Lot Streaming in Two-Stage Flow Shops and Assembly Systems

Niloy J. Mukherjee

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
Barbara M.P. Fraticelli
Raghu Pasupathy

September 2, 2014
Blacksburg, Virginia

Keywords: Lot Streaming, Flow Shop, Assembly Shop

Lot Streaming in Two-Stage Flow Shops and Assembly Systems

Niloy J. Mukherjee

ABSTRACT

The research work presented in this dissertation relates to lot streaming in two-stage flow shops and assembly shops. Lot streaming refers to the process of splitting a production lot into sublots, and then, processing the sublots on different machines simultaneously in an overlapping manner. Such a strategy allows finished material at each stage to be transferred downstream sooner than if production and transfer batches were restricted to be the same size. In the case when each sublot consists of just one item, a single-piece-flow is obtained. Such a continuous flow is a key element of the Toyota Production System. However, single-piece-flow increases the number of transfers and the total transportation cost (time). As a result, it may not be economically justifiable in many cases, and therefore, material may have to be transferred in batches (called transfer batches, or sublots). Lot streaming addresses the problems of determining optimal sublot sizes for use in various machine environments and optimizes different performance measures. Given this relationship between lot streaming and the Toyota Production System, lot streaming can be considered a generalization of lean principles.

In this dissertation, we first provide a comprehensive review of the existing literature related to lot streaming. We show that two-stage flow shop problems have been studied more frequently than other machine environments. In particular, single-lot two-machine flow shops have been very well researched and efficient solution techniques have been discovered for a large variety of problems.

While two-stage flow shop lot streaming problems have been studied extensively, we find that the existing literature assumes that production rates at each stage remain constant.

Such an assumption is not valid when processing rates change, for example, due to learning. Learning here, refers to the improvements in processing rates achieved due to experience gained from processing units. We consider the case when the phenomenon of learning affects processing and setup times in a two-stage flow shop processing a single lot, and when, sublot-attached setup times exist. The decrease in unit-processing time, or sublot-attached setup time, is given by Wright's learning curve. We find closed-form expressions or simple search techniques to obtain optimal sublot sizes that minimize the makespan when the effect of learning reduces processing times, sublot-attached setup times, or, both. Then, we provide a general method to transform a large family of scheduling problems related to lot streaming in the presence of learning, to their equivalent counterparts that are not influenced by learning. This transformation is valid for all integrable learning functions (including the Wright's learning curve). As a result, a large variety of new problems involving learning can be solved using existing solution techniques.

We then consider lot streaming in stochastic environments in the context of sourcing material. Such problems have been well studied in the literature related to lot streaming for cost-based objective functions when demand is continuous, and when processing times are deterministic, or, for material sourcing problems when the time required to procure a lot is stochastic but is independent of the lot size. We extend this study to the case when the time required to produce a given quantity of products is stochastic and dependent on the number of units produced. We consider the case when two sublots are used, and also compare the performance of lot streaming to the case when each sublot is sourced from an independent supplier.

Next, we address a new problem related to lot streaming in a two-stage assembly shop, where we minimize a weighted sum of material handling costs and makespan. We consider the case when several suppliers provide material to a single manufacturer, who then assembles units from different suppliers into a single item. We assume deterministic, but not necessarily constant, lead times for each supplier, who may use lot streaming to provide material to the manufacturer. Lead times are defined as the length of the time interval between a supplier beginning to process material and the time when the first sublot is delivered to the manufacturer; Subsequent sublots must be transported early enough so that the manufacturer is not starved of material. The supplier may reduce this lead time by using lot streaming, but at an increased material handling cost. The decrease in lead time is also affected by other factors such as lot attached/detached setup times, transportation times etc. We allow these factors to be different for each supplier, and each lot processed by the same supplier. We refer to this problem as the Assembly Lot Streaming Problem (ALSP). We show that the ALSP can be solved using two steps. The first step consists of solution to several two-stage, single-lot, flow shop, makespan minimization problems. The solution to these problems generate prospective sublot sizes. Solution methods outlined in the existing literature can be used to complete this step. The second step obtains optimal number of sublots and production sequence. For a given production sequence, this step can be executed in polynomial-time; otherwise, the second step problem is NP-hard and integer programming formulations and decomposition-based methodologies are investigated for their solution. We make very lim-

ited assumptions regarding the handling cost and the relationship between the supplier lead time and number of sublots used. As a result, our solution methodology has a wide scope.

Dedication

This dissertation is dedicated to my parents Sumitra and Bhaskar Mukherjee. Without their support and encouragement this research would not have been possible.

Acknowledgments

I would like to thank my advisor, Dr. Subhash C. Sarin, who has guided me over the past five years with great scholarship and patience. He has always been supportive and the freedom that he provided has allowed me to study a large variety of problems. I have enjoyed the insightful discussions about research that he provided and his careful review of my work has greatly helped my development as a researcher. I am also grateful to my other committee members, Dr. Douglas R. Bish, Dr. Barbara M. Fraticelli and Dr. Raghu Pasupathy, who have been very supportive, and have provided many useful suggestions. I also want to thank the Grado Department of Industrial and Systems Engineering of Virginia Tech. Without their continuous financial support it would have been impossible to dedicate my energies to research. I am particularly grateful for the Alexander E.Walter Fellowship and for the Dean's Fellowship. I am also grateful for the teaching opportunities they provided. I would also like to thank the professors who have prepared me for my research with rigorous courses during my studies at Virginia Tech. I also appreciate the efforts of our staff in Industrial and Systems engineering who have always been prompt and helpful. Without their professionalism it would have been impossible to complete my academic documents and teaching responsibilities. I have made many new friends during my stay in Blacksburg. Their friendship has enriched my experience and allowed me to learn more about various different areas of research. I am grateful for their company and support. Particularly, I would like to thank Daniel Steeneck, Sanchit Singh and Anupam Gupta for helping me with academic forms and computational runs whenever I was away.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Now-a-days many products are manufactured in small lots in response to customer orders. In order to be competitive, manufacturers must respond to orders quickly, and yet, minimize the cost of production. One strategy to reduce the makespan of a production lot is to use lot streaming. Lot streaming is the strategy of splitting a production lot into smaller sublots, which are transferred downstream even as other sublots are being processed, so that, a lot can be processed in an overlapping manner. Such a strategy increases the velocity of material flow in a production system. However, an increase in the number of sublots, and hence the number of transfers, increases the material handling cost of the system. Hence, it is important to consider the dual objectives of improving time-based performance and material handling costs when implementing lot streaming.

Customized, made-to-order, production often requires the production of new products. Consequently, manufacturers may experience a learning phase when producing these products. As a result of such a learning process, the production time for one unit reduces with increased experience. Considering the effect of learning in scheduling models allows models to depict actual production times more accurately, and hence, prevents unnecessary idle time. When a manufacturer sources material from several different suppliers in order to assemble them into a single product, suppliers may use lot streaming to increase the velocity of material flow to reduce production lead time. The makespan of a production lot, however, is determined by the slowest supplier. An increase in the velocity of material flow from a supplier will result in a shorter makespan for a production lot only if that supplier is currently the slowest among all suppliers. In order to prevent unnecessary expenditure on increased material handling, suppliers must take lot streaming decisions in co-ordination with each other. Such co-ordination between suppliers is another important facet of an efficient production system.

Apart from reducing the production makespan, lot streaming also reduces the cycle time and average work-in-process inventory. It also reduces the amount of storage space, and, if sublot sizes are small enough, the capacity of material handling equipment required. Lot streaming involves splitting a production lot (batch) of jobs into smaller-sized sublots (transfer batches),

and then, processing them in an overlapping fashion over the machines. The literature related to lot streaming has been reviewed in the past by Chang and Chiu (2005) and Sarin and Jaiprakash (2006). More recently, lot streaming has been reviewed by Cheng *et al.* (2013).

Next, we present a review of lot streaming (based on Cheng *et al.* (2013)) and then present some problems that have not been addressed in the literature. Finally we propose new problems that may be studied. We review the literature on the use of lot streaming in various machine configurations. Since flow shops and assembly systems are particularly relevant to the problems proposed here, we study the literature related to these production environments in greater detail. We propose a general method to solve scheduling problems when the effect of learning is present. Since this method is applicable in various production environments, we present a review of lot streaming problems studied in environments such as parallel machines, hybrid flow shops, job shops and open shops as well.

1.1 Classification Scheme

Lot streaming has been studied in a variety of production environments. In order to present the myriad lot streaming problems addressed in the literature, we employ the classification scheme for scheduling problems presented by Sarin and Jaiprakash (2006). In it, the various features of a lot streaming problem are defined as follows:

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

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

Each of the nine fields of this classification scheme are defined 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 on a machine, 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 to process a lot on a machine can be started only after the arrival of the lot at 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) or Removal Time (R). Transfer time refers to the time required to move a lot or a sublot from one machine to another. During this time, the machine is available for processing the next lot or sublot. Removal time refers to the time required to remove a lot or sublot from a machine, but during this time, the machine is not available to process the next lot or sublot.
- Objective Function. The performance-based objective functions generally considered are: makespan ($C_{\max}$), mean flow time ($\bar{F}$), total flow time ($\sum F$), and total tardiness, among others, while the cost-based objective functions include cost of production, inventory, setup, and transfer/removal, among others.

We illustrate the occurrences of some of these features in a two-machine flow shop and an assembly flow shop. The production environment shown in Figure 1.1.1 is a two-machine flow shop with one machine at stage 1 and the second machine at stage 2. A single lot is split into three sublots for processing at stage 1, and the sublots, are then, processed on the second machine at stage 2. Lot-attached and sublot-attached setup times are required at stage 1. Transfer and removal times are also incurred for the sublots from stage 1 to stage 2. Sublot-detached setup times are required at stage 2. Intermittent idling is permitted between the processing of the sublots on the machine at stage 2. The problem is to determine an optimal number of sublots and their sizes. If more than one production lot is considered, the optimal sequence in which to process the lots must also be considered.

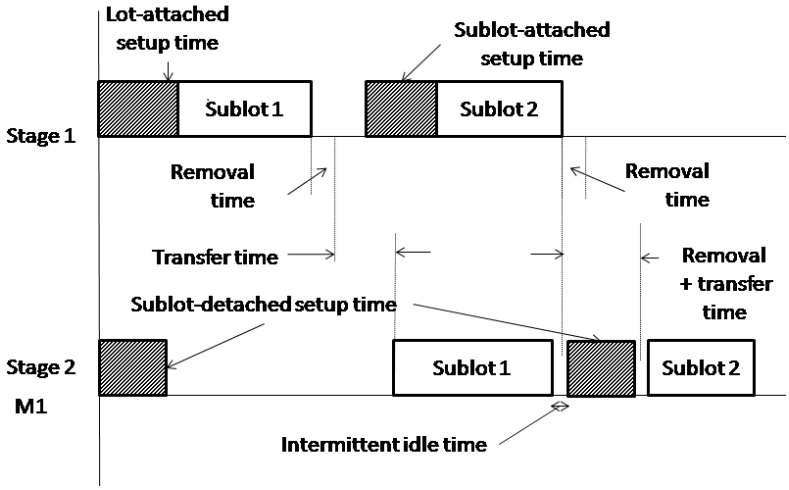


Figure 1.1.1: Lot streaming in a two-machine flow shop

An example of the assembly environment that we consider in this dissertation is shown in Figure 1.1.2, which consists of three parallel machines (suppliers) at the first stage and a single machine (assembly facility) at stage 2. The vendors supply components at stage 1 that are assembled at the second stage to obtain finished products. Note that a lot of size 50 is split into two sublots of sizes 10 and 40. The first sublot begins assembly (at stage 2) at time 70, only after the requisite components for sublot 1 have been manufactured by all the vendors at stage 1. Similarly, the second sublot begins assembly at time 110. Note that a lot-attached setup time is incurred on all the machines (at stage 1 and stage 2). The lot streaming-related features for this environment are: the number and sizes of sublots for each vendor. In case there are more than one lot processed, the sequence in which the various lots must be processed in order to optimize a performance measure must also be considered. Note that the sublot sizes are consistent across suppliers in the example presented here, i.e., the size of the first sublot is equal for each vendor and so is the size of the second sublot. This may not necessarily be optimal in all production environments. In such cases, the sublot sizes and number of sublots that each supplier uses must be optimized individually. Again lot/sublot-attached/detached setups may be involved during the processing of lots. In addition to all these features, lot streaming scheduling problems differ from their traditional counterparts because of the inherent flexibility of objective functions that they offer. For example, if the mean flow time is to be minimized, the completion time of a lot could be defined as the time when all the sublots belonging to that lot finish processing, or, as a function of the completion time of the individual sublots.

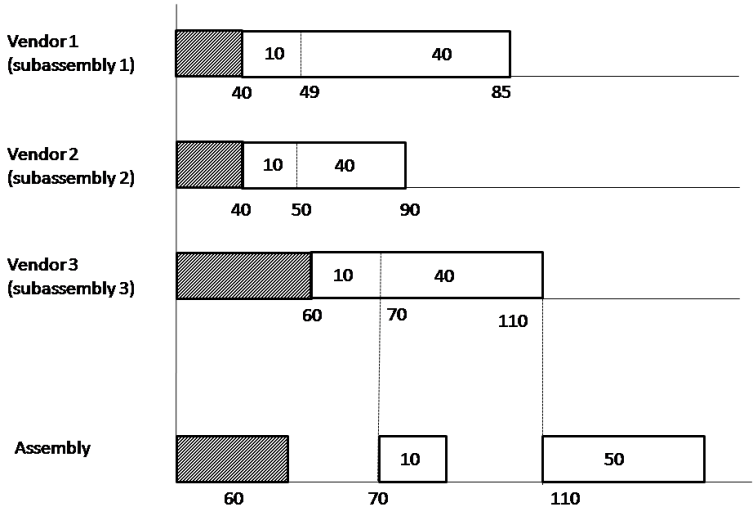


Figure 1.1.2: Two-stage assembly system

1.2 Lot Streaming for Time-based Objective Functions

A time-based performance measure is especially important for make-to-order environments, which involve objective functions like minimization of makespan, total flow time, cycle time, and meeting due dates. For this class of objective functions, lot streaming has been applied to machine configurations of flow shop, job shop, open shop, parallel machines, hybrid flow shop, and two-stage assembly systems. The lot streaming literature related to flow shops and two-stage assembly shops is particularly relevant to the the problem studied here. So, we review the literature related to these production structures in detail, while briefly mentioning the literature related to other production environments.

1.2.1 Two-stage flow shop and assembly systems

Two-stage assembly systems consist of a several parallel machines at the first stage which produce components that are assembled at a single assembly machine at the second stage. Such a production environment is closely related to two-stage flow shops since each supplier and assembly machine considered in isolation represent a two-stage flow shop. Lot streaming in flow shop environments has been well studied in the literature. We next present the related literature.

Two-stage flow shop

In a flow shop, each job follows the same order of processing over the machines. The majority of work done deals with minimizing the regular performance measures of makespan and mean flow time. The makespan of a schedule is the length of the interval over which the jobs are processed, including any idle time encountered between their processing. Kalir and Sarin (2000) have presented the extent of improvement of a time-based performance measure that may be obtained by using lot streaming. They show that when a single lot of U discrete jobs is processed in an m machine flow shop, dividing the lot into unit-sized sublots results in the greatest reduction in the makespan and flowtime. Such a lot streaming strategy reduces the makespan by a factor of $\frac{\sum_{i=1}^{m} p_i + (U-1)p_{max}}{U \sum_{i=1}^{m} p_i}$ and the mean flow time by $\frac{(U-1)p_{max} + 2\sum_{i=1}^{m} p_i}{2U \sum_{i=1}^{m} p_i}$, where p_i is the processing time per item (job) on machine i, U is the number of jobs in the lot, $p_{max} \equiv \max_i p_i$. Here, the mean flow time refers to the average completion time of the sublots (in the absence of lot streaming, it is simply the completion time of the lot).

Table 1.1: The two-machine, single-lot, flow shop lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
F2/1/E,C/II,NI/FixN/CV/-/-/$C_{\max}$	Potts and Baker (1989a)	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 (1993b)	Polynomial time algorithm and limited transporter capacity
F2/1/E/II/FixN/CV/-/-/F	Çetinkaya and Gupta (1994)	Optimal sublot sizes when first machine is the bottleneck
F2/1/E,C,V/II/FixN/CV/-/-/F	Sen *et al.* (1998)	Equal, consistent, and variable sublot sizes
F2/1/C/II/FixN/CV,DV/L(d)/-/$C_{\max}$	Sriskandarajah and Wagneur (1999)	Closed-form expression and heuristic method for integer sublot sizes
F2/1/C/II/FlexN/CV/S(a)/-/F	Bukchin *et al.* (2002b)	Frontier approach and tradeoff between alternative objective functions
F2/1/C/II/FixN/CV,DV/S(a)/-/C_{max}	Alfieri *et al.* (2012)	Discrete sublot sizes, sublot-attached setup, and optimal algorithm

Table 1.2: The assembly system, lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

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

The two-machine, single-lot, flow shop lot streaming problems addressed in the literature are depicted in Table 1.1. One of the earliest results with regard to time-based objective functions was developed for the case of a lot of identical jobs being processed over two machines and the objective of minimizing makespan when Trietsch (1987) and Potts and Baker (1989a) showed the optimality of geometric sublot sizes. In particular, they provided some basic properties for continuous optimal sublot sizes and makespan minimization problem when the number of sublots, denoted by n, is given. They showed that: (a) when the direction of material flow is reversed, the optimal sublot sizes $(s_1, \cdots, s_n)$ are the same as that for the original problem, but indexed in the reverse order, i.e., $(s_n, \cdots, s_1)$, (b) the no-idling restriction does not impact the optimal makespan value for both the continuous and discrete-sized sublots case, and (c) all the sublots are critical in an optimal solution. An immediate consequence of these results is that the optimal sublot sizes are geometric in nature with a ratio $\frac{p_2}{p_1}$, where p_i is the processing time per item on machine i, and that, no idle time exists between the processing of sublots on the second machine. To see this, since there is no inserted idle time on the second machine, we have

$$s_i = q s_{i-1}, \tag{1.1}$$

where $q = p_2/p_1$, and s_i is the size of the ith sublot. By (1.1), we obtain

$$s_i = q^{i-1} s_1. \tag{1.2}$$

If we let $U = \sum_{i=1}^{n} s_i$, we have

$$s_1 = \begin{cases} \frac{1-q}{1-(q)^n} U, & \text{if } q \neq 1 \\ \frac{U}{n}, & \text{otherwise.} \end{cases} \tag{1.3}$$

Combining (1.2) and (1.3), we have the following expression for the optimal sublot sizes:

$$s_i = \begin{cases} \frac{(q)^{i-1}-(q)^i}{1-(q)^n} U, & \forall i = 1, \ldots, n, \text{ if } q \neq 1, \\ \frac{U}{n}, \text{ otherwise.} \end{cases} \tag{1.4}$$

The makespan resulting from these sublot sizes is $p_1 s_1 + p_2 U$. Note that s_1 decreases with increase in n. So, if the number of sublots is not fixed a priori, then the makespan is minimized by using as many sublots as permitted by other constraints. Trietsch (1987) has considered the case when the continuous version of the problem serves as an approximation of a discrete lot streaming problem. The constraint $s_i \geq 1 \ , \forall i$, is added to the problem so that the resulting solution is close to the optimal solution for the discrete case. As a result of this constraint, the number of sublots is bounded by a closed-form expression that is $O(\log U)$ (see page 15 of Trietsch 1987), and hence, the optimal number of sublots, n, is polynomial in input length.

An illustration of geometric sublot sizes is given in Figure 1.2.1 for $q = p_2/p_1 = 2$.

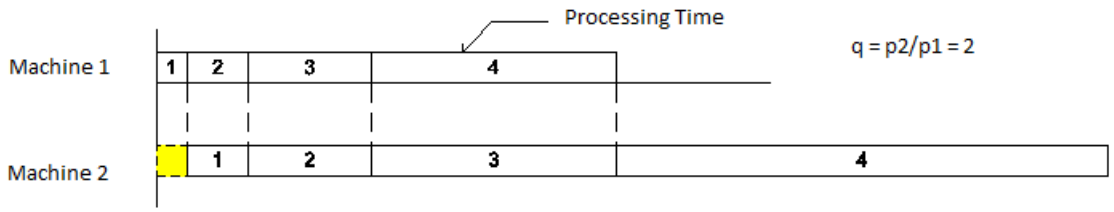


Figure 1.2.1: An example of optimal geometric sublot sizes

Trietsch and Baker (1993b) have considered the case of integer-sized sublots and present an iterative $O(n^2)$ time algorithm to obtain an optimal integer solution. They have also 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 for both the two-machine and three-machine cases. In addition, they develop polynomial-time algorithms to determine the optimal number of sublots and sublot sizes when vehicles used to transfer sublots require time to travel to the second machine, and then, return to the first machine, as well as the case when the capacity of the vehicle is limited. A modification of geometric sublots have also been shown to be optimal in the presence of sublot-attached setups (see research notes for Chapter 12 Baker and Trietsch 2009, and Alfieri *et al.* 2012), and for no-wait flow shops in the presence of lot-detached setups (Sriskandarajah and Wagneur, 1999). In each case, the optimality of geometric sublot sizes, or sublot sizes that are modifications of geometric sublot sizes, follows from the fact that all sublots are critical in an optimal schedule and there is no idle time between the processing of lots. While the optimal sublot sizes are nearly geometric, they are not necessarily equal to the sizes obtained by using (1.4). As an example, in the presence of sublot-attached setup times τ_1 and τ_2 at the first and second machine, respectively, we have a modified version of (1.1), given by $\tau_1 + p_1 s_i = \tau_2 + p_2 s_{i-1}$. Using $h = \frac{\tau_2 - \tau_1}{p_1}$, we have $s_i = h + q s_{i-1}$ $i = 2, \cdots, n$. Using $U = \sum_{i=1}^{n} s_i$ we get:

$$s_1 = \begin{cases} \left[\frac{U - \frac{nh}{1-q} + \frac{h}{1-q}\left(\frac{1-q^n}{1-q}\right)}{\left(\frac{1-q^n}{1-q}\right)} \right], & q \neq 1 \\ \left[\frac{U}{n} - (n-1)\frac{h}{2} \right], & q = 1. \end{cases} \tag{1.5}$$

Sen *et al.* (1998) have considered the two-machine lot streaming problem for the objective of minimizing weighted completion time when sublot sizes are equal, consistent, or variable. For a given schedule, the weighted flow time is defined as $\frac{1}{U}\left(\sum_{i=1}^{n} s_i C_i\right)$, where C_i is the completion time of the i^{th} sublot on the second machine. They extend the result of Çetinkaya and Gupta (1994) that equal sublot sizes are optimal for the case when $p_2/p_1 \leq 1$ to the case when $p_2/p_1 > 1$ and show that, for this case, geometric sublots must be used until some sublot v, and equal sublots are used thereafter. The optimal value of v is obtained by starting with $v = 1$, and checking if the condition $q s_v \geq s_{v+1} \geq s_v$ holds. We stop if this condition is true;

otherwise, v is incremented by 1 and the process is repeated.

