	10	1942.62	**11.878**	520.924
33	9	7	3	10	59.27	**7.728**	15.504
34	9	7	3	10	277.03	**13.511**	47.723
35	9	7	3	10	$3600^{\dagger}$	**16.467**	143.434
36	9	7	3	15	74.1	**4.532**	8.246
37	9	7	3	15	572.16	**7.127**	66.619
38	9	7	3	15	318.85	**7.15**	20.154
39	9	7	3	15	1325.46	**13.141**	70.821
40	9	7	3	15	470.52	**4.092**	14.977
41	9	7	3	20	1482.01	**4.422**	76.108
42	9	7	3	20	1805.23	**16.171**	34.04
43	9	7	3	20	1706.45	**8.879**	26.454
44	9	7	3	20	1590.82	**11.672**	76.16
45	9	7	3	20	$3600^{\dagger}$	33.124	**16.639**

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.5: CPU times (in seconds) required for direct solution of TSAS-MP-MIP by CPLEX and by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times (in seconds)		
					TSAS-MP-MIP	TSAS-MP-CG1	TSAS-MP-CG2
46	10	7	3	10	$3600^{\dagger}$	**44.031**	149.662
47	10	7	3	10	1841.23	**6.57**	544.338
48	10	7	3	10	$3600^{\dagger}$	**5.287**	111.645
49	10	7	3	10	12.2	**1.649**	4.311
50	10	7	3	10	$3600^{\dagger}$	**5.184**	719.183
51	10	7	3	15	299.72	**6.576**	61.627
52	10	7	3	15	1078.61	**123.002**	211.661
53	10	7	3	15	549.17	**11.105**	57.957
54	10	7	3	15	$3600^{\dagger}$	**487.468**	1247.59
55	10	7	3	15	1780.52	**6.943**	45.83
56	10	7	3	20	$3600^{\dagger}$	**25.764**	2000.51
57	10	7	3	20	$3600^{\dagger}$	**31.493**	77.42
58	10	7	3	20	$3600^{\dagger}$	**33.5**	980.935
59	10	7	3	20	$3600^{\dagger}$	**71.292**	340.9
60	10	7	3	20	$3600^{\dagger}$	**23.446**	1455.2

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.6: Comparison between optimality gap values for the solutions obtained by TSAS-MP-CG1 and TSAS-MP-CG2

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-MP-CG1	TSAS-MP-CG2
1	7	5	2	10	**0.000%**	0.118%
2	7	5	2	10	0.343%	**0.315%**
3	7	5	2	10	0.599%	**0.414%**
4	7	5	2	10	4.447%	**0.155%**
5	7	5	2	10	0.376%	**0.181%**
6	7	5	2	15	6.154%	**0.642%**
7	7	5	2	15	1.050%	**0.382%**
8	7	5	2	15	**0.120%**	0.974%
9	7	5	2	15	**0.019%**	0.687%
10	7	5	2	15	2.749%	**0.090%**
11	7	5	2	20	0.000%	0.000%
12	7	5	2	20	4.652%	**1.649%**
13	7	5	2	20	**0.040%**	0.397%
14	7	5	2	20	0.240%	**0.169%**
15	7	5	2	20	**1.006%**	1.862%

Table 3.7: Comparison between optimality gap values for the solutions obtained by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-MP-CG1	TSAS-MP-CG2
16	8	6	3	10	1.352%	**0.975%**
17	8	6	3	10	**0.011%**	0.591%
18	8	6	3	10	0.326%	**0.000%**
19	8	6	3	10	**0.286%**	0.306%
20	8	6	3	10	**0.009%**	0.282%
21	8	6	3	15	6.989%	**0.538%**
22	8	6	3	15	**0.578%**	0.591%
23	8	6	3	15	5.511%	**0.000%**
24	8	6	3	15	N/A	N/A
25	8	6	3	15	3.374%	**0.236%**
26	8	6	3	20	5.805%	**0.122%**
27	8	6	3	20	2.975%	**0.276%**
28	8	6	3	20	**0.037%**	0.299%
29	8	6	3	20	**0.000%**	0.378%
30	8	6	3	20	3.374%	**0.236%**

N/A: Not available since optimal solution cannot be obtained

Table 3.8: Comparison between optimality gap values for the solutions obtained by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-MP-CG1	TSAS-MP-CG2
31	9	7	3	10	8.227%	**0.401%**
32	9	7	3	10	**0.000%**	0.184%
33	9	7	3	10	0.770%	**0.000%**
34	9	7	3	10	2.001%	**0.000%**
35	9	7	3	10	N/A	N/A
36	9	7	3	15	0.401%	**0.308%**
37	9	7	3	15	**0.278%**	0.285%
38	9	7	3	15	**0.003%**	0.031%
39	9	7	3	15	4.353%	**0.014%**
40	9	7	3	15	0.961%	0.961%
41	9	7	3	20	0.348%	**0.137%**
42	9	7	3	20	0.113%	**0.104%**
43	9	7	3	20	2.606%	**0.339%**
44	9	7	3	20	5.700%	**0.197%**
45	9	7	3	20	N/A	N/A

N/A: Not available since optimal solution cannot be obtained

Table 3.9: Comparison between optimality gap values for the solutions obtained by TSAS-MP-CG1 and TSAS-MP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-MP-CG1	TSAS-MP-CG2
46	10	7	3	10	N/A	N/A
47	10	7	3	10	0.076%	**0.006%**
48	10	7	3	10	N/A	N/A
49	10	7	3	10	0.069%	**0.047%**
50	10	7	3	10	N/A	N/A
51	10	7	3	15	**0.024%**	1.451%
52	10	7	3	15	**0.000%**	0.190%
53	10	7	3	15	7.978%	**1.440%**
54	10	7	3	15	N/A	N/A
55	10	7	3	15	**0.016%**	0.541%
56	10	7	3	20	N/A	N/A
57	10	7	3	20	N/A	N/A
58	10	7	3	20	N/A	N/A
59	10	7	3	20	N/A	N/A
60	10	7	3	20	N/A	N/A

N/A: Not available since optimal solution cannot be obtained

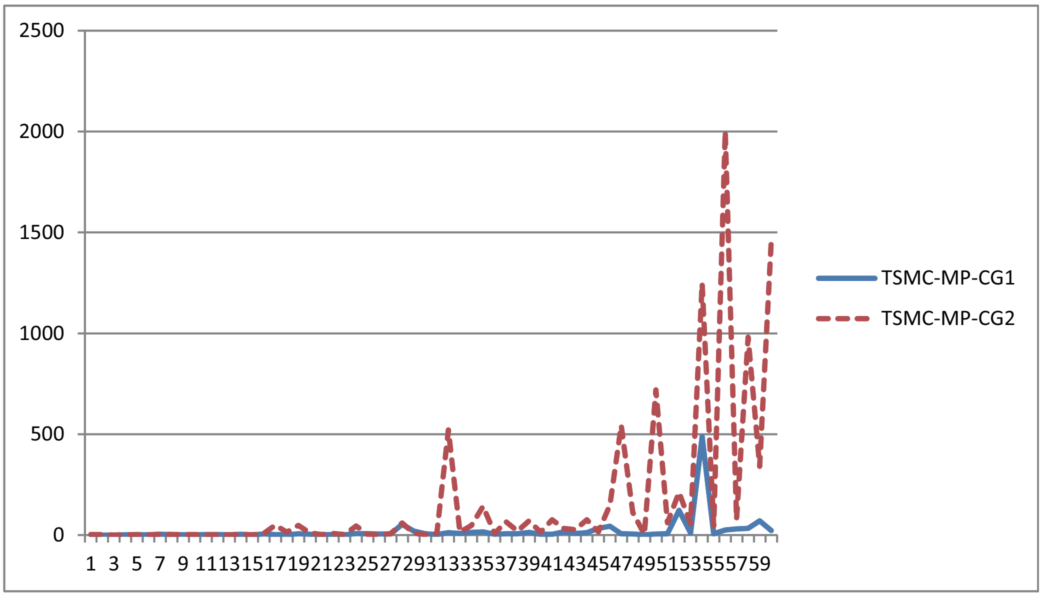

Figure 3.4: CPU times (in seconds) required by TSAS-MP-CG1 and TSAS-MP-CG2

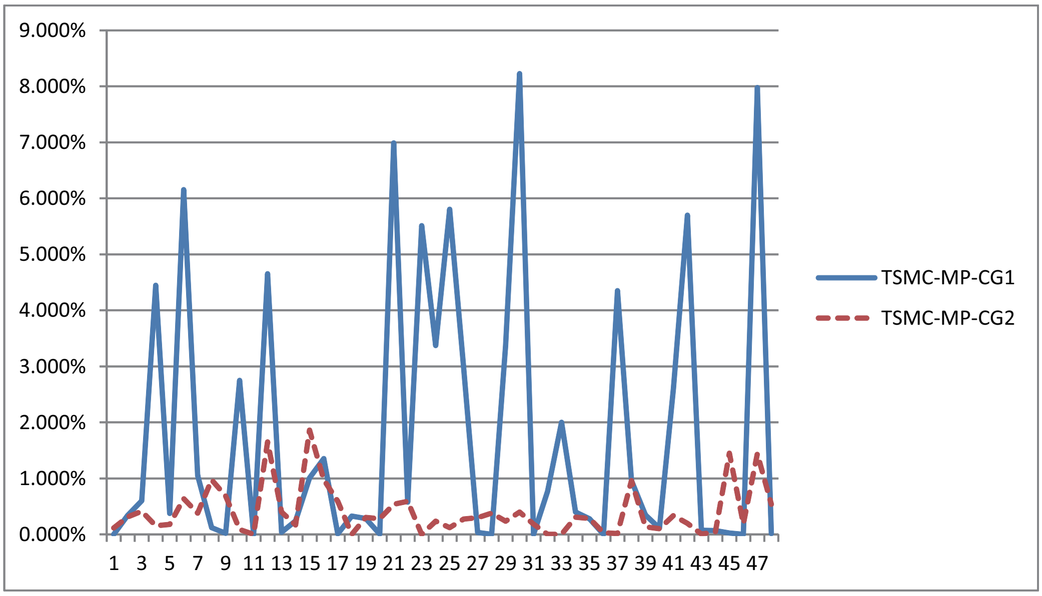

Figure 3.5: Optimality gap values for the solutions obtained by TSAS-MP-CG1 and TSAS-MP-CG2

3.4.2 Results for TSAS-CP-MIP, TSAS-CP-CG1 and TSAS-CP-CG2

Comparison of CPU time between MIP and column generation methods

As in the last section, we compare the performances of CPU times required by the direct solution by CPLEX of model TSAS-CP-MIP, and by TSAS-CP-CG1 and TSAS-CP-CG2. The smaller CPU times are highlighted in Tables 3.10, 3.11, 3.12, and 3.13. Since the cost-based model is much more complex than the makespan model, TSAS-CP-MIP could not solve 40 out of 60 instances within the specified time limit of 3600 seconds. TSAS-CP-CG2 performs better than TSAS-CP-CG1 for 31 out of 60 instances, while TSAS-CP-CG1 performs better for 26 out of 60 instances. The results of CPU times are depicted graphically in Figure 3.6. The average CPU time required by direct solution by CPLEX is 2678.5 seconds, while TSAS-CP-CG1 and TSAS-CP-CG2 require 860.5 and 412.3 seconds on average, respectively.

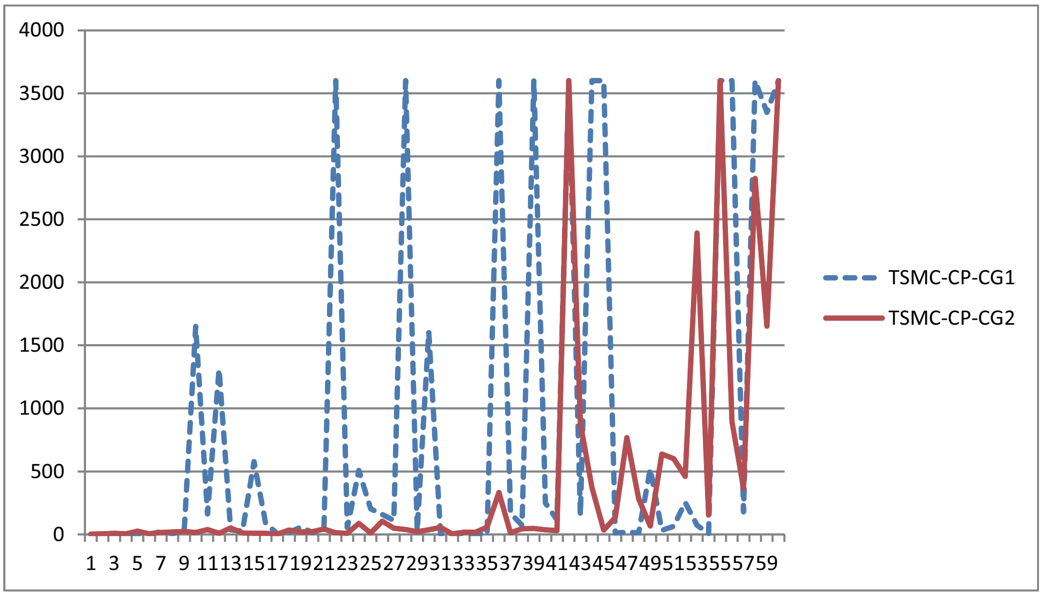

Figure 3.6: CPU times (in seconds) required by TSAS-CP-CG1 and TSAS-CP-CG2

Table 3.10: CPU times (in seconds) required for direct solution of TSAS-CP-MIP by CPLEX and by TSAS-CP-CG1 and TSAS-CP-CG2

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times		
					TSAS-CP-MIP	TSAS-CP-CG1	TSAS-CP-CG2
1	7	5	2	10	29.78	**0.978**	3.416
2	7	5	2	10	19.42	**1.423**	4.115
3	7	5	2	10	40.81	**7.018**	9.683
4	7	5	2	10	12	**2.075**	4.229
5	7	5	2	10	47.07	**1.909**	25.431
6	7	5	2	15	37.49	**2.446**	4.02
7	7	5	2	15	820.32	22.041	**15.098**
8	7	5	2	15	221.1	**7.399**	19.215
9	7	5	2	15	407.85	**20.799**	24.734
10	7	5	2	15	$3600^{\dagger}$	1651.03	**14.989**
11	7	5	2	20	1305.65	162.422	**39.725**
12	7	5	2	20	3600	1312.09	**7.993**
13	7	5	2	20	299.32	**36.461**	51.519
14	7	5	2	20	332.64	26.594	**11.316**
15	7	5	2	20	1429.44	577.436	**9.171**

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.11: CPU times (in seconds) required for direct solution of TSAS-CP-MIP by CPLEX and by TSAS-CP-CG1 and TSAS-CP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times		
					TSAS-CP-MIP	TSAS-CP-CG1	TSAS-CP-CG2
16	8	6	3	10	$3600^{\dagger}$	90.51	**8.821**
17	8	6	3	10	1852.81	3.906	**3.701**
18	8	6	3	10	$3600^{\dagger}$	**15.276**	34.879
19	8	6	3	10	919.36	58.058	**19.822**
20	8	6	3	10	$3600^{\dagger}$	**15.084**	21.601
21	8	6	3	15	1553.24	43.77	**43.07**
22	8	6	3	15	$3600^{\dagger}$	$3600^{\dagger}$	**12.678**
23	8	6	3	15	$3600^{\dagger}$	53.538	**9.596**
24	8	6	3	15	$3600^{\dagger}$	508.004	**86.831**
25	8	6	3	15	1556	203.858	**11.653**
26	8	6	3	20	309.55	157.736	**105.427**
27	8	6	3	20	1916.58	110.69	**49.101**
28	8	6	3	20	$3600^{\dagger}$	$3600^{\dagger}$	**39.55**
29	8	6	3	20	$3600^{\dagger}$	29.828	**21.897**
30	8	6	3	20	$3600^{\dagger}$	1607.98	**36.289**

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.12: CPU times (in seconds) required for direct solution of TSAS-CP-MIP by CPLEX and by TSAS-CP-CG1 and TSAS-CP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times		
					TSAS-CP-MIP	TSAS-CP-CG1	TSAS-CP-CG2
31	9	7	3	10	$3600^{\dagger}$	**1.711**	56.053
32	9	7	3	10	$3600^{\dagger}$	**1.336**	4.955
33	9	7	3	10	$3600^{\dagger}$	**5.025**	18.316
34	9	7	3	10	$3600^{\dagger}$	**4.887**	17.186
35	9	7	3	10	$3600^{\dagger}$	**3.675**	58.207
36	9	7	3	15	$3600^{\dagger}$	$3600^{\dagger}$	**332.86**
37	9	7	3	15	$3600^{\dagger}$	169.552	**13.11**
38	9	7	3	15	$3600^{\dagger}$	72.649	**45.759**
39	9	7	3	15	$3600^{\dagger}$	$3600^{\dagger}$	**47.386**
40	9	7	3	15	$3600^{\dagger}$	248.85	**37.768**
41	9	7	3	20	$3600^{\dagger}$	116.499	**29.63**
42	9	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	$3600^{\dagger}$
43	9	7	3	20	$3600^{\dagger}$	**151.478**	882.337
44	9	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	**374.267**
45	9	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	**35.367**

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 3.13: CPU times (in seconds) required for direct solution of TSAS-CP-MIP by CPLEX and by TSAS-CP-CG1 and TSAS-CP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	CPU Times		
					TSAS-CP-MIP	TSAS-CP-CG1	TSAS-CP-CG2
46	10	7	3	10	$3600^{\dagger}$	**10.815**	134.652
47	10	7	3	10	$3600^{\dagger}$	**18.514**	767.873
48	10	7	3	10	$3600^{\dagger}$	**12.653**	281.268
49	10	7	3	10	$3600^{\dagger}$	521.46	**67.444**
50	10	7	3	10	$3600^{\dagger}$	**32.234**	637.704
51	10	7	3	15	$3600^{\dagger}$	**63.506**	602.405
52	10	7	3	15	$3600^{\dagger}$	**254.513**	461.899
53	10	7	3	15	$3600^{\dagger}$	**72.502**	2390.98
54	10	7	3	15	$3600^{\dagger}$	**5.991**	154.389
55	10	7	3	15	$3600^{\dagger}$	$3600^{\dagger}$	$3600^{\dagger}$
56	10	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	**896.188**
57	10	7	3	20	$3600^{\dagger}$	**180.199**	365.079
58	10	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	**2824.1**
59	10	7	3	20	$3600^{\dagger}$	3348.84	**1651.3**
60	10	7	3	20	$3600^{\dagger}$	$3600^{\dagger}$	$3600^{\dagger}$

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Comparison of optimality gap between TSAS-CP-CG1 and TSAS-CP-CG2

Next, we study the optimality gap values between optimal solutions and those obtained for TSAS-CP-CG1 and TSAS-CP-CG2. These are shown in Tables 3.14 and 3.15. For the cost-based model, TSAS-CP-CG2 dominates TSAS-CP-CG1 for all instances. The optimality gap values could be obtained only for 19 out of 60 instances, since 41 instances could not be solved to optimality. A comparison of the optimality gap values for these 19 instances between TSAS-CP-CG1 and TSAS-CP-CG2 is depicted in Figure 3.7. Note that the worst optimality gap values for TSAS-CP-CG2 is within 1.8%, while that for model TSAS-CP-CG1 is above 6%. Therefore, the TSAS-CP-CG2 method is much more suitable for the cost-based objective function.

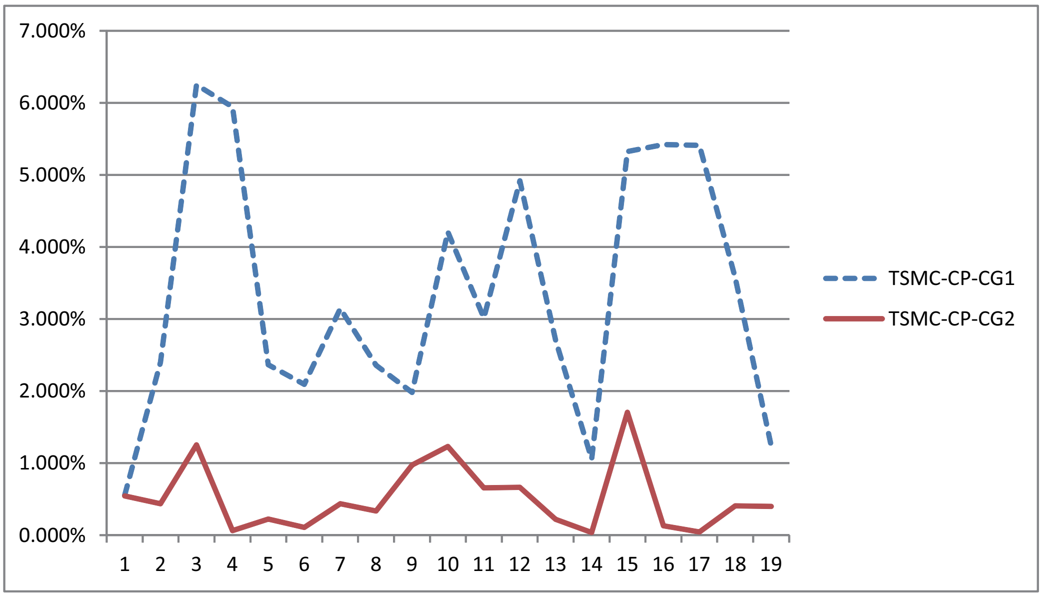

Figure 3.7: Comparison of optimality gap values for the solutions obtained by TSAS-CP-CG1 and TSAS-CP-CG2

Table 3.14: Comparison between optimality gap values for the solutions obtained by TSAS-CP-CG1 and TSAS-CP-CG2

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-CP-CG1	TSAS-CP-CG2
1	7	5	2	10	0.556%	**0.547%**
2	7	5	2	10	2.399%	**0.438%**
3	7	5	2	10	6.250%	**1.255%**
4	7	5	2	10	5.941%	**0.062%**
5	7	5	2	10	2.365%	**0.224%**
6	7	5	2	15	2.095%	**0.108%**
7	7	5	2	15	3.146%	**0.439%**
8	7	5	2	15	2.358%	**0.337%**
9	7	5	2	15	1.981%	**0.974%**
10	7	5	2	15	N/A	N/A
11	7	5	2	20	4.206%	**1.233%**
12	7	5	2	20	N/A	N/A
13	7	5	2	20	3.015%	**0.658%**
14	7	5	2	20	4.915%	**0.664%**
15	7	5	2	20	2.706%	**0.221%**

N/A: Not available since optimal solution cannot be obtained

Table 3.15: Comparison between optimality gap values for the solutions obtained by TSAS-CP-CG1 and TSAS-CP-CG2 (continued)

Instances	Lots	No of machines at Stage 1	No of machines at Stage 2	Maximum No of Sublots	Optimality Gap	
					TSAS-CP-CG1	TSAS-CP-CG2
16	8	6	3	10	N/A	N/A
17	8	6	3	10	1.058%	**0.036%**
18	8	6	3	10	N/A	N/A
19	8	6	3	10	5.323%	**1.705%**
20	8	6	3	10	N/A	N/A
21	8	6	3	15	5.420%	**0.130%**
22	8	6	3	15	N/A	N/A
23	8	6	3	15	N/A	N/A
24	8	6	3	15	N/A	N/A
25	8	6	3	15	5.408%	**0.045%**
26	8	6	3	20	3.575%	**0.406%**
27	8	6	3	20	1.240%	**0.398%**
28	8	6	3	20	N/A	N/A
29	8	6	3	20	N/A	N/A
30	8	6	3	20	N/A	N/A

N/A: Not available since optimal solution cannot be obtained

3.5 Conclusion

In this chapter, we have addressed a two-stage assembly system scheduling problem in the presence of lot streaming for the objective of minimizing the makespan and total cost. First, we formulated this problem as a mixed integer programming model, and enhanced its solvability by adding additional constraints. Due to the difficulty of solving this model directly for large instances, we developed two column generation-based methods. Our computational investigation has revealed that the proposed column generation methods generate solutions that are close to optimum. The optimality gap values are less than 8.3% (TSAS-MP-CG1) and 1.9% (TSAS-MP-CG2) for the makespan model, and are less than 6.3% (TSAS-CP-CG1) and 1.8% (TSAS-CP-CG2) for the cost-based model. Therefore, the TSAS-MP-CG2 and TSAS-CP-CG2 methods perform better for the makespan and cost-based objective functions, respectively. In addition, the column generation methods find solutions in a few seconds of CPU time while the direct solution of a mixed integer programming model of the problem by CPLEX requires a much larger CPU time.