This problem in the presence of sublot-attached setups and batch availability to minimize average flow time has been investigated by Bukchin *et al.* (2002b). Their solution technique relies on the determination of optimal solution for the bottleneck machine. Their investigation also reveals that the optimal solution for the objective of minimizing average flow time generally produces a good solution for the objective of minimizing the makespan.

The multiple-lot problem gives rise to another issue regarding the sequencing of the lots that must be addressed. In addition, the impact of multiple lots on the optimal sublot sizes for a single lot in seclusion needs to be investigated. In many cases, the optimal sublot sizes obtained in seclusion for a lot remain optimal for the multiple lot case as well. A common approach for the sequencing of lots has been the use of a modified version of Johnson's rule (Johnson, 1954) . Table 1.3 depicts the two-machine, multiple lot, flow shop lot streaming problems addressed in the literature. Vickson and Alfredsson (1992) have considered the multiple-lot problem for both two-machine and three-machine flow shops, when equal sublots are used. They have shown the existence of an optimal permutation schedule of lots for the two-machine case (for any regular measure of performance) and for the three-machine case (for makespan minimization). For unit-sized sublots, a modification of Johnson's rule is used to obtain an optimal sequence of lots in order to minimize makespan for the two-machine case. They have also presented an empirical study to show the effect of the number of transfer batches of equal sizes on makespan and total flow time. A modification of Johnson's rule has also been used for the case of unit-sized sublots and lot-attached setup on the first machine and a lot-detached setup on the second machine by Cetinkaya and Kayaligil (1992), and for an automated two-machine flow shop in the presence of only one transport agent and both sublot-detached and sublot-attached setups by Cetinkaya (2006). Cetinkaya (1994) consider the case of lot-detached setup and sublot-attached removal times on both the machines, and the objective of minimizing the makespan. An optimal polynomial-time algorithm is presented to determine an optimal sequence in which to process the lots and sublot sizes for a given number of sublots for each lot. The independence of sublot sizing and lot streaming continues to hold for this case as well. Vickson (1995) also adresses this problem, but in the presence of transfer times instead of removal times, by first determining optimal number of sublots and sublot sizes for individual lots, and then, using them in the lot sequencing problem. If we let τ_l^i, p_l^i, and s_j^i represent the lot-detached setup time, unit processing time, and the size of sublot j, respectively, for lot i on machine l, $l \in \{1, 2\}$, and let Δ^i, U^i and n^i represent the transfer time in between the two machines, lot size, and the the number of sublots, respectively, for lot i, then an optimal sequence for processing the lots is obtained by using a modified Johnson's rule, where the processing times p_1^i and p_2^i are replaced with $z_i = \max\left(0,\, \tau_1^i + \Delta^i - \tau_2^i + \max_{1 \le k \le n^i}\left(p_1^i \sum_{j=1}^{k} s_j^i - p_2^i \sum_{j=1}^{k-1} s_j^i\right)\right)$ and $z_i' = z_i - (\tau_1^i - \tau_2^i + p_1^i U^i - p_2^i U^i)$. They have considered the cases of continuous and discrete sublot sizes. Baker (1995) has presented a modified version of Johnson's rule 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. Sriskandarajah and Wagneur (1999) have also shown independence of the sublot sizing problem for no-wait flow shops. Kalir and Sarin (2001b) have developed a pseudo-polynomial-time algorithm for the sublot-attached setup, equal sublot-size, m-machine flow shop problem that can be used for two-machine flow shops as well. Due to the presence of sublot-attached setup times, unit-sized sublots are not neccessarily optimal for this problem. So, Johnson's rule is used iteratively. At each iteration, sublot sizes are determined and Johnson's rule is reapplied.

Table 1.3: The two-machine, multiple-lot, flow shop lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
F2/n/E/II/FlexN/CV/-/-/$C_{\max}$, $\sum F$	Vickson and Alfredsson (1992)	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)	Optimal solution procedure similar to Johnson's rule for unit sized transfer batch
F2/n/C/II/FixN/CV,DV/L(d)/R/$C_{\max}$	Cetinkaya (1994)	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)	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)	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)	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)	Optimal algorithm based on Johnson's rule (pseudopolynomial-time method). The two-machine case is considered as a special case of the general m-machine problem.
F2/n/C/II/FixN/DV/S(a),S(d)/T/$C_{\max}$	Cetinkaya (2006)	Dependent and independent sublot-attached setup and optimal algorithm

Two-stage assembly system

The two-stage assembly system that we consider can be defined as follows: The first stage consists of a set of suppliers (vendors), each of whom produces one component. These components are then transferred in sublots to stage 2, where they are assembled into finished products. Stage 2 consists of one or more assembly machines. Each of these machines represents an assembly facility devoted to a product type. We have earlier depicted in Figure 1.1.2 a two-stage assembly system in which stage 2 consists of one machine. Note that an assembly system is distinct from a hybrid flow shop since it involves the convergence of material from different machines, and not just the processing of material in parallel. For the assembly system shown, all the subassemblies are processed simultaneously by the vendors at stage 1.

The two-stage assembly system, lot streaming problems addressed in the literature are shown in Table 1.2. The single-lot problem has been addressed by Sarin *et al.* (2011). 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 an upper bound on the number of sublots, and also for integer sublot sizes. They show the existence of an optimal schedule with consistent sublot sizes over all first-stage machines, and that, all the sublots must be critical. In addition, they present a polynomial-time algorithm to obtain integer sublot sizes. Sarin and Yao (2011) 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.

1.2.2 Lot streaming in other production environments

Next, we briefly discuss the literature related to production environments other than two-stage flow shops and assembly systems. In this dissertation we present some results related to the effect of learning in these environments.

Three-machine flow shops

The three-machine, flow shop lot streaming problems addressed in the literature are shown in Table 1.4. The study of this machine configuration has led to identification of cases for which the solution to three-machine lot streaming problems can be obtained from that for the two-machine problem. Also, the network representation introduced for the analysis of

the three-machine lot streaming problem has been used for other environments as well. In general, optimal sublot sizes need not be consistent in this case. Baker and Jia (1993) have addressed the cases with consistent and equal sublot sizes but with no setups and idling among the sublots, for the objective of minimizing the makespan. They show 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.* (1994a) have introduced a network representation for this problem, in which a digraph is constructed where each machine i and sublot j is represented by a vertex (i, j), and a directed arc connects a vertex (i, j) to vertices $(i + 1, j)$, for $i < m$, and $(i, j + 1)$ for $j < n$. An example of such a representation is shown in Figure 1.2.2. Minimizing the makespan for this case is equivalent to minimizing the longest path from the first to the last vertices, namely (1,1) and (3,5), respectively, for the network shown in Figure 1.2.2. Clearly, this representation assumes a fixed number of sublots. With this representation, it is easier to determine conditions under which a given path dominates others. They have presented an O(log n) algorithm to determine minimum makespan in the three-machine flow shop environment, and also, have developed some results that are applicable for the case when the number of machines exceeds three. Let the unit processing times of machines 1,2 and 3 be denoted by p_1, p_2 and p_3, respectively. Then, for the case when $p_2^2 - p_1 p_3 \leq 0$, the optimal sublot sizes are consistent and are given by (1.4), but with q now defined as $q = \frac{p_2+p_3}{p_1+p_2}$. Otherwise, all three machines run continuously and the problem can be decomposed into two two-machine problems, each solved independently. Therefore, for this case, the sublot sizes may not be consistent. Glass and Potts (1998) also used this network representation to develop an algorithm based on dominance machines.

In the presence of lot-detached setups,Chen and Steiner (1997b) have presented structural properties of the optimal solution using this network representation, for consistent and continuous sublot sizes. Similar results for discrete sublot sizes are presented by Chen and Steiner (1997a).

The m-machine, single-lot, flow shop lot streaming problems addressed in the literature are shown in Table 1.5. For equal sublot sizes, Truscott (1985, 1986) has addressed this problem for sublot-attached setup time, transfer time, and capacity constraints, and, the objective of minimizing the makespan. Steiner and Truscott (1993) have considered the presence of transfer times and the objective of minimizing cycle time, flow time, and processing cost for this problem. Williams *et al.* (1997) and Glass and Potts (1998) employ a network representation for the analysis of this problem assuming consistent sublot sizes and a given number of sublots, and for the objective of minimizing the makespan. Williams *et al.* (1997) present an O(m^2) algorithm in the presence of two or three sublots, and a heuristic method for the case of n-sublots.

A key idea in solving single-lot multiple-machine problems is that of identifying a bottleneck machine that determines the makespan of a schedule. Unless there are multiple bottleneck machines, any improvement in the performance of the bottleneck machine results in a reduc-

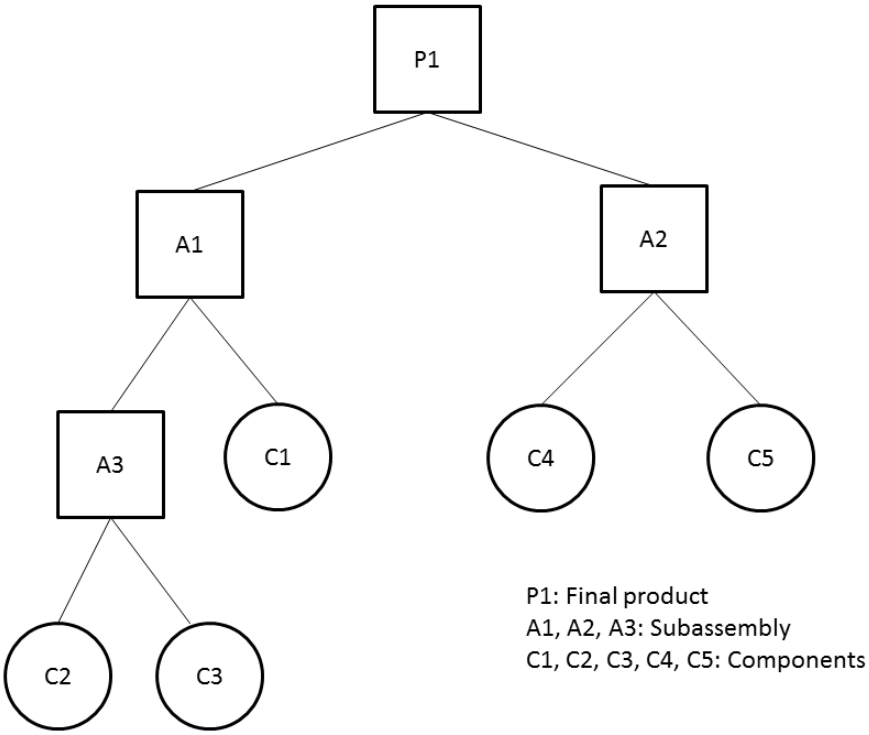


Figure 1.2.2: Network representation of a three-machine flow shop with one lot divided into five sublots.

tion of the minimum makespan value possible. For consistent sublot sizes, Baker and Pyke (1990) 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. They represent the makespan as an upper envelope of several functions, each corresponding to a machine. Kalir and Sarin (2001d) have presented a method to determine optimal number of sublots (n) in the presence of sublot transfer time and sublot-attached setup times for the case when there are multiple machines and a single lot. They show that the makespan is given by

$$\max_{1 \leq j \leq m} \left(M_j(n) \right),$$

where

$$M_j\left(n\right) = \left(n-1\right)\left(\frac{U}{n}\,p_j + \tau_j\right) + \sum_{k=1}^{m}\left(\frac{U}{n}\,p_k + \tau_k\right)$$

(here $M_j\left(n\right)$ serves as the upper envelope function, but, note that, the number of sublots is not restricted to two). They also show that $M\left(n\right)$ is a convex function of n, and either the optimal, continuous number of sublots,

$$n* = \sqrt{\frac{U\left(\sum_{k=1}^{m} p_k - p_j\right)}{\tau_j}} \text{ for some j,}$$

or, it lies at the intersection of two curves $M_k\left(n\right)$ and $M_l(n)$. These intersection points are given by

$$n_{kl} = \frac{U(p_k - p_l)}{\tau_k - \tau_l}.$$

In the above case, any $j \in \{1, \cdots, m\}$ that has the largest value of $M_j(n)$ for a given n is the bottleneck machine. Chen and Steiner (2003) Chen and Steiner (2003) have addressed a discrete version of this problem for a no-wait m-machine flow shop.

Table 1.4: The three-machine, flow shop lot streaming problems addressed in the literature(Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
F3/1/E,C,V/NI,II/FixN/CV,DV/-/-/$C_{\max}$	Glass *et al.* (1994a)	Network representation and O(log n) algorithm
F3/1/C/II/FixN/CV/-/-/$C_{\max}$	Glass and Potts (1998)	Network representation and optimal algorithm
F3/1/E,C/NI/FixN/CV/-/-/$C_{\max}$	Baker and Jia (1993)	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)	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 (1997b)	Network representation and optimal O(long) algorithm
F3/1/C/II/FixN/DV/L(a)/-/$C_{\max}$	Chen and Steiner (1997a)	Network representation and two approximation methods
F3/n/E/II/FlexN/CV/-/-/$C_{\max}$, $\sum F$	Vickson and Alfredsson (1992)	Branch and bound-based and local neighborhood search-based heuristic methods

Table 1.5: The m-machine, single-lot, flow shop lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
Fm/1/E/II/FixN/DV/S(a)/T/$C_{\max}$	Truscott (1985, 1986)	Mathematical programming models
Fm/1/C/II/FixN/DV/-/-/$C_{\max}$	Glass *et al.* (1994a)	Network representation and O(log n) algorithm
	Chen and Steiner (1997a)	Network representation and two approximation methods
Fm/1/C/NI/FixN/CV,DV/-/-/$C_{\max}$	Chen and Steiner (2003)	Network representation and approximation method in no-wait flow shop
Fm/1/C/II/FixN/CV/-/-/$C_{\max}$	Baker and Pyke (1990)	Heuristic method for more than two sublots
	Glass *et al.* (1994a)	Network representation and O(log n) algorithm
	Williams *et al.* (1997)	Network representation and heuristic method (feasible index search method)
	Glass and Potts (1998)	Network representation and optimal algorithm
	Topaloglu *et al.* (1994)	Closed form expressions when the maximum number of sublots is bounded by two
Fm/1/E/NI/FixN/CV/-/T/$C_{\max}$,$\sum F$	Steiner and Truscott (1993)	Optimal algorithm for different measures of performance
Fm/1/E/II/FlexN/CV/S(a)/T/$C_{\max}$	Kalir and Sarin (2001d)	Optimal algorithm in the presence of sublot-attached setup and transfer time
Fm/1/E/II/FixN/DV/-/-/$C_{\max}$,$\sum F$	Kalir and Sarin (2000)	Closed-form expressions for makespan and mean flow time reduction achieved by lot streaming
Fm/1/C/II/FlexN/CV/S(a)/-/Multiple Objectives	Bukchin and Masin (2004)	Frontier approach and tradeoff between alternative objective functions

Topaloglu *et al.* (1994) also use the idea of an upper envelope function and show that for the 2-sublot version of the problem, the sublot sizes that minimize the mean-flow-time can be obtained by enumerating the set of closed-form expressions for the minimum of individual convex functions, given by:

$$s_1^{k*} = \frac{1}{2} - \frac{\sum_{i=1}^{k-1} p_i}{2\left(2\sum_{i=k+1}^{m} p_i + p_k\right)},\ k = 1, \cdots, n,$$

and the intersection points of pairs of convex functions, each given by $y_k = a_k x_1^2 + b_k x_1 + c_k$, where $a_k = (2\sum_{l=k}^{m} p_l) - p_k$, $b_k = \left(\sum_{l=1}^{k} p_l - 2\sum_{l=k}^{m} p_l\right)$, and $c_k = \sum_{l=k}^{m} p_l$. Bukchin and Masin (2004) 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.

The multiple lot-multiple machine case is considerably harder to solve since the problem is NP-hard even without the use of lot streaming. The m-machine, multiple-lot, flow shop lot streaming problems addressed in the literature are presented in Table 1.6. A typical solution method for the problems in this category has been based on a heuristic method. Kumar *et al.* (2000) have addressed this problem for no-wait, flow shop and detached setup times. For continuous-sized sublots, they showed that an optimal sequence can be obtained by solving a traveling salesman problem (TSP) problem, whereas for integer-sized sublots they proposed a heuristic procedure. A genetic algorithm was also proposed to solve this problem that produced results comparable to those for the heuristic procedure, and it enabled determination of the number of sublots to use. Hall *et al.* (2003) have also addressed this problem, and have presented both an optimum seeking and heuristic method for its solution. For equal sublot sizes, Kalir and Sarin (2001c) have developed a heuristic procedure, called the bottleneck minimal idleness heuristic, to minimize the makespan, and have shown the effectiveness of using their method. The bottleneck is defined as the machine that requires the greatest total processing time. The secondary bottleneck is defined for each lot as the machine upstream of the bottleneck that has the largest unit processing time. Lots are sequenced in the lexicographic ordering of the closeness of the secondary bottleneck to the bottleneck machine. Kalir and Sarin (2001b) have presented an optimal algorithm for this problem based on the dominance of machines. Tseng and Liao (2008) have examined the same 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 objective of minimizing mean flow time, Kropp and Smunt (1990) have presented a quadratic programming formulation. Smunt *et al.* (1996) 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 have used both equal and

consistent sublot sizes, and they emphasize the importance of determining optimal number of sublots. For the case of continuous sublot sizes, Yoon and Ventura (2002a,b) 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 (Yoon and Ventura, 2002a), while pairwise interchange method and neighborhood search mechanisms are developed in Yoon and Ventura (2002b). Kim and Jeong (2009) 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) has considered sequence-dependent setups for this problem, and proposed a genetic algorithm-based heuristic method for its solution.

Table 1.6: The m-machine, multiple-lot, flow shop lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
Fm/n/E,C/II/FixN/CV/L(a)/-/F	Smunt *et al.* (1996)	Different splitting policy in stochastic flow shop environment
Fm/1/C/II/FixN/CV/L(a)/-/F	Kropp and Smunt (1990)	Optimal algorithm and heuristic method, deterministic and stochastic environment
Fm/n/C/II/FixN/CV-DV/S(a)/T/C_{max}	Glass and Possani (2011)	Optimal solutions for special cases (polynomial time method)
Fm/n/V/II/VarN/CV-DV/L(a)/-/C_{max}	Defersha and Chen (2010)	Heuristic Method for variable sublot sizes
Fm/n/C/II/FixN/CV-DV/-/-/C_{max},F	Feldmann and Biskup (2008)	Mixed integer program, interleaving of sublots of different lots
Fm/n/E/II/FlexN/CV,DV/S(a)/-/$C_{\max}$	Kalir and Sarin (2001b)	Polynomial time method
Fm/n/E/II/FlexN/CV/-/-/$C_{\max}$	Kalir and Sarin (2001c)	Heuristic method (bottleneck minimal idleness heuristic)
Fm/n/C/NI/FixN/CV,DV/L(d)/-/$C_{\max}$	Kumar *et al.* (2000)	Genetic algorithm-based heuristic methods
Fm/n/C/NI/FixN/CV,DV/L(d)/-/$C_{\max}$	Hall *et al.* (2003)	Optimal algorithm based on the generalized traveling salesman problem
Fm/n/E/II/FixN/CV/L(a)/-/Mean weighted absolute deviation from due date	Yoon and Ventura (2002a)	Heuristic method (genetic algorithm)
Fm/n/E/II,NI/FixN/CV/L(a)/-/Mean weighted absolute deviation from due date	Yoon and Ventura (2002b)	Heuristic method (pairwise interchange methods and neighborhood search mechanisms)
Fm/n/E/II/FixN/CV/-/-/Total weighted tardiness and earliness	Tseng and Liao (2008)	Heuristic method (discrete particle swarm optimization)
Fm/n/C/NI/FixN/DV/L(a)/-/$C_{\max}$	Kim and Jeong (2009)	Genetic algorithm-based heuristic method
Fm/n/C/II/FixN/DV/L(a),S(a)/-/$C_{\max}$	Martin (2009)	Genetic algorithm-based heuristic method

Parallel machines shop

Lot streaming has also been used in the parallel-machine environment in which a lot is split into sublots, which are, then, processed in an overlapping fashion on multiple machines. This environment is different from the others considered in this paper because a job of a lot is required to visit only one machine for its processing. Figure 1.2.3 shows processing of three lots on two parallel-machines. The first and third lots are split into two sublots, and a setup time is required every time a new sublot is processed on a machine. Table 1.7 lists the literature on lot streaming in the parallel machine environment.

Various approaches and objective functions have been used for splitting a lot for its processing on parallel machines. The objective functions that have been considered include: the sum of completion times, total tardiness, number of tardy jobs, and, maximum lateness. Yalaoui and Chu (2006) have addressed this problem for the objective of minimizing the sum of completion times. They extend the use of traditional shortest remaining processing time rule (SRPT), in which a lot is split into as many sublots as there are machines available, and each sublot is processed on a machine until and unless it is interrupted by the arrival of a different lot with a shorter remaining processing time. The interruption occurs on all machines. They prove the optimality of such a schedule. For cases where such an extended SRPT rule is not necessarily feasible, they use it to obtain a lower bound to employ in a branch-and-bound-based method. This method is further enhanced through the use of several dominance properties. They have also presented potential real-life applications of this problem.

Kim *et al.* (2004) have considered 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) employ a job-splitting method for the problem involving unrelated parallel machines and the objective of minimizing the total weighted tardiness. A mixed integer linear programming model is developed for the problem, and a tabu search-based heuristic method is proposed for its solution.

Suer *et al.* (1997) 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 the lot-splitting-based method performs better in the majority of cases. Lin and Jeng (2004) 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. Two dynamic programming-based algorithms are developed to obtain optimal solutions, but their solution time grows exponentially. They also propose several heuristic methods to find near-optimal solutions in a reasonable amount of time.

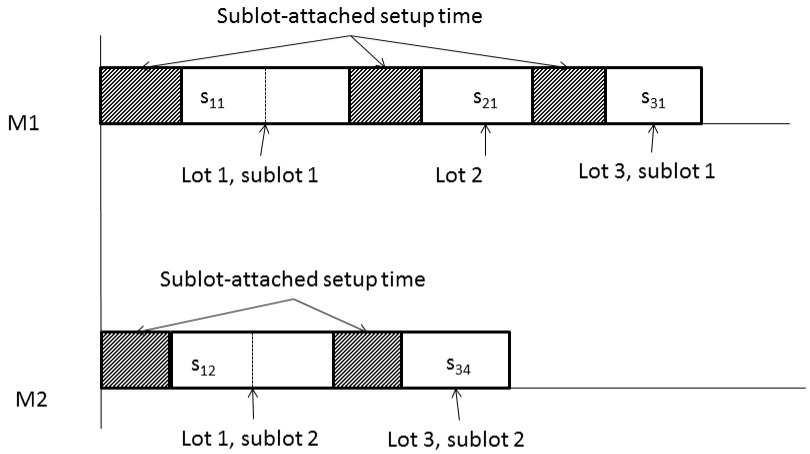


Figure 1.2.3: Lot streaming in parallel-machine environment

Hybrid flow shop

The hybrid flow shop consists of two or more stages, each of which comprises one or more parallel machines, with at least one stage consisting of multiple machines. Each lot (job) is processed on each stage sequentially but needs to be processed on at most one machine at any stage.

The hybrid flow shop, lot streaming problems addressed in the literature are shown in Table 1.8. These problems have been studied by Kim *et al.* (1997), Tsubone *et al.* (1996), Zhang *et al.* (2003, 2005), and Liu (2008). While a hybrid flow shop may consist of several stages, the existing literature is limited to the two-stage problem, with atleast one stage consisting of two or more machines. The hybrid flow shop is a general version of a flow shop and requires allocation of a sublot to one of parallel machines at a stage for processing. Consequently the results of the two-machine flow shop lot streaming problem are not directly applicable. Kim *et al.* (1997) have considered a two-stage hybrid flow shop with identical parallel machines at each stage and the objective of minimizing the makespan. They assume equal sublot sizes, lot-attached and lot-detached setup times, and the optimal solution is similar to that obtained using Johnson's rule. Tsubone *et al.* (1996) have considered the $1 + m$ (with one machine at stage 1 and m identical machines at stage 2) 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) and Liu (2008) have considered the $m + 1$ problem. Zhang *et al.* (2003) 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) assume equal sublot sizes and the objective of minimizing mean completion time. A lower bound and two heuristic methods are presented for the solution of this problem. Liu (2008) assumes a given number of sublots and the objective of minimizing the makespan. They prove that a rotation method of allocating the sublots of different lots over the m first-stage machines is optimal, and employ a linear programming model to obtain optimal sublot sizes. For the case of equal sublots, there exists an optimal schedule where all the first stage machines are balanced, i.e., have equal work content. They also present a heuristic method to determine number of sublots when all the sublots are of the same size.

Table 1.7: The parallel machines, lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

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

Table 1.8: The hybrid flow shop, lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

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)	Optimal algorithm similar to Johnson's rule
H(1+m)/n/-/II/-/S(a)/-/$C_{\max}$,$\sum F$,etc.,	Tsubone *et al.* (1996)	Simulation models to test different performance measures
H(m+1)/1/E/II/FlexN/CV/-/-/F	Zhang *et al.* (2003)	Heuristic methods
H(m+1)/n/E/II/FlexN/CV/S(a)/-/F	Zhang *et al.* (2005)	Mathematical programming model and heuristic method
H(m+1)/1/E/II/FlexN/CV/S(a)/-/$C_{\max}$	Liu (2008)	Heuristic method to test worst-case performance

Job Shop

The job shop, lot streaming problems addressed in the literature are shown in Table 1.9. One approach for using job splitting in job shops has been to permit intermingling of sublots of different lots. Often, an iterative procedure is used where the size and number of sublots are decided in one step, while their processing schedule is decided at a separate step. For the objective of minimizing makespan, Dauzere-Peres and Lasserre (1993, 1997) have presented an iterative procedure for the use of lot streaming in a general job shop with and without capacitated sublot sizes. They consider the case when the lots are first divided into sublots, and then, these sublots are treated as independent lots and are scheduled in the job shop, i.e., the sublots are allowed to intermingle. The sublot sizes are not allowed to change over different machines. 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. Since this problem is NP-hard, a heuristic based on the shifting bottleneck procedure (Adams *et al.*, 1988) is employed. Results of numerical experimentation are presented to show an improvement in makespan obtained because of lot streaming. Dauzere-Peres and Lasserre (1997) have also considered sublot-attached setup times and precedence constraints. Jeong *et al.* (1999) present an iterative procedure for lot splitting for the objective of minimizing the makespan when due dates must be satisfied. They analyze the influence of lot-detached setup times and allow consistent sublots without intermingling. In order to solve the sequencing and scheduling problem, and to reduce the number of batches, a modification of the shifting bottleneck procedure is developed. Once a schedule is developed, the number and size of sublots is decided. These steps are repeated until no further reduction in makespan is obtained, or, when a given target of reduction in makespan is achieved. Lot splitting is shown to reduce the makespan by up to 80% from the value obtained without lot splitting. Buscher and Shen (2009) present a three-phase algorithm to solve lot streaming problem in a job shop environment for minimizing makespan. The three-phase algorithm involves generation of equal sublot sizes, determination of schedules (by a tabu search method), and variation of sublot sizes. Individual sublots are permitted to intermingle. Performance of the basic tabu search method is confirmed by experimentation, and near-optimal solutions are generated in reasonable time.

Jacobs and Bragg (1988) and Kannan and Lyman (1994) have considered the objective of minimizing the mean flow time. Jacobs and Bragg (1988) 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 equal sublots in lot streaming, in order to reduce the flow time. Simulation experimentation is employed to evaluate the performance of the RL concept, and it is shown to significantly reduce flow times and their variability. Kannan and Lyman (1994) 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) 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) use multiple objectives with fixed-sized 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.

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 sizes of sublots. The open shop, lot streaming problems addressed in the literature are shown in Table 1.10. There are two cases considered in the literature. In the first case, all the sublots follow the same sequence of machines for their processing. Once such a sequence is determined, the problem reduces to a flow shop problem. In the second case, multiple routes exist and each sublot may follow an independent sequence. For a fixed number of equal sublots and no intermittent idling, Steiner and Truscott (1993) have shown that the optimal sequence of machines is pyramidal, i.e., there does not exist any machine in the sequence such that $p_{(i-1)} > p_{(i)} < p_{(i+1)}$, where $p_{(i)}$ is the processing time on the i'th machine in the optimal sequence. Sen and Benli (1998) extend these results to the case when sublots are not necessarily equal and intermittent idling is permitted. They also state that one of the important characteristics of an open shop is the one-to-one correspondence between an m-machine, n-product problem and an n-machine, m-product problem, that is, the machines and products could be switched. They note that the sequence in which the sublots are processed is known, i.e., the first sublot is followed by the second sublot, and so forth. By switching the machines and products, all that needs to be determined is an optimal permutation of machines that form a flow shop configuration. In addition, such a flow shop is an "ordered" flow shop since, if a machine i requires more time than a machine j to process a sublot, then, machine i will be slower for all other sublots as well. As shown in (Smith *et al.*, 1975), there exists an optimal pyramidal permutation for the ordered flow shop problem and a polynomial time algorithm to determine the optimal machine sequence for a given number and size of sublots. For multiple lots in a two-machine open shop, they have shown that lot streaming will reduce the makespan only if there is a job with large processing times. Glass *et al.* (1994a) have discussed the case of an m-stage production process, and have shown that in the presence of multiple routes, if $n \geq m$, the makespan can be minimized by using m equal sublots, with each sublot k following the machine sequence $k \rightarrow k+1 \rightarrow \cdots m \rightarrow \cdots k-1$. The resulting makespan is $m\frac{U}{m} \max_{1 \leq i \leq m}(p_i) = U \max_{1 \leq i \leq m}(p_i)$. Hall *et al.* (2005) have considered this problem for the no-wait case with sublot-attached setup

times. They formulate it as a traveling salesman problem with a pseudopolynomial number of cities.

Table 1.9: The job shop, lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
Jm/n/C/II/FixN/DV/-/-/$C_{\max}$	Dauzere-Peres and Lasserre (1993)	Heuristic method
Jm/n/C/II/FixN/DV/-/S(a)/$C_{\max}$	(Dauzere-Peres and Lasserre, 1997)	Optimal branch and bound-based algorithm, and, iterative heuristic with shifting bottleneck procedure
Jm/n/E,C,V/II/FixN/CV/L(a)/T/$C_{\max}$	Jeong *et al.* (1999)	Heuristic method (modified shifting bottleneck procedure)
Jm/n/C/II/FixN/CV/-/-/$C_{\max}$	Buscher and Shen (2009)	Heuristic method (tabu search)
Jm/n/C/II/FixN/CV/L(a)/-/F,Tardiness	Kannan and Lyman (1994)	Simulation for different family scheduling rules and transfer batch sizes
Jm/n/E,C/II/FixN/CV/L(a)/-/F	Smunt *et al.* (1996)	Different splitting policy in stochastic job shop environment
Jm/n/E/II/FixN/CV/L(a)/-/F,Tardiness	Jacobs and Bragg (1988)	Simulation model to test repetitive lots (RL) concept
Jm/n/C/II/FixN/CV/L(a)/-/Tardiness	Wagner and Ragatz (1994)	Repetitive lots (RL) on due date performance
Jm/n/E,C,V/II/FixN/CV/L(a)/-/Multiple Objectives	Jin *et al.* (1999)	Heuristic method (combination of Lagrangian relaxation and backward dynamic programming)
Jm/n/V/II/FixN/CV/S(a)/-/C_{max}	Defersha and Chen (2012)	Flexible job shop, sequence dependent setup time, LP-based sublot sizing
Jm/n/C/II/FixN,VarN/DV/-/-/C_{max}	Liu *et al.* (2012)	Numerical experimentation and heuristic search method to maximize exponentially decreasing value

Table 1.10: The open shop, lot streaming problems addressed in the literature (Cheng *et al.*, 2013)

Problem	Addressed by	Approach, Feature, and Contribution
Om/1/C/II/FixN/CV/-/-/$C_{\max}$	Glass *et al.* (1994a)	Network representation and optimal algorithm for multiple routes
Om/1/E/NI/FixN/CV/-/-/$C_{\max}$	Steiner and Truscott (1993)	Optimal algorithm to determine routing for equal sublots
Om/1/C/II/FixN/CV/-/-/$C_{\max}$	Sen and Benli (1998)	Optimal algorithm
O2/n/C/II/FixN/CV/-/-/$C_{\max}$	Sen and Benli (1998)	Optimal algorithm to determine routing for non-preemptive and preemptive cases
O2/n/C/NI/FixN/DV/S(a)/-/$C_{\max}$	Hall *et al.* (2005)	Optimal algorithm based on the generalized traveling salesman problem

1.2.3 Applications of Lot Streaming

The concept of lot streaming is similar to the idea of unit-piece flow in lean manufacturing (Womack and Jones, 2003), and in fact it can be considered as a generalization of the lean principles of material flow. Lot streaming helps in reducing makespan, WIP (work-in-process), and cycle time, among other measures of performance, by increasing the velocity of product flow. The key implementation strategy behind the flow principle of lean manufacturing is to transform a production system into a continuous flow system, one that involves smaller transfer lots (sublots), ideally of size one. However, unit-sized sublots may not always be economically feasible since increasing the number of sublots may increase material handling cost. Also, in the presence of transportation time, removal time, sublot-attached setup time, etc., unit-sized sublots may not be optimal for time-based objectives either. Nonetheless, a greater product flow velocity can be achieved by transferring sublots downstream that are smaller than the production batch. The problem, then, reduces to determining optimal number and sizes of sublots along with the sequence in which to process the batches. Such decisions assist in the implementation of the lean concept.

Another area of research that lot streaming relates to is material transfer co-ordination. Consider a two-stage assembly system. In such a system, material transfer decisions that maximize the velocity of material flow for one supplier may not be optimal for the entire system since there may exist another supplier that is a bottleneck for the entire system and increasing material flow from a supplier may merely increase handling cost, but not improve time-based performance. Two-stage assembly systems are frequently observed in industry. An instance of multiple lot assembly arises in the commercial truck manufacturing industry where customers place orders for small batches of trucks and orders may differ widely from each other. Customers often already have a fleet of trucks and want uniformity in the design of their trucks to make it easier to maintain spare parts inventory and perform maintenance checks or repairs. Therefore, even core elements of a truck's design may have to be adapted to customer requests, and it is not feasible to maintain an inventory of all possible sub-assemblies. Therefore, orders are placed with suppliers only after receiving a customer order. In addition, truck manufacturers have to quote short lead times in order to remain competitive. Other examples of two-stage assembly systems can be found in personal computer manufacturing (Potts *et al.*, 1995) , fire engine assembly (Lee *et al.*, 1993) and scheduling queries in distributed database systems (Allahverdi and Al-Anzi, 2006a).

1.2.4 Problems that have not been addressed in the literature

As can be seen in the survey of the literature in the prequel, lot streaming has been studied in a variety of production environments. In particular, lot streaming in two-stage flow shops has received the most attention. However, there are several two-stage lot streaming problems that have not been studied yet. Next, we outline some important problems that have yet to be addressed in the lot streaming literature:

- Frequently, when a new product is manufactured, initial production rates are lower than steady-state production rates. As the manufacturer gains experience, the production rate increases. Such a phenomenon is referred to as learning. The effect of learning can alter optimal sublot sizes for a lot streaming problem. However, the effect of learning has not been incorporated in any lot streaming model studied so far. Some areas of further investigation include:

  - Study of the impact of learning that influences only in processing times in lot streaming environments

  - Study of the impact of learning where it influences more than one processing step, sublot-attached setup times an processing times.

- Lot streaming has not been studied in the presence of stochastic processing times. In particular, whenever a manufacturer sources material from a supplier in two sublots, the performance of the system may be compared to dual sourcing, where the two sublots originate from different suppliers. In a stochastic environment, these two scenarios may have different performance characteristics. However, a comparison of these two cases has not been provided in the literature.

Another important area that has not been addressed in the literature is the case of two-stage assembly systems. These systems are ubiquitous in manufacturing environments, but the lot streaming literature related to them has been limited.

- Lot streaming in an assembly system has been studied by Sarin *et al.* (2011) in the past for the case when a single lot is manufactured and the multiple lot case has been studied by Sarin and Yao (2011) for the case when lot attached setup times are present. Both problems have been studied for the objective of minimizing makespan, and, under the assumption of equal number, and size, of sublots from each manufacturer. The case when each supplier is permitted to use different sublot sizes and different number of sublots has not been studied. Some new problem areas that may be researched are:

  - The relationship between two-stage flow shop problems and two-stage assembly system problems.

  - The influence of material handling costs on lot streaming decisions.

  - Efficient algorithms to optimize the size and number of sublots.

  - Integer programming formulations for the NP-hard cases of the two-stage assembly lot streaming problem.

1.3 Problem Studied in this Work

We address new research problems that address several of the new research areas identified in Section 1.2.4. In particular, we study lot streaming in two-stage flow shops and assembly

systems. Such two-stage production environments are observed when a supplier, at the first stage, produces material and supplies it to a manufacturer, at the second stage. While in some cases, the supplier may satisfy a manufacturer's demand from inventory, material may be produced after receiving an order due to several reasons. For example, in large commercial truck manufacturing, nearly all components are made to order by suppliers. One reason for doing so is that manufacturers have specific kitting (packaging) requirements and kitting material are provided to suppliers along with an order. While a supplier may produce goods in advance, the cost of packaging and storing in a warehouse, and then, repackaging using the manufacturer's kits usually proves uneconomical. This is particularly true in this industry because truck components are often very heavy and material handling and storage is expensive. Another reason that components are made to order is due to customization requirements. Often, customers who buy commercial trucks require a great deal of customization of the vehicles they purchase, either to satisfy the engineering requirements of the task they will perform, or to make new trucks conform to the design of the existing trucks in their fleet, so that maintenance is simplified. Due to customization, the design of truck components may vary greatly between orders. Hence, it may not be possible to maintain an inventory of all possible requirements. Additionally, truck designs are frequently updated, hence existing inventory may be rendered obsolete. Given the high cost of most components, this risk is unacceptable to suppliers.

In Chapter 2, we study lot streaming in the presence of learning in processing and setup times in a two-stage flow shop when sublot-attached setup times are present (see Figure 1.3.1). In our problem, we minimize the makespan, and, to that end, we obtain optimal sublot sizes and optimal number of sublots. Note that unlike most lot streaming problems, increasing the number of sublots may not necessarily result in a lower makespan when sublot-attached setup times are present. We extend our solution method to more general problems in Chapter 3 to address learning in general lot streaming problems where the effect of learning is observed in processing times. We introduce a transformation that allows the use of solution methods for contant-processing-time problems when learning is present. In addition, we provide details of the scope of problems to which it can be applied, and present a method to solve them.

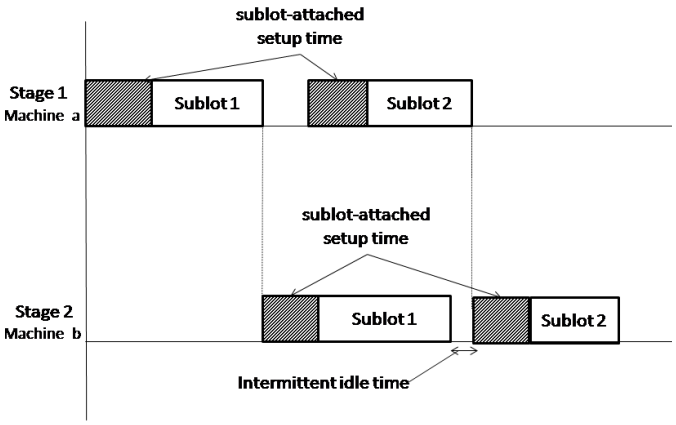
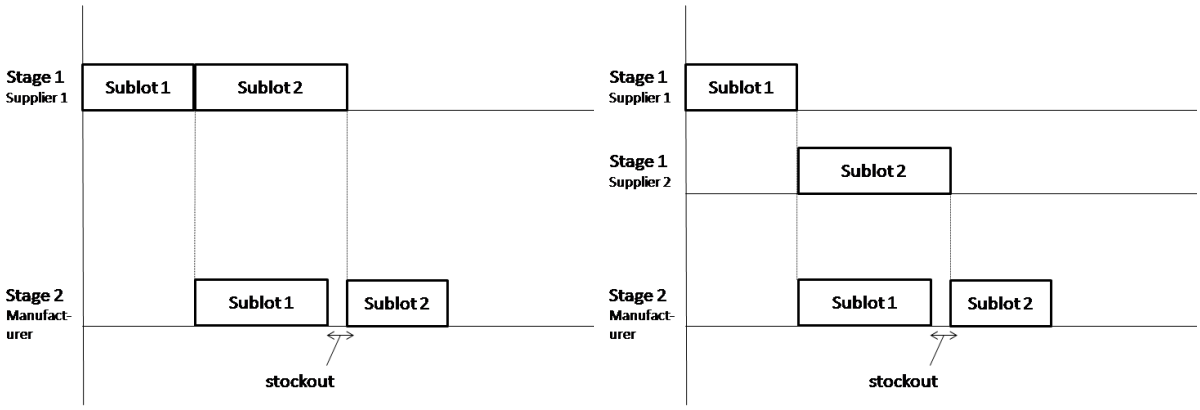


Figure 1.3.1: Two-stage flow shop with sublot-attached setup times

Then, in Chapter 4, we study lot streaming in the presence of stochastic processing times. We consider the case when material is supplied from one supplier at the first stage to a single manufacturer at the second stage. As discussed earlier, this constitutes a two-stage flow shop. The deterministic version of this problem has been studied by Trietsch (1987) and Potts and Baker (1989a) who show that optimal sublot sizes to minimize the makespan (which is linear in production lead time in this problem) are geometric in nature. However, when processing times are stochastic, we must consider the risk of a stockout at a manufacturer's facility between the exhaustion of the first sublot and the arrival of the second sublot. Reducing this risk by allowing the second sublot to arrive early increases the makespan (lead time) and average inventory. We provide closed-form expressions to evaluate these factors. We consider an alternative of dual sourcing in which two different suppliers at the first stage supply material, and, allow each supplier to produce one sublot. Then, we compare the performance of such a system to the case when a single supplier provides material using the same sublot sizes. Using two suppliers is referred to as dual sourcing, while using a single supplier, but with two sublots, can be termed single sourcing with lot streaming. We study single- and dual-sourcing for the cases when the suppliers' processing times are stochastic and when both the supplier and manufacturer have stochastic processing times. Figure 1.3.2 depicts the production environment considered in this chapter.