Chapter 4

Two-stage, Single-lot, Lot Streaming Problem for a $1 + 2$ Hybrid Flow Shop

In this chapter, we address a single-lot lot streaming problem for a two-stage hybrid flow shop, which consists of one machine at Stage 1 and two parallel (identical) machines at Stage 2. The objective is to minimize the makespan. A removal time is associated with each sublot at Stage 1. We present a mixed integer programming model for this problem. First, we consider the case of a given number of sublots for which we develop closed-form expressions to obtain optimal, continuous sublot sizes. Then, we consider determination of optimal number of sublots in addition to their sizes. We develop an upper bound on the number of sublots, and use a simple search procedure in conjunction with the closed-form expressions for sublot sizes to obtain an optimal solution. We also consider the problem of determining integer sublot sizes, and we propose a heuristic method that directly solves the mixed integer programming model after having fixed values of appropriate variables. The results of our computational investigation reveal the efficacy of the proposed method to obtain optimal, continuous sublot sizes, and also, that of the proposed heuristic method to obtain integer sublot sizes, which are within 0.2% of optimal solutions for the testbed of data used, each obtained within a few seconds of CPU time.

4.1 Introduction

We consider a *Hybrid Flow Shop*, which consists of one machine at Stage 1 and multiple parallel (identical) machines at Stage 2. A lot consisting of a number of identical items (products) is available for processing in this hybrid flow shop. Each item requires for processing only one of the parallel machines at Stage 2 after its processing at Stage 1.Such a machine configuration is often encountered in the electronics manufacturing environment for the assembly of printed circuit boards (PCB) [103]. At Stage 1, a magazine is configured with a requisite number of PCBs. Components are then inserted on a PCB by a robot at Stage 2. Several robotic stations are available at Stage 2 where PCBs can be processed simultaneously. Another example of a hybrid flow shop arises while executing a program on a parallel-task computer [9]. Running each program consists of two consecutive steps: loading the data from an external memory, and then, executing the program. Generally, while the loading of data from the external memory can only be done sequentially, the CPU executes the program concurrently on the available parallel processors. A hybrid flow shop is also encountered in a car painting factory, where the cars are degreased at Stage 1, and then, painted at Stage 2. Several identical paint booths are available at Stage 2 for painting a car. We designate this machine configuration as a $1 + m$ two-stage hybrid flow shop. Its structure is depicted in Figure 4.1. We apply the concept of lot streaming in which a sublot of items is transferred for processing at Stage 2 while the remainder of the items in the lot are still being processed at Stage 1. Thus, our aim is to determine both the number and sizes of sublots, and minimize the makespan, i.e., the maximum of the completion time of the items at the second stage.

Linn and Zhang (1999) [60] present an overview of hybrid flow shop scheduling problems. They classify the current research into three categories: (i) two-stage hybrid flow shops; (ii) three-stage hybrid flow shops; and (iii) k-stage ($k > 3$) hybrid flow shops. The two-stage hybrid flow shop has been addressed by Gupta (1988) [36], Kusiak (1989) [57], Sriskandarajah and Sethi (1989) [83], Gupta and Tunc (1991, 1994) [37, 38], Hoogeveen et al. (1996) [40],

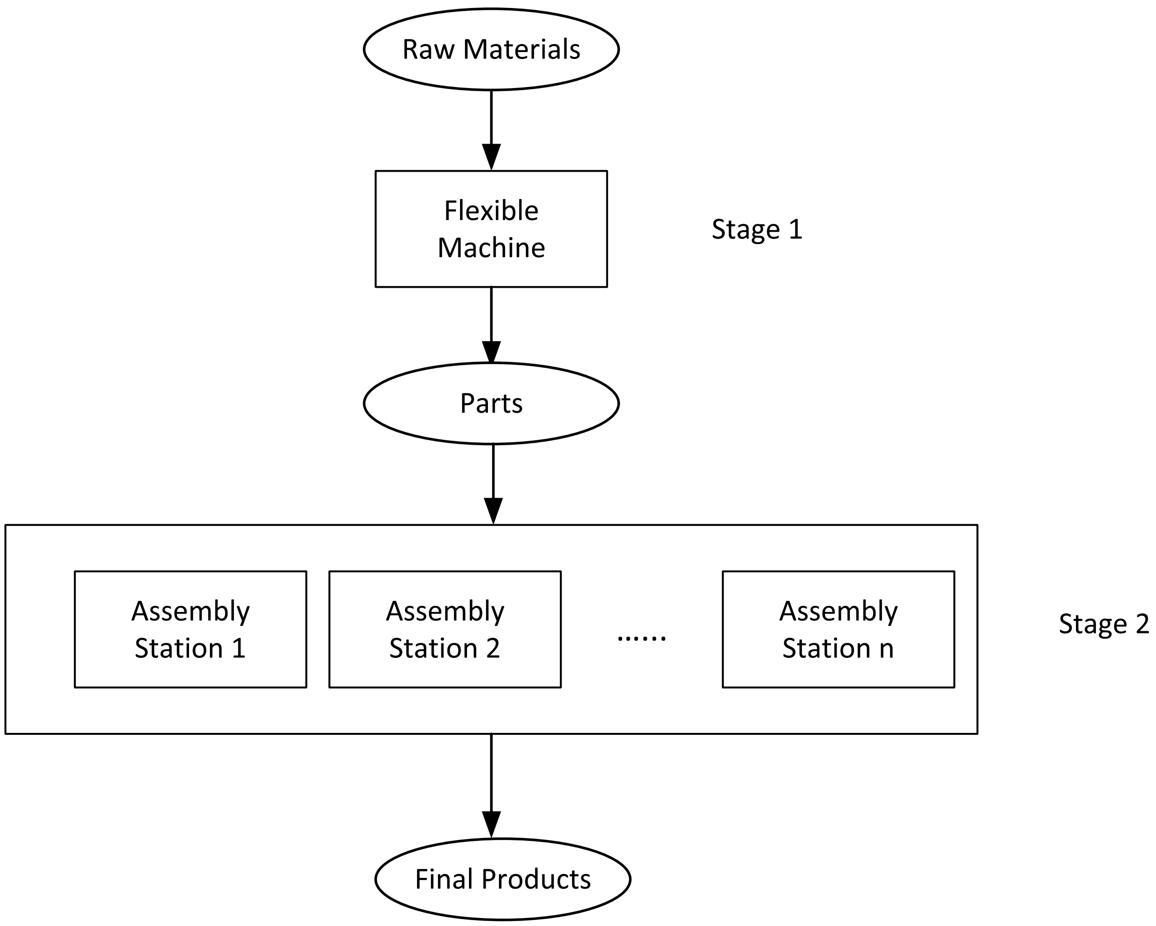

Figure 4.1: Structure of a 1+m hybrid flow shop

and Carpov et al. (2012) [9]. Gupta (1988) [36] has shown the two-stage hybrid flow shop problem for the case of only one machine at Stage 2 to be NP-complete, and has presented a heuristic method to minimize the makespan. Kusiak (1989) [57] has addressed a two-stage hybrid flow shop in the flexible manufacturing environment for the objective of minimizing the makespan. Sriskandarajah and Sethi (1989) [83] address the $1 + m$ hybrid flow shop problem and present the worst-case and average-case performance bounds of the proposed heuristic method. Gupta and Tunc (1991) [37] have developed two polynomial-bounded heuristic methods, and have shown their effectiveness in improving the performance of a branch-and-bound-based algorithm for this problem. Furthermore, they have considered separable setup and removal times, and for the case where the number of machines at the second stage is less than the number of jobs, they propose four different heuristic methods. Hoogeveen et al. (1996) [40] have shown the $1 + m$ problem for $m \geq 2$ to be NP-hard in the strong sense for the objective of minimizing the makespan. Carpov et al. (2012) [9] have studied this problem in the presence of precedence constraints for which they proposed two lower bounds. They also developed heuristic methods for its solution. Linn and Zhang (1999) [60] and Ruiz et al. (2010) [73] provide a review of two-stage problems with multiple parallel machines at both the stages, and for multiple-stage (three or more-stage) hybrid flow shops.

Our work is different from that presented above because of the consideration of lot streaming. The use of lot streaming in a two-stage hybrid flow shop system has been studied by Tsubone et al. (1996) [98], Zhang et al. (2003, 2005) [109, 110], and Liu (2008) [61]. Tsubone et al. (1996) [98] have considered the $1 + m$ problem and have used a simulation model to study the impact of lot sizing from the viewpoint of optimizing total flow time, makespan, the capacity utilization, and the maximum work-in-process level. Zhang et al. (2003, 2005) [109, 110] and Liu (2008) [61] have considered the $m + 1$ problem. Zhang et al. (2003) [109] assume one of the stages to be a bottleneck and sublot sizes to be integers. The problem is, then, formulated as a mixed integer linear programming model, and two heuristic methods are proposed that allocate the sublots as evenly as possible to the machines at Stage 1, and they are shown to produce near-optimal solutions. Zhang et al. (2005) [110] assume equal sublot sizes and the objective of minimizing mean completion time. A good lower bound and two heuristic methods are presented. Liu (2008) [61] assumes a given number of sublots and the objective of minimizing the makespan. They prove a property for this case by which an optimal solution can be obtained by allocating the sublots in "rotation" over the machines at Stage 1. The optimal, continuous sublot sizes are obtained using a linear program. They also present a heuristic method to determine number of sublots when all the sublots are of the same size. In this chapter, we present a closed-form expression to obtain optimal sublot sizes for a given number of sublots when $m = 2$. We also determine optimal number of sublots when the number of sublots is not known apriori. Moreover, we address the problem of determining integer sublot sizes. Figure 4.2 depicts a numerical example for $1 + 2$ hybrid flow shop problem with and without lot streaming. For three sublots, a lot consisting of 100 items is split into three sublots of sizes 30, 30, and 40 items. The makespan value of 140 under lot streaming is smaller than 200 obtained without lot streaming. Note that the makespan value can further be reduced by determining optimal sublot sizes and number of sublots as well.

The rest of this chapter is organized as follows. In Section 4.2, we formally define our problem, introduce the notation, present some preliminary results and a mixed-integer programming model for the problem. In Section 4.3, we develop closed-form expressions to obtain optimal, continuous sublot sizes for a given number of sublots. We address the problem of determining both optimal number of sublots and sublot sizes in Section 4.4. The problem of determining integer sublot sizes is addressed in Section 4.5. We present the results of our computational investigation in Section 4.6, and finally, make concluding remarks in Section 4.7.

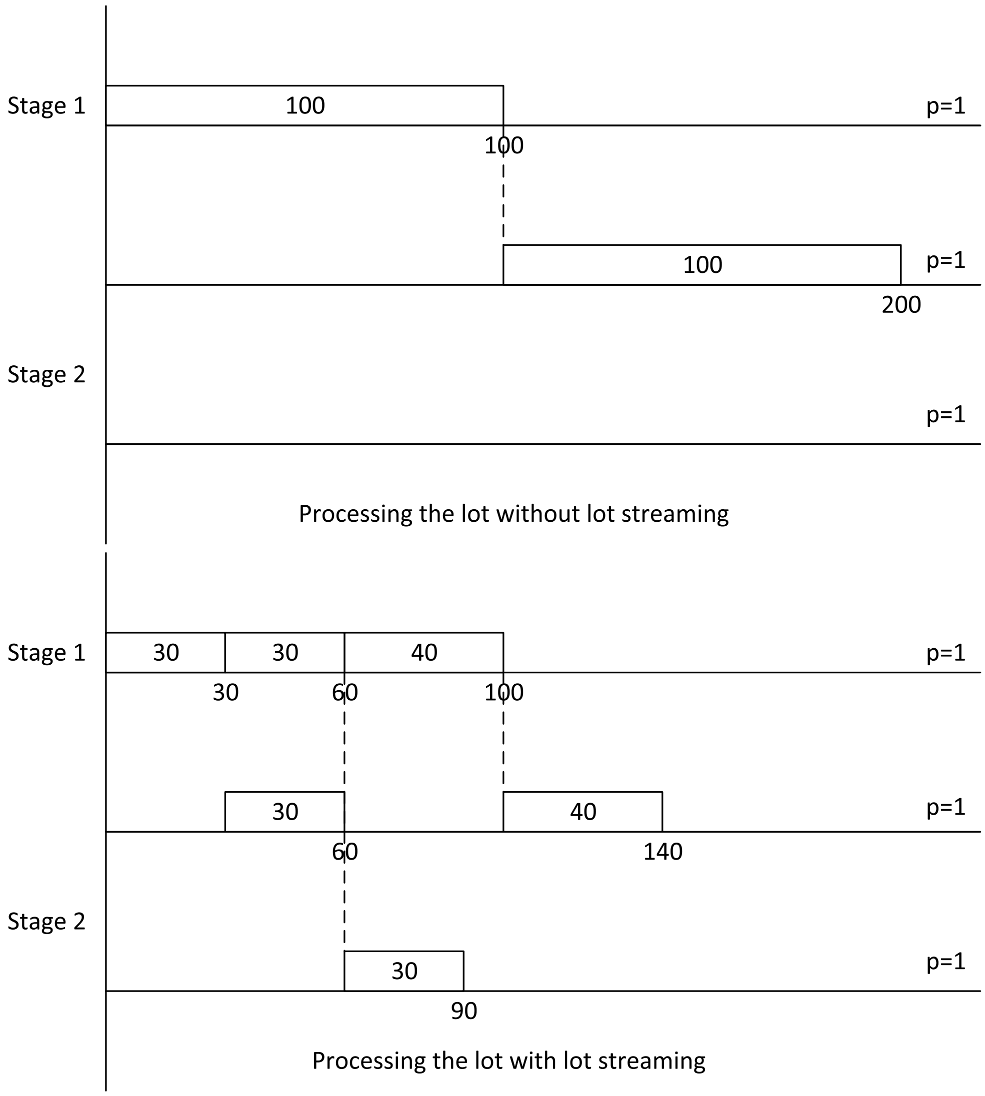

Figure 4.2: Example of processing a lot with and without lot streaming method

4.2 Problem Statement, Notation, Some Preliminaries, and Model Formulation

In this section, we present the problem statement, notation, assumptions, and some preliminary results, followed by a mixed integer programming model for the problem.

4.2.1 Problem Statement

The problem that we address can be concisely stated as follows: *Given a two-stage hybrid flow shop consisting of one machine at Stage 1 and two parallel machines at Stage 2, a lot consisting of U items (products) for processing on single machine at Stage 1 and one of the two parallel (identical) machines at Stage 2, a maximum number of sublots, and sublot-attached removal times required for sublots on the machines at Stage 1, determine optimal: (i) number of sublots, (ii) sublot sizes, and (iii) allocation of sublots to the parallel machines at Stage 2, in order to minimize the makespan, which is the completion time of the last sublot at Stage 2.* We designate this problem as a two-stage hybrid flow shop, lot streaming problem with flexible number of sublots (TSHFLSP-FlexN). We also consider the case when the number of sublots is pre-specified, and designate it as TSHFLSP-FixN. We consider the cases of both continuous and integer sublot sizes.

4.2.2 Notation

Consider the following notation:

Parameters:

- U : Number of items (products) in the lot;
- $N_{\max}$: Maximum number of sublots allowed;;

- n : Number of sublots used;
- m : Number of parallel machines at Stage 2 (even though the solution method has been developed for $m = 2$, we present some results and a model formulation for general m);
- f : Processing time per item at Stage 1;
- p_k : Processing time per item on machine k at Stage 2, $\forall k = 1, \ldots, m$; (the model formulation is presented for general p_k; however, for the development in the sequel, we assume $p_1 = p_2 = p$.)
- t : Removal time incurred per sublot on the machine at Stage 1;
- M: A large positive number;

Variables:

- s_{ig} : Size of sublot i used at Stage 1, $\forall i = 1, \ldots, n$;
- s_{ik} : Size of sublot i on machine k at Stage 2, $\forall i = 1, \ldots, n$, $\forall k = 1, \ldots, m$;
- R_i : Completion time of sublot i at Stage 1, $\forall i = 1, \ldots, n$;
- C_{ik} : Completion time of sublot i on machine k at Stage 2, $\forall i = 1, \ldots, n, \forall k = 1, \ldots, m$;
- $C_{\max}$: Makespan value;
- $X_{ik} = \begin{cases} 1, & \text{if sublot } i \text{ is assigned to parallel machine } k \text{ at Stage 2,} \\ 0, & \text{otherwise, } \forall i = 1, \ldots, n, k = 1, \ldots, m. \end{cases}$
- $Y_i = \begin{cases} 1, & \text{if size of sublot } i \text{ is greater than 0,} \\ 0, & \text{otherwise, } \forall i = 1, \ldots, n. \end{cases}$

4.2.3 Assumptions

We make the following assumptions:

- The items in the lot are identical.
- Machine processing times are deterministic.
- A removal time is incurred for every sublot at Stage 1, and it is independent of the sublot size.
- All the machines are available at time zero and remain continuously available.
- One machine can process only one sublot at any time.
- The processing of a sublot i on a machine at Stage 2 can begin only after its completion at Stage 1.

4.2.4 Some Preliminaries

Proposition 4.1. *There exists an optimal solution in which the sublot sizes used for processing the lot at Stage 1 and Stage 2 are consistent.*

Proof: The result holds by the fact that if a sublot size at Stage 2 is smaller than that at Stage 1, then it can be started earlier at Stage 2 upon the completion of requisite number of items at Stage 1. This would result in a better or identical makespan value. On the other hand, if the sublot size at Stage 2 is larger than that at Stage 1, then the sublot size at Stage 1 can be increased without impacting the makespan. Such a move may eventually lead to a reduction in makespan because of the elimination of a removal time. Thus, the sublots at Stage 2 can be made equal to those at Stage 1 without worsening the makespan. □

As a result of Proposition 4.1, we can drop the subscripts "g" and "k" from s_{ig} and s_{ik}, respectively, and use s_i to represent the size of sublot i on all the machines.

Equivalence of $1+m$ and $m+1$ Problems

The $m+1$ problem with sublot-attached setup time at Stage 1 (see Figure 4.3(b)) is the reverse of the $1+m$ problem with sublot-attached removal time (See Figure 4.3(a)), as their critical paths are identical. Furthermore, the critical path remains the same if we have removal time instead of sublot-attached setup time for the $m+1$ problem (See Figure 4.3(c)). Therefore, these problems are equivalent.

4.2.5 Model Formulation

Next, we present a mixed-integer programming model for the TSHFLSP-FlexN.

Model TSHFLSP-FlexN:

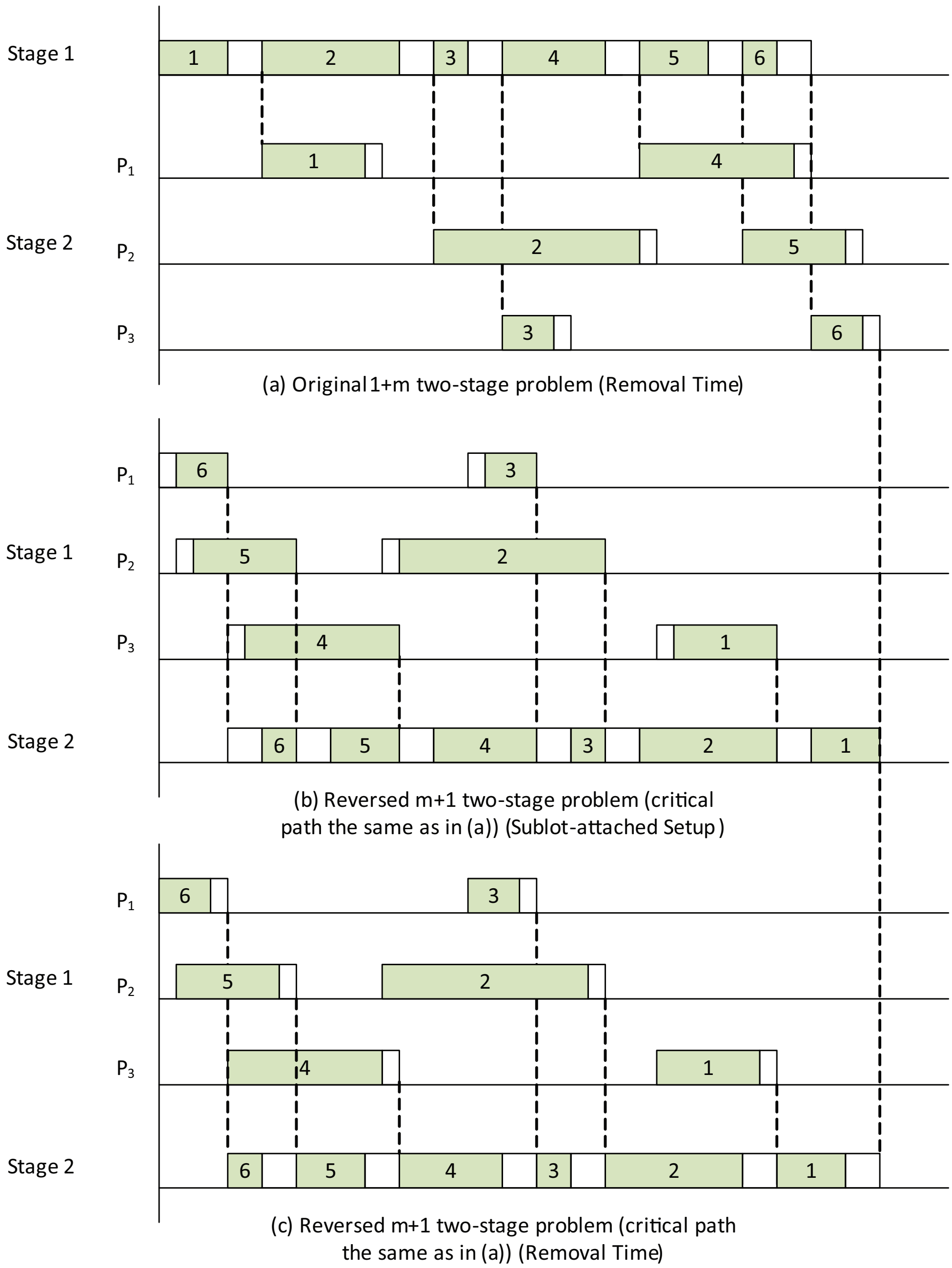

Figure 4.3: Original and reverse problems

Minimize

$$C_{\max} \tag{4.1}$$

Subject to

$$C_{\max} \geq C_{ik} - M(1 - Y_i), \qquad \forall i = 1, \ldots, N_{\max}, \forall k = 1, \ldots, m, \tag{4.2}$$

$$R_1 \geq t + fs_1, \tag{4.3}$$

$$R_i \geq t + R_{i-1} + fs_i, \qquad \forall i = 2, \ldots, N_{\max}, \tag{4.4}$$

$$C_{ik} \geq R_i + p_k s_i + (X_{ik} - 1)\, M, \qquad \forall i = 1, \ldots, N_{\max}, \forall k = 1, \ldots, m, \tag{4.5}$$

$$C_{ik} \geq C_{jk} + p_k s_i + (X_{ik} - 1)\, M, \qquad \forall i = 1, \ldots, N_{\max}, \forall j = 1, \ldots, i-1, \forall k = 1, \ldots, m, \tag{4.6}$$

$$\sum_{k=1}^{m} X_{ik} = 1, \qquad \forall i = 1, \ldots, N_{\max}, \tag{4.7}$$

$$s_i \leq Y_i U \qquad \forall i = 1, \ldots, N_{\max}, \tag{4.8}$$

$$\sum_{i=1}^{N_{max}} s_i = U, \tag{4.9}$$

$$C_{\max} \geq 0,$$

$$s_i \geq 0, \text{and integer}, \qquad \forall i = 1, \ldots, N_{\max},$$

$$R_i \geq 0, Y_i \in \{0, 1\}, \qquad \forall i = 1, \ldots, N_{\max},$$

$$C_{ik} \geq 0, X_{ik} \in \{0, 1\}. \qquad \forall i = 1, \ldots, N_{\max}, \forall k = 1, \ldots, m.$$

Constraints (4.2) define the makespan, which will only be affected by the sublots whose sizes are greater than zero. Constraints (4.3) and (4.4) capture relationships among the completion times of the sublots on the machine at Stage 1, while Constraints (4.5) and (4.6) do the same for the completion times of each sublot at Stage 1 and Stage 2 and between the completion times of the sublots on each of the machines at Stage 2, respectively. Constraints (4.7) assert that each sublot is assigned to only one machine at the second stage, while

Constraints (4.8) guarantee that if a sublot is not used, then the size of that sublot is zero. Constraint (4.9) ensures the summation of the sizes of all the sublots to be the size of the lot.

For the sake of convenience and without loss of generality, we scale the time unit by a factor of f. Consequently, the unit processing time at Stage 1 will be 1, the removal time will be t/f, and the unit processing time on a machine at Stage 2 will be p/f. We continue to use the notation t and p, which now represent removal time and unit processing on a machine at Stage 2 in the transformed time units. In view of this transformed scale, a reasonable value of M for use in Constraints (4.2), (4.5) and (4.6) is as follows:

$$M = n \cdot t + U + \max_{k=1,\dots,m} p_k \cdot U, \tag{4.10}$$

which is the total removal and processing times at Stage 1 plus the total processing time on the slowest machine at Stage 2.

4.3 Determination of Optimal Sublot Sizes for a Given Number of Sublots (n)

First, we consider the case where the number of sublots is given. For this case, the problem is to determine sublot sizes and allocate sublots for processing on the machines at Stage 2 so as to minimize makespan.

4.3.1 Solution of TSHFLSP-FixN

In the presence of parallel machines at Stage 2, it clearly follows that both of these machines will process sublots in order to minimize the makespan. For the $m+1$ problem, Zhang et al (2005) [110] and Liu (2008) [61] have shown that, for a given number of sublots and continuous sublot sizes, the "alternative" assignment method, in which the sublots are processed in an alternate fashion on the machines at Stage 1, results in an optimal solution. Since the $1+m$ and $m+1$ problems are equivalent, the same holds true for the $1+m$ problem as well. Zhang et al (2005) [110] and Liu (2008) [61] have used a linear program to determine optimal sublot sizes. However, we develop closed-form expressions for their determination, which are more efficient to implement than the use of a linear program.

Consider the sizes of the sublots in the reverse order, i.e., let $s_1^{'} = s_n$, $s_2^{'} = s_{n-1}, \ldots, s_n^{'} = s_1$. We address TSHFLSP-FixN without removal times and with removal times in Sections 4.3.2 and 4.3.3, respectively.

4.3.2 TSHFLSP-FixN without Removal Times

Proposition 4.2. *There exist optimal sublot sizes that can be obtained by solving the following expressions:*

$$s_2^{'} = s_1^{'}\left(1 + \frac{1}{p}\right), \tag{4.11}$$

$$s_i^{'} = \frac{s_{i-1}^{'} + s_{i-2}^{'}}{p}, \quad i = 3, \ldots, n, \tag{4.12}$$

$$\sum_{i=1}^{n} s_i^{'} = U. \tag{4.13}$$

Proof. We prove the result by induction.

Clearly, for $n = 1$, $s_1^{'*} = U$. When $n = 2$, the solution obtained using the above equations is: $s_1^{'*} = \frac{p}{2p+1}U$, $s_2^{'*} = \frac{p+1}{2p+1}U$ with makespan,

$$C_{\max} = \max\left\{\frac{p+1}{2p+1}U + \frac{p+1}{2p+1}U \cdot p,\ U + \frac{p}{2p+1}U \cdot p\right\} = \frac{(p+1)^2}{2p+1}U.$$

Suppose the solution is not optimal. Let the optimal solution be obtained for $\hat{s}_1' = \frac{p}{2p+1}U - x$ and $\hat{s}_2' = \frac{p+1}{2p+1}U + x,\ x \neq 0$. We have the makespan,

$$\begin{aligned} C_{\max}' &= \max\left\{\frac{p+1}{2p+1}U + x + \left(\frac{p+1}{2p+1}U + x\right) \cdot p,\ U + \left(\frac{p}{2p+1}U - x\right) \cdot p\right\} \\ &= \max\left\{\frac{(p+1)^2}{2p+1}U + (p+1)\,x,\ \frac{(p+1)^2}{2p+1}U - px\right\}. \end{aligned}$$

If $x > 0$, $C_{\max}' = \frac{(p+1)^2}{2p+1}U + (p+1)\,x > C_{\max}$.

If $x < 0$, $C_{\max}' = \frac{(p+1)^2}{2p+1}U - px > C_{\max}$.

Therefore, $s_1^{'*} = \frac{p}{2p+1}U$, $s_2^{'*} = \frac{p+1}{2p+1}U$ is the optimal solution when $n = 2$.

Let $s_l' = \frac{s_{l-1}' + s_{l-2}'}{p}$ be optimal for $\forall l = 2, \ldots, k$. We also use contradiction to show that it holds for $l = k + 1$ as well.

If $s_{k+1}' = \frac{s_k' + s_{k-1}'}{p}$ is not optimal, let $\hat{s}_{k+1}' = s_{k+1}' + x$ $(x \neq 0)$ be optimal. Note that the makespan is only affected by the idle time and processing times on the machines at Stage 2. Since the total processing time is fixed, consider the idle time. We have the following two situations.

1. $x < 0$. In this case, the idle time on the Stage 2 machines with sublots of sizes

$s_i^{'}, i = 1, \ldots, n$, minus that with sublot sizes $\hat{s}_i^{'}, i = 1, \ldots, n$, is given by

$$
\begin{aligned}
& -2x + px + \frac{s_k^{'}}{\sum_{i=1}^{k} s_i^{'}} x + \frac{s_k^{'} + s_{k-1}^{'}}{\sum_{i=1}^{k} s_i^{'}} x \\
= & \left(-2 + p + 1 - \frac{s_1^{'} + s_2^{'} + \cdots + s_{k-1}^{'}}{\sum_{i=1}^{k} s_i^{'}} + 1 - \frac{s_1^{'} + s_2^{'} + \cdots + s_{k-2}^{'}}{\sum_{i=1}^{k} s_i^{'}} \right) x \\
= & \left(p - \frac{s_{k-2}^{'} + s_{k-1}^{'} + s_{k-2}^{'} + s_{k-3}^{'} + \cdots + s_2^{'} + s_1^{'} + s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} \right) x \\
= & \left(p - \frac{p \cdot \left(s_k^{'} + s_{k-1}^{'} + \cdots + s_3^{'} \right) + s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} \right) x \qquad \text{(by (4.12))} \\
= & p \cdot x \cdot \frac{s_2^{'} + s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} - \frac{s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} x \\
= & p \cdot x \cdot \frac{\left(1 + \frac{1}{p}\right) s_1^{'} + s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} - \frac{s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} x \qquad \text{(by (4.11))} \\
= & \frac{2p \cdot s_1^{'}}{\sum_{i=1}^{k} s_i^{'}} x < 0.
\end{aligned}
$$

Therefore, the makespan will not improve if $x < 0$.

2. $x > 0$. In this case, the idle time on the Stage 2 machines with sublots of sizes $s_i^{'}, i = 1, \ldots, n$, minus that with sublot sizes $\hat{s}_i^{'}, i = 1, \ldots, n$, is given by

$$
-x + \frac{s_k^{'}}{\sum_{i=1}^{k} s_i^{'}} x < 0.
$$

Therefore, the makespan will not improve if $x > 0$. □

Closed-form expression for sublot-sizes when $p = 1$

When $p = 1$, by Proposition 4.2, we have $s_2^{'} = 2s_1^{'}$, $s_i^{'} = s_{i-1}^{'} + s_{i-2}^{'}$ and $\sum_{i=1}^{n} s_i^{'} = U$. Obviously, the size of the sublots follow the revised Fibonacci sequence since $s_i^{'} = s_{i-1}^{'} + s_{i-2}^{'}$. Next, we present a closed-form expression to calculate optimal sublot sizes.

Proposition 4.3. *If $p = 1$, then the optimal sublot sizes are given by the following expres-*

sions.

$$s_i^{'} = \frac{\frac{1}{\sqrt{5}}\left(a^{i+1} - b^{i+1}\right)}{\frac{1}{\sqrt{5}}\left(a^{n+3} - b^{n+3}\right) - 2}U, \quad \forall i = 1, \ldots, n, \tag{4.14}$$

where

$$a = \frac{1+\sqrt{5}}{2}, \text{ and}$$
$$b = \frac{1-\sqrt{5}}{2}.$$

Proof. We prove the result by showing that these values of $s_i^{'}, i = 1, \ldots, n$, satisfy the conditions on $s_i^{'}, i = 1, \ldots, n$ specified by Proposition 4.2. Clearly, by computing $s_i^{'}$ for $i = 1, 2$, we have $s_1^{'} = \frac{U}{a}$, and $s_2^{'} = \frac{2U}{a}$, and therefore $s_2^{'} = 2s_1^{'}$. Next, we show that $s_i^{'} = s_{i-1}^{'} + s_{i-2}^{'}, \forall i = 3, \ldots, n$, and $\sum_{i=1}^{n} s_i^{'} = U$. First, note that $1 + a = a^2$, and $1 + b = b^2$, $a + b = 1$, and $a - b = \sqrt{5}$. Let $Q = \frac{1}{\sqrt{5}}\left(a^{n+3} - b^{n+3}\right) - 2$.

We prove the result by induction. By computing $s_3^{'}$ from above, we have $s_3^{'} = \frac{3U}{Q}$. Therefore, $s_1^{'} + s_2^{'} = s_3^{'}$. Assume $s_l^{'} = s_{l-1}^{'} + s_{l-2}^{'}$ when $3 \leq l \leq k$, we want to show that $s_{k+1}^{'} = s_k^{'} + s_{k-1}^{'}$. We have

$$\begin{aligned}
s_k^{'} + s_{k-1}^{'} =& \frac{U}{Q}\left[\frac{1}{\sqrt{5}}\left(a^{k+1} - b^{k+1}\right) + \frac{1}{\sqrt{5}}\left(a^k - b^k\right)\right] \\
=& \frac{U}{Q}\left[\frac{1}{\sqrt{5}}a^k\left(1+a\right) - \frac{1}{\sqrt{5}}b^k\left(1+b\right)\right] \\
=& \frac{U}{Q}\left[\frac{1}{\sqrt{5}}a^k\left(a^2\right) - \frac{1}{\sqrt{5}}b^k\left(b^2\right)\right] \\
=& \frac{U}{Q}\left[\frac{1}{\sqrt{5}}\left(a^{k+2} - b^{k+2}\right)\right] \\
=& s_{k+1}^{'}.
\end{aligned}$$

To show $\sum_{i=1}^{n} s_i^{'} = U$, we proceed as follows.

Since $s_i^{'} = \frac{\frac{1}{\sqrt{5}}\left(a^{i+1} - b^{i+1}\right)}{\frac{1}{\sqrt{5}}\left(a^{n+3} - b^{n+3}\right) - 2}U$, it is enough to show that $\frac{1}{\sqrt{5}}\sum_{i=1}^{n}(a^{i+1} - b^{i+1}) = \frac{1}{\sqrt{5}}\left(a^{n+3} - b^{n+3}\right) -$

2. We have,

$$
\begin{aligned}
LHS =& \sum_{i=1}^{n} (a^{i+1} - b^{i+1}) \\
=& \frac{1}{\sqrt{5}} \left(\frac{a^2 - a^{n+2}}{1-a} - \frac{b^2 - b^{n+2}}{1-b} \right) \\
=& \frac{1}{\sqrt{5}} \left(\frac{a^{n+3} - a^3}{a^2 - a} - \frac{b^{n+3} - b^3}{b^2 - b} \right) \qquad \left(a^2 = a+1, \text{and } b^2 = b+1 \Rightarrow a^2 - a = 1, \text{and } b^2 - b = 1\right) \\
=& \frac{1}{\sqrt{5}} \left(a^{n-3} - b^{n-3}\right) - \frac{1}{\sqrt{5}} \left(a^3 - b^3\right) \\
=& \frac{1}{\sqrt{5}} \left(a^{n-3} - b^{n-3}\right) - 2 \qquad \left(a^3 - b^3 = \left(\frac{1+\sqrt{5}}{2}\right)^3 + \left(\frac{1-\sqrt{5}}{2}\right)^3 = \frac{16\sqrt{5}}{8} = 2\sqrt{5} \right). \qquad \square
\end{aligned}
$$

Corollary 4.1. *All the sublots are critical and the optimal makespan value,*

$$M_{\text{opt}} = U + ps_1^{'}. \tag{4.15}$$

By (4.11) and (4.12), and assignment of sublots to alternate machines of Stage 2, a machine is always available when a sublot completes operation at Stage 1, and also, there is no idle time between the processing of adjacent sublots on a machine of Stage 2. Therefore, all sublots are critical. Hence, the optimal makespan value is given by (4.15).

Example 4.1

Consider the following example: $U = 200$, and $n = 8$. By equation (4.14), we have $a = 1.61803$, and $b = -0.61803$. Therefore,

$$s_1^{'} = \frac{\frac{1}{\sqrt{5}}\left(a^2 - b^2\right)}{\frac{1}{\sqrt{5}}\left(a^{11} - b^{11}\right) - 2} \cdot 200 = 2.29885$$

$$s_2^{'} = \frac{\frac{1}{\sqrt{5}}\left(a^3 - b^3\right)}{\frac{1}{\sqrt{5}}\left(a^{11} - b^{11}\right) - 2} \cdot 200 = 4.5977$$

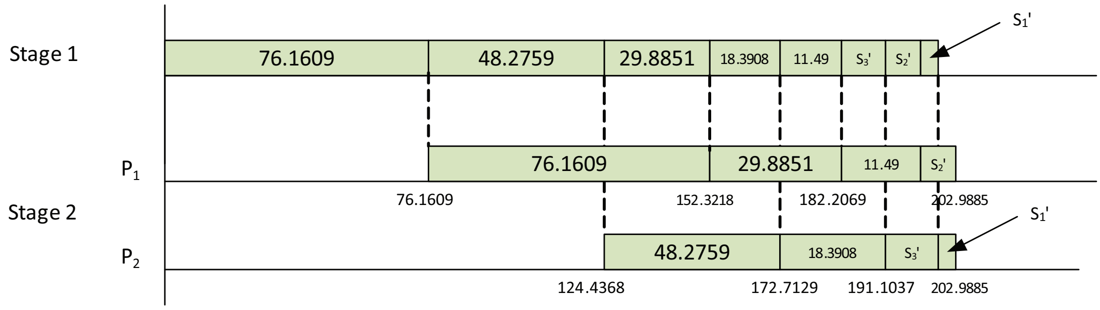

Figure 4.4: Optimal schedule for Example 4.1

Similarly, $s_3^{'} = 6.89655$, $s_4^{'} = 11.4943$, $s_5^{'} = 18.3908$, $s_6^{'} = 29.8851$, $s_7^{'} = 48.2759$, and $s_8^{'} = 76.1609$. Therefore, by equation (4.15), we have the optimal makespan value,

$$M^* = U + ps_1^{'} = 200 + 2.29885 = 202.29885.$$

A graphical depiction of the solution is given in Figure 4.4.

4.3.3 TSHFLSP-FixN with Removal Times

Next, we consider the case when a sublot-attached removal time is present after the processing of each sublot at Stage 1.

Proposition 4.4. *For a given number, n, of sublots and presence of sublot-attached removal times at Stage 1, the optimal sublot sizes can be obtained by the following expressions, if the sizes of all the sublots are greater than zero.*

$$s_2^{'} = s_1^{'}\left(1+\frac{1}{p}\right)+\frac{t}{p}, \tag{4.16}$$

$$s_i^{'} = \frac{s_{i-1}^{'}+s_{i-2}^{'}}{p}+\frac{2t}{p}, \quad i=3\ldots,n, \tag{4.17}$$

$$\sum_{i=1}^{n} s_i^{'} = U. \tag{4.18}$$

Proof. We prove the result by induction.

Clearly, for $n=1$, $s_1^{'*}=U$. When $n=2$, the solution obtained by using the above equations is: $s_1^{'*}=\frac{pU-t}{2p+1}$, $s_2^{'*}=\frac{(p+1)U+t}{2p+1}$, with makespan,

$$C_{\max}^{*} = \max\left\{t+\frac{(p+1)U+t}{2p+1}+\frac{(p+1)U+t}{2p+1}\cdot p,\ U+2t+\frac{pU-t}{2p+1}\cdot p\right\} = \frac{(p+1)^2U+3pt+2t}{2p+1}$$

Suppose the solution is not optimal. Let the optimal solution be obtained for $\hat{s}_1^{'}=\frac{pU-t}{2p+1}-x$ and $\hat{s}_2^{'}=\frac{pU-t}{2p+1}U+x,\ x\neq 0$. We have makespan,

$$\begin{aligned} C_{\max}^{'} &= \max\left\{\frac{(p+1)U+t}{2p+1}+x+\left(\frac{(p+1)U+t}{2p+1}+x\right)\cdot p,\ U+2t+\left(\frac{pU-t}{2p+1}-x\right)\cdot p\right\} \\ &= \max\left\{\frac{(p+1)^2U+3pt+2t}{2p+1}+(p+1)x,\ \frac{(p+1)^2U+3pt+2t}{2p+1}-px\right\} \end{aligned}$$

If $x>0$, $C_{\max}^{'}=\frac{(p+1)^2U+3pt+2t}{2p+1}+(p+1)x>C_{\max}^{*}$.

If $x<0$, $C_{\max}^{'}=\frac{(p+1)^2U+3pt+2t}{2p+1}-px>C_{\max}^{*}$.

Therefore, $s_1^{'*}=\frac{pU-t}{2p+1}$, $s_2^{'*}=\frac{(p+1)U+t}{2p+1}$ is the optimal solution when $n=2$.

Let $s_l^{'}=\frac{s_{l-1}^{'}+s_{l-2}^{'}}{p}+\frac{2t}{p}$ be optimal, for $\forall l=2,\ldots,k$. We want to show that it holds for $l=k+1$ as well. We also prove this by contradiction.

If $s'_{k+1} = \frac{s'_k + s'_{k-1}}{p} + \frac{2t}{p}$ is not optimal, let $\hat{s}'_{k+1} = s'_{k+1} + x$, $(x \neq 0)$ be optimal. Note that the makespan is only affected by the idle time and processing times on the machines at Stage 2. Since the total processing time is fixed, consider the idle time. We have the following two situations.

1. $x < 0$. In this case, the idle time on Stage 2 machines with sublots of sizes $s'_i, i = 1, \ldots, n$, minus that with sublot sizes $\hat{s}'_i, i = 1, \ldots, n$, is given by

$$
\begin{aligned}
& -2x + px + \frac{s'_k + t}{\sum_{i=1}^k s'_i + kt} x + \frac{s'_k + s'_{k-1} + 2t}{\sum_{i=1}^k s'_i + kt} x \\
=& \left(-2 + p + 1 - \frac{s'_1 + s'_2 + \cdots + s'_{k-1} + (k-1)t}{\sum_{i=1}^k s'_i + kt} + 1 - \frac{s'_1 + s'_2 + \cdots + s'_{k-2} + (k-2)t}{\sum_{i=1}^k s'_i + kt} \right) x \\
=& \left(p - \frac{s'_{k-2} + s'_{k-1} + s'_{k-2} + s'_{k-3} + \cdots + s'_2 + s'_1 + s'_1 + (2k-3)t}{\sum_{i=1}^k s'_i + kt} \right) x \\
=& \left(p - \frac{p \cdot \left(s'_k + s'_{k-1} + \cdots + s'_3\right) - (2k-4)t + s'_1 + (2k-3)t}{\sum_{i=1}^k s'_i + kt} \right) x \qquad \text{(by (4.17))} \\
=& p \cdot x \cdot \frac{s'_2 + s'_1}{\sum_{i=1}^k s'_i + kt} - \frac{s'_1 + t}{\sum_{i=1}^k s'_i + kt} x \\
=& p \cdot x \cdot \frac{\left(1 + \frac{1}{p}\right) s'_1 + \frac{t}{p} + s'_1}{\sum_{i=1}^k s'_i + kt} - \frac{s'_1 + t}{\sum_{i=1}^k s'_i + kt} x \qquad \text{(by (4.16))} \\
=& \frac{2p \cdot s'_1}{\sum_{i=1}^k s'_i + kt} x < 0.
\end{aligned}
$$

   Therefore, the makespan will not improve if $x < 0$.

2. $x > 0$. In this case, the idle time on Stage 2 machines with sublots of sizes $s'_i, i = 1, \ldots, n$, minus that with sublot sizes $\hat{s}'_i, i = 1, \ldots, n$, is given by

$$
-x + \frac{s'_k + t}{\sum_{i=1}^k s'_i + kt} x < 0.
$$

   Therefore, the makespan will not improve if $x > 0$. $\square$

Remark 4.1. *In case the sublot sizes given by Expressions (4.16), (4.17), and (4.18) are not all positive, it implies that we cannot find a feasible solution in which all n sublots have positive values, i.e., at least one of the sublot sizes must be zero.*

There are the following properties of the solution given by Expressions (4.16), (4.17), and (4.18).

Corollary 4.2. *All the sublots are critical, and the optimal makespan value,*

$$M_{\text{opt}} = U + nt + ps_1'. \tag{4.19}$$

Proof. The argument follows along the lines for the proof of Corollary 4.1. Therefore, the optimal makespan value,

$$M_{\text{opt}} = U + nt + ps_1'. \qquad \square$$

Corollary 4.3. *The completion times of the last sublots on the machines at Stage 2 are the same if $n \geq 2$.*

Proof. We consider the following two cases.

1. Number of sublots is even.

The completion time of the last sublot on machine 1,

$$C_1 = s_n' + t + ps_n' + 2t + ps_{n-2}' + 2t + \cdots + ps_4' + 2t + ps_2' + t,$$

The completion time of the last sublot on machine 2,

$$C_2 = s_n' + t + s_{n-1}' + t + ps_{n-1}' + 2t + ps_{n-3}' + 2t + \cdots + ps_3' + 2t + ps_1' + t.$$

Therefore,

$$
\begin{aligned}
C_1 - C_2 =& (ps_n^{'} - s_{n-1}^{'}) - ps_{n-1}^{'} + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots + ps_2^{'} - ps_1^{'} - t \\
=& (s_{n-2}^{'} + 2t - ps_{n-1}^{'}) + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots + ps_2^{'} - ps_1^{'} - t \qquad \text{(by (4.17))} \\
=& - s_{n-3}^{'} + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots + ps_2^{'} - ps_1^{'} - t \qquad \text{(by (4.17))} \\
& \text{(which upon repeated application of (4.17) and simplification)} \\
=& - s_1^{'} + ps_2^{'} - ps_1^{'} - t \\
=& 0. \qquad \text{(by (4.16)).} \qquad \square
\end{aligned}
$$

2. Number of sublots is odd.

The completion time of the last sublot on machine 1,

$$C_1^{'} = s_n^{'} + t + ps_n^{'} + 2t + ps_{n-2}^{'} + 2t + \cdots + ps_3^{'} + 2t + ps_1^{'} + t,$$

The completion time of the last sublot on machine 2,

$$C_2^{'} = s_n^{'} + t + s_{n-1}^{'} + t + ps_{n-1}^{'} + 2t + ps_{n-3}^{'} + 2t + \cdots + ps_4^{'} + 2t + ps_2^{'} + t.$$

Therefore,

$$
\begin{aligned}
C_1^{'} - C_2^{'} =& (ps_n^{'} - s_{n-1}^{'}) - ps_{n-1}^{'} + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots - ps_2^{'} + ps_1^{'} + t \\
=& (s_{n-2}^{'} + 2t - ps_{n-1}^{'}) + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots - ps_2^{'} + ps_1^{'} + t \qquad \text{(by (4.17))} \\
=& - s_{n-3}^{'} + ps_{n-2}^{'} - ps_{n-3}^{'} + ps_{n-4}^{'} + \cdots - ps_2^{'} + ps_1^{'} + t \qquad \text{(by (4.17))} \\
& \text{(which upon repeated application of (4.17) and simplification)} \\
=& s_1^{'} - ps_2^{'} + ps_1^{'} + t \\
=& 0. \qquad \text{(by (4.16))}. \qquad \square
\end{aligned}
$$

Development of closed-form expressions for sublot sizes when $p \neq 2$

By (4.16) and (4.17), all $s_i^{'}$, $\forall i = 1, \ldots, n$, can be expressed as a function of $s_1^{'}$. By substituting for $s_i^{'}$, $\forall i = 2, \ldots, n$, in (4.18), we have

$$A \cdot s_1^{'} + B = U, \tag{4.20}$$

where A is the summation of all the coefficients of $s_1^{'}$ and B is the summation of all the constant terms of $s_i^{'}$. If $B < U$, $s_1^{'}$ will be positive and all $s_i^{'}$, $i = 2 \ldots, n$, will be positive as well since (4.16) and (4.17) only involve positive coefficients and constants. Otherwise, when $B \geq U$, the Remark 4.1 applies here as well. Next, we determine both A and B by exploiting the underlying recurrence relations.