(a) Single sourcing (with lot streaming) (b) Dual sourcing

Figure 1.3.2: Sourcing strategies

Finally, in Chapter 5, we study the use of lot streaming in assembly systems with s distinct suppliers that produce sub-assemblies at the first stage, and a second stage, consisting of a single assembly line, where the sub-assemblies are assembled into a final product. We study this two-stage assembly problem for the case when l lots are produced by s suppliers, and suppliers may use lot streaming to reduce their lead time. Using lot streaming may result in a reduction of the makespan resulting from the production of all lots. However, it would also result in an increase in material handling cost. Hence, a weighted sum of the makespan and material handling costs is minimized. We refer to this problem as the Assembly Lot Streaming Problem (ALSP). We assume that the material handling cost increases with the number of sublots used, and it is given. Also, we assume that the maximum number of sublots, denoted by n_{iv} $i \in 1 \cdots l$, $v \in 1 \cdots s$, that any lot may use, is provided. In order to obtain an optimal schedule, (i) we must obtain the optimal sequence in which the lots are processed, (ii) the optimal number of sublots used by each lot from each supplier, and, (iii) the optimal sublot sizes. We refer to this problem as the Assembly Lot Streaming Problem (ALSP). Note that the size of the input data is polynomial in lsN, where N is the largest value of n_{iv}. An example of a schedule for the ALSP is shown in Figure 1.3.3. Scheduling multiple lots in an assembly system was first studied by Lee *et al.* (1993) for the objective of minimizing makespan for the case of number of suppliers, $s = 2$. Potts *et al.* (1995) considered the problem for an arbitrary s and proved the problem to be NP-hard in the strong sense, even for $s = 2$. They showed that there exists an optimal permutation schedule, and an arbitrary sequence has a worst-case ratio bound of 2 . They presented a heuristic with a worst-case bound of $2 - 1/s$. Hariri and Potts (1997) also study this problem and develop a branch-and-bound algorithm as well as obtain a lower bound and several dominance properties to improve its performance. Haouari and Daouas (1999) also propose a branch-and-bound algorithm for $m = 2$. Powerful heuristics for this problem have been proposed

by Sun *et al.* (2003). Allahverdi and Al-Anzi (2006a) obtained a dominance relationship for the case when setup times are considered separately from processing times. They also propose evolutionary heuristics to solve this problem. Some other objective functions that have been studied for this system are maximum lateness (Allahverdi and Al-Anzi, 2006b), maximum lateness with setup times (Al-Anzi and Allahverdi, 2007), a bi-criteria of maximum lateness and makespan (Al-Anzi and Allahverdi, 2009), total completion time (Al-Anzi and Allahverdi, 2006), total weighted flow-time time (Tozkapan *et al.*, 2003), a bi-criteria of makespan and mean completion time (Allahverdi and Al-Anzi, 2008) and total completion time with setup times (Allahverdi and Al-Anzi, 2009). Recently, Sarin *et al.* (2011) have studied the use of lot streaming when a single lot of identical items is produced in a two-stage assembly system and a setup time is incurred at the beginning of production on any machine. Sarin and Yao (2011) have addressed the same problem for the case of multiple lots. Given the number of sublots, they develop an algorithm to obtain optimal integer-sized sublot sizes. An instance of this problem occurs in commercial truck manufacturing where trucks are ordered in small batches by customers and components are made to order by manufacturers. Other examples of two stage assembly systems can be found in personal computer manufacturing (Potts *et al.*, 1995) , fire engine assembly (Lee *et al.*, 1993) and scheduling queries in distributed database systems (Allahverdi and Al-Anzi, 2006a).

There exists a close relationship between the ALSP and makespan minimization problems for single lot, two-machine, flow shops, where the machine at the first stage represents a supplier and the machine at the second stage represents the assembly operation. We refer to this problem as the Single Lot Flow Shop Lot Streaming Problem (SLFS-LSP) In Chapter 5, we show that in any optimal schedule, there exists atleast one lot for which the sublot sizes used by a supplier at the first stage are identical to the sublot sizes in an optimal schedule for the corresponding SLFS-LSP. We refer to this lot as the "bottleneck lot". For other suppliers, sublot sizes corresponding to an optimal schedule for the SLFS-LSP are optimal, but, due to the presence of slack, not necessarily unique. Since the bottleneck lot cannot be identified a-priori, the first step required to solve the ALSP is to solve the SLFS-LSP for each lot $i = 1, \cdots, l$, corresponding to each supplier $v = 1, \cdots, s$, for all values of $n = 1, \cdots, n_{iv}$. Note that, this step requires polynomial-time, provided the minimum makespan and optimal sublot sizes for each SLFS-LSP can be obtained in $O(N^{\alpha})$ time, where α is finite. The algorithm used to solve this problem is dependent on the production environment considered, for example, on the presence of factors like setup times, transportation times, the effect of learning etc., which impact the solution method used. As we show,the ALSP can be solved in polynomial-time if the production sequence is known, and it is NP-hard if an optimal sequence must be obtained. The methodology used to solve this second step is independent of the production environment, provided the total time required by the supplier or the assembly stage to process a lot is independent of the number of shipments used to transfer that lot from the supplier to the manufacturer. It requires the minimum makespan values of the SLFS-LSP problems obtained at the first step. Consequently, the solution method for the second step developed here is applicable to a wide variety of problems, provided the problems at the first are solvable.

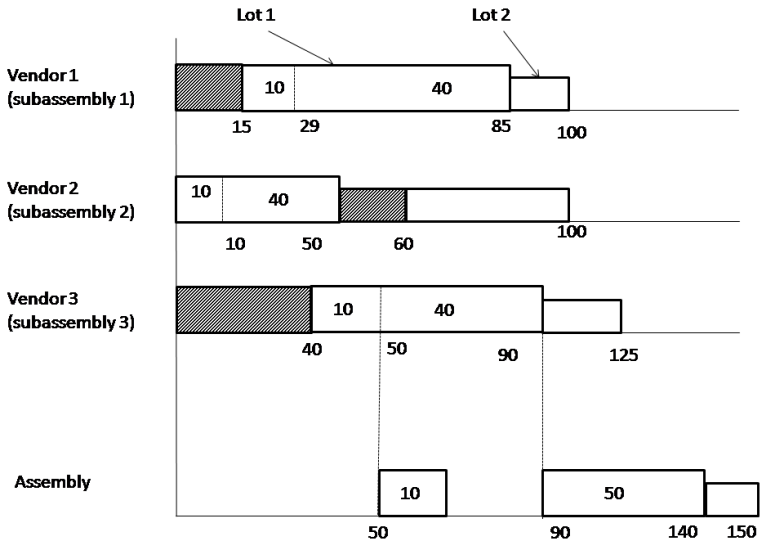


Figure 1.3.3: Lot streaming in a two-stage assembly shop.

Chapter 2

Lot Streaming in Two-stage Flow Shops in the Presence of Learning and Sublot-attached Setup Times

2.1 Introduction

In this chapter, we address the problem of lot streaming in a single lot flow shop in the presence of learning and sublot-attached setup times. For brevity, this problem is referred to as Lot streaming with Sublot Attached Setups (L-SAS). In addition, we study the same problem when the effect of learning is observed in processing times (referred to as L-SAS with Learning in Processing times, or, L-SASLP) and the case when learning is observed in sublot-attached setup times (referred to as L-SAS with Learning in sublot-attached Setup time, or, L-SASLS). We also study the case when learning is observed in both processing- and setup-times. Since only a single lot is considered in this chapter, we simplify the notation used by denoting the number of sublots used by n and the upper bound on n by N. Usually, the minimum makespan decreases with an increase in n. However, this is not true when sublot-attached setup times are present. It is possible that the minimum makespan value reaches a minimum for some value of n, say n_{opt}, and then increases. In such cases, all values of $n = n_{opt}+1, \cdots, N$ could be ignored, since they have a higher makespan as well as a higher material handling cost than that obtained for n_{opt}. In this chapter, we address the problem of determining both the maximum number of sublots n_{opt} that is economically viable, i.e., the value of n after which the makespan increases, and optimal sublot sizes for processing a single lot in a two-machine flow shop for the criterion of minimizing the makespan. A setup time is encountered every time a sublot is processed on a machine, and it is incurred only after the arrival of the sublot on that machine. We designate it as sublot-attached setup. This is in contrast to a sublot-detached setup that may be performed ahead of the arrival of the sublot on that machine. Also, we address the case when there exists a learning effect in processing

times and sublot-attached setup times. We provide closed-form expressions to determine optimal sublot sizes when sublot sizes are permitted to be continuous and propose simple search procedures to obtain optimal number of sublots, n_{opt}, for the makespan criterion. Also, we propose a method to determine integer sublot sizes.

The two new features in the context of the two-machine, flow-shop, lot streaming problem that we consider are: sublot-attached setups and the effect of learning on the processing and setup times. An instance of sublot-attached setup arises in semiconductor manufacturing where a processing tool, like a photolithography workstation, requires qualification whenever a new batch of jobs is to be processed on it, in order to keep the process in control. Kalir and Sarin (Kalir and Sarin, 2001a and Kalir and Sarin, 2003) have considered sublot-attached setup times for an m-machine flow-shop lot streaming problem involving a single lot and multiple lots, respectively, but for the case of equal-sized sublots. The problem addressed by Bukchin *et al.* (2002a) is similar to the problem we consider in this paper except for the objective function, which in their case is minimization of average flow time as against the minimization of makespan for our problem. Baker and Trietsch (2009) and Alfieri *et al.* (2012) have presented a method to obtain optimal discrete sublot sizes in the presence of sublot-attached setup times, and they have also proposed a heuristic that is based on continuous sublot sizes.

Our motivation for including the second feature is to study the impact of learning in the lot streaming environment, i.e., to determine how the optimal number of sublots and optimal sublot sizes vary as the processing and setup times change due to experience gained with every additional unit produced. The learning-effect becomes important whenever the setup and processing of a job are performed manually. An example of such a scenario arises in a small-scale wafer fabrication facility that mainly caters to military demand. Such a facility is operated manually because of relatively small-sized orders of products, which keep evolving with military needs.

Closed-form solutions have been developed for some lot streaming problems involving two, three and m-machine flow shop problems (Potts and Baker, 1989a; Glass *et al.*, 1994b; Kalir and Sarin, 2001a). Most of the literature on finding closed-form solutions assumes a continuously divisible lot. Moreover, it has been shown that continuous sublot sizes provide a good approximation when the production lot consists of discrete parts (Bukchin *et al.*, 2002a; Shallcross, 1992). In this paper as well, we assume a continuously divisible lot to develop closed-form solutions for optimal sublot sizes. However, we also present a method to determine integer sublot sizes.

A learning effect is observed when a worker's ability increases with experience. The mathematical form of a learning curve was introduced by Wright-Patterson (1936) who observed the cycle time to reduce by a constant percentage every time the quantity produced was doubled. Yelle (1979) has reviewed various modifications of Wright-Patterson's learning curve. The log-linear form of the learning curve remains popular, and it has been supported empirically in the literature (Dar-El, 2000). If we represent the time required to execute the

first cycle by τ^1 and that required by a subsequent cycle i by τ^i, $i = 2, 3, \cdots$, then Wright-Patterson's learning curve can be expressed as a function of the number of cycles i and a constant d as follows: $\tau^i = \tau^1 i^{-d}$, $i = 2, 3 \cdots$, where $d\,(> 0)$ is called the learning constant, and it captures a worker's ability to learn. A larger value of d represents faster learning, and hence, a quicker reduction in cycle time. The value of d is less than one in practice (Baloff, 1971). Dar-El (2000) and Dar-El *et al.* (1995a,b) have related the standard processing time, denoted by STD, to the first execution time by the expression: $STD = {}^{\tau^1}/_{(57-60\,2^{-d})}$. A variety of scheduling problems in the presence of learning have been studied in the literature. The impact of learning in job processing times in the context of a single machine scheduling problem has been studied by Biskup (1999a) for the joint objective of minimizing the sum of flow times and deviation from a common due date. Mosheiov (2001) has shown that this problem can be solved in polynomial time for the objective of minimizing the makespan. The problem of minimizing the total completion time for a two-machine flow shop has been addressed by Lee and Wu (2004), who proposed a heuristic to solve this problem, and also, developed some dominance rules to enhance the effectiveness of a branch-and-bound method for this problem. Biskup (2008a) has provided a review of scheduling in the presence of learning. However, the study of lot streaming in the presence of learning is new, and it has not been addressed in the literature.

In what follows, we present the notation, assumptions and some preliminary results in Section 2.2. The problem of determining optimal sublot sizes and number of sublots for constant job processing and sublot-attached setup times for a continuously divisible lot is discussed in Section 2.3, while the same problem in the presence of learning in processing and setup times is analyzed in Section 2.4. A method to determine integer sublot sizes is then presented in Section 2.5. Finally, some concluding remarks are made in Section 2.6.

2.2 Notation, Assumptions and Some Preliminaries

Consider the following notation:

Subscripts:

a = Machine 1.

b = Machine 2.

Parameters:

p_a = Unit processing time on machine 1.

p_b = Unit processing time on machine 2.

U = lot size.

N = Maximum possible number of sublots. Note that all sublot sizes do not necessarily take positive

values and some can be zero.

τ_a = Sublot-attached setup time on machine 1 that is encountered only if sublot size is positive.

τ_b = Sublot-attached setup time on machine 2 that is encountered only if sublot size is positive.

Decision Variables

n = Number of sublots with positive sizes.

s_k = Size of the k^{th}sublot, $k = 1, 2, \cdots, n$; it can take continuous values

(the case of discrete sublot sizes is discussed in Section 2.5).

Furthermore, we define the ratio of the processing time of a job on machine 2 to that on machine 1 by

$$q = p_b/p_a. \tag{2.1}$$

and,

$$T = (\tau_b - \tau_a)/p_a. \tag{2.2}$$

When $q = 1$, we have $p_b = p_a$, and we denote both p_a and p_b by p in that case. Note that, when $\tau_a \neq \tau_b$, T can take a positive or negative value. Unless mentioned otherwise, we use M to denote the makespan of a schedule. We assume that the material is processed on machine 2 in the order in which it arrives. Sublot sizes are said to be consistent if they remain the same on both the machines. In the absence of any setup, Potts and Baker (1989a) have shown that there exists an optimal schedule with consistent sublot sizes for the two-machine flow shop, makespan minimization, lot streaming problem. It is easy to show that this result holds true when sublot-attached setups are present as well.

Property 2.1: There exists an optimal schedule with consistent sublot sizes for the L-SAS problem.

Proof. We prove this result by transforming an optimal schedule for which the sublot sizes are not consistent to a schedule with consistent sublots without increasing the makespan.

Let the given optimal schedule consist of n sublots on machine 1 and n' sublots on machine 2. Sublot i on machine 1 can supply material to more than one sublot on machine 2. We denote the first sublot on machine 2 that receives material from sublot i by $j^{min}(i)$ and the last sublot that receives material from sublot i by $j^{max}(i)$, i.e., sublot i supplies material to sublots $j^{min}(i), j^{min}(i) + 1, \cdots, j^{max}(i)$ of machine 2. Similarly, a sublot on machine 2 may receive material from more than one sublot on machine 1. Since, by assumption, the material is processed on machine 2 in the order in which it arrives from machine 1, the earliest sublot on machine 2 that can receive material from sublot $i + 1$ (of machine 1) is

$j^{max}(i)$. Therefore, $j^{max}(i) \leq j^{min}(i+1)$. If $j^{max}(i) = j^{min}(i+1)$, there must exist some material that is processed in sublot i (on machine 1) and then transferred to sublot $j^{max}(i)$ (on machine 2). We can transfer this material from sublot i (on machine 1) to sublot $i+1$ (on machine 1). Clearly, this step does not increase the makespan since sublot $j^{max}(i)$ (on machine 2) does not begin processing until all the necessary material from sublot $i+1$ (on machine 1) has arrived. Repeating this step would produce a schedule in which each sublot on machine 1 supplies material to exactly one sublot on machine 2. Now, if more than one sublot on machine 1 supplies material to a sublot j on machine 2, then these sublots can be merged. Clearly, this would also not alter the completion time of the last sublot to transfer material to sublot j (on machine 2), and hence would not impact the makespan. Repeating this step would produce a schedule in which where each sublot on machine 2 receives material from exactly one sublot on machine 1. Hence, by repeating the above steps, we obtain a schedule with consistent sublot sizes without increasing the makespan. □

The reversibility property in the context of scheduling is said to hold if the makespan obtained by scheduling the jobs in the forward sequence from machine 1 to machine 2 is the same as that obtained by scheduling them backward from machine 2 to machine 1. The reversibility property holds true for the two-machine flow shop, makespan minimization, lot streaming problem (Potts and Baker (1989a)). This property holds true in the presence of sublot-attached setup times as well because the critical path remains the same in the forward and backward directions, and it is stated as follows:

Property 2.2: The L-SAS problem is reversible.

Proof. The length of a critical path determines the makespan for a flow-shop. For the problem on hand, the length of a critical path is given by the following expression,

$$M = \underset{i=1,\cdots,n}{max} \left(\sum_{k=1}^{i} (\tau_a + p_a s_k) + \sum_{k=i}^{n} (\tau_b + p_b s_k) \right). \tag{2.3}$$

For the reverse problem, the order of the sublots is reversed, i.e., if we denote the sublot sizes for the reverse problem by $s_k^{'}$, we have $s_k^{'} = s_{n+1-k}$, $k = 1, \cdots, n$. So, the critical path is obtained by the following expression,

$$M = \underset{i=1,\cdots,n}{max} \left(\sum_{k=1}^{i} (\tau_b + p_b s_k^{'}) + \sum_{k=i}^{n} (\tau_a + p_a s_k^{'}) \right).$$

By substituting $s_k^{'}$ with s_{n+1-k} we have,

$$M = \underset{i=1,\cdots,n}{max} \left(\sum_{k=1}^{i} (\tau_b + p_b s_{n+1-k}) + \sum_{k=i}^{n} (\tau_a + p_a s_{n+1-k}) \right).$$

After rearranging the indices,

$$M = \underset{i\,=\,1,\cdots,n}{max} \left(\sum_{k=n-i+1}^{n} (\tau_b + p_b s_k) + \sum_{k=1}^{n-i+1} (\tau_a + p_a s_k) \right),$$

which is the same as,

$$M = \underset{i\,=\,1,\cdots,n}{max} \left(\sum_{k=i}^{n} (\tau_b + p_b s_k) + \sum_{k=1}^{i} (\tau_a + p_a s_k) \right). \tag{2.4}$$

Since (2.3) and (2.4) are identical, both the problems have the same makespan for a given set of sublots, provided the sequence is reversed. Hence, the optimal solution for the forward problem yields an optimal solution for the reversed problem as well. □

As a result of Property 2.2, we can assume, without loss of generality, the machine with the larger sublot-attached setup time to be machine 2, and hence, $T \geq 0$. Since the case of $\tau_a = \tau_b = 0$ has been studied by Potts and Baker (1989a), here, we consider the case when at least one of τ_a and $\tau_b (\geq \tau_a)$ is positive, which implies that at least one of τ_a and T is positive. In addition, we assume that $p_a, p_b > 0$, and that, there exists an infinite buffer space between the machines. The first sublot is assumed to be processed on machine 1 at time zero. By definition, a sublot-attached setup time exists only for a positive sublot size, and it is completed immediately before the start of the processing of a sublot. Furthermore, we assume that a sublot on machine 2 can begin processing as soon as the sublot on machine 1 containing the items constituting that sublot on machine 2 has finished processing.

We also use the notion of a *compact schedule*. A schedule is designated as compact if there exists no idle times between the processing of the sublots, and if the completion time of the k^{th} sublot on the first machine is equal to the completion time of the $k - 1^{th}$ sublot on the second machine, where $k = 2, 3, \cdots, n$ and $s_k > 0$ for $k = 1, 2, 3, \cdots, n$.