In order to calculate A, let y_i be the coefficient of $s_i^{'}$, i.e., $y_1 = 1$, $y_2 = 1 + \frac{1}{p}$, and $y_i = \frac{y_{i-1}+y_{i-2}}{p}$, $\forall i = 3, \ldots, n$. Let $\sigma_i = \sum_{k=1}^{i} y_k$, i.e., $\sigma_1 = y_1 = 1$, $\sigma_2 = y_1 + y_2 = 1 + 1 + \frac{1}{p} = 2 + \frac{1}{p}$. We have,

$$
\begin{aligned}
\sigma_i - y_1 - y_2 =& \sum_{k=3}^{i} y_k \\
=& \frac{\sigma_{i-1} + \sigma_{i-2} - y_1}{p}, \text{(after substituting for } y_k,\ k = 3, \ldots i \text{ from above and simplifying)}
\end{aligned}
$$

$$\Rightarrow \sigma_i = \frac{\sigma_{i-1} + \sigma_{i-2}}{p} + y_1 + y_2 - \frac{y_1}{p},$$
$$\Rightarrow \sigma_i = \frac{\sigma_{i-1} + \sigma_{i-2}}{p} + 2.$$

Therefore, σ_i follows a recurrence relation, but with a constant 2. In order to eliminate the constant, we define a new variable,

$$\beta_i = \sigma_i + a, \ \forall i = 1, \ldots, n.$$

In order to solve for a, we have,

$$\beta_i - a = \frac{\beta_{i-1} + \beta_{i-2} - 2a}{p} + 2,$$

$$\Rightarrow pa - 2a + 2p = 0,$$
$$\Rightarrow a = \frac{2p}{2-p}.$$

Substituting a into the expression for β_i, we have $\beta_1 = \sigma_1 + a = 1 + \frac{2p}{2-p}$, $\beta_2 = \sigma_2 + a = 2 + \frac{1}{p} + \frac{2p}{2-p}$, and $\beta_i = \frac{\beta_{i-1}+\beta_{i-2}}{p}$, $\forall i = 3, \ldots, n$. We can use the following recurrence relation to solve for β_i and A (see Chiang (1984) [17]):

$$\beta_i = C\lambda_1^i + D\lambda_2^i, \tag{4.21}$$

where λ_1 and λ_2 are the roots of

$$r^2 - \frac{1}{p}r - \frac{1}{p} = 0. \tag{4.22}$$

Therefore,

$$\lambda_1 = \frac{1-\sqrt{1+4p}}{2p}, \tag{4.23}$$

$$\lambda_2 = \frac{1+\sqrt{1+4p}}{2p}. \tag{4.24}$$

The values of C and D can be obtained by using the values of β_1 and β_2. Thus,

$$\begin{cases} C \cdot \lambda_1 + D \cdot \lambda_2 = \beta_1 & = 1 + \frac{2p}{2-p} \\ C \cdot \lambda_1^2 + D \cdot \lambda_2^2 = \beta_2 & = 2 + \frac{1}{p} + \frac{2p}{2-p} \end{cases}$$

$$\Longrightarrow \begin{cases} C = & -\frac{16p^2+8p^3-8p^4-16p^2\sqrt{1+4p}+8p^3\sqrt{1+4p}}{32p\sqrt{1+4p}-32p^2\sqrt{1+4p}+8p^3\sqrt{1+4p}} \\ D = & -\frac{p+p^2+p\sqrt{1+4p}}{(-2+p)\sqrt{1+4p}} \end{cases}$$

Substituting for λ_1, λ_2, C and D into (4.21), we have

$$\begin{aligned} \beta_i \;=\; & \frac{1}{(2-p)\sqrt{1+4p}} \left(-2^{-i} p \left(-(\frac{1-\sqrt{1+4p}}{p})^i + \sqrt{1+4p}(\frac{1-\sqrt{1+4p}}{p})^i + (\frac{1+\sqrt{1+4p}}{p})^i \right.\right. \\ & \left.\left. + \sqrt{1+4p}(\frac{1+\sqrt{1+4p}}{p})^i + p \left(-(\frac{1-\sqrt{1+4p}}{p})^i + (\frac{1+\sqrt{1+4p}}{p})^i \right) \right) \right), i = 3, \ldots, n. \end{aligned}$$

Therefore,

$$
\begin{aligned}
A &= \sigma_n \\
&= \beta_n - \frac{2p}{2-p} \\
&= \frac{2^n p}{(-2+p)\sqrt{1+4p}} \left(2^{1+n}\sqrt{1+4p} + (\frac{1-\sqrt{1+4p}}{p})^n - \sqrt{1+4p}(\frac{1-\sqrt{1+4p}}{p})^n \right. \\
&\quad -(\frac{1+\sqrt{1+4p}}{p})^n - \sqrt{1+4p}(\frac{1+\sqrt{1+4p}}{p})^n \\
&\quad \left. +p\left(-(\frac{1-\sqrt{1+4p}}{p})^n + (\frac{1+\sqrt{1+4p}}{p})^n\right)\right).
\end{aligned}
$$

In order to determine B, let q_i be the constant term of $s_i^{'}$, i.e., $q_1 = 0$, $q_2 = \frac{t}{p}$, and $q_i = \frac{q_{i-1}+q_{i-2}}{p} + \frac{2t}{p}, \forall i = 3, \ldots, n$. To eliminate the constant term $\frac{2t}{p}$, we define a variable, $T_i = q_i + b$,

$$
T_i - b = \frac{T_{i-1} + T_{i-2} - 2b}{p} + \frac{2t}{p}
$$

$$
\begin{aligned}
&\Rightarrow pb - 2b + 2t = 0, \\
&\Rightarrow b = \frac{2t}{2-p}.
\end{aligned}
$$

Therefore, we have $T_1 = q_1 + b = \frac{2t}{2-p}$, $T_2 = q_2 + b = \frac{t}{p} + \frac{2t}{2-p}$, and $T_i = \frac{T_{i-1}+T_{i-2}}{p}, \forall i = 3, \ldots, n$.

Let $\pi_i = \sum_{k=1}^{i} T_k$, i.e., $\pi_1 = T_1 = \frac{2t}{2-p}$, $\pi_2 = T_1 + T_2 = \frac{t}{p} + \frac{4t}{2-p}$. We have,

$$
\begin{aligned}
\pi_i - T_1 - T_2 &= \sum_{i=3}^{n} T_i \\
&= \frac{\pi_{i-1} + \pi_{i-2} - T_1}{p},
\end{aligned}
$$

$$\Rightarrow \pi_i = \frac{\pi_{i-1} + \pi_{i-2}}{p} + T_1 + T_2 - \frac{T_1}{p},$$
$$\Rightarrow \pi_i = \frac{\pi_{i-1} + \pi_{i-2}}{p} + \frac{4t}{2-p} + \frac{t}{p} - \frac{2t}{p\,(2-p)}.$$

Therefore, π_i follows a recurrence relation, but with a constant $\frac{4t}{2-p} + \frac{t}{p} - \frac{2t}{p(2-p)}$. To eliminate the constant, we define a new variable,

$$\Omega_i = \Omega_i + c,\ \forall i = 1, \ldots, n.$$

In order to solve for c, we have

$$\Omega_i - c = \frac{\Omega_{i-1} + \Omega_{i-2} - 2c}{p} + \frac{4t}{2-p} + \frac{t}{p} - \frac{2t}{p\,(2-p)},$$

$$\Rightarrow pc - 2c + \frac{4+p}{2-p} + t - \frac{2t}{2-p} = 0,$$
$$\Rightarrow c = \frac{4+p}{(2-p)^2} + \frac{t}{2-p} - \frac{2t}{(2-p)^2}.$$

Substituting c into the expression for Ω_i, we have $\Omega_1 = \pi_1 + c = \frac{3t}{2-p} + \frac{4+p-2t}{(2-p)^2}$, $\Omega_2 = \pi_2 + c = \frac{t}{p} + \frac{5t}{2-p} + \frac{4+p-2t}{(2-p)^2}$, and $\Omega_i = \frac{\Omega_{i-1}+\Omega_{i-2}}{p}$, $\forall i = 3, \ldots, n$. We can use the following recurrence relation to solve for Ω_i and B.

$$\Omega_i = G\lambda_1^i + H\lambda_2^i, \tag{4.25}$$

where λ_1 and λ_2 are the roots of (4.22) given by (4.23) and (4.24).

The values of G and H can be solved for by using the values of Ω_1 and Ω_2. Thus,

$$
\begin{cases}
G \cdot \lambda_1 + H \cdot \lambda_2 = \Omega_1 & = \frac{3t}{2-p} + \frac{4+p-2t}{(2-p)^2} \\
G \cdot \lambda_1^2 + H \cdot \lambda_2^2 = \Omega_2 & = \frac{t}{p} + \frac{5t}{2-p} + \frac{4+p-2t}{(2-p)^2}
\end{cases}
$$

$$
\Longrightarrow
\begin{cases}
G = & -\frac{1}{\sqrt{1+4p}} p^2 \left(\frac{\left(1+\sqrt{1+4p}\right)^2 \left(-\frac{2t}{(2-p)^2} + \frac{3t}{2-p} + \frac{4pt}{(2-p)^2}\right)}{4p^2} \right. \\
& \left. -\frac{1}{2p}\left(1+\sqrt{1+4p}\right)\left(-\frac{2t}{(2-p)^2} + \frac{5t}{2-p} + \frac{t}{p} + \frac{4pt}{(2-p)^2}\right)\right) \\
H = & -\frac{-4pt-7p^2t-4p\sqrt{1+4p}t-p^2\sqrt{1+4p}t}{(-2+p)^2\sqrt{1+4p}\left(1+\sqrt{1+4p}\right)}
\end{cases}
$$

Substituting for λ_1, λ_2, G and H into (4.25), we have

$$
\begin{aligned}
\Omega_i \;=\; & \frac{-1}{(-2+p)^2\sqrt{1+4p}\left(1+\sqrt{1+4p}\right)} \left(2^{-i}pt \left(\left(1+\sqrt{1+4p}\right) \left(\left(\frac{1-\sqrt{1+4p}}{p} \right)^i \right.\right.\right. \\
& \left. -4 \left(\frac{1+\sqrt{1+4p}}{p} \right)^i \right) + p \left(-5 \left(\frac{1-\sqrt{1+4p}}{p} \right)^i + \sqrt{1+4p} \left(\frac{1-\sqrt{1+4p}}{p} \right)^i \right. \\
& \left.\left.\left. -7 \left(\frac{1+\sqrt{1+4p}}{p} \right)^n - \sqrt{1+4p} \left(\frac{1+\sqrt{1+4p}}{p} \right)^n \right)\right)\right).
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
B &= \sum_{k=1}^{n} q_k \\
&= \sum_{k=1}^{n} T_k + n \cdot \frac{2t}{p-2} \\
&= \pi_n + n \cdot \frac{2t}{p-2} \\
&= \Omega_n - c + n \cdot \frac{2t}{p-2} \\
&= \Omega_n - \frac{4+p}{(2-p)^2} + \frac{t}{2-p} - \frac{2t}{(2-p)^2} + n \cdot \frac{2t}{p-2} \\
&= \frac{1}{(-2+p)^2\sqrt{1+4p}\left(1+\sqrt{1+4p}\right)} \left(2^{-n} t \left(2^{1+n} n (-2+p) \left(1+4p+\sqrt{1+4p}\right)\right.\right. \\
&\quad -p\left(\left(1+\sqrt{1+4p}\right)\left(3 \cdot 2^n + \left(\frac{1-\sqrt{1+4p}}{p}\right)^n - 4\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right. \\
&\quad +p\left(3 \cdot 2^{2+n} - 5\left(\frac{1-\sqrt{1+4p}}{p}\right)^n + \sqrt{1+4p}\left(\frac{1-\sqrt{1+4p}}{p}\right)^n\right) \\
&\quad \left.\left.\left. -7\left(\frac{1+\sqrt{1+4p}}{p}\right)^n - \sqrt{1+4p}\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right)\right).
\end{aligned}
$$

Note that, both A and B are positive. If $B < U$, we have $s_1^{'} > 0$ (by (4.20)), and all $s_i^{'} > 0$, $\forall i = 2, \ldots, n$. (see (4.16) and (4.17)); otherwise, we decrease n by 1 and repeat the process. Therefore,

$$
\begin{aligned}
s_1^{'} =& \frac{U-B}{A} \\
=& \Bigg(U - \frac{1}{(-2+p)^2\sqrt{1+4p}\left(1+\sqrt{1+4p}\right)}\left(2^{-n}t\left(2^{1+n}n\left(-2+p\right)\left(1+4p+\sqrt{1+4p}\right)\right.\right. \\
& -p\left(\left(1+\sqrt{1+4p}\right)\left(3\cdot 2^n+\left(\frac{1-\sqrt{1+4p}}{p}\right)^n-4\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right. \\
& +p\left(3\cdot 2^{2+n}-5\left(\frac{1-\sqrt{1+4p}}{p}\right)^n+\sqrt{1+4p}\left(\frac{1-\sqrt{1+4p}}{p}\right)^n\right) \\
& \left.\left.\left.-7\left(\frac{1+\sqrt{1+4p}}{p}\right)^n-\sqrt{1+4p}\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right)\right)\Bigg)/ \\
& \Bigg(\frac{2^n p}{(-2+p)\sqrt{1+4p}}\Bigg(2^{1+n}\sqrt{1+4p}+(\frac{1-\sqrt{1+4p}}{p})^n-\sqrt{1+4p}(\frac{1-\sqrt{1+4p}}{p})^n \\
& -(\frac{1+\sqrt{1+4p}}{p})^n-\sqrt{1+4p}(\frac{1+\sqrt{1+4p}}{p})^n+ \\
& p\left(-(\frac{1-\sqrt{1+4p}}{p})^n+(\frac{1+\sqrt{1+4p}}{p})^n\right)\Bigg)\Bigg).
\end{aligned} \tag{4.26}
$$

The value of optimal makespan, M^*, can be determined by Expression (4.19).

Remark 4.2. *Expression (4.26) reduces to Expression (4.14) when $t = 0$ and $p = 1$.*

Development of closed-form expressions for sublot sizes when $p = 2$

If $p = 2$, the above expressions do not hold because of the presence of $(p - 2)$ in the denominator. Expressions (4.16) - (4.18), can now be expressed as,

$$s_2^{'} = \frac{3}{2}s_1^{'} + \frac{t}{2}, \tag{4.27}$$

$$s_i^{'} = \frac{s_{i-1}^{'} + s_{i-2}^{'}}{2} + t, \quad i = 3 \ldots n, \tag{4.28}$$

$$\sum_{i=1}^{n} s_i^{'} = U. \tag{4.29}$$

Let $F_i = s_i^{'}+it^{'}$, where $t^{'} = -\frac{2}{3}t$. Then, $F_1 = s_1^{'}-\frac{2}{3}t$ and $F_2 = s_2^{'}-\frac{4}{3}t = \frac{3}{2}s_1^{'}+\frac{t}{2}-\frac{4}{3}t = \frac{3}{2}s_1^{'}+\frac{5}{6}t$. Since $s_i^{'} = \frac{s_{i-1}^{'}+s_{i-2}^{'}}{2} + t, i = 3, \ldots, n$, we have

$$\begin{aligned} \frac{F_{i-1} + F_{i-2}}{2} &= \frac{s_{i-1}^{'} + (i-1)\,t^{'} + s_{i-2}^{'} + (i-2)\,t^{'}}{2} \\ &= s_i^{'} - t + it^{'} - \frac{3}{2}t' \\ &= s_i^{'} + it^{'} \\ &= F_i. \end{aligned}$$

Once again, we can use the following recurrence relation to solve for F_i.

$$F_i = C^{'}\lambda_1^{'i} + D^{'}\lambda_2^{'i},$$

where $\lambda_1^{'}$ and $\lambda_2^{'}$ are the root of

$$r^2 - \frac{1}{2}r - \frac{1}{2} = 0.$$

Therefore, $\lambda_1^{'} = 1$ and $\lambda_2^{'} = -\frac{1}{2}$. The value of $C^{'}$ and $D^{'}$ can be obtained by using the values of T_1 and T_2. Thus,

$$\begin{cases} C^{'} \cdot \lambda_1^{'} + D^{'} \cdot \lambda_2^{'} = F_1 & = s_1^{'} - \frac{2}{3}t \\ C^{'} \cdot \lambda_1^{'2} + D^{'} \cdot \lambda_2^{'2} = F_2 & = \frac{3}{2}s_1^{'} + \frac{5}{6}t \end{cases}$$

$$\Longrightarrow \begin{cases} C^{'} = & \frac{4}{3}s_1^{'} - \frac{7}{9}t \\ D^{'} = & \frac{2}{3}s_1^{'} - \frac{2}{9}t \end{cases}$$

By substituting the values of $\lambda_1^{'}$, $\lambda_2^{'}$, $C^{'}$ and $D^{'}$ into (4.28), we have

$$\begin{aligned} F_i &= C^{'}\lambda_1^{'i} + D^{'}\lambda_2^{'i} \\ &= \frac{4}{3}s_1^{'} - \frac{7}{9}t + \left(\frac{2}{3}s_1^{'} - \frac{2}{9}t\right)\left(-\frac{1}{2}\right)^i. \end{aligned}$$

Therefore,

$$\begin{aligned} s_i^{'} &= F_i + \frac{2}{3}it \\ &= \frac{4}{3}s_1^{'} - \frac{7}{9}t + \left(\frac{2}{3}s_1^{'} - \frac{2}{9}t\right)\left(-\frac{1}{2}\right)^i + \frac{2}{3}it. \end{aligned}$$

By substituting the value of $s_2^{'}, s_3^{'}, \cdots, s_n^{'}$ into (4.29), we have

$$\sum_{i=1}^{n} s_i^{'} = \left(\frac{4}{3}s_1^{'} - \frac{7}{9}t\right)n + \left(\frac{2}{3}s_1^{'} - \frac{2}{9}t\right)\left(\frac{\left(-\frac{1}{2}\right)^n - 1}{3}\right) + \frac{1}{3}tn\,(n+1) = U.$$

Therefore,

$$s_1^{'} = \frac{U + \frac{7}{9}tn - \frac{2}{27}t\left(1 - \left(-\frac{1}{2}\right)^n\right) - \frac{1}{3}tn\,(n+1)}{\frac{4}{3}n - \frac{2}{9}\left(1 - \left(-\frac{1}{2}\right)^n\right)}. \tag{4.30}$$

When $s_1^{'} > 0$, the optimal makespan,

$$M^* = U + n \cdot t + 2s_1^{'} = U + nt + \frac{U + \frac{7}{9}tn - \frac{2}{27}t\left(1 - \left(-\frac{1}{2}\right)^n\right) - \frac{1}{3}tn\,(n+1)}{\frac{2}{3}n - \frac{1}{9}\left(1 - \left(-\frac{1}{2}\right)^n\right)}.$$

Proposition 4.5. *The size of the last sublot $s_1^{'}$ decreases monotonically with increment in the number of sublots, n.*

Proof. By (4.20), we have $s_1^{'} = \frac{U-B}{A}$, where A is the summation of the coefficients of $s_1^{'}$ and B is the summation of all the constant terms of $s_1^{'}$. By definition, both A and B are monotonically increasing in the number of sublots. Therefore, $s_1^{'}$ monotonic decreases with increment in the number of sublots. □

Proposition 4.6. *For a given number of sublots n, if the size of the last sublot, $s_1^{'}$, calculated using (4.16), (4.17), and (4.18), is greater than zero, while for $n+1$ sublots, the size of the last sublot is less than or equal to zero, then the makespan of the system will not improve for number of sublots greater than or equal to n', where $(n' - n) \cdot t \geq ps_1^{'}$, with the restriction that the sublot sizes are greater than or equal to zero.*

Proof. For a given number of sublot n, since $s_1^{'}$ is positive, all sublot sizes are positive. By Corollary 4.2, the optimal makespan, $M^*(n) = U + nt + ps_1^{'}$. If all the sublot sizes are greater than or equal to zero, a lower bound on the completion time of the last sublot (consider processing only on Stage 1),

$$LB\left(n'\right) = U + n' \cdot t \tag{4.31}$$

Comparing the lower bound on the makespan for number of sublots n' with the optimal makespan for number of sublot n, we have

$$LB\left(n'\right) - M^*\left(n\right) = U + n' \cdot t - U - n \cdot t - p \cdot s_1^{'} = \left(n' - n\right) \cdot t - p \cdot s_1^{'}. \tag{4.32}$$

Therefore, $LB\left(n'\right) \geq M^*\left(n\right)$, if $\left(n' - n\right) \cdot t \geq p \cdot s_1^{'}$.

Since the optimal makespan for n' sublots $M^*\left(n'\right) \geq LB\left(n'\right)$, we have $M^*\left(n'\right) \geq M^*\left(n\right)$. □

Example 4.2

Consider the example problem for which the data is given in Table 4.1. By Equation (4.26),

$$\begin{aligned}
s_1^{'} = \quad & \Bigg(U - \frac{1}{(-2+p)^2\sqrt{1+4p}\left(1+\sqrt{1+4p}\right)}\left(2^{-n}t\left(2^{1+n}n\left(-2+p\right)\left(1+4p+\sqrt{1+4p}\right)\right.\right. \\
& - p\left(\left(1+\sqrt{1+4p}\right)\left(3\cdot 2^n + \left(\frac{1-\sqrt{1+4p}}{p}\right)^n - 4\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right. \\
& + p\left(3\cdot 2^{2+n} - 5\left(\frac{1-\sqrt{1+4p}}{p}\right)^n + \sqrt{1+4p}\left(\frac{1-\sqrt{1+4p}}{p}\right)^n\right) \\
& \left.\left.\left.-7\left(\frac{1+\sqrt{1+4p}}{p}\right)^n - \sqrt{1+4p}\left(\frac{1+\sqrt{1+4p}}{p}\right)^n\right)\right)\right)\Bigg) / \\
& \left(\frac{2^n p}{(-2+p)\sqrt{1+4p}}\left(2^{1+n}\sqrt{1+4p} + (\frac{1-\sqrt{1+4p}}{p})^n - \sqrt{1+4p}(\frac{1-\sqrt{1+4p}}{p})^n\right.\right. \\
& - (\frac{1+\sqrt{1+4p}}{p})^n - \sqrt{1+4p}(\frac{1+\sqrt{1+4p}}{p})^n + \\
& \left.\left.p\left(-(\frac{1-\sqrt{1+4p}}{p})^n + (\frac{1+\sqrt{1+4p}}{p})^n\right)\right)\right) \\
= \quad & 9.29582.
\end{aligned}$$

Table 4.1: Data used in Example 4.2

Number of items in the lot U	200
Number of sublots n	6
Removal time per sublot at Stage 1, t	5
Processing time per item at Stage 2, p	1.5

By Equation (4.16),

$$s_2^{'} = s_1^{'}\left(1 + \frac{1}{1.5}\right) + \frac{5}{1.5} = 18.8264.$$

Similarly, by Equation (4.17), we have

$$s_3^{'} = \frac{s_2^{'} + s_1^{'}}{1.5} + \frac{2 * 5}{1.5} = 25.4148,$$

$$s_4^{'} = \frac{s_3^{'} + s_2^{'}}{1.5} + \frac{2 * 5}{1.5} = 36.1608,$$

$$s_5^{'} = \frac{s_4^{'} + s_3^{'}}{1.5} + \frac{2 * 5}{1.5} = 47.717,$$

$$s_6^{'} = \frac{s_5^{'} + s_4^{'}}{1.5} + \frac{2 * 5}{1.5} = 62.5852.$$

By equation (4.19), the optimal makespan,

$$M^* = U + nt + ps_1^{'} = 200 + 6 * 5 + 1.5 * 9.29582 = 243.94373.$$

The optimal schedule is shown in Figure 4.5.