For the case when learning is present, we assume identical learning constants over the machines albeit different for processing and setup times, (see Dar-El and Rabinovitch (1988) and Dar-El (2000)).

2.3 Optimal Solution for the L-SAS Problem When the Lot is Continuously Divisible

We begin in Section 2.3.1 by showing that there exists an optimal schedule for this problem that is a compact schedule. Then, in Section 2.3.2, we obtain optimal sublot sizes and the minimum makespan value for a compact schedule, when the number of sublots, n, is given.

A method to determine the number of sublots that minimize the makespan is presented in Section 2.3.3. To that end, we determine a bound value, $n_{infeasible}$, for the number of sublots for which a compact schedule is infeasible if $n \geq n_{infeasible}$. We also show that there exists a unique minimum makespan value for compact schedules, and it is achieved either at some unique $n_{opt} < n_{infeasible}$ or at both n_{opt} and $n_{opt} + 1$.

2.3.1 Properties of the optimal solution

A feasible schedule for our problem consists of $n \in \{1, 2, \cdots, N\}$ sublots with their sublot sizes given by s_k, $k = 1, \cdots, n$, such that $s_1 = U$ if $n = 1$ and $U > s_k > 0$ with $\sum_{k=1}^{n} s_k = U$ for $n \geq 2$. Clearly, postponing the processing of any sublot on machine 1 cannot improve the makespan; hence, we assume that there does not exist an idle time between the processing of sublots on machine 1. We denote the makespan obtained for such a schedule by $M(n, s_1, \cdots, s_n)$ and the completion time of sublot k on machine 2 by C_k. We have, $C_2 = (\tau_b + p_b s_2) + max(\sum_{k=1}^{2}(\tau_a + p_a s_k), (\tau_a + p_a s_1) + (\tau_b + p_b s_1))$, and $C_j = (\tau_b + p_b s_j) + max(\sum_{k=1}^{j}(\tau_a + p_b s_k), C_{j-1})$ $\forall j = 3, \cdots, n$. In particular,

$$M(n, s_1, \cdots, s_n) = p_b s_n + max(\sum_{k=1}^{n}(p_a s_k + \tau_a), C_{n-1}). \tag{2.5}$$

Remark 2.1. The makespan function given by (2.5) is continuous in $(s_1, \cdots, s_n)$ since the maximum and summation functions are continuous, and the composite of a finite number of continuous functions is also continuous. The set of possible sublot sizes for a given number of sublots $n \geq 2$ is open and bounded. For $n = 1$, there exists a single feasible schedule, and hence, it is optimal.

First, we show the following result.

Proposition 2.1. *There exists an optimal compact schedule with $n_{opt} \in \{1, \cdots, N\}$ sublots.*

Proof. Note that $\inf_{n, s_1, \cdots, s_n} M(n, s_1, \cdots, s_n) = \inf_{n} \left(\inf_{s_1, \cdots, s_n} M(n, s_1, \cdots, s_n) \right)$. We prove the result by showing that, for a given n, $\min_{s_1, \cdots, s_n} M(n, s_1, \cdots, s_n)$ exists only if $\inf_{s_1, \cdots, s_n} M(n, s_1, \cdots, s_n)$ is achieved for a compact schedule with n sublots; otherwise, $\inf_{s_1, \cdots, s_n} M(n, s_1, \cdots, s_n) \geq \inf_{s_1, \cdots, s_{n'}} M(n', s_1, \cdots, s_{n'})$ for some $n' < n$. We show this in two steps: (i) if there exists an optimal schedule for a given n with $s_k > 0$, $\forall k = 1, \cdots n$, then it must be a compact schedule, and (ii) if there does not exist an optimal schedule for a given n with $s_k > 0$, $\forall k = 1, \cdots n$, then $s_k \to 0$ for at least one $k \in \{1, \cdots, n\}$ as $M(n, s_1, \cdots, s_n) \to \inf_{s_1, \cdots, s_n} M(n, s_1, \cdots, s_n)$.

We prove step (i) by contradiction. Suppose there exists an optimal schedule having idle times between the processing of the sublots on machine 2. By assumption, $s_i > 0$, for $i = 1, 2, \cdots, n$. Machine 1 is the bottleneck until some sublot $k \leq n$, machine 2 is the bottleneck

thereafter, and there exists an idle time on machine 2 after the completion of sublot r, where $r < k$. The makespan, M, for this scenario can be expressed as:

$$M = [k\tau_a + p_a \sum_{i=1}^{k} s_i] + [(n-k+1)\tau_b + p_b \sum_{i=k}^{n} s_i]. \tag{2.6}$$

Consider the following new sublot sizes obtained by replacing s_r and s_i with $s_r^{'}$ and $s_i^{'}$ respectively, using $\Delta, \theta > 0$ such that $\sum_{i=1}^{n} \theta q^{i-1} = \Delta$ and Δ sufficiently small to maintain feasibility.

We have,

$$s_r^{'} = s_r + \Delta - \theta q^{r-1}, \text{ and} \tag{2.7}$$

$$s_i^{'} = s_i - \theta q^{i-1} \quad \forall\, i \neq r,\, i = 1, 2, \ldots, n\ , \tag{2.8}$$

Since Δ, and therefore θ, can be made arbitrarily small, the new sublot sizes $s_r^{'}$ and $s_i^{'}$, $i \neq r$, $i = 1, 2, \ldots, n$ are non-negative.

For these sublot sizes, we obtain the makespan,

$$M^{'} = [k\tau_a + p_a \sum_{i=1}^{k} (s_i - \theta q^{i-1})] + [(n-k+1)\tau_b + p_b \sum_{i=k}^{n} (s_i - \theta q^{i-1})] + p_a\Delta, \tag{2.9}$$

and

$$M' - M = p_a\Delta - p_a \sum_{i=1}^{k} \theta q^{i-1} - p_b \sum_{i=k}^{n} \theta q^{i-1}. \tag{2.10}$$

By replacing p_b with $p_a q$, we get

$$M' - M = p_a\Delta - p_a \sum_{i=1}^{k} \theta q^{i-1} - p_a \sum_{i=k+1}^{n+1} \theta q^{i-1}.$$

Since $\sum_{i=1}^{n} q^{i-1}\theta = \Delta$, we have

$$M' - M = -p_a\theta q^n,$$

which contradicts the fact that the given solution is optimal.

Now, consider step (ii). By Remark 2.1, $M(n, s_1, \cdots, s_n)$ is continuous in $(s_1, \cdots, s_n)$, $n \geq 2$, and the set of feasible sublot sizes is open and bounded. Let $\bar{M}(n, s_1, \cdots, s_n)$ be the continuous extension of $M(n, s_1, \cdots, s_n)$ to the closure of the set of feasible solutions. Since the closure of the set of feasible sublot sizes is closed and bounded, $\bar{M}(n, s_1, \cdots, s_n)$ attains a minimum at some point $(s_1^*, \cdots, s_n^*)$ by Weierstrass's theorem. By continuity, and the fact that a minimum is not obtained in the interior of the set, $\bar{M}(n, s_1^*, \cdots, s_n^*) = \underset{s_1,\cdots,s_n}{inf} M(n, s_1, \cdots, s_n)$. Since the infimum is not achieved at any interior point, $(s_1^*, \cdots, s_n^*)$ must be a boundary point. But, any point on the boundary of the feasible region has at least one $s_{k'} = 0$, $k' \in \{1, \cdots, n\}$. Thus, $\bar{M}(n, s_1^*, \cdots, s_n^*)$ corresponds to the makespan of a schedule with n sublot-attached setup times and $n' < n$ positive sublot sizes. Since the setup times associated with the sublots of zero size are redundant, they can be deleted to yield a feasible schedule with $n' < n$ sublots and sublot sizes $s_k > 0, \forall k = 1, \cdots, n'$, whose makespan value is no larger than $\underset{s_1,\cdots,s_n}{inf} M(n, s_1, \cdots, s_n)$. If an optimal schedule exists for n', then by step (i) it is a compact schedule and, since it is optimal, it's makespan value is no larger than that of any other feasible schedule and $\underset{s_1,\cdots,s_{n'}}{min} M(n', s_1, \cdots, s_{n'}) \leq \underset{s_1,\cdots,s_n}{inf} M(n, s_1, \cdots, s_n)$. Otherwise, if an optimal schedule does not exist for n' sublots, the above process can be repeated. Since there exists an optimal solution for $n = 1$, there must exist an $1 \leq n'' < n$ for which an optimal schedule exists, which is a compact schedule. Therefore, $\underset{n}{inf} \left(\underset{s_1,\cdots,s_n}{inf} M(n, s_1, \cdots, s_n) \right)$ is achieved for a compact schedule and for some n_{opt} sublots. □

2.3.2 Determination of sublot sizes and makespan

Next, we find sublot sizes, s_i, $i = 1, \cdots, n$, for a compact schedule. We consider the cases for q=1 and $q \neq 1$ separately.

By the definition of a compact schedule, we have:

$$p_a s_k + \tau_a = p_b s_{k-1} + \tau_b, \qquad k = 2, \cdots, n \tag{2.11}$$

By (2.11),(2.2) and (2.1),we obtain the size of sublot k,

$$s_k = q s_{k-1} + T, \qquad k = 2, \cdots, n, \tag{2.12}$$

or

$$s_k = \begin{cases} q^{k-1} s_1 + T \left(\frac{1-q^{k-1}}{1-q} \right), & k = 2, \cdots, n, \ q \neq 1 \\ s_1 + (k-1)T, & k = 2, \cdots, n, \ q = 1 \end{cases}. \tag{2.13}$$

Note that

$$\sum_{i=1}^{n} s_i = U. \tag{2.14}$$

By substituting the values of s_k, $k = 2, \cdots, n$, from (2.13) in (2.14) and solving for s_1, we have

$$s_1 = \begin{cases} \left[\frac{U - \frac{nT}{1-q} + \frac{T}{1-q}\left(\frac{1-q^n}{1-q}\right)}{\left(\frac{1-q^n}{1-q}\right)} \right], \ q \neq 1 \\ \left[\frac{U}{n} - (n-1)\frac{T}{2} \right], \ q = 1 \end{cases}. \tag{2.15}$$

Furthermore, the makespan can be expressed as follows,

$$M = n\tau_a + p_a U + \tau_b + p_b s_n = n\tau_b + p_b U + \tau_a + p_a s_1. \tag{2.16}$$

Remark 2.2. Note that, if T and s_1 are positive, then all the s_k, $k = 2, \cdots, n$, are positive. Recall that $T \geq 0$, by assumption. Therefore, we need to show $s_1 \geq 0$ to ensure that all the sublot sizes are non-negative.

In the absence of sublot-attached setup times, T=0, and consequently, sublot sizes s_k, $k = 1, \cdots, n$, (given by (2.15) and (2.13)), reduce to geometric sublot sizes, the result shown by Potts and Baker (1989a) for the case when sublot-attached setup times are not present. In this respect, the above expressions generalize their result.

Consider the following property of s_1, which essentially follows from its definition given in Expression (2.15).

Proposition 2.2. *The value of s_1 is monotone decreasing in n, and there exists an n, designated $n_{infeasible}$, such that $s_1 < 0$ for $n \geq n_{infeasible}$.*

Proof. Consider $s_1(n) - s_1(n+1)$ for $q \neq 1$.

Since,

$s_1(n) = \left[\frac{U - \frac{nT}{1-q} + \frac{T}{1-q}\left(\frac{1-q^n}{1-q}\right)}{\left(\frac{1-q^n}{1-q}\right)} \right]$, and

$s_1(n+1) = \left[\frac{U - \frac{(n+1)T}{1-q} + \frac{T}{1-q}\left(\frac{1-q^{n+1}}{1-q}\right)}{\left(\frac{1-q^{n+1}}{1-q}\right)} \right]$, by (2.15)

we have

$s_1(n) - s_1(n+1) = U\left(\frac{1}{\frac{1-q^n}{1-q}} - \frac{1}{\frac{1-q^{n+1}}{1-q}} \right) + T\left(\frac{1+n}{1-q^{n+1}} - \frac{n}{1-q^n} \right)$.

Note that $\left(\frac{1}{\frac{1-q^n}{1-q}} - \frac{1}{\frac{1-q^{n+1}}{1-q}} \right) > 0$ for $q \neq 1$.By rearranging the second term, we obtain :

$T\left(\frac{1-q^n+n-nq^n\ -n+nq^{n+1}}{(1-q^{n+1})(1-q^n)}\right)$

$= \frac{T(1-q)}{(1-q^{n+1})(1-q^n)}\left(\frac{1-q^n}{1-q} - nq^n\right)$

$= \frac{T}{(1-q^{n+1})(1-q^n)}\left(\frac{(1-q)^2}{1-q}\right)(1+q+\cdots+q^{n-1}-nq^n)$

$= \frac{T(1-q)^2}{(1-q^{n+1})(1-q^n)}\left(\frac{(1-q^n)+(q-q^n)\cdots(q^{n-1}-q^n)}{1-q}\right) > 0.$

Therefore, $s_1(n) > s_1(n+1)$, which implies that s_1 is monotone decreasing in n. Hence, if $s_1(n) < 0$, then $s_1(n+1) < 0$.

To prove the second part, consider the expression of s_1 given in (2.15). Note that

$$\lim_{n\to\infty}\left[\frac{U-\frac{nT}{1-q}+\frac{T}{1-q}\left(\frac{1-q^n}{1-q}\right)}{\left(\frac{1-q^n}{1-q}\right)}\right] = (U(1-q)+\frac{T}{1-q}-\infty) < 0, \quad \text{for} \quad q<1.$$

and

$$\frac{T}{1-q} - \lim_{n\to\infty}\left(\frac{nT}{1-q^n}\right), \quad \text{for} \quad q>1,$$

$$\text{since } \frac{1-q^n}{1-q} \to \infty, \quad \text{for} \quad q>1.$$

Using L'Hopital's rule, $\frac{T}{1-q} - \lim\frac{nT}{1-q^n} = \frac{T}{1-q} - \lim\frac{T}{-q^n\ln(q)} = \frac{-T}{q-1} < 0$, which implies that s_1 becomes negative as $n \to \infty$.

Next consider the case when $q = 1$. We have $s_1 = \frac{U}{n} - (n-1)\frac{T}{2}$ by (2.15). Clearly, the positive term $\frac{U}{n}$ decreases with n and the absolute value, $(n-1)\frac{T}{2}$, of the negative term increases with n. Therefore, the value of s_1 is monotone decreasing. Moreover, the value of s_1 is negative when $\frac{n(n-1)T}{2} > U$.

Hence, in any case, the value of s_1 converges to a negative number or diverges to -∞ as $n \to \infty$, thereby implying that there exists a value of n, designated $n_{infeasible}$, such that $s_1 < 0$ for all $n \geq n_{infeasible}$ and $s_1 \geq 0$ if $n < n_{infeasible}$. □

We can compute the values of $s_k,\ k = 2,\cdots,n$, by substituting for s_1 from (2.15) in (2.13). By substituting the value of s_n, computed from (2.13), in (2.16), we have,

$$M = \begin{cases} n\tau_a + p_aU + \tau_b + p_bT\left(\frac{1-q^{n-1}}{1-q}\right) + \frac{p_bTq^{n-1}}{1-q} + \frac{p_bUq^{n-1}-p_bUq^n-p_bq^{n-1}nT}{1-q^n} & for\ \text{q}\neq 1 \\ n\tau_a + pU + \tau_b + \frac{pU}{n} + \frac{p(n-1)T}{2}, & \text{for q=1} \end{cases} \quad (2.17)$$

2.3.3 The number of sublots for which the makespan is minimized

By Proposition 2.1, there exists an optimal schedule that is a compact schedule. In addition, by Remark 2.2 and Proposition 2.2, there exists an $n_{infeasible}$ such that compact schedules are feasible if and only if $n < n_{infeasible}$. We also have $n \leq N$. Therefore, we can restrict our attention to compact schedules with the number of sublots $n \leq min(N, n_{infeasible} - 1)$.

Consider the makespan Expression (2.17), now written as a function of n:

$$M(n) = n\tau_a + p_a U + \tau_b + p_b T \left(\frac{1-q^{n-1}}{1-q}\right) + \frac{p_b T q^{n-1}}{1-q} + \frac{p_b U q^{n-1} - p_b U q^n - p_b q^{n-1} n T}{1-q^n}, \text{ for } q \neq 1, \tag{2.18}$$

and, by substituting $T = (\tau_b - \tau_a)/p$ in (2.17) for $q = 1$, and simplifying,

$$M(n) = n\frac{\tau_b + \tau_a}{2} + \frac{pU}{n} + \frac{\tau_b + \tau_a}{2} + pU, \text{ for } q = 1. \tag{2.19}$$

First, we relax the integrality restriction on n and allow it to be a continuous variable whose range is $[1, N]$. With this relaxation, we show that $M(n)$ is a strongly quasi-convex function of n for $q \neq 1$ and a strictly convex function of n for $q = 1$. These properties of $M(n)$ aid in determining optimal, integer number of sublots effectively.

Case q=1

Remark 2.3. $M(n)$ is a strictly convex function of n when $q = 1$.

Recall that both p_a and p_b are denoted by p in this case, and clearly, the second derivative of Expression (2.19) with respect to n, $\frac{2pU}{n^3} > 0$. Therefore, M(n) is a strictly convex function.

Now, by setting the first derivative of $M(n)$ to zero, the minimum value is obtained for

$$n^* = \sqrt{\frac{2pU}{\tau_b + \tau_a}}. \tag{2.20}$$

Next, we show that $s_1 > 0$ when $\lfloor n^* \rfloor$ sublots are used.

Proposition 2.3. *The number of sublots, $n = \lfloor n^* \rfloor$, affords a feasible compact schedule when $q = 1$.*

Proof. Consider

$$\begin{aligned}\frac{\partial M(n)}{\partial n} &= \frac{\tau_b+\tau_a}{2}-\frac{pU}{n^2}\\ &= \tau_a+\frac{\tau_b-\tau_a}{2}-\frac{pU}{n^2}.\end{aligned}$$

Substituting $T=(\tau_b-\tau_a)/p$ and $\frac{U}{n}=s_1+(n-1)\frac{T}{2}$ for $q=1$ from (2.15), we obtain,

$$\begin{aligned}\frac{\partial M(n)}{\partial n} &= \tau_a+\frac{pT}{2}-\frac{p}{n}\left(s_1+(n-1)\frac{T}{2}\right)\\ &= \tau_a+\frac{p}{n}\left(\frac{T}{2}-s_1\right).\end{aligned}$$

Clearly, if $s_1<0$, then $\frac{\partial M(n)}{\partial n}>0$ since p¿0 and at least one of τ_a and T is positive by assumption. However, we have shown above that M(n) is strictly convex, and $\lfloor n^*\rfloor<n^*$. Therefore, $\frac{\partial M(\lfloor n^*\rfloor)}{\partial n}<0$, which implies that $s_1>0$ for $n=\lfloor n^*\rfloor$. □

Since $M(n)$ is a strictly convex function, it's integer minimum solution is obtained either for $n=\lfloor n^*\rfloor$ or $n=\lceil n^*\rceil$, provided $\lceil n^*\rceil<n_{infeasible}$. So, the minimum feasible value of $M(n)$, for integer n, is given by $min(M(\lfloor n^*\rfloor), M(\lceil n^*\rceil))$ if $s_1>0$ for $n=\lceil n^*\rceil$, or else, it is given by $M(\lfloor n^*\rfloor)$. The value of s_1 can be calculated using (2.15) for q=1. Let the value of n so obtained be denoted by n^{**}. By Proposition 2.1 and the strict convexity of $M(n)$ for $q=1$, the optimal number of sublots is given by $n_{opt}=min(N,n^{**})$.

Case $q\neq 1$

We show the desired result that M(n) is a strongly quasi-convex function of n in Proposition 2.6. We prove this result by showing the following two facts: (i) there exists at most two values of n for which $\frac{\partial M(n)}{\partial n}=0$, which we show in Proposition 2.4, and (ii) once $\frac{\partial M(n)}{\partial n}$ turns positive for some n', it stays positive for all $n>n'$, which we show in Proposition 2.5.

First differentiate (2.18) with respect to n. We have

$$\frac{\partial M(n)}{\partial n}=\tau_a+\left[\frac{\left({}^{1}/_{q^n}-1\right)\left({}^{-p_bT}/_{q}\right)+\left({}^{p_bU}/_{q}-p_bU-{}^{p_bnT}/_{q}\right)\frac{ln(q)}{q^n}}{\left({}^{1}/_{q^n}-1\right)^2}\right]. \tag{2.21}$$

By setting Expression (2.21) to zero and rearranging, we have:

$$n=\frac{q}{p_bT}\left[\frac{\tau_a\left(1-q^n\right)^2}{q^n}-\frac{\left(1-q^n\right)p_bT}{q}\right]\frac{1}{ln\,q}-\frac{q}{T}\left(U-\frac{U}{q}\right). \tag{2.22}$$

The following result follows immediately from this expression.

Proposition 2.4. *There exists at most two values of n for which* $\frac{\partial M(n)}{\partial n} = 0$.

Proof. Consider the second derivative with respect to n of the right side of (2.22), given by:$\frac{\tau_a q^{n+1} \ln q}{p_b T} \left(\frac{\tau_b}{\tau_a} + \frac{1}{q^{2n}} \right)$. Note that, it has the same sign as that of $ln(q)$ for all n. Therefore, the right side of (2.22) is either a strictly convex or a strictly concave function. Since a strictly convex/concave function can intersect a straight line (given by the left side of (2.22)) at no more than two points, the result follows. □

Next, we study the characteristics of $\left(\frac{\partial M(n)}{\partial n} \right)$ (Expression (2.21)), which help in determining the value of n for which the makespan is minimized.

By rearranging the Expression for $\left(\frac{\partial M(n)}{\partial n} \right)$ and expressing it in terms of s_1, we have:

$$\frac{\partial M(n)}{\partial n} = \tau_a + \frac{p_b ln(q)\ (q^n - 1)}{q^{n+1} \left(1 - \frac{1}{q^n}\right)^2} \left(\frac{T(ln(q) + 1 - q)}{(1-q)ln(q)} - s_1 \right). \tag{2.23}$$

Remark 2.4. (i) The term $\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}$ is monotone decreasing and is always positive for $q \neq 1$.

This follows by the fact that, for $q \neq 1$, $\frac{\partial}{\partial n} \left(\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2} \right) = \frac{-p_b (ln(q))^2}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}$, which is always negative. Also, $ln(q)$ and $(q^n - 1)$ are of the same sign, and therefore, $\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}$ is always positive.

(ii) The terms $(ln(q) + 1 - q)$ and $(1-q)ln(q)$ are negative for all values of $q \neq 1$.

Remark 2.5. In view of Proposition 2.2, since s_1 becomes negative for large values of n, $\frac{\partial M(n)}{\partial n} > 0$ as $n \to \infty$.

Proposition 2.5. *If there exists a value of* $n = n'$ *for which* $\frac{\partial M(n')}{\partial n} > 0$, *then* $\frac{\partial M(n)}{\partial n} > 0$ *for all* $n > n'$.

Proof. This can be shown by using Remark 2.4 and Proposition 2.2 and considering the cases $\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n) \right) < 0$ and $\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n) \right) \geq 0$.

By assumption, $\frac{\partial M(n)}{\partial n} > 0$ for $n = n'$. Then, for any $n > n'$, we have

$\left(\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2} \right) < \left(\frac{p_b ln(q)\left(q^{n'}-1\right)}{q^{n'+1}\left(1-\frac{1}{q^{n'}}\right)^2} \right)$ (by Remark 2.4(i)).

Also, $s_1(n) < s_1(n')$ by Proposition 2.2. Therefore,

$\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n)\right) > \left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n')\right)$.

Now, if $\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n)\right) \geq 0$, then $\frac{\partial M}{\partial n}(n)$ cannot be negative since

$\left(\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}\right) > 0$ from above and $\tau_a > 0$. Therefore, the result holds true. On the other hand,

if $\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n)\right) < 0$, then

$\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n')\right) < \left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n)\right) < 0$ since

$s_1(n') < s_1(n)$.

Therefore,

$$abs\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n)\right) < abs\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1(n')\right) \tag{2.24}$$

Also, $0 < \left(\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}\right) < \left(\frac{p_b ln(q)\left(q^{n'}-1\right)}{q^{n'+1}\left(1-\frac{1}{q^{n'}}\right)^2}\right)$.

For this case, we can rewrite (2.23) as follows:

$$\frac{\partial M(n')}{\partial n} = \tau_a - \frac{p_b ln(q)\ \left(q^{n'}-1\right)}{q^{n'+1}\left(1-\frac{1}{q^{n'}}\right)^2} abs\left(\frac{T(ln(q)+1-q)}{(1-q)ln(q)} - s_1\right) \tag{2.25}$$

By (2.25), and the fact that $\left(\frac{p_b ln(q)\,(q^n-1)}{q^{n+1}\left(1-\frac{1}{q^n}\right)^2}\right)$ is monotone decreasing in n and positive (from above), the result holds true for this case as well. □

We are now ready to show the following result.

Proposition 2.6. *The makespan M(n) is a strongly quasi-convex function of n when* $q \neq 1$.

Proof. Let $n_1 < n' < n_2$. We have

$M(n_2) = M(n_1) + \int_{n_1}^{n_2} \frac{\partial M(n)}{\partial n} dn$.

Consider the following three possible cases:

(i) $\frac{\partial M(n_1)}{\partial n} > 0$. Then, $\frac{\partial M(n)}{\partial n} > 0, \forall n > n_1$ (by Proposition 2.5). Hence, $\int_{n_1}^{n_2} \frac{\partial M(n)}{\partial n} dn > \int_{n_1}^{n'} \frac{\partial M(n)}{\partial n} dn$. Therefore, $M(n') < M(n_2) \leq max(M(n_2), M(n_1))$.

(ii) $\frac{\partial M(n_2)}{\partial n} < 0$. Then $\frac{\partial M(n_1)}{\partial n} \leq 0$ (by Proposition 2.5). By Proposition 2.4 and Remark 2.5, $\frac{\partial M(n)}{\partial n} = 0$ at most at one point in the interval $[n_1, n_2]$ i.e., $\frac{\partial M(n)}{\partial n} < 0$ almost everywhere in the interval $[n_1, n_2]$. Therefore, $\int_{n_1}^{n_2} \frac{\partial M(n)}{\partial n} dn < \int_{n_1}^{n'} \frac{\partial M(n)}{\partial n} dn < 0$. Hence, $M(n') < M(n_1) \leq max(M(n_1), M(n_2))$.

(iii) $\frac{\partial M(n_2)}{\partial n} > 0$ and $\frac{\partial M(n_1)}{\partial n} < 0$. By Proposition 2.5, there exists one point n_{opt} such that $\frac{\partial M(n)}{\partial n} < 0 \, for \, n \in [n_1, n_{opt}]$, almost everywhere, and $\frac{\partial M(n)}{\partial n} > 0 \, for \, n \in [n_{opt}, n_2]$. If $n' \leq n_{opt}$, then $\int_{n_1}^{n'} \frac{\partial M(n)}{\partial n} dn < 0$. Hence, $M(n_1) > M(n_1) + \int_{n_1}^{n'} \frac{\partial M(n)}{\partial n} dn = M(n')$. If $n' > n_{opt}$, then $\int_{n'}^{n_2} \frac{\partial M(n)}{\partial n} dn > 0$. Therefore, $M(n_2) > M(n_2) - \int_{n'}^{n_2} \frac{\partial M(n)}{\partial n} dn = M(n')$. Hence, $M(n') < max(M(n_1), M(n_2))$.

Since $M(n') < max(M(n_1), max(M(n_2))$ is true for all possible cases, the makespan function is strongly quasi-convex and has a unique minimum. □

It follows that if $\frac{\partial M}{\partial n} > 0$ at $n = 1$, then lot splitting is not a viable option; otherwise, $M(n)$ has a unique minima, which is obtained at n^*. It may also have an inflexion point before this minima. It remains to be shown that $\lfloor n^* \rfloor < n_{infeasible}$. We do this next.

Proposition 2.7. *The number of sublots $n = \lfloor n^* \rfloor$ gives a feasible compact schedule when $q \neq 1$.*

Proof. By Remark (2.2), the values of s_k, $k = 2, \cdots, n$, are positive if s_1 is positive. Therefore, it is sufficient to prove that $\frac{\partial M(n)}{\partial n}$ cannot be zero when s_1 is negative in order to ensure that the solution obtained from (2.22) and the method described above, is feasible.

Consider Expression (2.23). In view of Remark 2.4 (ii), and the fact that at least one of τ_a and T is positive by assumption, $\frac{\partial M}{\partial n}$ takes on a positive value when s_1 is negative. However, M(n) is strongly quasi-convex and $\lfloor n^* \rfloor < n^*$. Therefore, $\frac{\partial M(\lfloor n^* \rfloor)}{\partial n} \leq 0$, which, by Expression (2.23) and Remark 2.4, implies that $s_1 > 0$ for $n = \lfloor n^* \rfloor$. Hence, the solution generated is feasible. □

We have the following solution procedure to obtain an optimal schedule:

Step 1 If $\frac{\partial M}{\partial n} > 0$ for $n = 1$, set $n_{opt} = 1$ and go to Step 3. Otherwise, obtain $\min(\lfloor n^* \rfloor, N)$ by evaluating $M(n)$ for $n = 1, \cdots, N$.

Step 2 If $\lfloor n^* \rfloor < N$, evaluate s_1 for $\lfloor n^* \rfloor + 1$. If $s_1 > 0$ and $M(\lfloor n^* \rfloor + 1) < M(\lfloor n^* \rfloor)$, set $n_{opt} = \lfloor n^* \rfloor + 1$, otherwise set $n_{opt} = \lfloor n^* \rfloor$. If $\lfloor n^* \rfloor \not< N$ set $n_{opt} = N$.

Step 3 Determine sublot sizes for the compact schedule consisting of n_{opt} sublots using Expression (2.15) and (2.12).

In order to analyze the complexity of the algorithm, consider step 1. In case $\frac{\partial M}{\partial n} > 0$ for $n = 1$, it takes a fixed amount of time. Otherwise, for the case of $q = 1$, the value of n^* is given by Expression 2.20, and step 1 can again be computed in fixed time. If $q \neq 1$, we have, $\frac{\partial M(\lfloor n^* \rfloor)}{\partial n} \leq 0$ and $\frac{\partial M(\lfloor n^* \rfloor + 1)}{\partial n} = \frac{\partial M(\lceil n^* \rceil)}{\partial n} > 0$, since $M(n)$ is quasi-convex. Therefore, $min\left(\lfloor n^* \rfloor, N\right)$ can be found by using a binary search over $1, \cdots, N$ in $O(\log N)$ time. Step 2 requires a fixed number of computations. Step 3 involves n_{opt} evaluations of closed-form expressions. Since $n_{opt} \leq N$, this step can be completed in at most $O(N)$ time. Therefore, the complexity of the solution procedure is $O(N)$. Note that this is equal to the complexity of the solution procedure presented by Potts and Baker (1989a) for the case without sublot-attached setup times, since their algorithm consists of evaluating step 3, with $n_{opt} = N$ and $\tau_a = \tau_b = 0$. As mentioned in Trietsch and Baker (1993a), this constitutes a polynomial-time algorithm.

We illustrate the above procedure by using an example problem.

Example 2.1. Consider the following data: $p_a = 3.1$, $p_b = 3.1$, $\tau_a = 1$, $\tau_b = 4$, and $U = 10$. We assume $N = 10$. Note that $p_b = p_a$, and therefore, $q = 1$. By (2.20), $n^* = 3.52$. By (2.15), we have $s_1 = 1.05 > 0$ for $\lceil n^* \rceil = 4$, which implies that a compact schedule is feasible when $\lceil n^* \rceil = 4$ sublots are used. Computing the makespan values for $\lceil n^* \rceil = 4$ and $\lfloor n^* \rfloor = 3$ using (2.19), we have, $M(3) = 51.33$ and $M(4) = 51.25$. Therefore, the number of sublots that yields the minimum makespan, $n_{opt} = 4$. The optimal sublot sizes along with their start times and completion times are provided in Table 2.1. We also considered the values of τ_b equal to 8 and 16. The corresponding solutions are also presented in Table 2.1. Note that, as the value of τ_b increases, the optimal number of sublots decreases.

2.4 Impact of Learning

When the processing times and setup times are influenced by learning, they decrease with experience. As mentioned earlier, learning curves have been described by the mathematical function, $Y = KX^{-d}$, where Y is the production time per unit, K is the time required to produce the first unit, X is the cumulative number of units produced, and d is a constant, with value less than one, and it determines the learning rate ((Baloff, 1971),(Yelle, 1979)). Since the production lot is assumed to be continuous, we use the continuous relaxation of this function and use S, the cumulative completed work content, in place of X. Thus, the unit processing time after having completed total work content S on a machine is given by:

$$p = p_0 S^{-d}, \tag{2.26}$$

where p_0 is the initial processing time. Similarly, for the impact of learning on the setup time, if τ_0 denotes the initial setup time, and $d'(< 1)$ represents it's learning constant, then the setup time after n setups is given by:

Table 2.1: Optimal schedules for Example 2.1

Values of τ_b, n_{opt}	Sublot number	Sublot size	Machine 1		Machine 2	
			Start time	Completion time	Start time	Completion time
$\tau_b = 4$ $n_{opt} = 4$	1	1.05	0	4.25	4.25	11.5
	2	2.02	4.25	11.5	11.5	21.75
	3	2.98	11.5	21.75	21.75	35
	4	3.95	21.75	35	35	**51.25**
$\tau_b = 8$ $n_{opt} = 3$	1	1.08	0	4.33	4.33	15.67
	2	3.33	4.33	15.67	15.67	34
	3	5.59	15.67	34	34	**59.33**
$\tau_b = 16$ $n_{opt} = 2$	1	2.58	0	9	9	33
	2	7.42	9	33	33	**72**

$$\tau = \tau_0 n^{-d'}. \tag{2.27}$$

2.4.1 Learning in processing time

First, we examine the case of learning in processing times. Let $\mathcal{S}_{(U,\tau_a,\tau_b,p_a,p_b,N)}$ and $\mathcal{S}'_{(U,\tau_a,\tau_b,p_a,p_b,N,d)}$ represent the set of feasible schedules for the case without learning and with learning in processing times, respectively. We show that for each schedule $s' \in \mathcal{S}'_{(U,\tau_a,\tau_b,p_a,p_b,N,d)}$, there exists a schedule $s \in \mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ with an equal makespan, and vice versa. Therefore, this problem can be transformed to an equivalent problem involving constant processing times, and thus, can be solved by using the method outlined in Section 2.3. The setup times and the upper bound on the number of sublots, N, are the same for both the original and the transformed problems. Recall that, by the reversibility property (Property 2.2.2), for the case $\tau_b < \tau_a$, it is possible to search for the number of sublots at which the makespan is minimized for the L-SAS problem in $\mathcal{S}_{(\frac{U^{1-d}}{1-d},\tau_\mathbf{b},\tau_\mathbf{a},p_a,p_b,N)}$ instead, and then transform the resulting optimal solution to a schedule in $\mathcal{S}_{(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N)}$ (this simply requires reversing the order in which the sublots are numbered). We consider the simultaneous impact of learning on processing and setup times in Section 2.4.3.

Let the sublot sizes for the case with learning in processing times be represented by $u_1, u_2, \cdots, u_n$. Then, the time taken to process the i^{th} sublot is given by:

$$\int_{u_1+u_2+\cdots+u_{i-1}}^{u_1+u_2+\cdots+u_i} p_0 S^{-d} dS = p_0 \frac{U_i^{1-d} - U_{i-1}^{1-d}}{1-d}, \tag{2.28}$$

where

$$\begin{aligned} \sum_{k=1}^{i} u_k &= U_i \text{ and} \\ U_0 &= 0. \end{aligned}$$

Now, we find a schedule in $\mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ that has an equal makespan. Consider a schedule in $\mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ with the same number of sublots, n, and sublot sizes given by s_k, such that:

$$s_1 = \frac{U_1^{1-d}}{1-d},$$

and

$$s_k = \frac{U_k^{1-d} - U_{k-1}^{1-d}}{1-d}, \qquad k = 2, 3, \cdots n. \tag{2.29}$$

Since the schedules in $\mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ are not influenced by learning, the processing time of sublot k is $p_0 s_k$, $k = 1 \cdots, n$.

In the presence of learning, the processing time on either machine for sublot u_i is given by:

$$\int_{U_{i-1}}^{U_i} p_0 S^{-d} dS = p_0 \quad \frac{U_i^{1-d} - U_{i-1}^{1-d}}{1-d}.$$

Furthermore,

$$\begin{aligned} p_0 \frac{U_i^{1-d} - U_{i-1}^{1-d}}{1-d} &= p_0 \frac{\left(\left((1-d)\sum_{k=1}^{i} s_k\right)^{1/1-d}\right)^{1-d} - \left(\left((1-d)\sum_{k=1}^{i-1} s_k\right)^{1/1-d}\right)^{1-d}}{1-d} \\ &= p_0 \frac{\left(\left((1-d)\sum_{k=1}^{i} s_k\right)\right) - \left(\left((1-d)\sum_{k=1}^{i-1} s_k\right)\right)}{1-d} \\ &= p_0 (\sum_{k=1}^{i} s_k - \sum_{k=1}^{i-1} s_k) \\ &= p_0 s_i. \end{aligned}$$

Therefore, the processing time of the i^{th} sublot, u_i, for the case with learning is equal to the processing time of the i^{th} sublot, of size s_i, for the case without learning. Since sublot attached setup times are also equal for both cases, the makespan for both the schedules are equal. Also note that $\sum_{i=1}^{n} s_i = \frac{U_1^{1-d}}{1-d} + \sum_{i=1}^{n} \frac{U_i^{1-d} - U_{i-1}^{1-d}}{1-d} = \frac{U^{1-d}}{1-d}$. Therefore, for each schedule $s \in \mathcal{S}'_{(U,\tau_a,\tau_b,p_a,p_b,N,d)}$, there exists a schedule $s' \in \mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ with an equal makespan. The converse holds true since the mapping of $u_1,\ u_2,\ \cdots,\ u_n$ to $s_1,\ s_2,\ \cdots,\ s_n$ (2.29) is clearly bijective. Therefore, the problem of finding an optimal schedule for the case when there exists learning in processing times can be solved by first finding an optimal schedule for the case without learning using the lot size of $\frac{U^{1-d}}{1-d}$, and then, transforming the optimal solution so obtained back to the case with learning. We next show how such a transformation of the optimal solution is performed. Suppose this problem yields n sublots of sizes $s_1, s_2, \cdots, s_n$. Then, the cumulative lot sizes for the original problem can be given by:

$$U_i = \left((1-d) \sum_{k=1}^{i} s_k \right)^{1/1-d}, \ i = 1, \cdots, n. \tag{2.30}$$

and the actual sublot sizes by

$$\begin{aligned} u_1 &= U_1, \ \textit{and} \\ u_i &= U_i - U_{i-1}, \ i = 2, \cdots, n. \end{aligned} \tag{2.31}$$

Remark 2.6. Note that, in the absence of learning in processing times, $d = 0$. Therefore, $\frac{U^{1-d}}{1-d}$ reduces to U. Moreover, by (2.30) and (2.31), $u_k = s_k$, $k = 1, \cdots, n$, as expected.

We illustrate the above procedure by the following example.

Example 2.2. Consider the following data: $p_a = 3$, $p_b = 6$, $\tau_a = 4$, $\tau_b = 19$, $d = 0.312$ and $U = 80$. We assume $N = 80$.

The transformed lot size for the equivalent problem without learning is given by:

$$\begin{aligned} U' &= \frac{U^{1-d}}{1-d} \\ &= 29.63. \end{aligned}$$

The unit processing times and setup times for the transformed problem are the same as their initial values in the original problem. For the transformed problem, we obtain the optimal makespan of 242.91 for number of sublots, $n_{opt} = 3$. The optimal sublot sizes are $s_1 = 1.38,\ s_2 = 7.75$ and $s_3 = 20.5$.

We can, now, obtain cumulative sublot sizes for the original problem as follows: $U_1 = ((1-d)s_1)^{\frac{1}{1-d}} = 0.923$; $U_2 = ((1-d)(s_1+s_2))^{\frac{1}{1-d}} = 14.45$; and $U_3 = ((1-d)(s_1+s_2+s_3))^{\frac{1}{1-d}} = 80$. Consequently, the individual sublot sizes are given by: $u_1 = U_1 = 0.923$; $u_2 = U_2 - U_1 = 13.524$; and $u_3 = U_3 - U_2 = 65.553$ with the makespan value the same as that obtained for the transformed problem.

Note that schedules are no longer reversible in the presence of learning. For illustration, consider Example 3.2 with the direction of material flow reversed. This results in the swapping of the values of τ_a and τ_b and the values of p_a and p_b, i.e. $p_a = 6$, $p_b = 3$, $\tau_a = 19$ and $\tau_b = 4$ with all other values unchanged. The optimal solution for the corresponding problem without learning is given by $s_1 = 20.5$, $s_2 = 7.75$, $s_3 = 1.38$ (this is simply the reverse ordering of the sublots from the previous solution). Transforming to a schedule in $\mathcal{S}'_{(U,\tau_a,\tau_b,p_a,p_b,N,d)}$, we have, $U_1 = ((1-d)s_1)^{\frac{1}{1-d}} = 46.83$; $U_2 = ((1-d)(s_1+s_2))^{\frac{1}{1-d}} = 74.64$; and $U_3 = ((1-d)(s_1+s_2+s_3))^{\frac{1}{1-d}} = 80$. Therefore, the optimal solution to the problem with learning is $u_1 = U_1 = 46.83$; $u_2 = U_2 - U_1 = 27.81$; and $u_3 = U_3 - U_2 = 5.36$. These sublot sizes are <u>not</u> the same as those obtained in the previous example; however, the optimal makespan value is the same for both cases.

The complexity of this algorithm is identical to that for the problem without learning since the transformation into a problem in $\mathcal{S}_{\left(\frac{U^{1-d}}{1-d},\tau_a,\tau_b,p_a,p_b,N\right)}$ requires a fixed computation, and the reverse transformation to the original problem requires a sequential evaluation of Expressions (2.30) and (2.31), each requiring $O(N)$ time.