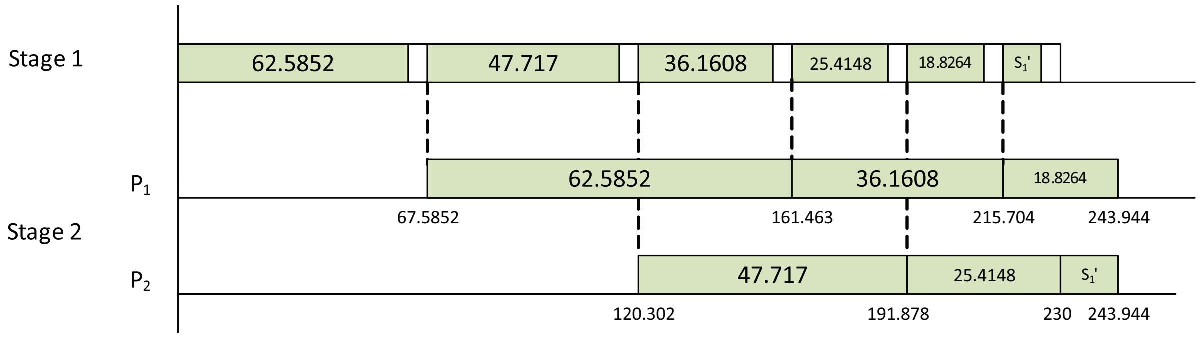

Figure 4.5: Optimal schedule for Example 4.2

4.4 Determination of Both Number of Sublots and Continuous Sublot Sizes for TSHFLSP-FlexN

Next, we relax the assumption of a known number of sublots and determine both optimal number of sublots and sublot sizes. First, we determine N_{pos}, which servers as an upper bound on number of sublots to obtain a feasible solution. Then, we present the overall scheme to solve the TSHFLSP-FlexN in Section 4.4.2.

4.4.1 Determination of N_{pos} and N^*_{max} for TSHFLSP-FlexN

Because of the presence of removal times, the makespan will increase as the number of sublots exceeds beyond a certain value. Therefore, we can use a binary search in concert with the result of Proposition 4.5 to determine the maximal number of sublots beyond which s'_1 will be negative. We designate the value so obtained as N_{pos}. The procedure is as follows:

Listing 4.1: Pseudocode of the procedure

```
Let the minimum number of sublots be, LB (=1 at start), and the
        maximum number of sublots be UB
```

```
For LB less than UB
        If the size of the last sublot , s'_1 > 0 when the number
                of sublots equals to UB , then
                        N_pos = UB
            Stop
        End If
        If the size of the last sublot s'_1 ≤ 0 when the number
                of sublots equals to LB , then
                        N_pos = LB
            Stop
        End if
        Let Mid = ⌊(LB+UB)/2⌋
        If the size of the last sublot s'_1 ≤ 0 when the number
                of sublots equals to Mid , then
                        UB = Mid − 1
        Else
                        LB = Mid + 1
        End if
End for
```

We can also use Proposition 4.6 to determine the number of sublots, $\bar{N}$, beyond which $C_{\max}$ cannot decrease. In particular, if $\hat{N}$ is a number of sublots for which $s_1^{'} \geq 0$, then

$$\bar{N} = \hat{N} + \left\lfloor \frac{ps_1^{'}}{t} \right\rfloor, \tag{4.33}$$

where $\lfloor x \rfloor$ is the largest integer less than x. Then, the maximum possible number of sublots,

$$N^*_{\max} = \mathrm{Min}(N_{\mathrm{pos}}, \bar{N}). \tag{4.34}$$

4.4.2 Proposed Algorithm

First, let $s_1^{'}(n)$ be the size of the last sublot obtained by using Expression (4.26) if $p \neq 2$, or by using Expression (4.30) if $p = 2$, for a given n; $M_{\text{opt}}(n) = U + nt + p \cdot s_1^{'}(n)$ (see 4.19) be the optimal makespan for n sublots; M^* be the optimal makespan; and N^* be the optimal number of sublots. The proposed algorithm, designated Algorithm TF-C, to determine optimal number of sublots, continuous sublot sizes, and makespan for TSHFLSP-FlexN is as follows:

Algorithm TF-C

Step 1. Let $M^* = M_{\text{opt}}(1)$, $N^* = 1$, and $n = 1$. Go to Step 2.

Step 2. Use the pseudocode presented in Section 4.4.1 to determine N_{pos}. Let $N^*_{\text{max}} = N_{\text{pos}}$, and go to Step 3.

Step 3. If $n \geq N^*_{\text{max}}$, go to Step 6; otherwise, go to Step 4.

Step 4. Calculate $M_{\text{opt}}(n)$, and $N^*_{\text{max}} = \text{Min}(N_{\text{pos}}, \bar{n})$, where $\bar{n}$ is obtained by Expression 4.33, and go to Step 5.

Step 5. If $M^* > M_{\text{opt}}(n)$, then let $M^* = M_{\text{opt}}(n)$, $N^* = n$. Let $n = n + 1$, and go to Step 3.

Step 6. We have the optimal solution M^* and optimal number of sublots N^*. Go to Step 7.

Step 7. Determine optimal sizes for N^* sublots using Expression (4.26), if $p \neq 2$, or (4.30), if $p = 2$, and Expressions (4.16), and (4.17). Stop.

4.5 Integer Sublot Sizes and Number of Sublots for TSHFLSP-FlexN

The developments in Sections 4.3 and 4.4 rely on the fact that the "alternative" assignment method results in an optimal solution. Based on this property, we have developed closed-form expressions to determine continuous sublot sizes. This property inherently fixes the sequence in which to process the sublots. However, it is not possible to do so for integer sublot sizes. Therefore, for this case, we need to additionally determine the sequence in which to process the sublots at Stage 1, and also, determine their allocation to the machines at Stage 2. This makes the problem more difficult to solve. The 1+2 hybrid flow shop problem for a given number of jobs and the objective of minimizing makespan is a special case of our problem obtained by fixing number of sublots and sublot sizes, and it has been shown to be NP-hard by Gupta (1988) [36]. Therefore, the TSHFLSP-FlexN problem is also NP-hard. We present an optimum-seeking method and a heuristic method to obtain integer sublot sizes and number of sublots for this problem. These methods rely on the results presented in Section 4.3.

Optimal solution using "alternate" assignment method

We have stated earlier (see Section 4.3.1) the optimality of "alternative" assignment method for a given number of sublots. We can thus fix the assignment variables, X_{ik}, $i = 1, \dots N^*_{\max}$, $k = 1, \dots m$, where $N^*_{\max}$ is determined in Section 4.4.1. We have the following algorithm.

Algorithm TF-I

Step 1. Apply “alternative” assignment method, and for all $i = 1, \ldots, N_{\text{pos}}$, let

$$\begin{cases} X_{i1} = 1, \\ \text{and} \\ X_{i2} = 0 \end{cases} \quad \text{when } i \text{ is an odd number, and} \quad \begin{cases} X_{i1} = 0, \\ \text{and} \\ X_{i2} = 1 \end{cases} \quad \text{when } i \text{ is an even number.}$$

Step 2. Solve model TSHFLSP-FlexN to obtain integer sublot sizes and number of sublots using CPLEX.

Heuristic TF-HI

In addition to applying the ”alternative” assignment method to fix the allocation of sublots to the machines at Stage 2, in this method, we also fix the number of sublots to N^*, which is obtained for TF-C. We have the following steps.

Heuristic TF-HI

Step 1. Calculate the optimal number of sublots, N^*, by using Algorithm TF-C.

Step 2. Let $N_{\max} = N^*$.

Step 3. Apply "alternatve" assignment method, and for all $i = 1, \ldots, N^*$, let

$$\begin{cases} X_{i1} = 1, \\ \text{and} \\ X_{i2} = 0 \end{cases} \quad \text{when } i \text{ is an odd number and} \begin{cases} X_{i1} = 0, \\ \text{and} \\ X_{i2} = 1 \end{cases} \quad \text{when } i \text{ is an even number.}$$

Step 4. Let $Y_i = 1$ for $i = 1, \ldots, N^*$.

Step 5. Solve model TSHFLSP-FlexN to obtain integer sublot sizes using CPLEX.

4.6 Computational Investigation

In this section, we present the results of our computational investigation to compare: (i) the performance of Algorithm TF-C with the direct solution of model TSHFLSP-FlexN by CPLEX, and (ii) the performance of the proposed Algorithm TF-I and that of the heuristic method TF-HI with the direct solution of model TSHFLSP-FlexN by CPLEX.

4.6.1 Continuous Sublot Sizes

The test data that we used for our experimentation is presented in Table 4.2. We used three instances of U consisting of 500, 1000, and 2000 items. For each instance of number of items, when we fixed $t = 1$, four different ranges of processing times, p, were used to obtain high to low speed of machines at Stage 2. Similarly, when we fixed $p = 1$, four different ranges of sublot-attached removal times, t, were used to obtain small to large level of removal times.

For each combination of U, p, and t, we used two levels of the maximum number of sublots, $N_{\max}$, namely, low and high. This amounted to a total of 3*4*2*2=48 instances. For each instance, we used 5 replications by randomly generating 5 values of processing times or sublot-attached setup times over the range specified for that instance. All numerical tests were executed on a Sony computer with Intel i7 Q740 CPU and 8GB DDR3 memory.

Table 4.2: Data used for computational investigation for continuous sublot sizes

U	500,1000,2000
t	Uniform distribution [0.02,0.2], [0.2,1.0], [1.0,5.0], [5.0,50.0]
p	Uniform distribution [0.02,0.2], [0.2,1.0], [1.0,5.0], [5.0,50.0]
$N_{\max}$	20, 40

Tables 4.3 and 4.4 present the minimum, average and maximum values of CPU times required for the direct solution by CPLEX 11.2 of the TSHFLSP-FlexN model (with the variables X_{ik}, $i = 1, \dots N^*_{\max}$, $k = 1, \dots m$, fixed according to the "alternative assignment" method) to obtain continuous sublot sizes. These depict the best, average, and worst-case performance of this method. We also present the CPU times required by the proposed Algorithm TF-C. Note that the CPU time required by the proposed Algorithm TF-C is less than 0.01 seconds for all datasets. On the other hand, the CPU time required for the direct solution by CPLEX is quite large and increases with increment in the value of $N_{\max}$. Some of the higher CPU times are obtained for higher ratios of p and t. For instance, see the results for datasets 2, 6, 10, 18, 22, 24, 26, 30, 32, 38, 40, 46, and 48.

4.6.2 Integer Sublot Sizes

Next, we present the results of our computational investigation to obtain integer sublot sizes by Algorithm TF-I and heuristic method, TF-HI. We also study the performances of the

Table 4.3: CPU times (in seconds) required by the direct solution of TSHFLSP-FlexN by CPLEX 11.2 and the proposed Algorithm TF-C to obtain continuous sublot sizes

Dataset	U	p	t	$N_{\max}$	CPU Time (secs)			
					CPLEX			Algorithm TF-C
					Min	Average	Max	
1	500	1	[0.02,0.2]	20	0.499	0.555	0.593	< 0.01
2	500	1	[0.02,0.2]	40	6.178	42.233	80.060	< 0.01
3	500	1	[0.2,1]	20	0.406	0.440	0.452	< 0.01
4	500	1	[0.2,1]	40	1.794	48.732	85.894	< 0.01
5	500	1	[1,5]	20	0.390	0.418	0.437	< 0.01
6	500	1	[1,5]	40	6.786	14.489	38.673	< 0.01
7	500	1	[5,50]	20	0.312	0.449	0.546	< 0.01
8	500	1	[5,50]	40	0.468	1.151	3.214	< 0.01
9	1000	1	[0.02,0.2]	20	0.499	0.549	0.577	< 0.01
10	1000	1	[0.02,0.2]	40	1.622	26.395	68.001	< 0.01
11	1000	1	[0.2,1]	20	0.499	1.117	1.685	< 0.01
12	1000	1	[0.2,1]	40	6.583	78.999	132.897	< 0.01
13	1000	1	[1,5]	20	0.421	0.549	0.733	< 0.01
14	1000	1	[1,5]	40	8.861	15.934	26.692	< 0.01
15	1000	1	[5,50]	20	0.468	0.933	1.420	< 0.01
16	1000	1	[5,50]	40	0.858	2.752	9.235	< 0.01
17	2000	1	[0.02,0.2]	20	0.484	0.537	0.655	< 0.01
18	2000	1	[0.02,0.2]	40	7.285	48.061	121.618	< 0.01
19	2000	1	[0.2,1]	20	0.484	0.612	0.718	< 0.01
20	2000	1	[0.2,1]	40	14.992	32.904	51.262	< 0.01
21	2000	1	[1,5]	20	0.515	0.537	0.562	< 0.01
22	2000	1	[1,5]	40	12.028	41.484	70.809	< 0.01
23	2000	1	[5,50]	20	0.499	0.543	0.624	< 0.01
24	2000	1	[5,50]	40	3.838	185.529	461.763	< 0.01

Table 4.4: CPU times (in seconds) required by the direct solution of TSHFLSP-FlexN by CPLEX 11.2 and the proposed Algorithm TF-C to obtain continuous sublot sizes (continued)

Dataset	U	p	t	$N_{\max}$	CPU Time (secs)			
					CPLEX			Algorithm TF-C
					Min	Average	Max	
25	500	[0.02,0.2]	1	20	0.406	0.449	0.484	< 0.01
26	500	[0.02,0.2]	1	40	8.299	27.693	58.407	< 0.01
27	500	[0.2,1]	1	20	0.406	0.456	0.484	< 0.01
28	500	[0.2,1]	1	40	8.081	46.339	114.489	< 0.01
29	500	[1,5]	1	20	0.515	1.641	2.590	< 0.01
30	500	[1,5]	1	40	57.767	289.114	1118.530	< 0.01
31	500	[5,50]	1	20	1.186	1.513	1.856	< 0.01
32	500	[5,50]	1	40	27.503	94.715	173.645	< 0.01
33	1000	[0.02,0.2]	1	20	0.437	0.484	0.515	< 0.01
34	1000	[0.02,0.2]	1	40	4.992	6.596	10.234	< 0.01
35	1000	[0.2,1]	1	20	0.359	0.496	0.655	< 0.01
36	1000	[0.2,1]	1	40	5.678	8.683	16.926	< 0.01
37	1000	[1,5]	1	20	1.451	1.697	1.934	< 0.01
38	1000	[1,5]	1	40	45.396	372.037	898.129	< 0.01
39	1000	[5,50]	1	20	1.279	1.416	1.669	< 0.01
40	1000	[5,50]	1	40	22.948	417.596	1131.180	< 0.01
41	2000	[0.02,0.2]	1	20	0.437	0.521	0.624	< 0.01
42	2000	[0.02,0.2]	1	40	13.806	43.571	72.962	< 0.01
43	2000	[0.2,1]	1	20	0.390	0.480	0.608	< 0.01
44	2000	[0.2,1]	1	40	1.435	24.386	59.655	< 0.01
45	2000	[1,5]	1	20	1.466	1.713	1.872	< 0.01
46	2000	[1,5]	1	40	29.671	674.449	2206.480	< 0.01
47	2000	[5,50]	1	20	1.232	1.507	1.934	< 0.01
48	2000	[5,50]	1	40	16.833	213.051	503.322	< 0.01

proposed heuristic method and a lower bound, which is the optimal solution for continuous sublots sizes. All datasets used for integer sublot sizes are identical to those for continuous sublot sizes, except that we changed the two levels of maximum number of sublots, $N_{\max}$, to 15 and 30. The heuristic method was programmed using Visual C++ (Version 2008). Besides the average CPU time, we also report on ratio of heuristic solution and optimal solution values, and the ratio of lower bound and optimal solution values. The results of CPU times are presented in Tables 4.5 and 4.6. Note that the heuristic method TF-HI requires less CPU time for all datasets compared with that required by Algorithm TF-I. TF-I could not solve the problem for dataset 32 within 3600 seconds of CPU time for all 5 replications, while Heuristic TF-HI requires only 2160.187 seconds on average. The results on the ratios of the values of heuristic solution and optimal solution and lower bound and optimal solution, which are shown in Tables 4.7 and 4.8, reveal that the value of the heuristic solution is within 0.15% of the optimal solution.

4.7 Conclusion

In this chapter, we have addressed a single-lot lot streaming problem in a two-stage $1 + 2$ hybrid flow shop in the presence of sublot-attached removal time for each sublot at Stage 1. The objective is to minimize the makespan. We present a mixed integer programming model TSHFLSP-FlexN for the general problem of determining number of sublots, sublot sizes, and allocation of sublots for processing on the parallel machines at Stage 2. For a given number of sublots, optimal, continuous sublot sizes can be obtained by using an "alternate" assignment method. We have developed closed-form expressions for obtaining optimal makespan and sublot sizes for the $1 + 2$ problem in the presence of sublot-attached removal times at Stage 1. When the number of sublots is also to be determined, we have presented an optimum seeking method, TF-C, for continuous sublot sizes. For integer sublot sizes, we have presented both an optimum seeking method, TF-I, and a heuristic method, TF-HI. Our computational investigation has revealed the efficacy of the proposed method to

Table 4.5: Comparison between CPU times (in seconds) required by Algorithm TF-I and the proposed heuristic method TF-HI

Data	U	p	t	$N_{\max}$	TF-I			TF-HI		
					Min	Ave	Max	Min	Ave	Max
1	500	1	[0.02,0.2]	15	0.640	0.671	0.764	0.187	0.209	0.218
2	500	1	[0.02,0.2]	30	2.714	109.606	447.052	0.203	0.218	0.250
3	500	1	[0.2,1]	15	0.406	0.615	1.030	0.187	0.212	0.218
4	500	1	[0.2,1]	30	2.012	2.203	2.434	0.203	0.215	0.234
5	500	1	[1,5]	15	0.406	0.449	0.515	0.218	0.218	0.218
6	500	1	[1,5]	30	2.293	2.484	2.839	0.187	0.203	0.218
7	500	1	[5,50]	15	0.281	0.340	0.437	0.187	0.200	0.203
8	500	1	[5,50]	30	0.515	1.666	2.761	0.172	0.193	0.203
9	1000	1	[0.02,0.2]	15	0.530	0.646	0.796	0.203	0.225	0.250
10	1000	1	[0.02,0.2]	30	2.699	19.622	84.771	0.203	0.222	0.234
11	1000	1	[0.2,1]	15	0.437	0.721	1.014	0.203	0.222	0.250
12	1000	1	[0.2,1]	30	0.952	2.137	3.042	0.203	0.218	0.250
13	1000	1	[1,5]	15	0.328	0.427	0.640	0.187	0.200	0.203
14	1000	1	[1,5]	30	1.014	2.630	4.930	0.187	0.193	0.203
15	1000	1	[5,50]	15	0.328	0.359	0.390	0.187	0.197	0.203
16	1000	1	[5,50]	30	2.153	3.055	5.585	0.187	0.197	0.203
17	2000	1	[0.02,0.2]	15	0.577	0.627	0.718	0.203	0.209	0.234
18	2000	1	[0.02,0.2]	30	3.541	252.194	818.896	0.234	0.234	0.234
19	2000	1	[0.2,1]	15	0.671	0.764	0.983	0.218	0.240	0.265
20	2000	1	[0.2,1]	30	1.934	5.884	11.310	0.203	0.215	0.234
21	2000	1	[1,5]	15	0.328	0.452	0.640	0.203	0.209	0.218
22	2000	1	[1,5]	30	1.841	2.895	5.382	0.203	0.203	0.203
23	2000	1	[5,50]	15	0.359	0.412	0.515	0.187	0.200	0.218
24	2000	1	[5,50]	30	0.905	1.966	2.387	0.187	0.206	0.234

Table 4.6: Comparison between CPU times (in seconds) required by Algorithm TF-I and the proposed heuristic method TF-HI (continued)

Data	U	p	t	$N_{\max}$	TF-I			TF-HI		
					Min	Ave	Max	Min	Ave	Max
25	500	[0.02,0.2]	1	15	0.468	0.505	0.546	0.296	0.309	0.312
26	500	[0.02,0.2]	1	30	2.933	3.707	5.023	0.281	0.296	0.312
27	500	[0.2,1]	1	15	0.406	0.565	0.655	0.265	0.284	0.296
28	500	[0.2,1]	1	30	2.059	2.555	3.323	0.296	0.306	0.312
29	500	[1,5]	1	15	1.186	2161	$3600^{\dagger}$	0.359	1113.934	$3600^{\dagger}$
30	500	[1,5]	1	30	4.586	2214	$3600^{\dagger}$	0.437	2160.452	$3600^{\dagger}$
31	500	[5,50]	1	15	33.852	2887	$3600^{\dagger}$	0.312	2424.096	$3600^{\dagger}$
32	500	[5,50]	1	30	$3600^{\dagger}$	$3600^{\dagger}$	$3600^{\dagger}$	0.265	2160.187	$3600^{\dagger}$
33	1000	[0.02,0.2]	1	15	0.281	0.315	0.343	0.187	0.206	0.218
34	1000	[0.02,0.2]	1	30	1.498	2.125	2.855	0.203	0.206	0.218
35	1000	[0.2,1]	1	15	0.359	0.465	0.764	0.203	0.222	0.250
36	1000	[0.2,1]	1	30	1.763	1.938	2.122	0.203	0.215	0.234
37	1000	[1,5]	1	15	1.310	128.289	568.156	0.312	10.452	47.206
38	1000	[1,5]	1	30	3.650	2173	$3600^{\dagger}$	0.374	2161	$3600^{\dagger}$
39	1000	[5,50]	1	15	2.480	13.482	36.551	0.234	0.437	0.562
40	1000	[5,50]	1	30	47.034	2201	$3600^{\dagger}$	0.265	0.615	0.827
41	2000	[0.02,0.2]	1	15	0.296	0.318	0.328	0.203	0.215	0.234
42	2000	[0.02,0.2]	1	30	1.622	2.718	4.384	0.218	0.218	0.218
43	2000	[0.2,1]	1	15	0.343	0.477	0.624	0.203	0.215	0.218
44	2000	[0.2,1]	1	30	1.825	2.468	3.385	0.203	0.228	0.250
45	2000	[1,5]	1	15	0.608	1.248	2.215	0.234	0.368	0.671
46	2000	[1,5]	1	30	2.933	1447	$3600^{\dagger}$	0.328	1440.331	$3600^{\dagger}$
47	2000	[5,50]	1	15	1.825	2.861	5.398	0.530	0.555	0.577
48	2000	[5,50]	1	30	28.158	83.052	182.896	0.234	0.256	0.265

$3600^{\dagger}$: Exceeds pre-specified time limit of 3600 seconds.