2.4.2 Learning in setup times

Next, we consider the case of learning in setup times. Let τ_a^i and τ_b^i be the sublot-attached setup times for the i^{th} sublot on machines 1 and 2, respectively. For the learning constant of d',we have,

$$\tau_a^i = \tau_a i^{-d'}, \text{ and} \tag{2.32}$$

$$\tau_b^i = \tau_b i^{-d'}. \tag{2.33}$$

First, we show that there exists an optimal schedule that is a compact schedule. Then, we obtain sublot sizes for compact schedules and show that they are feasible (non-negative) if and only if the number of sublots is less than a value, $n_{infeasible}$. This is followed by presentation of a method to obtain the value of n at which the makespan is minimized.

Properties of an optimal solution

Similar to the case without learning, any feasible schedule consists of $n \in \{1, 2, \cdots, N\}$ sublots with sublot sizes given by s_k, $k = 1, \cdots, n$ such that $s_1 = U$ if $n = 1$, and $U > s_k > 0$ such that $\sum_{k=1}^{n} s_k = U$ if $n \geq 2$. Clearly, postponing the processing of any sublot on machine 1 can not improve the makespan, hence, we assume that there exists no idle time between the processing of the sublots on machine 1. As before, we denote the makespan obtained for such a schedule by $M(n, s_1, \cdots, s_n)$, and the completion time of sublot j on machine 2 by C_j. We have, $C_2 = (\tau_b^2 + p_b s_2) + max(\sum_{k=1}^{2}(\tau_a^k + p_a s_k), (\tau_a^1 + p_a s_1) + (\tau_b^1 + p_b s_1))$ and. $C_j = (\tau_b^j + p_b s_j) + max(\sum_{k=1}^{j}(\tau_a^k + p_b s_k), C_{j-1})\ \ \forall j = 3, \cdots, n$. Therefore, the makespan $M(n, s_1, \cdots, s_n)$ is given by the following expression:

$$M(n, s_1, \cdots, s_n) = \tau_b^n + p_b s_n + max(\sum_{k=1}^{n}(p_a s_k + \tau_a^k), C_{n-1}). \tag{2.34}$$

Clearly, Remark 2.1 applies for (2.34) as well.

Proposition 2.8. *There exists an optimal compact schedule with $n_{opt} \in \{1, \cdots, N\}$ sublots for the L-SASLS problem.*

Proof. The proof of this proposition follows along the arguments used in the proof of Proposition 2.1 except that the setup time terms in the makespan expressions are now $\sum_{i=1}^{k} \tau_a^i$ instead of $k\tau_a$ for machine 1 and $\sum_{i=k}^{n} \tau_b^i$ instead of $(n - k + 1)\tau_b$ for machine 2. □

Determination of sublot sizes

For a compact schedule with n sublots, the makespan M is given by the following expression:

$$M(n) = [\tau_a^1 + p_a s_1] + [\sum_{1}^{n} \tau_b^i + p_b U], \tag{2.35}$$

where s_i is the size of sublot i. We also use variable $s_i(n)$ to denote the size of sublot i given n sublots in a compact schedule. In the presence of learning in setup times, a compact schedule is obtained if

$$\tau_a^i + p_a s_i = \tau_b^{i-1} + p_b s_{i-1}\ , i = 2, \cdots, n.$$

If we define T_i as

$$T_i = \frac{\tau_b^{i-1} - \tau_a^i}{p_a},\ i = 2, \cdots, n, \tag{2.36}$$

then we have

$$s_i = qs_{i-1} + T_i, \quad i = 2, 3, \cdots, n. \tag{2.37}$$

Remark 2.7. If $T_i < 0$, then $T_j < 0 \; \forall \; j > i$.

Proof. This follows by the fact that the learning constant is the same for both the machines. By (2.36),(2.32) and (2.33), we can express T_i as

$$T_i = \frac{\tau_a (i-1)^{-d}}{p_a} \left(\frac{\tau_b}{\tau_a} - \left(1 - \frac{1}{i}\right)^d \right) \; i = 2, 3, \cdots, n. \tag{2.38}$$

Note that, $\frac{\tau_a (i-1)^{-d}}{p_a}$ is always positive. For $i = 2, 3, \ldots, n$, the function $\left(1 - \frac{1}{i}\right)^d$ is monotone increasing. Therefore, if $T_i < 0$, then $\left(1 - \frac{1}{i}\right)^d > \frac{\tau_b}{\tau_a}$. This implies that $\left(1 - \frac{1}{i'}\right)^d > \frac{\tau_b}{\tau_a} \; \forall i' > i$, and the result follows. □

To determine sublot sizes, we proceed as follows. First, we determine s_1. By (2.37), we have

$$\begin{array}{lcccccc} s_1 & = & s_1 & & & & \\ s_2 & = & qs_1 + & T_2 & & & \\ s_3 & = & q^2 s_1 + & qT_2 + & T_3 & & \\ \vdots & & & & & & \\ s_n & = & q^{n-1} s_1 + & q^{n-2} T_2 + & q^{n-3} T_3 & \cdots & \end{array} \tag{2.39}$$

Adding all rows in (2.39) and using (2.14), we have,

$$s_1 = \begin{cases} \frac{1-q}{1-q^n}\left(U - \sum_{i=2}^{n} \frac{1-q^{n-i+1}}{1-q} T_i\right), & q \neq 1 \\ \frac{1}{n}\left(U - \sum_{i=2}^{n} (n-i+1) T_i\right), & q = 1 \end{cases}. \tag{2.40}$$

Consequently, $s_k \; k = 2, \cdots, n$, can be calculated by (2.37).

Remark 2.8. Expressions (2.40) and (2.37) reduce to Expressions (2.15) and (2.12), respectively, if T is constant, and are therefore, further generalizations of those expressions.

However, we need to add conditions for the feasibility of the solution given by (2.40) and (2.37). Consider the following result.

Proposition 2.9. *If $s_1 > 0$ and $s_n > 0$, then $s_k > 0, \quad \forall k = 1, \cdots, n$.*

Proof. We prove this result by contradiction. Suppose there exists a sublot s_i such that $s_i < 0$ and $s_{i-1} > 0$.

Since

$$s_i = qs_{i-1} + T_i \ ,$$

we must have,

$$T_i < 0 \ .$$

Now, because $s_n > 0$, there must exist a j, $i < j \leq n$, such that $s_j > 0$ when $s_{j-1} < 0$.

Since

$$s_j = qs_{j-1} + T_j \ ,$$

it implies that

$$T_j > 0.$$

But, this is not possible because of $j > i$, $T_i < 0$, and Remark 2.7 . □

Next, we show the optimality of a compact schedule for the case of learning in setup times.

Proposition 2.10. *If a compact schedule is not feasible for $n_{infeasible}$ sublots, then all compact schedules containing $n \geq n_{infeasible}$ sublots will also be infeasible.*

Proof. For a compact schedule to be infeasible, the size of either the first sublot s_1 or the last sublot s_n must be negative, by Proposition 2.9. We use this fact to prove the result.

First, consider the case of s_n being negative when n sublots are used. To prove the result, it is enough to show that if $s_n < 0$, then $s_{n+1} < 0$. By (2.37), we have

$$\begin{array}{rcllllll} s_n & = & s_n & & & & & \\ s_{n-1} & = & \frac{1}{q}s_n - & \frac{1}{q}T_n & & & & \\ s_{n-2} & = & \frac{1}{q^2}s_n - & \frac{1}{q^2}T_n - & \frac{1}{q}T_{n-1} & & & \\ \vdots & & & & & & & \\ s_1 & = & \frac{1}{q^{n-1}}s_n - & \frac{1}{q^{n-1}}T_n - & \frac{1}{q^{n-2}}T_{n-1} & \cdots & -\frac{1}{q}T_2 \end{array} \ . \tag{2.41}$$

Adding all the rows in Expression (2.41) and using (2.14) , we have after rearranging,

$$s_n\left(1+\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^{n-1}}\right) = U + T_2\frac{1}{q} + T_3\left(\frac{1}{q}+\frac{1}{q^2}\right)+\cdots+T_n\left(\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^{n-1}}\right). \tag{2.42}$$

If $s_n(n)$ is negative, then the right side of the above equation is also negative. Therefore, at least a T_i, $2 \leq i \leq n$, must be negative. By Remark 2.7, this implies that $T_{n'} < 0$ for all $n' > i$..

For $n+1$ sublots, the size of the last sublot can be expressed as:

$$s_{n+1}(\text{n+1})\left(1+\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^n}\right) = U + T_2\frac{1}{q}+\cdots+T_{n+1}\left(\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^{n-1}}+\frac{1}{q^n}\right),$$

Substituting $s_n(\text{n})$ in the above equation and simplifying,

$$s_{n+1}(\text{n+1})\frac{\left(1+\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^n}\right)}{\left(1+\frac{1}{q}+\frac{1}{q^2}+\cdots+\frac{1}{q^{n-1}}\right)} = s_n(\text{n}) + \frac{T_{n+1}}{q}, \tag{2.43}$$

If $s_n(n)$ is negative, then T_{n+1} is also negative (from above), and therefore, $s_{n+1}(n+1)$ is negative. Hence, the schedule is infeasible for $n+1$ sublots.

Next, consider the case when s_1 is negative. Rewriting (2.41)

for compact schedules with n and $n+1$ sublots, we get,

$$\begin{aligned} U &= (1+q+\cdots+q^{n-1})s_1(n) + (1+q+\cdots+q^{n-2})T_1+\cdots+T_n, \text{and} && (2.44)\\ U &= (1+q+\cdots+q^{n-1}+q^n)s_1(n+1) + (1+q+\cdots+q^{n-2}+q^{n-1})T_1+\cdots \\ &\quad \cdots+(1+q)T_n+T_{n+1} && (2.45)\end{aligned}$$

Subtracting (2.44) from (2.45) and rearranging, we have,

$$(1+q+\cdots+q^{n-1})(s_1(n+1)-s_1(n)) = -(q^n s_1(n+1)+q^{n-1}T_1+\cdots+T_{n+1}) \tag{2.46}$$

Using $n+1$ instead of n in (2.39), and evaluating $s_{n+1}(n+1)$ we get,

$$s_{n+1}(n+1) = q^n s_1(n+1)+q^{n-1}T_1+\cdots+T_{n+1} \tag{2.47}$$

Substituting (2.47) in (2.46) and rearranging, we can find the change in the size of the first sublot, s_1, when an additional sublot is used, given by:

$$s_1(n+1)-s_1(n) = \begin{cases} -\left(\frac{1-q}{1-q^n}\right)s_{n+1}(n+1) & q\neq 1 \\ -\frac{1}{n}s_{n+1}(n+1) & q=1 \end{cases} \tag{2.48}$$

If $s_1(n+1) < 0$, then the schedule remains infeasible. On the other hand, if the value of $s_1(\text{n})$ is negative and the value of $s_1(\text{n+1})$ is positive, the left side of the above Expression (2.48) is positive. Therefore, $s_{n+1}(\text{n+1}) < 0$, which implies (from above) that $s_{n'}(\text{n'}) < 0 \quad \forall n' \geq n+1$. So, compact schedules remain infeasible even if the value of the first sublot s_1 becomes positive. □

Solution procedure

In this section, we find the number of sublots, n_{opt} that minimizes makespan. By Proposition 2.8, we need to search over only compact schedules. An upper bound on n_{opt} is N.

Clearly, a feasible compact schedule exists for $n = 1$. Hence, in view of Proposition 2.10, to determine n_{opt}, we evaluate the makespan M(n) for compact schedules and search over n in increments of 1, starting from $n = 1$, until the first value of n (call it $n_{infeasible}$) is obtained for which the compact schedule is infeasible. If no such value is reached, we stop at $n = N$. Therefore, the optimal value of $n_{opt} \in \{1, \ldots, min(N, n_{infeasible} - 1)\}$. We have the following solution procedure to determine n_{opt} and optimal sublot sizes:

Step 1 Set the number of sublots $n = 1$, $s_1(n) = U$, $s_n(n) = U$, and evaluate the makespan, $M(n)$, using Expression (2.35).

Step 2 Increment the number of sublots. Evaluate the last sublot $s_n(n)$ using (2.43). If it is negative, go to Step 6; otherwise, go to Step 3.

Step 3 Evaluate the size of the first sublot $s_1(n)$ using (2.48). If this is negative, go to Step 6; otherwise, go to Step 4.

Step 4 Find the makespan using Expression (2.35).

Step 5 If the number of sublots $n < N$, go to Step 2; otherwise, go to Step 6.

Step 6 Find the optimal number of sublots n_{opt}, which gives the minimum makespan value.

Step 7 Determine sublot sizes for the corresponding compact schedule consisting of n_{opt} sublots using Expression (2.37).

Note that, the evaluation of an expression is required in each of the Steps 2, 3 and 4, and these steps are executed at most N times. Moreover, Step 6 requires at most N comparisons, and Step 7 needs evaluation of at most $N-2$ sublots. Therefore, the complexity of the algorithm is $O(N)$, which is the same as the complexity for the problem without learning. As an illustration of the solution procedure, consider the following example.

Example 2.3. Let the lot size, $U = 20$; learning constant, $d' = 0.322$; unit processing times on machines 1 and 2, $p_a = 1$,$p_b = 1.1$; an upper bound on the number of sublots, $N = 20$; and setup times of the first setups on machines 1 and 2 ,$\tau_a = 3$, $\tau_b = 1$.

The makespan value for different number of sublots, n, is shown in Table 2.2. Note that a local minima is obtained when $n = 4$. However, the search is continued since the sublot sizes are still feasible. The makespan increases with increment in number of sublots. The last feasible solution is obtained when $n = 7$. Compact schedules are infeasible for $n > 7$. Therefore, thel number of sublots at which the minimum makespan is obtained is 4.

Table 2.2: Variation of makespan with number of sublots for Example 2.3.

No. of sublots	Makespan	Size of first sublot	Size of last sublot	Comment
1	46	20	20	Increasing the number of sublots decreases the makespan.
2	37	10.19	9.81	
3	34.83	7.32	6.02	
4	**34.30**	6.16	3.85	A minimum makespan is reached.
5	34.39	5.65	2.35	
6	34.76	5.46	1.18	
7	35.27	5.43	0.21	
8	35.85	5.50	**-0.64**	A compact schedule is infeasible, and the search is terminated.

We conducted numerical experimentation to investigate the performance of the above procedure and to determine the impact of learning on the solution obtained. We generated the input data as follows. The processing times were varied from 1 to 9 in steps of 1, and the sublot-attached setup times were varied between 1 and 100 in steps of 6 to obtain $9 \times 9 \times 17 \times 17 = 23,409$ different instances of data. We used three lot sizes, namely, $U = 1, 5$ and 10 and an upper bound on the number of sublots $N = 1, 5$ and 10, respectively. The results are shown in Table 2.3. The average CPU time required to solve an instance is 0.00006 seconds. The time required to solve all of these instances decreased slightly with increment in the value of the learning constant, d', but increased with increment in the size of a lot, U. For larger lot sizes, the maximum feasible number of sublots is often much larger than the n_{opt}. An upper bound on n_{opt} is given by $\hat{N} = min\,(N, n_{infeasible} - 1)$. Some sample solutions to various instances of data are provided in Table 2.4 for $d' = 0.322,\ U = 10$ and $N = 10$. The value of $\hat{N}$ is shown in the last column.

Table 2.3: Total CPU times (in secs) required for 23,409 test cases when d' and U are varied (for Example 2.3).

Learning Constant d'	Lot Size $U = 1$	Lot Size $U = 5$	Lot Size $U = 10$
0.1	1.134	1.409	1.61
0.322	1.133	1.26	1.492
0.7	1.123	1.211	1.461

Table 2.4: Optimal solutions for $d' = 0.322$, $U = 10$, and different values of p_a, p_b, τ_a *and* τ_b.

p_a	p_b	τ_a	τ_b	n_{opt}	$s_1(n)$	$s_n(n)$	$M(n)$	$\hat{N}$
1	1	1	1	4	2.29	2.65	16.43	10
3	3	1	1	8	1.14	1.3	39.76	10
7	5	19	7	2	6.52	3.48	127.22	3
8	3	85	85	1	10	10	280	10

2.4.3 Simultaneous consideration of the effects of learning in processing times and setup times

When the learning effect simultaneously exists both in the setup and processing times, the problem can be reduced to the case of learning in setup times only by using the transformation defined in (2.29) and the transformed lot size, $U' = \frac{U^{1-d}}{1-d}$. An upper bound on the number of sublots is N. The solution procedure outlined in Section 2.4.2 is then applied to the transformed problem with lot size U' in place of U. An example is provided below to illustrate the solution procedure.

Example 2.4. Consider the following problem data: $p_a = 4$; $p_b = 8$; $\tau_a = 7$; $\tau_b = 1$; learning constant for processing times, $d = 0.5$; learning constant for setup times $d' = 0.322$; $U = 10$, and $N = 10$.

The lot size ($U = 10$) is transformed to that for the equivalent problem (without learning in processing times),

$$\begin{aligned} U' &= \frac{U^{1-d}}{1-d} \\ &= 6.32\,. \end{aligned}$$

The minimum makespan value (65.65) is obtained for this data when $n = 4$. The sublot sizes obtained are: $s_1 = 1.23$, $s_2 = 1.3$, $s_3 = 1.58$, and $s_4 = 2.21$.

The sublot sizes for the original problem are obtained using (2.30) and (2.31).We have: $s_1 = 0.38$, $s_2 = 1.23$, $s_3 = 2.62$ and $s_4 = 5.78$.

To study the impact of learning, this problem was solved for different values of d and d'. The minimum makespan values for each of these cases is shown in Table 2.5. If a learning effect is present but is not considered when modeling a system, the sublot sizes and number of sublots so obtained will not be optimal; the resultant makespan will be larger than that for the true optimal solution. For the example on hand, these suboptimal makespan values are shown in Table 2.6. For instance, if $d = 0.6$ and $d' = 0.6$, the makespan value obtained by using a

constant model based on the initial processing and setup times is 72.41. This is in contrast to the makespan of 63.78, which would be obtained if learning effects were considered. Thus, inclusion of the effects of learning in analysis would result in a reduction of 13.5% in the makespan value, in this case. A similar effect is observed if we use standard times instead of initial times considered in Table 2.6. Standard processing and setup times can be obtained by multiplying initial unit processing and setup times by a correction factor of $1/(57-60\,2^{-d})$ and $1/(57-60\,2^{-d'})$, respectively (Dar-El (2000) and Dar-El *et al.* (1995a,b)). The results are shown in Table 2.7. When calculating standard processing and setup times, we assume the value of the correction factor to be 1 if $d = 0$, (or $d' = 0$). Note that, the use of both initial processing and setup times (Table 2.6) and standard processing and setup times (Table 2.7) result in significantly higher makespan values than the optimal makespan values (Table 2.5). Note that the computational complexity of the procedure described above is $O(N)$, since the complexity of optimizing the transformed problem is $O(N)$ and transformation of the problem to the case without learning requires $O(N)$ time, as shown in Section 2.4.1.

Table 2.5: Optimal makespan values (for different values of d and d' for Example 2.4).

Learning constants	Makespan value			
	d=0	d=0.15	d=0.322	d=0.6
d'=0	98.07	84.25	73.49	67.25
d'=0.15	96.96	83.22	72.46	66.28
d'=0.322	95.66	82.07	71.45	65.28
d'=0.6	93.87	80.39	69.85	63.78

Table 2.6: Makespan values when the optimal solution of a constant model is used in the presence of learning (for Example 2.4).

Learning constants	Makespan value			
	d=0	d=0.15	d=0.322	d=0.6
d'=0	98.07	85.26	75.89	73.79
d'=0.15	97.63	84.82	75.45	73.36
d'=0.322	97.21	84.41	75.04	72.94
d'=0.6	96.68	83.88	74.51	72.41

Table 2.7: Makespan values when the optimal solution of a constant model with standard setup and processing times is used in the presence of learning (for Example 2.4).

Learning constants	Makespan value			
	d=0	d=0.15	d=0.322	d=0.6
d'=0	98.07	96.37	92.32	83.36
d'=0.15	128	84.82	81.24	80.03
d'=0.322	114.07	87.94	75.04	75.44
d'=0.6	107.33	85.20	72	72.41

2.5 Determination of Integer Sublot Sizes

We adapt the method proposed by Trietsch and Baker (1993a) to determine integer sublot sizes for our problem by accommodating features of sublot-attached setup times and effect of learning. We present this method for the most general case when the effect of learning is present in both unit processing and setup times.

Given a schedule with n sublots, we denote the cumulative sublot size of k sublots by S_k $k = 1, \cdots n$, i.e., $S_k = s_1 + \cdots + s_k$. Clearly, $0 \leq S_k \leq U$. The total time required to process sublots $1, 2, \cdots k$ on machine 1 is given by $\sum_{i=1}^{k} \tau_a^i + p_a \left(\frac{S_k^{1-d}}{1-d}\right)$. The time required to process sublots $k, k+1, \cdots, n$ on machine 2 is given by $\sum_{i=k}^{n} \tau_b^i + p_b \left(\frac{U^{1-d} - S_{k-1}^{1-d}}{1-d}\right)$. For a makespan value of M to be feasible, we must have, $M - \sum_{i=k}^{n} \tau_b^i - p_b \left(\frac{U^{1-d} - S_{k-1}^{1-d}}{1-d}\right) \geq \sum_{i=1}^{k} \tau_a^i + p_a \left(\frac{S_k^{1-d}}{1-d}\right)$. Therefore, if a feasible schedule exists for a given value of M and S_{k-1}, we have

$$p_a \left(\frac{S_k^{1-d}}{1-d}\right) \leq M - \sum_{i=k}^{n} \tau_b^i - p_b \left(\frac{U^{1-d} - S_{k-1}^{1-d}}{1-d}\right) - \sum_{i=1}^{k} \tau_a^i, \tag{2.49}$$

and an upper bound on the value of S_k given by

$$min \left(U, \left(M - \sum_{i=k}^{n} \tau_b^i + p_b \left(\frac{U^{1-d} - S_{k-1}^{1-d}}{1-d}\right) - \sum_{i=1}^{k} \tau_a^i\right)^{\frac{1}{1-d}} \left(\frac{1-d}{p_a}\right)^{\frac{1}{1-d}}\right). \tag{2.50}$$

Note that the expression on the right side of (2.49) can take a negative value, which would imply that a feasible schedule does not exist for the values of M and S_{k-1} considered.