Table 4.7: Comparison of ratios of the values of the proposed heuristic solution (obtained for TF-HI) and optimal solution (obtained using TF-I), and lower bound and optimal solution

Data	U	p	t	$N_{\max}$	$\frac{\text{TF-HI}}{\text{TF-I}} \times 100\%$			$\frac{\text{Lower Bound}}{\text{TF-I}} \times 100\%$		
					Min	Ave	Max	Min	Ave	Max
1	500	1	[0.02,0.2]	15	1.0000	1.0000	1.0000	0.9985	0.9986	0.9987
2	500	1	[0.02,0.2]	30	1.0000	1.0000	1.0000	0.9983	0.9986	0.9988
3	500	1	[0.2,1]	15	1.0000	1.0000	1.0000	0.9987	0.9991	0.9996
4	500	1	[0.2,1]	30	1.0000	1.0000	1.0001	0.9987	0.9990	0.9993
5	500	1	[1,5]	15	1.0000	1.0000	1.0000	0.9986	0.9990	0.9992
6	500	1	[1,5]	30	1.0000	1.0000	1.0001	0.9987	0.9989	0.9992
7	500	1	[5,50]	15	1.0000	1.0000	1.0000	0.9986	0.9991	0.9994
8	500	1	[5,50]	30	1.0000	1.0000	1.0000	0.9992	0.9993	0.9994
9	1000	1	[0.02,0.2]	15	1.0000	1.0000	1.0001	0.9994	0.9994	0.9995
10	1000	1	[0.02,0.2]	30	1.0000	1.0000	1.0001	0.9991	0.9993	0.9995
11	1000	1	[0.2,1]	15	1.0000	1.0000	1.0000	0.9991	0.9994	0.9996
12	1000	1	[0.2,1]	30	1.0000	1.0000	1.0000	0.9992	0.9994	0.9996
13	1000	1	[1,5]	15	1.0000	1.0000	1.0000	0.9993	0.9994	0.9995
14	1000	1	[1,5]	30	1.0000	1.0000	1.0001	0.9994	0.9995	0.9997
15	1000	1	[5,50]	15	1.0000	1.0000	1.0001	0.9996	0.9997	0.9998
16	1000	1	[5,50]	30	1.0000	1.0000	1.0000	0.9994	0.9995	0.9996
17	2000	1	[0.02,0.2]	15	1.0000	1.0000	1.0001	0.9997	0.9998	0.9998
18	2000	1	[0.02,0.2]	30	1.0000	1.0000	1.0001	0.9996	0.9996	0.9997
19	2000	1	[0.2,1]	15	1.0000	1.0000	1.0001	0.9996	0.9997	0.9998
20	2000	1	[0.2,1]	30	1.0000	1.0000	1.0000	0.9997	0.9997	0.9998
21	2000	1	[1,5]	15	1.0000	1.0000	1.0001	0.9997	0.9997	0.9998
22	2000	1	[1,5]	30	1.0000	1.0000	1.0000	0.9997	0.9998	0.9998
23	2000	1	[5,50]	15	1.0000	1.0000	1.0001	0.9997	0.9998	0.9999
24	2000	1	[5,50]	30	1.0000	1.0000	1.0000	0.9997	0.9998	0.9998

Table 4.8: Comparison of ratios of the values of the proposed heuristic solution (obtained for TF-HI) and optimal solution (obtained using TF-I), and lower bound and optimal solution (continued)

Data	U	p	t	$N_{\max}$	$\frac{\text{TF-HI}}{\text{TF-I}} \times 100\%$			$\frac{\text{Lower Bound}}{\text{TF-I}} \times 100\%$		
					Min	Ave	Max	Min	Ave	Max
25	500	[0.02,0.2]	1	15	1.0000	1.0000	1.0000	0.9996	0.9998	1.0000
26	500	[0.02,0.2]	1	30	1.0000	1.0000	1.0000	0.9998	0.9999	1.0000
27	500	[0.2,1]	1	15	1.0000	1.0000	1.0000	0.9987	0.9994	0.9998
28	500	[0.2,1]	1	30	1.0000	1.0000	1.0000	0.9988	0.9994	0.9997
29	500	[1,5]	1	15	1.0000	1.0001	$1.0005^{\dagger}$	0.9974	0.9982	0.9990
30	500	[1,5]	1	30	1.0000	1.0000	$1.0001^{\dagger}$	0.9986	0.9989	0.9993
31	500	[5,50]	1	15	1.0000	$1.0007^{\dagger}$	$1.0015^{\dagger}$	0.9998	0.9998	0.9998
32	500	[5,50]	1	30	$1.0001^{\dagger}$	$1.0002^{\dagger}$	$1.0002^{\dagger}$	-*	-*	-*
33	1000	[0.02,0.2]	1	15	1.0000	1.0000	1.0000	0.9998	0.9999	1.0000
34	1000	[0.02,0.2]	1	30	1.0000	1.0000	1.0000	0.9999	0.9999	1.0000
35	1000	[0.2,1]	1	15	1.0000	1.0000	1.0000	0.9991	0.9996	0.9999
36	1000	[0.2,1]	1	30	1.0000	1.0000	1.0000	0.9995	0.9996	0.9999
37	1000	[1,5]	1	15	1.0000	1.0000	1.0000	0.9989	0.9994	0.9999
38	1000	[1,5]	1	30	1.0000	1.0000	$1.0000^{\dagger}$	0.9991	0.9994	0.9996
39	1000	[5,50]	1	15	1.0000	1.0000	1.0000	0.9998	0.9998	1.0000
40	1000	[5,50]	1	30	1.0000	1.0001	$1.0003^{\dagger}$	0.9998	0.9999	0.9999
41	2000	[0.02,0.2]	1	15	1.0000	1.0000	1.0000	0.9998	0.9999	1.0000
42	2000	[0.02,0.2]	1	30	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000
43	2000	[0.2,1]	1	15	1.0000	1.0000	1.0001	0.9997	0.9999	1.0000
44	2000	[0.2,1]	1	30	1.0000	1.0000	1.0001	0.9998	0.9999	0.9999
45	2000	[1,5]	1	15	1.0000	1.0000	1.0000	0.9994	0.9997	0.9999
46	2000	[1,5]	1	30	1.0000	1.0000	$1.0000^{\dagger}$	0.9996	0.9997	0.9998
47	2000	[5,50]	1	15	1.0000	1.0000	1.0001	0.9999	0.9999	1.0000
48	2000	[5,50]	1	30	1.0000	1.0000	1.0000	0.9997	0.9999	1.0000

†: We report TF-HI over lower bound when optimal solution was not obtained within 3600 seconds of CPU time.
*: Not applicable because optimal solution was not obtained within 3600 seconds of CPU time.

obtain optimal, continuous sublot sizes, and of the heuristic method to obtain near-optimal integer sublot sizes (within 0.15% optimality gap) while requiring a few seconds of CPU time.

Chapter 5

Two-stage, Multiple-lot, Lot Streaming Problem for a $1+2$ Hybrid Flow Shop

In this chapter, we extend the single-lot lot streaming problem presented in Chapter 4 for a $1+2$ hybrid flow shop to the case of the multiple lots, where each lot contains items of a unique product type. We consider two objective functions: minimizing the makespan, and minimizing the sum of the completion times for all the lots. The additional decision for the sequencing of the lots makes the problem more difficult to analyze. Our approach for this problem exploits some basic results derived for the TSHFLSP-FlexN problem in Chapter 4. We present heuristic solution methods for both the instances of the problem. Our computational investigation reveals the efficacy of these methods.

5.1 Introduction

Our work in this chapter extends the single-lot lot streaming problem for a two-stage hybrid flow shop presented in Chapter 4 to the multiple-lot case, for the objective of minimizing both the makespan and the sum of the completion times of all the lots. In the presence of multiple lots, we need to simultaneously determine an optimal sequence in which to process the lots, number of sublots for each lot, sublot sizes, and the allocation of sublots to the machines at the second stage. Literature review for multiple-lot problems using lot streaming technique has been presented in Chapter 2 for both time-based and cost-based situations.

Our work in this chapter is different from that presented in the literature because of the simultaneous consideration of lot streaming and multiple lots in a hybrid flow shop. The configuration of the multiple-lot $1 + 2$ hybrid flow shop is illustrated in Figure 5.1. The first stage of the system only consists of one machine, while there are two parallel machines at the second stage. All the lots (product types) are processed on the machine at Stage 1, and then, they are transferred to one of the parallel machines for processing at Stage 2. The production lots consist of 100 and 60 items. Sublot-attached setup is incurred on the machine at Stage 1. For the example, these values are assumed to be 10 units for lot A, and 5 units for lot B. The processing times for lot A are 1 unit per item on the machine at Stage 1 and 0.5 units per item on the machines at Stage 2, while for lot B, the processing times are 1 unit per item on the machine at Stage 1 and 2 units per item on the machines at Stage 2. Figure 5.1(a) depicts Schedule I in which the processing of lot A precedes that of lot B. The sublot sizes used for lot A are 20 and 80, and those for lot B are 20 and 40. Schedule II is shown in Figure 5.1(b) in which the processing sequence of the lots is altered, while the sublot sizes used for both the lots are kept the same as in Schedule I. As a result of this change in sequence, the makespan value reduces from 270 to 230 units. If we change the sublot sizes for lot B to be 20, 20 and 20, and the sublot sizes for lot A to be 40, 40 and 20, but keep the sequence to be lot A followed by lot B, the makespan further reduces from 235 to 215 unit times (see Figure 5.1(c)). Note that even though Schedule III is better than Schedules I and II, it need not be the optimal solution. Our aim is to determine the sequence of the lots, the number of sublots in each lot, sublot sizes, and the allocation of the sublots to the machines at the second stage, so as to minimize the makespan or the sum of the completion times of the lots.

This chapter is organized as follows. In Section 5.2, we describe the problem that we address in this chapter and present some basic properties. A mixed integer programming formulation for the problem, is then, presented in Section 5.3. In Section 5.4, we introduce different dispatching rules, and present heuristic methods for the solution of the problem. We present a computational investigation to test the efficacy of the proposed heuristic methods in Section 5.5. Some concluding remarks are, then, made in Section 5.6.

5.2 Problem Definition, Basic Properties, Notation, and Assumptions

In this section, we formally define the problem, present basic properties, and introduce the notation and assumptions.

5.2.1 Problem Definition

The problem that we address can be concisely stated as follows: *A two-stage hybrid flow shop consists of one machine at Stage 1 and two parallel machines at Stage 2. Q production lots are available for processing in the system. Each lot j consists of U_j items. Given a maximum number of sublots for each lot j and presence of sublot-attached setup time before processing of each sublot on the machine at Stage 1, determine the optimal sequence of the lots, the optimal number of sublots for each lot, the optimal sublot sizes, and the allocation of the sublots to the machines at Stage 2, in order to minimize the makespan or sum of the completion times.* We designate the problem for makespan objective function as TSMLLSP-

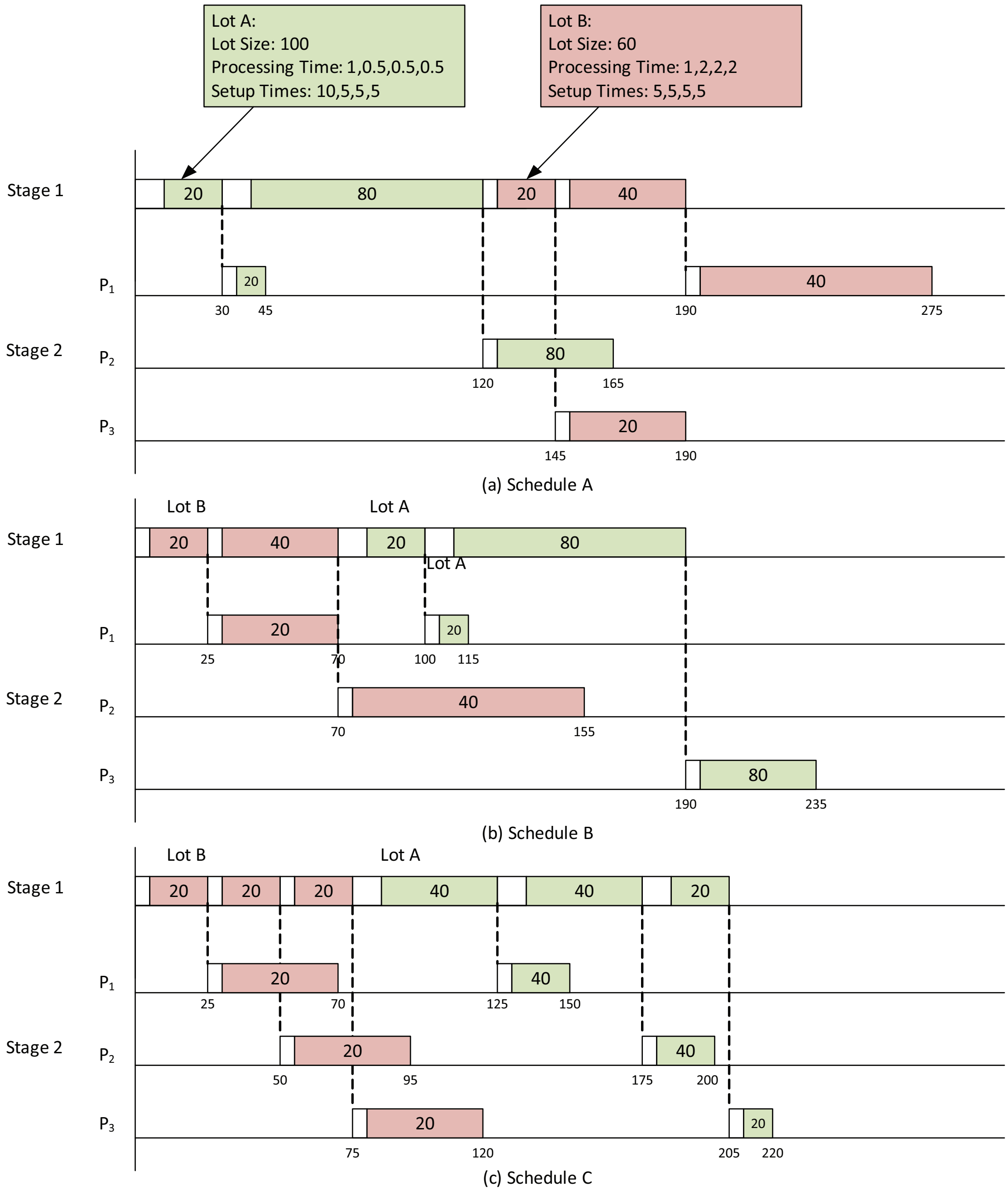

Figure 5.1: Example of a multiple-lot 1 + 2 hybrid flow shop

M (two-stage multiple-lot lot streaming problem for makespan objective) and that for the sum of the completion times of the lots as TSMLLSP-C (two-stage multiple-lot lot streaming problem for completion time objective). We consider the case of both continuous and integer sublot sizes.

5.2.2 Properties

Proposition 5.1. *There exists an optimal solution the sequence in which the lots are processed in both the stages is identical and the sublot sizes used for processing different lots on the machine(s) at the first and second stages are consistent.*

It is easy to see that if the sequences of the lots on the machines at Stage 1 and Stage 2 are different, then we can modify the sequences and make them identical without worsening the makespan. A similar argument holds for the sublot sizes as well. We also consider identical sequence for the TSMLLSP-C problem.

Proposition 5.2. *TSMLLSP-M problem is NP-complete. TSMLLSP-C problem is NP-hard in the strong sense.*

Proof: Gupta (1988) [36] has shown the two-stage hybrid flow shop makespan problem to be NP-complete if at least one of the stages has 2 or more machines. This problem is a special case of TSMLCSP-M. Du and Leung (1993) [24] have shown that the two machine flow shop problem to minimize mean flow time is NP-hard in the strong sense, which is a special case of our TSMLLSP-C problem. □

5.2.3 Notation

Consider the following notation:

Parameters:

- Q: Number of production lots.
- U_j: Number of items in lot $j, \forall j = 1, \ldots, Q$.
- $N^j_{\max}$: Maximum number of sublots allowed for lot $j, \forall j = 1, \ldots, Q$.
- n_j : Number of sublots for lot $j, \forall j = 1, \ldots, Q$.
- m : Number of parallel machines at Stage 2.
- f_j : Processing time per item of lot j on the machine at Stage 1, $\forall j = 1, \ldots, Q$.
- $p_{k,j}$: Processing time per item of lot j on parallel machine k at Stage 1, $\forall j = 1, \ldots, Q$, $\forall k = 1, \ldots, m$.
- t_j : Sublot-attached setup times for sublots of lot j on the machine at Stage 1.
- M: A large positive number, $\forall j = 1, \ldots, Q$.

Variables:

- s_{ij} : Size of sublot i of lot $j, \forall i = 1, \ldots, N^j_{\max}$, $\forall j = 1, \ldots, Q$.
- R_{ij} : Completion time of sublot i of lot j at Stage 1, $\forall i = 1, \ldots, N^j_{\max}$, $\forall j = 1, \ldots, Q$.
- C_{ijk} : Completion time of sublot i of lot j on parallel machine k at Stage 2, $\forall i = 1, \ldots, N^j_{\max}$, $\forall j = 1, \ldots, Q$, $\forall k = 1, \ldots, m$.
- V_j : Completion time of lot $j, \forall j = 1, \ldots, Q$.
- $C_{\max}$: Makespan value.
- $X_{ijk} = \begin{cases} 1 & \text{if sublot } i \text{ of lot } j \text{ is assigned to parallel machine } k \\ 0 & \text{otherwise, } \forall i = 1, \ldots, N^j_{\max},\ \forall j = 1, \ldots, Q,\ k = 1, \ldots, m. \end{cases}$

- $Y_{ij} = \begin{cases} 1 & \text{if size of sublot } i \text{ of lot } j \text{ is greater than } 0 \\ 0 & \text{otherwise, } \forall i = 1, \ldots, N^j_{\max}, \ \forall j = 1, \ldots, Q. \end{cases}$

- $Z_{ij} = \begin{cases} 1 & \text{if lot } i \text{ is processed before lot } j \\ 0 & \text{otherwise, } \forall i = 1, \ldots, N^j_{\max}, \ \forall j = 1, \ldots, Q. \end{cases}$

5.2.4 Assumptions

We assume that all the items of a lot are identical and require identical processing time. The lots are processed sequentially at both stages, i.e., if a lot starts processing, then the remaining lots can only start after this lot has finished processing. All the other assumptions are the same as those presented in Chapter 4 for the single-lot problem.

5.3 A Mixed Integer Programming Formulation

5.3.1 Model TSMLLSP-M

Our mathematical model for the TSMLLSP-M is as follows:

Minimize

$$C_{\max}$$

Subject to

$$C_{\max} \geq C_{ijk} - (2 - Y_{ij} - X_{ijk})M, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q$$
$$k = 1, \ldots, m \tag{5.1}$$

$$R_{0l} \geq R_{N_{\max}^j, j} + (Z_{jl} - 1)M, \quad \forall j = 1, \ldots, Q,\ l = 1, \ldots, Q, j \neq l \tag{5.2}$$

$$R_{ij} \geq R_{i-1,j} + t_j Y_{ij} + f_j s_{ij}, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q \tag{5.3}$$

$$C_{0lk} \geq C_{N_{\max}^j, j, k} + (Z_{jl} - 1)M, \quad \forall j = 1, \ldots, Q,\ l = 1, \ldots, Q, j \neq l$$
$$k = 1, \ldots, m \tag{5.4}$$

$$C_{ijk} \geq R_{ij} + p_{jk} s_{ij} + (X_{ijk} - 1)M, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q$$
$$k = 1, \ldots, m, \tag{5.5}$$

$$C_{ijk} \geq C_{i-1,j,k} + p_{jk} s_{ij} + (X_{ijk} - 1)M, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q$$
$$k = 1, \ldots, m, \tag{5.6}$$

$$s_{ij} \leq U_j Y_{ij}, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q, \tag{5.7}$$

$$\sum_{k=1}^{m} X_{ijk} = 1, \quad \forall i = 1, \ldots, N_{\max}^j,\ j = 1, \ldots, Q, \tag{5.8}$$

$$\sum_{i=1}^{N_{\max}^j} s_i = U_j, \quad \forall j = 1, \ldots, Q, \tag{5.9}$$

$$Z_{jl} + Z_{lj} = 1 \quad \forall j = 1, \ldots, Q,\ l = 1, \ldots, Q, j \neq l \tag{5.10}$$

$$Z_{jl} + Z_{lq} + Z_{qj} \leq 2 \quad \forall j = 1, \ldots, Q,\ l = 1, \ldots, Q,$$
$$q = 1, \ldots, Q, j \neq l, l \neq q, q \neq j. \tag{5.11}$$

$$
\begin{aligned}
&s_{i,j} \geq 0, && \forall i = 1, \ldots, N^j_{\max},\ j = 1, \ldots, Q, && (5.12)\\
&R_{i,j} \geq 0, && \forall i = 0, \ldots, N^j_{\max},\ j = 1, \ldots, Q, && (5.13)\\
&C_{i,j,k} \geq 0, && \forall i = 0, \ldots, N^j_{\max},\ j = 1, \ldots, Q, && (5.14)\\
& && k = 1, \ldots, m, && (5.15)\\
&C_{\max} \geq 0, && && (5.16)\\
&X_{i,j,k} \in \{0,1\}, && \forall i = 0, \ldots, N^j_{\max},\ j = 1, \ldots, Q, && (5.17)\\
& && k = 1, \ldots, m, && (5.18)\\
&Y_{i,j} \in \{0,1\}, && \forall i = 1, \ldots, N^j_{\max},\ j = 1, \ldots, Q, && (5.19)\\
&Z_{j,l} \in \{0,1\}. && \forall j = 1, \ldots, Q,\ l = 1, \ldots, Q, j \neq l. && (5.20)
\end{aligned}
$$

Constraints (5.1) guarantee $C_{\max}$ to be greater than or equal to the completion times of all the sublots of the lots on the parallel machines at Stage 2. Constraints (5.2) and (5.3) ensure relationships among the completion times of the sublots on the machine at Stage 1. Note that a dummy sublot (0) is used to capture the completion time of the previous lot assigned to the same machine. Constraints (5.4), (5.5), and (5.6) ensure the starting time of a sublot of a lot on a parallel machine to be greater than or equal to the completion time of the previous sublot on the same machine ((5.6)), and greater than or equal to the completion time of the same sublot on the machine at Stage 1 ((5.5)). Again, a dummy sublot is used to capture the completion time of the previous lot assigned to the same machine ((5.4)). Constraints (5.7) guarantee that the size of a sublot of each lot is not greater than the size of that lot. Constraints (5.8) ensure that each sublot is assigned to only one machine at the second stage. Constraints (5.9) are based on the fact that the summation of the sizes of all the sublots of a lot is equal to the size of that lot. Constraints (5.10) assure that either job j is processed before job l or job l is processed before job j. Constraint (5.11) are the subtour elimination constraints to avoid the possibility that job j if precedes job l, and job l precedes job m, then job m precedes job j (see Sarin et al. (2005) [76]). The remaining constraints (5.12) - (5.20) specify the domains of all the variables.

Note that a large positive number M appears in constraints (5.1), (5.2), (5.4), (5.5) and (5.6). A reasonable value of M to use is as follows:

$$\hat{M} = \sum_{j=1}^{Q} \left(N^j_{\max} \cdot t_j + f_j U_j + \max_{k=1\ldots m} p_{j,k} \cdot U_j \right),$$

which is the total setup and processing times on the machine at Stage 1 plus the total processing time on the slowest machine at Stage 2.

5.3.2 Model TSMLLSP-C

For the sum of completion times, we have the following objective function:

$$\text{Minimize } \sum_{j=1}^{Q} V_j, \tag{5.21}$$

where,

$$V_j \geq C_{i,j,k} \qquad \forall i = 0, \ldots, N^j_{\max},\ j = 1, \ldots, Q,\ k = 1, \ldots, m. \tag{5.22}$$

We, thus, have the following model:

Model TSMLLSP-C: (5.21) s.t. (5.2) - (5.20), (5.22).

In line with our development in Chapter 4, we consider Stage 2 to consist of only two parallel machines, i.e., $p_{j1} = p_{j2} = p_j$.

5.4 Heuristic Solution Method and Lower Bounds

Our heuristic methods are based on fixing values of appropriate integer variables, and then, solving the model formulation presented in Section 5.3.

Assignment of sublots to alternate machine

The assignment of the sublots of a lot to alternate machine in rotation gives an optimal solution (see Chapter 4). In this method, we use the same allocation scheme for the sublots of each lot even though it need not guarantee optimality.

Reduce number of sublots of each lot by fixing n_j

As presented in Chapter 4, TSHFLSP-FixN is easier to solve than the TSHFLSP-FlexN. Therefore, in this method, we use TF-C presented in Chapter 4 to fix $n_j, \forall j = 1, \ldots, Q$, to appropriate values.

Dispatching rules to fix the sequence of lots

We use two dispatching rules: Shortest processing times (SPT) and longest processing times (LPT), to fix the sequence of lots. The processing time for a lot is the total completion times of the lot at both the stages plus the total setup time at Stage 1 assuming the maximum number of sublots is used. For example, the total completion time for lot $j = U_j * (p_j + f_j) + N^j_{\max} * t_j$.

- SPT: Sequence the lots by the shortest processing times first rule;
- LPT: Sequence the lots by the longest processing times first rule.

5.4.1 TSMLLSP-M Problem

We propose four heuristic methods based on the combination of the above rules for the solution of TSMLLSP-M as follows:

Heuristic Method 1: MS 1

- Fix the assignment of the sublots of a lot using “Alternative Assignment” method;
- Fix the sequence of the lots using “LPT”;
- Flexible number of sublots for each lot.

Heuristic Method 2: MS 2

- Fix the assignment of the sublots of a lot using “Alternative Assignment” method;
- Fix the sequence of the lots using “SPT”;
- Fix the number of sublots of each lot by applying TF-C method presented in Chapter 4.

Heuristic Method 3: MS 3

- Fix the assignment of the sublots of a lot using “Alternative Assignment” method;
- Fix the sequence of the lot using “LPT”;
- Fix the number of sublots of each lot by applying TF-C method presented in Chapter 4.

Heuristic Method 4: MS 4

- Fix the assignment of the sublots of a lot using "Alternative Assignment" method;
- Fix the sequence of the lots using "LPT";
- Fix the number of sublots of each lot as follows: let the number of sublots for the first $Q-2$ lots equals to $\lfloor N^j_{\max}/2 \rfloor$, determine those for the last two lots based on TF-C presented in Chapter 4.

5.4.2 TSMLLSP-C Problem

We propose three heuristic methods based on the combination of above rules for the solution of the TSMLLSP-C as follows:

Heuristic Method 1: TC 1

- Fix the assignment of the sublots of a lot using "Alternative Assignment" method;
- Fix the sequence of the lots using "SPT";
- Flexible number of sublots for each lot.

Heuristic Method 2: TC 2

- Fix the assignment of the sublots of a lot using "Alternative Assignment" method;
- Fix the sequence of the lots using "SPT";
- Fix the number of sublots of each lot by applying TF-C method presented in Chapter 4.

Heuristic Method 3: TC 3

- Fix the assignment of the sublots of a lot using "Alternative Assignment" method;
- Fix the sequence of the lots using "LPT";
- Fix the number of sublots of each lot by applying TF-C method presented in Chapter 4.

5.4.3 Lower Bounds

For the TSMLLSP-M, a lower bound can be defined as the smallest processing times on the machine at Stage 1 as follows:

$$LB_{\mathrm{M}} = \sum_{j=1}^{Q} t_j + f_j U_{j.}$$

For the TSMLLSP-C problem, if we only consider the total completion times at Stage 1, the SPT schedule will result in the optimal solution. Let the lots be ordered in the non-decreasing values of $r_j = t_j + f_j U_j$, and let $r_1^{'}$ be the smallest r_j $\forall j = 1, \ldots, Q$, $r_Q^{'}$ be the largest r_j, $\forall j = 1, \ldots, Q$, where $r_1^{'} \leq r_2^{'} \leq \cdots \leq r_Q^{'}$. Then, we have a lower bound as follows:

$$LB_{\mathrm{C}} = Qr_1^{'} + (Q-1)\, r_2^{'} + \cdots + 2r_{Q-1}^{'} + r_Q^{'}.$$

5.5 Computational Investigation

In this section, we present the results of our computational investigation of the CPU times required to obtain integer-sized sublots by the direct solution of Model TSMLLSP-M by CPLEX and the four heuristic methods presented in Section 5.4.1, and by direct solution of Model TSMLLSP-C by CPLEX and three heuristic methods presented in Section 5.4.2. We also compare the performances of the proposed heuristic methods and the lower bound. These methods were programmed using Visual C++ (Version 2008) and AMPL CPLEX Solver (version 11.2). The data used for experimentation (number of lots, number of items in lot j, sublot-attached setup time for lot j, processing times per item at Stage 1 and Stage 2, and maximum number of sublots for lot j) are presented in Table 5.1. We used three instances of the number of lots Q, which are 5, 10, and 15 lots. For each lot j, we considered two ranges of number of items, U_j, in the lot, namely, low and high. For each combination of Q, U_j, we used three levels of the maximum number of sublots, $N_{\max}^j$, namely, low, medium, and high. This amounted to a total of 3*3*2=18 instances. For each instance, we used 5 replications by randomly generating 5 values of number of items in each lot, sublot-attached setup times and processing times at Stage 1 and Stage 2 over the range specified for that instance. All numerical tests were executed on a Sony computer with Intel i7 Q740 CPU and 8GB DDR3 memory.

Table 5.1: Data used for computational investigation

Number of lots Q	5, 10, 15
Number of items in lot j (U_j)	Uniform distribution [500,1000], [1000,2000]
Sublot-attached setup time for lot j (t_j)	Uniform distribution [0.1,2.0]
Processing time per item (Stage 1) for lot j (f_j)	Uniform distribution [0.1,2.0]
Processing time per item (Stage 2) for lot j (p_j)	Uniform distribution [0.1,2.0]
Maximum number of sublots for lot j ($N_{\max}^j$)	4, 7, 10

5.5.1 TSMLLSP-M Problem

Table 5.2 presents the minimum, average, and maximum values of CPU times (in seconds) required for the direct solution of TSMLLSP-M by CPLEX and the proposed heuristic methods (MS 1, MS 2, MS 3, and MS 4). It depicts the best, average, and worst-case performances of these methods. If the CPU time required for a method is greater than 3600 seconds, we have marked it as 3600+ in the table. Also, the best value in each category is highlighted. Note that direct solution of TSMLLSP-M by CPLEX require more than 3600 seconds in all instances. MS 1 requires more CPU times than that required by MS 2, MS 3, and MS 4. For datasets 12 and 18, the maximum CPU times required by MS 1 are more than 3600 seconds, while MS 2, MS 3, and MS 4 can solve all instances within 0.3 seconds. On average, MS 4 requires less CPU time than MS 2 and MS 3.

Since TSMLLSP-M cannot be solved to optimality in a reasonable amount of time, we compare the heuristic solution values with lower bound. The results of minimum, average, and maximum ratios of the values obtained by the proposed heuristic methods (MS 1, MS 2, MS 3, and MS 4) and lower bound are presented in Table 5.3. The smallest ratio in each category is highlighted. The average ratios are shown graphically in Figure 5.2. MS 1 performs the best with a gap of less than 1% from lower bound on average. MS 4 (with average and maximum gaps from lower bound within 1.5%) performs better than MS 2 (with average gaps from lower bound within 2.5%, maximum gaps from lower bound within 3.5%) and MS 3 (with average optimality gaps from lower bound within 2%, maximum gaps from lower bound within 4%). MS 1 seems to generate the best solutions while MS 4 is a close second. However, MS 1 requires the largest CPU time among the four heuristics. Therefore, MS 4 is a great overall method.

Table 5.2: Comparison of CPU times (in seconds) required by the direct solution of TSMLLSP-M by CPLEX (MIP-M) and heuristic methods (MS 1, MS 2, MS 3, and MS 4)

Dataset	No of lots	U_j	$N^j_{\max}$	MIP-M	MS 1			MS 2			MS 3			MS 4		
					Min	Ave	Max	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
1	5	[500,1000]	4	3600+	0.23	0.27	0.30	**0.19**	**0.20**	0.22	0.20	0.21	0.22	0.20	0.22	0.23
2	5	[500,1000]	7	3600+	1.11	27.42	71.60	**0.20**	0.22	0.23	0.22	0.24	0.30	0.22	0.22	0.23
3	5	[500,1000]	10	3600+	65.05	1421.64	3600.00	0.22	0.24	0.28	0.22	0.25	0.28	0.22	**0.23**	**0.25**
4	5	[1000,2000]	4	3600+	0.25	0.28	0.33	0.20	0.21	0.22	0.20	0.21	0.22	0.20	0.21	0.22
5	5	[1000,2000]	7	3600+	0.31	33.77	156.44	0.22	0.23	0.25	0.22	0.23	0.25	**0.20**	**0.22**	0.28
6	5	[1000,2000]	10	3600+	0.31	53.71	215.55	0.22	0.24	0.27	0.22	0.24	0.25	0.22	**0.22**	**0.23**
7	10	[500,1000]	4	3600+	0.34	0.38	0.45	0.23	**0.24**	**0.25**	0.23	0.25	0.28	0.23	0.25	0.28
8	10	[500,1000]	7	3600+	4.99	32.78	77.19	0.22	0.24	0.27	0.22	0.23	0.23	0.22	**0.22**	0.23
9	10	[500,1000]	10	3600+	19.83	308.34	917.22	0.23	0.24	0.28	0.22	0.24	0.27	0.22	**0.23**	**0.25**
10	10	[1000,2000]	4	3600+	0.37	0.50	0.86	0.23	0.25	0.27	0.22	0.23	0.25	0.22	**0.22**	**0.23**
11	10	[1000,2000]	7	3600+	4.15	574.21	1830.66	0.23	0.25	0.27	0.23	0.25	0.28	**0.22**	**0.22**	**0.23**
12	10	[1000,2000]	10	3600+	2.96	875.67	3600.00	0.23	0.27	0.31	**0.20**	0.22	0.23	0.22	0.22	**0.22**
13	15	[500,1000]	4	3600+	0.47	0.55	0.67	0.22	**0.23**	**0.25**	0.23	0.26	0.30	0.22	0.25	0.27
14	15	[500,1000]	7	3600+	1.06	58.46	196.59	0.23	0.26	0.28	0.23	0.26	0.30	**0.22**	**0.24**	0.27
15	15	[500,1000]	10	3600+	1.28	38.87	104.10	0.27	0.28	0.31	0.23	0.26	0.28	**0.22**	**0.24**	**0.25**
16	15	[1000,2000]	4	3600+	0.50	0.56	0.61	0.22	0.24	0.27	0.22	0.24	0.25	0.22	**0.23**	0.25
17	15	[1000,2000]	7	3600+	15.46	51.90	113.04	0.23	0.24	0.25	0.22	0.25	0.28	0.22	**0.23**	**0.23**
18	15	[1000,2000]	10	3600+	5.15	734.21	3600.00	0.27	0.28	0.33	0.22	0.25	0.28	0.22	**0.23**	**0.23**

Table 5.3: Comparison of ratios of the solution values obtained by proposed heuristic methods (MS 1, MS 2, MS 3, and MS 4) and lower bound

Data	No of lots	U_j	$N^j_{\max}$	$\frac{\text{MS 1}}{\text{Lower Bound}}$			$\frac{\text{MS 2}}{\text{Lower Bound}}$			$\frac{\text{MS 3}}{\text{Lower Bound}}$			$\frac{\text{MS 4}}{\text{Lower Bound}}$		
				Min	Ave	Max	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
1	5	[500,1000]	4	**1.0009**	**1.0041**	**1.0097**	1.0045	1.0172	1.0307	1.0029	1.0072	1.0132	1.0020	1.0057	1.0109
2	5	[500,1000]	7	**1.0009**	**1.0089**	**1.0279**	1.0109	1.0199	1.0493	1.0060	1.0153	1.0357	1.0032	1.0115	1.0309
3	5	[500,1000]	10	**1.0012**	**1.0060**	**1.0106**	1.0070	1.0139	1.0192	1.0074	1.0155	1.0198	1.0046	1.0107	1.0150
4	5	[1000,2000]	4	**1.0018**	**1.0049**	**1.0106**	1.0073	1.0197	1.0312	1.0030	1.0064	1.0118	1.0024	1.0056	1.0111
5	5	[1000,2000]	7	**1.0003**	**1.0030**	**1.0093**	1.0042	1.0095	1.0272	1.0026	1.0057	1.0139	1.0017	1.0042	1.0112
6	5	[1000,2000]	10	**1.0004**	**1.0019**	**1.0062**	1.0042	1.0110	1.0192	1.0041	1.0070	1.0105	1.0026	1.0048	1.0088
7	10	[500,1000]	4	**1.0004**	**1.0084**	**1.0212**	1.0149	1.0245	1.0347	1.0026	1.0119	1.0238	1.0013	1.0096	1.0220
8	10	[500,1000]	7	**1.0002**	**1.0011**	**1.0018**	1.0061	1.0096	1.0142	1.0060	1.0089	1.0128	1.0033	1.0043	1.0060
9	10	[500,1000]	10	**1.0007**	**1.0018**	**1.0040**	1.0081	1.0102	1.0145	1.0081	1.0110	1.0167	1.0040	1.0060	1.0096
10	10	[1000,2000]	4	**1.0015**	**1.0047**	**1.0109**	1.0073	1.0160	1.0339	1.0035	1.0065	1.0126	1.0023	1.0053	1.0115
11	10	[1000,2000]	7	**1.0001**	**1.0007**	**1.0019**	1.0026	1.0049	1.0072	1.0021	1.0038	1.0057	1.0011	1.0020	1.0033
12	10	[1000,2000]	10	**1.0002**	**1.0008**	**1.0021**	1.0034	1.0059	1.0091	1.0035	1.0059	1.0093	1.0016	1.0032	1.0055
13	15	[500,1000]	4	**1.0006**	**1.0028**	**1.0056**	1.0048	1.0178	1.0392	1.0031	1.0063	1.0088	1.0016	1.0040	1.0066
14	15	[500,1000]	7	**1.0004**	**1.0020**	**1.0034**	1.0061	1.0073	1.0093	1.0068	1.0081	1.0097	1.0028	1.0042	1.0056
15	15	[500,1000]	10	**1.0002**	**1.0007**	**1.0019**	1.0083	1.0097	1.0106	1.0082	1.0096	1.0104	1.0038	1.0046	1.0053
16	15	[1000,2000]	4	**1.0004**	**1.0025**	**1.0063**	1.0099	1.0118	1.0166	1.0022	1.0043	1.0080	1.0010	1.0031	1.0069
17	15	[1000,2000]	7	**1.0003**	**1.0010**	**1.0031**	1.0040	1.0069	1.0158	1.0031	1.0045	1.0066	1.0014	1.0023	1.0044
18	15	[1000,2000]	10	**1.0000**	**1.0001**	**1.0002**	1.0033	1.0051	1.0090	1.0032	1.0049	1.0087	1.0016	1.0024	1.0044

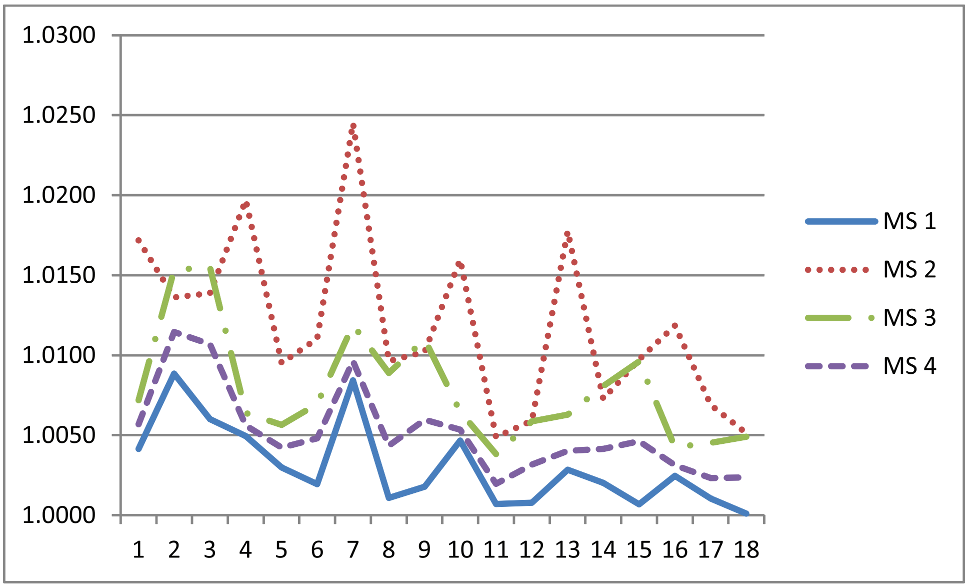

Figure 5.2: Comparison of ratios of the average solution values obtained by the proposed heuristic methods (MS 1, MS 2, MS 3, and MS 4) and lower bound

5.5.2 TSMLLSP-C Problem

As in the last subsection, Table 5.4 presents the minimum, average, and maximum values of CPU times (in seconds) required for the direct solution of TSMLLSP-C by CPLEX and proposed heuristic methods (TC 1, TC 2, and TC 3). It depicts the best, average, and worst-case performances of these methods. The best value in each category is highlighted. As before, if the CPU time required for the method is greater than 3600 seconds, we have marked it as 3600+. TC 1 requires more CPU times than those required by TC 2, and TC 3. Especially for datasets 3, 6, 8, 9, 11, 12, 14, 15, 17, and 18, TC 1 cannot solve some instances within 3600 seconds of CPU time. For dataset 18, the maximum CPU times required by TC 3 is 3600+, while TC 2 can solve all instances within 111 seconds. On average, TC 2 requires less CPU time than that required by TC 1 and TC 3.

Table 5.4: Comparison of CPU times (in seconds) required by the direct solution of TSMLLSP-C by CPLEX (MIP-C) and heuristic methods (TC 1, TC 2, and TC 3)

Dataset	No of lots	U_j	$N^j_{\max}$	MIP-C	TC 1			TC 2			TC 3		
					Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
1	5	[500,1000]	4	3600+	0.23	0.28	0.36	0.22	0.25	0.28	0.22	**0.22**	**0.23**
2	5	[500,1000]	7	3600+	0.53	7.74	34.55	0.25	**0.30**	**0.37**	0.25	0.32	0.39
3	5	[500,1000]	10	3600+	3600+	3600+	3600+	0.30	**0.46**	**0.89**	**0.27**	0.47	0.95
4	5	[1000,2000]	4	3600+	0.23	0.28	0.33	0.22	0.23	0.27	0.22	0.23	**0.25**
5	5	[1000,2000]	7	3600+	0.39	7.87	36.40	0.25	0.27	**0.31**	0.25	0.27	0.34
6	5	[1000,2000]	10	3600+	17.46	2098	3600+	0.28	0.35	0.47	0.28	**0.34**	0.47
7	10	[500,1000]	4	3600+	0.51	0.58	0.69	0.27	0.31	0.39	**0.25**	0.31	**0.37**
8	10	[500,1000]	7	3600+	644	3009	3600+	0.56	1.41	**4.63**	**0.55**	**1.40**	4.65
9	10	[500,1000]	10	3600+	3600+	3600+	3600+	**1.33**	**20.15**	89.97	1.84	20.68	**89.93**
10	10	[1000,2000]	4	3600+	0.47	0.56	0.69	0.28	**0.32**	0.39	0.28	0.34	0.39
11	10	[1000,2000]	7	3600+	5.27	2678	3600+	**0.44**	**0.58**	**0.66**	0.58	0.66	0.78
12	10	[1000,2000]	10	3600+	3600+	3600+	3600+	0.92	**2.73**	**4.23**	**0.83**	5.94	22.50
13	15	[500,1000]	4	3600+	0.73	0.96	1.14	**0.33**	0.43	0.50	0.34	**0.41**	**0.48**
14	15	[500,1000]	7	3600+	3600+	3600+	3600+	**0.84**	**1.05**	1.34	0.97	1.07	**1.29**
15	15	[500,1000]	10	3600+	3600+	3600+	3600+	**1.58**	**36.50**	**110.39**	2.00	2161	3600+
16	15	[1000,2000]	4	3600+	0.80	0.92	1.09	**0.31**	**0.46**	**0.64**	0.33	0.47	0.66
17	15	[1000,2000]	7	3600+	3600+	3600+	3600+	**0.59**	**1.04**	1.58	0.83	1.07	**1.51**
18	15	[1000,2000]	10	3600+	3600+	3600+	3600+	**1.40**	**5.62**	**16.60**	1.61	727.80	3600+

We also compare the solution values obtained by TC 1, TC 2, and TC 3 and lower bound. The results of minimum, average, and maximum ratios of the solution values of the proposed heuristic methods (TC 1, TC 2, and TC 3) and lower bound are presented in Table 5.5. The smaller ratio in each category is highlighted. The average ratios are shown graphically in Figure 5.3. Note that TC 1 performs better if the instances can be solved within 3600 seconds of CPU times. The gap from lower bound is within 3%. However, when TC 1 cannot solve an instance, TC 2 (with average gaps from lower bound within 3%, and maximum gaps from lower bound within 5%) performs much better than TC 3 (with average gaps from lower bound within 69%, and maximum gaps from lower bound within 223%).

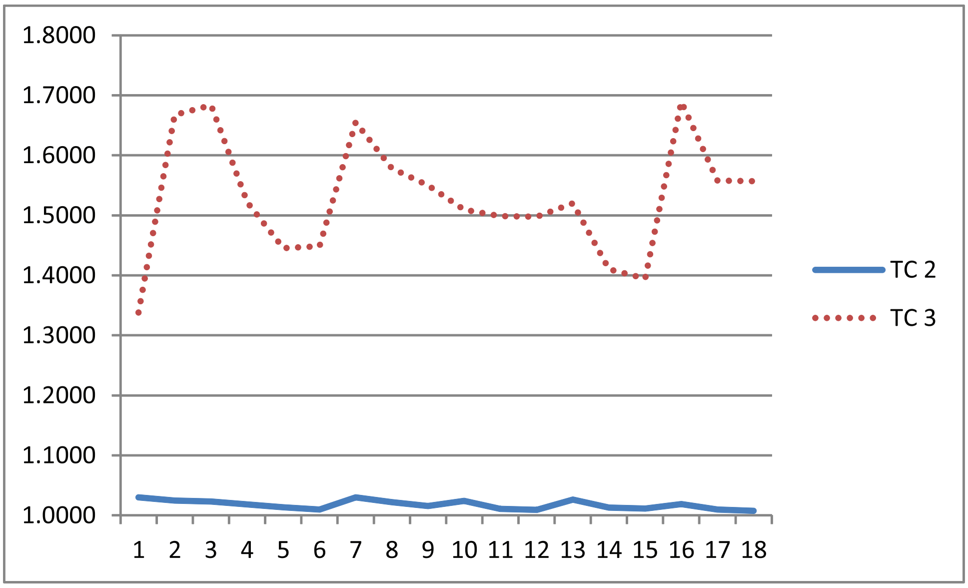

Figure 5.3: Comparison of ratios of the average solution values obtained by the proposed heuristic methods (TC 2 and TC 3) and lower bound

Table 5.5: Comparison of ratios of the solution values obtained by proposed heuristic methods (TC 1, TC 2, and TC 3) and lower bound

Dataset	No of lots	U_j	$N^j_{\max}$	$\frac{\text{TC 1}}{\text{Lower Bound}}$			$\frac{\text{TC 2}}{\text{Lower Bound}}$			$\frac{\text{TC 3}}{\text{Lower Bound}}$		
				Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
1	5	[500,1000]	4	**1.0115**	**1.0297**	**1.0469**	1.0118	1.0298	1.0469	1.1336	1.3378	1.6883
2	5	[500,1000]	7	**1.0131**	**1.0242**	**1.0336**	1.0135	1.0245	1.0337	1.1418	1.6676	2.2217
3	5	[500,1000]	10	N/A	N/A	N/A	**1.0094**	**1.0232**	**1.0358**	1.5921	1.6836	1.8231
4	5	[1000,2000]	4	1.0014	1.0181	1.0279	1.0014	1.0181	1.0279	1.3999	1.5215	1.6179
5	5	[1000,2000]	7	**1.0036**	**1.0135**	1.0252	1.0037	1.0136	1.0252	1.1133	1.4453	1.9241
6	5	[1000,2000]	10	**1.0061**	**1.0086**	**1.0101**	1.0063	1.0097	1.0108	1.2585	1.4484	1.6484
7	10	[500,1000]	4	1.0225	**1.0299**	1.0398	1.0225	1.0300	1.0398	1.3770	1.6550	2.1525
8	10	[500,1000]	7	1.0423	1.0423	**1.0423**	**1.0122**	**1.0220**	1.0433	1.2370	1.5784	1.9184
9	10	[500,1000]	10	N/A	N/A	N/A	**1.0105**	**1.0158**	**1.0243**	1.4174	1.5497	1.7108
10	10	[1000,2000]	4	1.0128	**1.0242**	1.0307	1.0128	1.0243	1.0307	1.2457	1.5093	1.8189
11	10	[1000,2000]	7	1.0115	1.0143	1.0172	**1.0052**	**1.0110**	1.0172	1.2440	1.4987	1.7984
12	10	[1000,2000]	10	N/A	N/A	N/A	**1.0048**	**1.0093**	**1.0166**	1.2628	1.4977	1.9238
13	15	[500,1000]	4	**1.0156**	**1.0260**	**1.0354**	1.0158	1.0262	1.0355	1.3430	1.5202	1.7928
14	15	[500,1000]	7	N/A	N/A	N/A	**1.0117**	**1.0131**	**1.0157**	1.0723	1.4103	1.6155
15	15	[500,1000]	10	N/A	N/A	N/A	**1.0068**	**1.0116**	**1.0164**	1.3610	1.3946	1.4282
16	15	[1000,2000]	4	**1.0118**	**1.0188**	1.0246	1.0119	1.0189	1.0246	1.3198	1.6894	1.8631
17	15	[1000,2000]	7	N/A	N/A	N/A	**1.0073**	**1.0097**	**1.0133**	1.4221	1.5581	1.8867
18	15	[1000,2000]	10	N/A	N/A	N/A	**1.0047**	**1.0074**	**1.0110**	1.4587	1.5568	1.6565

N/A: Solution not found within 3600 seconds of CPU time

5.6 Concluding Remarks

In this chapter, we have extended the single-lot lot streaming problem in a $1 + 2$ hybrid flow shop presented in Chapter 4, to the case of multiple lot, where each lot consists of a unique product type. We consider two objective functions, namely, minimize makespan and minimize the sum of completion time of the all lots. The problem is more difficult to solve because of the added requirement for the sequencing of the lots. Some basic results and properties derived for the TSHFLSP-FixN and TSHFLSP-FlexN problems presented in Chapter 4 are used for this problem as well. For solution methodology, we present four heuristic methods, using these properties, for the TSMLLSP-M problem and three heuristic methods for the TSMLLSP-C problem. Computational investigation of the proposed methods for the TSMLLSP-M problem reveals their efficacy in solving the problem provide near-optimal solution (with solution value within 1.5% of a lower bound obtained within a few seconds of CPU time). For the TSMLLSP-C problem, method TC 2 obtains solutions that are within 3% of a lower bound on average and within 5% in the worst case, within 111 seconds of CPU time.

Chapter 6

Summary, Conclusion and Future Research

In this dissertation, we have investigated the use of lot streaming in a two-stage assembly system and a two-stage hybrid flow shop to improve system performance. In particular, we have addressed three problems, namely, lot streaming in a two-stage assembly system, a single-lot lot streaming problem in a two-stage $1 + 2$ hybrid flow shop, and a multiple-lot lot streaming problem in a two-stage 1+2 hybrid flow shop.

First, in Chapter 2, we have presented an extensive review of literature on lot streaming problems in different machining environments for both performance-based and cost-based objectives. This includes problems in flow shop, open shop, job shop, parallel machines, hybrid flow shop, and an assembly system.

Then, in Chapter 3, we have considered the lot streaming problem in a two-stage assembly system. Given a set of lots, each consisting of a known number of products, and a maximum number of sublots, the problem is to split each lot into equal-sized sublots, sequence the lots for processing in a two-stage assembly system, in the presence of lot- and sublot-attached setup times on the machines at both the stages and sublot-attached time and cost incurred for the transfer of a sublot from Stage 1 to Stage 2, so as to optimize a performance measure. We consider both minimization of makespan and total cost. First, we formulate this problem as a mixed integer programming model, and enhance its solvability by adding additional constraints. Due to the difficulty of solving this model directly for large instances, we consider two column generation-based methods. Our computational investigation has shown that the proposed column generation methods generate solutions that are close to optimum. The optimality gap is less than 8.3% and 1.9% for the first and second methods, CG1 and CG2, respectively, for the makespan models, and is less than 6.3% and 1.8% for the first and second methods, CG1 and CG2, respectively, for the cost models. Therefore, the CG2 method performs better for both makespan and cost-based objective functions. In addition, the column generation procedures find solutions in a few seconds of CPU time while the direct solution of a mixed integer programming model of the problem by CPLEX requires a much larger CPU time.

In Chapter 4, we have addressed a single-lot lot streaming problem in a two-stage $1 + 2$ hybrid flow shop in the presence of sublot-attached removal time for each sublot at Stage 1. The objective is to minimize the makespan. We present a mixed integer programming model TSHFLSP-FlexN for the general problem of determining number of sublots, sublot sizes, and allocation of sublots for processing on the parallel machines at Stage 2. For a given number of sublots, optimal, continuous sublot sizes are obtained by using an "alternate" assignment method. We have developed closed-form expressions for obtaining optimal makespan and sublot sizes for the $1+2$ problem in the presence of sublot-attached removal times at Stage 1. When the number of sublots is also to be determined, we have presented an optimum seeking method, TF-C, for continuous sublot sizes. For integer sublot sizes, we have presented both an optimum seeking method, TF-I, and a heuristic method, TF-HI. Our numerical experimentation has revealed the efficacy of the proposed methodology to obtain optimal, continuous sublot sizes, and of the heuristic method to obtain near-optimal integer sublot sizes (within 0.15% optimality gap) while requiring a few seconds of CPU time.

In Chapter 5, we have extended the single-lot lot streaming problem in a $1 + 2$ hybrid flow shop, presented in Chapter 4, to the case of multiple lots, where each lot consists of a unique product type. We consider two objective functions, namely, minimize the makespan and minimize the sum of completion time of the all lots. This problem is more difficult to solve because of the added requirement of sequencing the lots. Some basic results and properties derived for the problems presented in Chapter 4 are used for this problem as well. For solution methodology, we present four heuristic methods, using these properties, for the makespan problem and three heuristic methods for the total completion time problem. Computational investigation of the proposed methods for the makespan problem reveals their efficacy in solving the problem. Near-optimal solutions (with solution value within 1.5% of a lower bound) are obtained within a few seconds of CPU time. For the total completion time problem, solutions that are within 3% of a lower bound on average and within 5% in the worst case, are obtained within 111 seconds of CPU time.

For future work, there are several directions that can be followed. For the assembly system problem, we may consider a more general environment consisting of more than two stages, and also permit a sublot to be assigned to any of the assembly locations. The column generation-based method can further be enhanced by exploiting some inherent structural properties. For the two-stage hybrid flow shop problems, there are two viable directions for future work: (i) development of an algorithm for the general $1 + m$ or $m + 1$ problem; and (ii) consideration of a more general hybrid flow shop, for example, multiple machines at both the stages.

Bibliography

[1] K. R. Baker. Lot streaming in the two-machine flow shop with setup times. *Annals of Operations Research*, 57(1):1–11, 1995.

[2] K. R. Baker and D. Jia. A comparative study of lot streaming procedures. *Omega*, 21(5):561–566, 1993.

[3] K. R. Baker and D. E. Pyke. Solution procedures for the lot-streaming problem. *Decision Sciences*, 21(3):475–491, 1990.

[4] D. E. Blumenfeld, L. D. Burns, and C. F. Daganzo. Synchronizing production and transportation schedules. *Transportation Research Part B: Methodological*, 25(1):23–37, 1991.

[5] R. W. Bogaschewsky, U. D. Buscher, and G. Lindner. Optimizing multi-stage production with constant lot size and varying number of unequal sized batches. *Omega*, 29(2):183–191, 2001.

[6] J. Bukchin and M. Masin. Multi-objective lot splitting for a single product m-machine flowshop line. *IIE Transactions*, 36(2):191–202, 2004.

[7] J. Bukchin, M. Tzur, and M. Jaffe. Lot splitting to minimize average flow-time in a two-machine flow-shop. *IIE Transactions*, 34(11):953–970, 2002.

[8] U. Buscher and L. Shen. An integrated tabu search algorithm for the lot streaming problem in job shops. *European Journal of Operational Research*, 199(2):385–399, 2009.

[9] S. Carpov, J. Carlier, D. Nace, and R. Sirdey. Two-stage hybrid flow shop with precedence constraints and parallel machines at second stage. *Computers and Operations Research*, 39(3):736–745, 2012.

[10] F. C. Cetinkaya. Lot streaming in a two-stage flow shop with set-up, processing and removal times separated. *Journal of the Operational Research Society*, 45(12):1445–1455, 1994.

[11] F. C. Cetinkaya. Unit sized transfer batch scheduling in an automated two-machine flow-line cell with one transport agent. *The International Journal of Advanced Manufacturing Technology*, 29(1):178–183, 2006.

[12] F. C. Cetinkaya and M. S. Kayaligil. Unit sized transfer batch scheduling with setup times. *Computers and Industrial Engineering*, 22(2):177–183, 1992.

[13] F. T. Chan, T. C. Wong, and L. Y. Chan. Lot streaming for product assembly in job shop environment. *Robotics and Computer-Integrated Manufacturing*, 24(3):321–331, 2008.

[14] J. Chen and G. Steiner. Approximation methods for discrete lot streaming in flow shops. *Operations Research Letters*, 21(3):139–145, 1997.

[15] J. Chen and G. Steiner. Lot streaming with detached setups in three-machine flow shops. *European Journal of Operational Research*, 96(3):591–611, 1997.

[16] J. Chen and G. Steiner. On discrete lot streaming in no-wait flow shops. *IIE Transactions*, 35(2):91–101, 2003.

[17] A. C. Chiang. *Fundamental Methods of Mathematical Economics, third edition*. McGraw-Hill, New York, NY, USA, 1984.

[18] C. Chiang. Order splitting under periodic review inventory systems. *International Journal of Production Economics*, 70(1):67–76, 2001.

[19] H. N. Chiu and J. H. Chang. Cost models for lot streaming in a multistage flow shop. *Omega*, 33(5):435–450, 2005.

[20] H. N. Chiu, J. H. Chang, and C. H. Lee. Lot streaming models with a limited number of capacitated transporters in multistage batch production systems. *Computers and Operations Research*, 31(12):2003–2020, 2004.

[21] S. Dauzere-Peres and J. Lasserre. An iterative procedure for lot streaming in job-shop scheduling. *Computers and Industrial Engineering*, 25(1-4):231–234, 1993.

[22] S. Dauzere-Peres and J. Lasserre. Lot streaming in job-shop scheduling. *Operations Research*, 45(4):584–595, 1997.

[23] Z. Drezner, A. Z. Szendrovits, and G. O. Wesolowsky. Multi-stage production with variable lot sizes and transportation of partial lots. *European Journal of Operational Research*, 17(2):227–237, 1984.

[24] J. Z. Du and J. Y. Leung. Minimizing mean flow time in two-machine open shops and flow shops. *Journal of Algorithms*, 14(1):24–44, 1993.

[25] K. Ertogral, M. Darwish, and M. Ben-Daya. Production and shipment lot sizing in a vendorcbuyer supply chain with transportation cost. *European Journal of Operational Research*, 176(3):1592–1606, 2007.

[26] A. Eynan and C. L. Li. Lot-splitting decisions and learning effects. *IIE Transactions*, 29(2):139–46, 1997.

[27] E. J. Fern. Six steps to the future: How mass customization is changing our world. *http://www.asapm.org/asapmag/articles/SixSteps.pdf*.

[28] C. A. Glass, J. N. Gupta, and C. N. Potts. Lot streaming in three-stage production processes. *European Journal of Operational Research*, 75(2):378–394, 1994.

[29] C. A. Glass and C. N. Potts. Structural properties of lot streaming in a flow shop. *Mathematics of Operations Research*, 23(3):624–639, 1998.

[30] S. K. Goyal. Note on "manufacturing cycle time determination for a multi-stage economic production quantity model". *Management Science*, 23(3):332–333, 1976.

[31] S. K. Goyal. Determination of optimum production quantity for a two-stage production system. *Operational Research Quarterly*, 28(4):865–870, 1977.

[32] S. K. Goyal. Economic batch quantity in a multi-stage production system. *International Journal of Production Research*, 16(4):267, 1978.

[33] S. K. Goyal. A joint economic-lot-size model for purchaser and vendor. *Decision Science*, 19(1):236–241, 1988.

[34] S. K. Goyal and A. Z. Szendrovits. A constant lot size model with equal and unequal sized batch shipments between production stages. *Engineering Costs and Production Economics*, 10(1):203–210, 1986.

[35] S. C. Graves and M. M. Kostreva. Overlapping operations in material requirements planning. *Journal of Operations Management*, 6(3-4):283–294, 1986.

[36] J. N. Gupta. Two-stage, hybrid flowshop scheduling problem. *The Journal of the Operational Research Society*, 39(4):359–364, 1988.

[37] J. N. Gupta and E. A. Tunc. Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. *International Journal of Production Research*, 29(7):1489, 1991.

[38] J. N. Gupta and E. A. Tunc. Scheduling a two-stage hybrid flowshop with separable setup and removal times. *European Journal of Operational Research*, 77(3):415–428, 1994.

[39] R. M. Hill. Batching policies for linearly increasing demand with a finite input rate. *International Journal of Production Economics*, 43(2-3):149–154, 1996.

[40] J. A. Hoogeveen, J. K. Lenstra, and B. Veltman. Preemptive scheduling in a two-stage multiprocessor flow shop is NP-hard. *European Journal of Operational Research*, 89(1):172–175, 1996.

[41] C. K. Huang. An optimal policy for a single-vendor single-buyer integrated productioncinventory problem with process unreliability consideration. *International Journal of Production Economics*, 91(1):91–98, 2004.

[42] F. R. Jacobs and D. J. Bragg. Repetitive lots: Flow-time reductions through sequencing and dynamic batch sizing. *Decision Sciences*, 19(2):281–294, 1988.

[43] H. Jeong, J. Park, and R. C. Leachman. A batch splitting method for a job shop scheduling problem in an mrp environment. *International Journal of Production Research*, 37(15):3583–3598, 1999.

[44] B. Jin, P. B. Luh, and L. S. Thakur. An effective optimization-based algorithm for job shop scheduling with fixed-size transfer lots. *Journal of Manufacturing Systems*, 18(4):284–300, 1999.

[45] S. M. Johnson. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1(1):61–68, 1954.

[46] A. A. Kalir and S. C. Sarin. Evaluation of the potential benefits of lot streaming in flow-shop systems. *International Journal of Production Economics*, 66(2):131–142, 2000.

[47] A. A. Kalir and S. C. Sarin. Constructing near optimal schedules for the flow-shop lot streaming problem with sublot-attached setups. *Journal of Combinatorial Optimization*, 7(1):23–44, 2001.

[48] A. A. Kalir and S. C. Sarin. A near-optimal heuristic for the sequencing problem in multiple-batch flow-shops with small equal sublots. *Omega*, 29(6):577–584, 2001.

[49] A. A. Kalir and S. C. Sarin. Optimal solutions for the single batch, flow shop, lot-streaming problem with equal sublots. *Decision Sciences*, 32(2):387–398, 2001.

[50] V. R. Kannan and S. B. Lyman. Impact of family-based scheduling on transfer batches in a job shop manufacturing cell. *International Journal of Production Research*, 32(12):2777–2794, 1994.

[51] J. S. Kim, S. H. Kang, and S. M. Lee. Transfer batch scheduling for a two-stage flowshop with identical parallel machines at each stage. *Omega*, 25(5):547–555, 1997.

[52] K. Kim and I. Jeong. Flow shop scheduling with no-wait flexible lot streaming using an adaptive genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, 44(11):1181–1190, 2009.

[53] S. L. Kim and D. Ha. A jit lot-splitting model for supply chain management: Enhancing buyercsupplier linkage. *International Journal of Production Economics*, 86(1):1–10, 2003.

[54] Y. D. Kim, S. O. Shim, S. B. Kim, Y. C. Choi, and H. M. Yoon. Parallel machine scheduling considering a job-splitting property. *International Journal of Production Research*, 42(21):4531–4546, 2004.

[55] D. H. Kropp and T. L. Smunt. Optimal and heuristic models for lot splitting in a flow shop. *Decision Sciences*, 21(4):691–709, 1990.

[56] S. Kumar, T. P. Bagchi, and C. Sriskandarajah. Lot streaming and scheduling heuristics for m-machine no-wait flowshops. *Computers and Industrial Engineering*, 38(1):149–172, 2000.

[57] A. Kusiak. Aggregate scheduling of a flexible machining and assembly system. *IEEE Transactions on Robotics and Automation*, 5(4):451–459, 1989.

[58] C. L. Li and W. Q. Xiao. Lot streaming with supplier manufacturer coordination. *Naval Research Logistics (NRL)*, 51(4):522–542, 2004.

[59] B. M. Lin and A. A. Jeng. Parallel-machine batch scheduling to minimize the maximum lateness and the number of tardy jobs. *International Journal of Production Economics*, 91(2):121–134, 2004.

[60] R. Linn and W. Zhang. Hybrid flow shop scheduling: A survey. *Computers and Industrial Engineering*, 37(1-2):57–61, 1999.

[61] J. Liu. Single-job lot streaming in m+1 two-stage hybrid flowshops. *European Journal of Operational Research*, 187(3):1171–1183, 2008.

[62] R. Logendran and F. Subur. Unrelated parallel machine scheduling with job splitting. *IIE Transactions*, 36(4):359–372, 2004.

[63] U. Manoj, J. Gupta, S. Gupta, and C. Sriskandarajah. Supply chain scheduling: Just-in-time environment. *Annals of Operations Research*, 161(1):53–86, 2008.

[64] C. H. Martin. A hybrid genetic algorithm/mathematical programming approach to the multi-family flowshop scheduling problem with lot streaming. *Omega*, 37(1):126–137, 2009.

[65] J. P. Moily. Optimal and heuristic procedures for component lot-splitting in multi-stage manufacturing systems. *Management Science*, 32(1):113–125, 1986.

[66] I. V. Nieuwenhuyse and N. Vandaele. Determining the optimal number of sublots in a single-product, deterministic flow shop with overlapping operations. *International Journal of Production Economics*, 92(3):221–239, 2004.

[67] I. V. Nieuwenhuyse and N. Vandaele. The impact of delivery lot splitting on delivery reliability in a two-stage supply chain. *International Journal of Production Economics*, 104(2):694–708, 2006.

[68] B. J. Pine II, B. Victor, and A. C. Boynton. Making mass customization work. *Harvard Business Review*, 71(5):108–118, 1993.

[69] C. N. Potts and K. R. Baker. Flow shop scheduling with lot streaming. *Operations Research Letters*, 8(6):297–303, 1989.

[70] R. V. Ramasesh, H. Fu, D. K. Fong, and J. C. Hayya. Lot streaming in multistage production systems. *International Journal of Production Economics*, 66(3):199–211, 2000.

[71] H. Rau and B. C. OuYang. An optimal batch size for integrated productioncinventory policy in a supply chain. *European Journal of Operational Research*, 185(2):619–634, 2008.

[72] H. Rau, M. Y. Wu, and H. M. Wee. Integrated inventory model for deteriorating items under a multi-echelon supply chain environment. *International Journal of Production Economics*, 86(2):155–168, 2003.

[73] R. Ruiz and J. A. Vazquez-Rodriguez. The hybrid flow shop scheduling problem. *European Journal of Operational Research*, 205(1):1–18, 2010.

[74] S. C. Sarin and P. Jaiprakash. *Flow Shop Lot Streaming Problems.* Springer, 2006.

[75] S. C. Sarin, A. A. Kalir, and M. Chen. A single-lot, unified cost-based flow shop lot-streaming problem. *International Journal of Production Economics*, 113(1):413–424, 2008.

[76] S. C. Sarin, H. D. Sherali, and A. Bhootra. New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. *Operations Research Letters*, 33(1):62–70, 2005.

[77] S. C. Sarin and L. Yao. Multiple-lot, lot streaming in a two-stage assembly system. *Research Report*, 2011.

[78] S. C. Sarin, L. Yao, and D. Trietsch. Single-batch lot streaming in a two-stage assembly system. *International Journal of Planning and Scheduling*, 1(1-2):90–108, 2011.

[79] A. Sen and O. S. Benli. Lot streaming in open shops. *Operations Research Letters*, 23(3-5):135–142, 1998.

[80] A. Sen, E. Topaloglu, and O. S. Benli. Optimal streaming of a single job in a two-stage flow shop. *European Journal of Operational Research*, 110(1):42–62, 1998.

[81] G. D. Silveira, D. Borenstein, and F. Fogliatto. Mass customization: Literature review and research directions. *International Journal of Production Economics*, 72(1):1–13, 2001.

[82] T. L. Smunt, A. H. Buss, and D. H. Kropp. Lot splitting in stochastic flow shop and job shop environments. *Decision Sciences*, 27(2):215–238, 1996.

[83] C. Sriskandarajah and S. P. Sethi. Scheduling algorithms for flexible flowshops: Worst and average case performance. *European Journal of Operational Research*, 43(2):143–160, 1989.

[84] C. Sriskandarajah and E. Wagneur. Lot streaming and scheduling multiple products in two-machine no-wait flowshops. *IIE Transactions*, 31(8):695–707, 1999.

[85] K. E. Stecke and Z. Kuying. Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode. *Manufacturing and Service Operations Management*, 9(2):206–224, 2007.

[86] G. Steiner and W. G. Truscott. Batch scheduling to minimize cycle time, flow time, and processing cost. *IIE Transactions*, 25(5):90–97, 1993.

[87] G. A. Suer, F. Pico, and A. Santiago. Identical machine scheduling to minimize the number of tardy jobs when lot-splitting is allowed. *Computers and Industrial Engineering*, 33(1-2):277–280, 1997.

[88] A. Z. Szendrovits. Manufacturing cycle time determination for a multi-stage economic production quantity model. *Management Science*, 22(3):298–308, 1975.

[89] A. Z. Szendrovits. On the optimality of sub-batch sizes for a multi-stage epq model-a rejoinder. *Management Science*, 23(3):334–338, 1976.

[90] A. Z. Szendrovits. An inventory model for interrupted multi-stage production. *International Journal of Production Research*, 25(1):129, 1987.

[91] A. Z. Szendrovits and Z. Drezner. Optimizing multi-stage production with constant lot size and varying numbers of batches. *Omega*, 8(6):623–629, 1980.

[92] A. Z. Szendrovits and M. Golden. The effect of two classes of process organizations on multi-stage production/inventory systems. *INFOR*, 22(1):40–55, 1984.

[93] D. Trietsch. *Optimal transfer lots for batch manufacturing: A base case and extensions.* Technical Report NPS-54-89-011, Naval Postgraduate School, Monterey, CA, 1987.

[94] D. Trietsch and K. R. Baker. Basic techniques for lot streaming. *Operations Research*, 41(6):1065–1076, 1993.

[95] W. G. Truscott. Scheduling production activities in multi-stage batch manufacturing systems. *International Journal of Production Research*, 23(2):315, 1985.

[96] W. G. Truscott. Production scheduling with capacity-constrained transportation activities. *Journal of Operations Management*, 6(3-4):333–348, 1986.

[97] C. T. Tseng and C. J. Liao. A discrete particle swarm optimization for lot-streaming flowshop scheduling problem. *European Journal of Operational Research*, 191(2):360–373, 2008.

[98] H. Tsubone, M. Ohba, and T. Uetake. The impact of lot sizing and sequencing on manufacturing performance in a two-stage hybrid flow shop. *International Journal of Production Research*, 34(11):3037–3053, 1996.

[99] R. G. Vickson. Optimal lot streaming for multiple products in a two-machine flow shop. *European Journal of Operational Research*, 85(3):556–575, 1995.

[100] R. G. Vickson and B. E. Alfredsson. Two- and three-machine flow shop scheduling problems with equal sized transfer batches. *International Journal of Production Research*, 30(7):1551–1574, 1992.

[101] B. J. Wagner and G. L. Ragatz. The impact of lot splitting on due date performance. *Journal of Operations Management*, 12(1):13–25, 1994.

[102] E. F. Williams, S. Tfekcli, and M. Akansel. O(m2) algorithms for the two and three sublot lot streaming problem. *Production and Operations Management*, 6(1):74–96, 1997.

[103] R. J. Wittrock. An adaptable scheduling algorithm for flexible flow lines. *Operations Research*, 36(3):445–453, 1988.

[104] T. C. Wong, F. T. S. Chan, and L. Y. Chan. A resource-constrained assembly job shop scheduling problem with lot streaming technique. *Computers and Industrial Engineering*, 57(3):983–995, 2009.

[105] F. Yalaoui and C. Chu. New exact method to solve the pm/rj/cj schedule problem. *International Journal of Production Economics*, 100(1):168–179, 2006.

[106] S. H. Yoon and J. A. Ventura. An application of genetic algorithms to lot-streaming flow shop scheduling. *IIE Transactions*, 34(9):779–787, 2002.

[107] S. H. Yoon and J. A. Ventura. Minimizing the mean weighted absolute deviation from due dates in lot-streaming flow shop scheduling. *Computers and Operations Research*, 29(10):1301–1315, 2002.

[108] T. Zhang, L. Liang, Y. Yu, and Y. Yu. An integrated vendor-manged inventory model for a two-echelon system with order cost reduction. *International Journal of Production Economics*, 109(1-2):241–253, 2007.

[109] W. Zhang, J. Liu, and R. J. Linn. Model and heuristics for lot streaming of one job in m-1 hybrid flowshops. *International Journal of Operations and Quantitative Management*, (9):49–64, 2003.

[110] W. Zhang, C. Yin, J. Liu, and R. J. Linn. Multi-job lot streaming to minimize the mean completion time in m-1 hybrid flowshops. *International Journal of Production Economics*, 96(2):189–200, 2005.