We set $S_0 = 0$, and use the upper bound (2.50) iteratively. If the right side of (2.49) becomes negative, we increase the value of M. Our method to obtain integer-sized sublots is as follows.

Step 1: Set M equal to the minimum makespan value obtained for the continuous version of the problem, and $S_0 = 0$.

Step 2: Set $k = 1$. If the right side of (2.49) is positive, assign S_k the largest integer value smaller than the upper bound obtained from (2.50); otherwise, set $S_k = S_{k-1}$ and increase the value of M by $\sum_{i=k}^{n} \tau_b^i + p_b \left(\frac{U^{1-d} - S_{k-1}^{1-d}}{1-d} \right) + \sum_{i=1}^{k} \tau_a^i - M$. If $S_k = U$, go to Step 5. Otherwise, check if $k < N$; if so, set $k = k+1$ and repeat, otherwise go to Step 3.

Step 3: Set $G = U - S_n$.

Step 4: Set $i = 1$. If $S_i - S_{i-1} = 0$ and $G > 0$, increase S_j by 1 for all $j = i, \cdots, n$ and decrease G by 1. If $G > 0$ and $i < n$, increase i by 1 and repeat.

Step 5: Delete all sublots which have zero size in the resulting schedule, and re-number the sublots. Let n' be the last sublot. If $S_{n'} = U$, stop.

Step 6: Select sublot k, $k \in \{1, \cdots, n\}$ such that incrementing s_k by 1 results in the smallest increase in the makespan value. Decrease G by 1. If $G > 0$ repeat; otherwise, stop.

Note that sublot sizes are always assigned integer values and, after Step 5, no sublot has size zero. Therefore, the above steps result in positive integer sublot sizes. The minimum makespan for the continuous sublot sizes serves as a lower bound for the optimal makespan of integer sublot sizes. If M^c and M^I denote the makespan values obtained for continuous and integer sublot sizes, respectively, then we define an upper bound on the optimality gap, OG, as follows: $OG = \frac{M^I - M^C}{M^C} \times 100$. We conducted a numerical experimentation to study the performance of the above heuristic using the following data: U was set to 100, the values of τ_a, τ_b, p_a, p_b were varied from 1 to 10 in steps of 1, and d, $d' \in \{0,\ 0.15,\ 0.322,\ 0.6\}$. The average optimality gap was determined over 10,000 observations for each case under different values of d and d'. These values are shown in Table 2.8. Note that the integer sublot sizes are almost optimal. The worst average optimality gap obtained is 3.83%.

Table 2.8: Average optimality gap (OG) for the proposed heuristic method.

Learning constants	Average OG over 10,000 observations			
	d=0	d=0.15	d=0.322	d=0.6
d'=0	0.35	0.42	0.54	1.52
d'=0.15	0.41	0.48	0.68	2.05
d'=0.322	0.45	0.57	0.89	2.74
d'=0.6	0.51	0.73	1.24	3.83

2.6 Concluding Remarks

In this Chapter, we have addressed the SLFS-LSP when a sublot-attached setup time is incurred every time a sublot is processed on the machines. In addition, we have considered the presence of learning in the processing and setup times. These constitute two new features that have not been simultaneously considered in the literature for the two-machine flowshop, unequal-sublot size, lot streaming problem. Our work has generalized some known results in view of these features. We have shown that the optimal schedule is a compact schedule, and have provided closed-form expressions to obtain optimal sublot sizes in the presence of both of these features when the sublot sizes are assumed to be continuous. We have developed simple search procedures to obtain the number of sublots that minimizes makespan. Also, we have presented a method to determine integer sublot sizes.

Chapter 3

Lot Streaming in the Presence of Learning

3.1 Introduction

In the previous chapter, we developed a transformation of a single-lot, two-stage flow shop lot streaming problem, where the production rates are affected by learning, to a corresponding problem with constant processing times. In this chapter, we generalize this transformation to other lot streaming problems. Since production lots usually consist of a large number of jobs, they are frequently assumed to be continuously divisible for analytical tractability. Such an assumption results in a good approximation for the discrete case (Bukchin *et al.*, 2002a; Shallcross, 1992). We will assume continuously divisible sublot sizes for the problems considered here. As mentioned in Chapter 1, in the lot streaming problems studied in the literature, processing times have typically been assumed to remain constant over time. However, when manual labor is used to perform tasks, or during the ramp-up stage of a manufacturing process, a worker's productivity increases with experience. This phenomenon is referred to as the influence of learning. The impact of learning is gaining importance in today's dynamic manufacturing environment because of the frequent introduction of new products and/or processes (see Biskup (2008b)).

The mathematical form of a learning curve was introduced by Wright-Patterson (1936) and a review of various modifications of Wright-Patterson's learning curve has been presented by Yelle (1979). Several different learning curves have been studied in the literature. More recently, Badiru (May) has described ten different learning curve models (including the Wright-Patterson's model) that have been used in the literature to model learning. Janiak and Rudek (2009) also point out that learning often occurs in three phases: incipient, rapid-learning, and, maturity, which results in an S-shaped learning curve Carr (1946). Despite the various alternatives proposed, the log-linear form of the Wright-Patterson's learning curve

remains popular, and it has been supported empirically in the literature (Dar-El, 2000). If we represent the initial unit processing time by $p_{initial}$, and that required after the completion of amount of work X by p_X, where $X \geq 0$, then the Wright-Patterson's learning curve can be expressed as a function of X and a constant d as follows: $p_X = p_{initial} X^{-d}$, where $d\,(> 0)$ is called the learning constant (or learning index), and it captures a worker's ability to learn. A larger value of d represents faster learning, and hence, a quicker reduction in cycle time. The value of d is less than one in practice (Baloff, 1971). Smunt (1999) has noted that learning is a continuous process, and X can be considered a continuous variable even when discrete jobs are processed. This results in more accurate predictions. Dar-El (2000) and Dar-El *et al.* (1995a,b) have related the standard processing time, denoted as STD, to the first execution time by the expression: $STD = {}^{p_{initial}}\!/_{(57-60\,2^{-d})}$. More generally, the learning curve can be expressed as a function $f(X)$, and therefore, the processing time required for each job by $p_X = p_{initial} f(X)$, where $f(0) = 1$, and $f(X)$ is non-increasing in X.

The impact of learning on production scheduling problems was first studied by Biskup (1999b). He addressed the problem of scheduling multiple lots on a single machine when the processing time of a lot depends on its position in the production sequence. If the initial time to process a job, i, is p_i, then the time required to process it at the vth position in the sequence, is given by $p_{iv} = p_i v^{-d}$. This is referred to as *position-based learning*. Another form of learning that has been considered in the literature is the *sum-of-processing-time-based learning*. The interested reader is referred to a survey of learning in scheduling by Biskup (2008b).

Lee and Wu (2009) present a general learning model that incorporates both position-learning and processing-time-based learning in a multi-machine case. In their model, the processing time for the jth job, when processed in the vth position on machine i, is given by:

$$p_{i[v]} = p_{ij} \left\{ \alpha(v) + \sum_{k=1}^{v-1} \beta_{[k]} p_{i[k]} \right\}^{-d},$$

where $\alpha(v)$ represents the impact of position on learning ($\alpha(v) = v$ for the Wright-Patterson learning curve), and the remaining term $\sum_{k=1}^{v-1} \beta_{[k]} p_{i[k]}$ represents the weighted sum of the processing times of the previous jobs.

While the impact of learning on single-job scheduling problems has been studied in the literature Biskup (2008a), the influence of learning on lot streaming problems has not yet been addressed. In this chapter, we consider a general learning curve that continuously decreases the processing time of a production lot (consisting of several identical items), which is broken into sublots, for transferring to downstream machines as they are processed, even if the entire lot has not completed processing on a machine. As shown in Chapter 1, there exists a large amount of literature on lot streaming that determines optimal number and sizes of sublots, but under the assumption of constant processing times. Our aim is to transform (under certain restrictions) such a lot streaming problem in the presence of learning in processing times (defined by an arbitrary learning curve), into its equivalent

counterpart consisting of constant processing times (with some change in input data), so that the existing solution techniques can be used for its solution. We seek a transform with the following two properties: (a) it preserves the nature of the production scenario; for example, a two-machine flow shop problem with learning in processing time is transformed to a two-machine flow shop problem with constant processing times, and (b) it can be used for all regular measures of performance, i.e., the same transform can be used for any objective function that is non-decreasing in completion times of the lots, and their sublots on each machine. The transform that we present, in addition to having the above properties, transforms parameter values independently; for example, a change in processing time values for machine, i, in the original problem alters only the processing time of machine i in the transformed problem, and similarly, a change in the size of lot j alters only the size of that lot in the transformed problem. We also identify the necessary and sufficient conditions for its implementation.

In what follows, we introduce, in Section 2, the notation and the assumptions that we make. The proposed transform and the necessary and sufficient conditions for its implementation are presented in Section 3.3. In this section, we also illustrate the use of this transform on some lot streaming problems encountered in different multistage scheduling environments. The case of its use for single-stage processing encountered in parallel machine environments is presented in Section 4. Finally, some concluding remarks are made in Section 3.5.

3.2 Assumptions and Notation

As mentioned earlier, production lots are frequently assumed to be continuously divisible in the literature related to lot streaming (see for example (Potts and Baker, 1989a; Çetinkaya and Gupta, 1994; Vickson and Alfredsson, 1992; Glass and Potts, 1998; Glass *et al.*, 1994a)) since this assumption makes problems more tractable, while still yielding good quality solutions (Bukchin *et al.*, 2002a; Shallcross, 1992). In addition, the process of learning can also be considered to be continuous (Smunt, 1999). In this chapter, we restrict our attention to such continuous sublot sizes and learning functions. Also, note that, lot streaming problems are multi-stage problems by definition.

Consider the following notation:

Input Data:

m	Number of machines (workers).
w	Number of lots that need to be processed.
$\hat{U_j}$	Size of lot j, $j = 1, \cdots, w$ (lots are assumed to be continuously divisible).
p_{ij}	Initial unit processing time for lot j on machine i, $i = 1, \cdots, m$ and $j = 1, \cdots, w$.
$f_j(x)$	Learning function that captures the influence of learning after having processed x items of lot j, $j = 1, \cdots, w$.
$F_j(x)$	The integral of the learning function $\left(= \int_0^X f_j(x)dx\right)$.

The unit processing time for lot j on machine i after having processed $X(\leq U_j)$ items of the lot is given by $p_{ij}f_j(X)$. We also have $f_j(0) = 1$. For the Wright-Patterson's learning curve, we have $f(X) = X^{-d}$, where d is the learning constant, and $F(X) = \dfrac{X^{1-d}}{1-d}$.

In this chapter, we consider a general lot streaming scheduling problem having the following properties:

(a) The objective function is a regular measure, i.e., purely a function of the completion times of the lots and sublots, and/or the number of sublots.

(b) All the jobs of a lot follow the same machine sequence.

(c) The influence of learning on the unit processing time of a machine (worker) is independent for different lots, and, for different operations performed on a given lot.

(d) The sublot sizes are consistent, i.e., the same sublot sizes of a lot are used on all the machines.

(e) The effects of learning are homogeneous, i.e., the learning function for lot j, $f_j(x)$, is the same on every machine. Note that this still permits different learning functions for different lots.

We denote the set of lot streaming problems that satisfy the above properties by $\mathcal{L}$, and the counterpart of these problems, the set that involves no learning, but is otherwise identical to the problems in $\mathcal{L}$ with regard to machine environment, etc., by $\mathcal{C}$. We use $L(\mathbf{P}, \mathbf{u}, f)$ and $C(\mathbf{P}, \mathbf{u})$ to denote individual problem instances in $\mathcal{L}$ and $\mathcal{C}$, respectively, where $\mathbf{P}$ represents an $m \times w$ matrix of initial processing times, $\mathbf{u}$ a vector of lot sizes, and, f a learning function. We develop a transformation for any problem $L(\mathbf{P}, \mathbf{u}, f) \in \mathcal{L}$ to a problem $C(\mathbf{P}, \mathbf{u'}) \in \mathcal{C}$, so that problem in $\mathcal{L}$ can be solved if a solution of its counterpart in $\mathcal{C}$ exists.

Our objective is to obtain an optimal number of sublots, their sizes, and, the optimal sequence of processing the lots. The number of sublots of lot j is denoted by n_j, and the size of its sublot k by u_{jk}. We also use U_{jk} for cumulative sublot sizes, where $U_{jk} = \sum_{\alpha=1}^{k} u_{j\alpha}$. For clarity, we will use s_{jk} to represent the size of sublot k of lot j when the processing times are constant.

3.3 Transformation of Problems in $\mathcal{L}$ to problems in $\mathcal{C}$

Next, we present our main result that affords the transformation of a problem in $\mathcal{L}$ to a problem in $\mathcal{C}$ involving constant processing times.

Proposition 3.1. *If we let $L(\boldsymbol{P}, \boldsymbol{u}, f)$ be a scheduling problem influenced by learning, and $C(\boldsymbol{P}, \boldsymbol{u'})$ be the corresponding problem with constant processing times, where $U'_j = F_j(U_j)\,\forall j = 1, \cdots, w$, then, a lot streaming schedule, $\mathcal{S}_{\mathcal{C}}$, for $C(\boldsymbol{P}, \boldsymbol{u'})$ that consists of n_j sublots for lot $j = 1, \cdots, w$, with sublot sizes that are consistent and given by $s_{j1}, \cdots, s_{jn}$, is optimal if and only if there exists an optimal schedule, $\mathcal{S}_{\mathcal{L}}$, for $L(\boldsymbol{P}, \boldsymbol{u}, f)$ that also consists of n_j sublots, for lot $j = 1, \cdots, w$, with cumulative sublot sizes given by*

$$U_{jk} = F_j^{-1}\left(\sum_{\alpha=1}^{k} s_{j\alpha}\right),\ k = 1, \cdots, n_j, \tag{3.1}$$

and individual sublot sizes given by

$$u_{jk} = U_{jk} - U_{j\,k-1},\ k = 1, \cdots, n_j, \tag{3.2}$$

where $U_{j0} = 0$, with an identical sequence for lot processing.

Proof. First, note that the time required to process sublot k of lot j, on any machine i, in schedule $\mathcal{S}_{\mathcal{L}}$ for $L(\mathbf{P}, \mathbf{u}, f)$, is given by,

$$\int_{u_{j1}+u_{j2}+\cdots+u_{j\,k-1}}^{u_{j1}+u_{j2}+\cdots+u_{jk}} p_{ij} f_j(x)dx = p_{ij}\left(F_j(U_{jk}) - F_j(U_{j\,k-1})\right). \tag{3.3}$$

Next, substituting (3.1), we have,

$$\begin{aligned} p_{ij}\left(F_j(U_k) - F_j(U_{k-1})\right) &= p_{ij}\left(\sum_{\alpha=1}^{k} s_{j\alpha} - \sum_{\alpha=1}^{k-1} s_{j\alpha}\right) \\ &= p_{ij} s_{jk}, \end{aligned}$$

which is the processing time for sublot k in schedule $\mathcal{S}_{\mathcal{C}}$ for $C(\mathbf{P}, \mathbf{u'})$. Since the production sequence is the same in both schedules, the completion times of the lot and sublots are

identical for both cases. Hence, both schedules would have the same objective function value. Now, by contradiction, suppose there exists a schedule $\mathcal{S}_{\mathcal{C}}$ for $C(\mathbf{P}, \mathbf{u'})$ that results in a strictly better objective function value than the optimal value for $L(\mathbf{P}, \mathbf{u}, f)$. Then, by (3.1) and (3.2), we can obtain a schedule for $L(\mathbf{P}, \mathbf{u}, f)$ with an objective function value equal to that for $\mathcal{S}_{\mathcal{C}}$, which contradicts optimality. The converse, as well as the uniqueness of corresponding optimal solutions, follows from the existence of a one-to-one relationship between schedules $\mathcal{S}_{\mathcal{C}}$ and $\mathcal{S}_{\mathcal{L}}$ because $F_j(X)$ (and hence $F_j^{-1}(X)$ and (3.1)) is bijective as $f(x)$ is a positive valued function on the interval $[0, U_j]$. This fact also guarantees the existence of the inverse of $F(X)$. □

Next, we consider the scope of the problems to which the transformation presented in Proposition 3.1 can be applied. By definition, we can apply this transformation if the properties presented in Section 3.2 hold. Therefore, those properties (conditions) are sufficient. Next, we consider the necessity of these conditions. Recall that, the variables in a lot streaming scheduling problem are the number and sizes of sublots that are used, and their completion times. In case the objective function cannot be expressed as a function of the number of sublots, and/or the completion times of the lots and sublots, then it must be a function of sublot sizes. Proposition 3.1 cannot hold in that case since the transformed problem would have different sublot sizes, and hence, would have a different objective function value. Therefore, Condition (a) is also necessary. Note that, for some problems, the objective function can be expressed either in terms of sublot sizes, or, their completion times. The transformation of Proposition 3.1 would hold in this case provided the objective is expressed as a function of completion times. However, Condition (b) is not necessary. Consider the following example involving an open shop problem where each lot follows an independent route, in violation of Condition (b).

Example 3.1. The single lot, m-machine, open shop problem.

The single-lot, m-machine, open shop, makespan minimization, problem has been addressed by Glass *et al.* (1994b), for the case when the maximum permissible number of sublots, $n \geq m$. They show that the optimal solution consists of sublot sizes, $s_i = \frac{\hat{U}}{m}, 1 \leq i \leq m$, and $s_i = 0$ for all other sublots. The proof of optimality rests on the fact that there exists at least one machine that is never idle, and hence, the lower bound on the makespan is achieved in such a schedule. For the learning case, we need to obtain a schedule in which each machine processes m sublots, and each sublot requires the same time for processing. This can be achieved by solving the constant-processing-time problem with $F(\hat{U}_j)$ units. The optimal sublot sizes for this problem are $\dfrac{F(\hat{U}_j)}{m}$. We then use (3.1) and (3.2) to obtain sublot sizes for the original problem, which have equal sublot processing times.

Condition (c) is also a necessary condition, since, otherwise, the total processing time of a lot in the transformed problem would depend on its position in the production sequence. This is not true for any constant-processing-time scheduling problem. Note that the transformation

of Proposition 3.1 reduces a problem to its constant-processing-time counterpart in polynomial time. Therefore, the necessity of this condition is to be expected, since the scheduling problems for which the learning gains for different lots are not independent, are much harder to solve. For example, the multiple-lot two-machine flow shop, makespan minimization problem with lot streaming, can be solved in polynomial time by using a modification of Johnson's rule (see Vickson and Alfredsson (1992)), and hence, this problem with independent learning effects can also be solved in polynomial time by using Proposition 3.1. On the other hand, this problem is NP-Hard, even without lot streaming, when job processing times are a step function of the position of the job Rudek (2011). By assumption, the last Condition (d) is also necessary for the use of Proposition 3.1 since Expressions (3.1) and (3.2) use the same sublot size on all the machines.

Next, we show that Condition (e) is necessary for the transformation to hold. In fact, we present a stronger result that this condition is necessary for any similar transformation. Note that our transformation converts a scheduling problem in which the processing time of a lot is affected by learning, to an equivalent, constant-processing-time problem with the same initial processing times and machine configuration, but altered lot sizes. It also preserves the nature of the objective function. The transformed problem is closely related to the original problem since each feasible schedule for the original problem has a corresponding schedule in the transformed problem with identical completion times of the lots and sublots, as well as the same number of sublots for each lot. Hence, for any objective function satisfying Condition (a), both give the same objective value. Such a transformation to an equivalent constant-processing-time problem is not possible if the learning function is not homogeneous, i.e., if the learning function is different on different machines. Next, we present this formally.

Proposition 3.2. *For the case when the learning function is different for two or more machines, there does not exist a corresponding constant-processing-time scheduling problem with identical completion times of the lots and sublots (the number of which is also identical), and therefore, identical optimal objective function value.*

Proof. By contradiction, suppose such a transform exists when learning function is different for two or more machines. Let i and j be two machines with different learning functions (denoted by $f_i(X)$ and $f_j(X)$) for lot k over $0 < X < U_k$, where U_k is the size of lot k.

For the problem with learning, the ratio of the time required to process the first sublot, of size u_{k1}, on machine i, relative to the time required to process the entire lot on machine i, is given by $\dfrac{p_i F_i(u_{k1})}{p_i F_i(U_k)} = \dfrac{F_i(u_{k1})}{F_i(U_k)}$, and similarly, it is $\dfrac{F_j(u_{k1})}{F_j(U_k)}$, for machine j. Note that these ratios are not identical for both the machines for all values of u_{k1}. But, this is not true for any constant-processing-time problem, which is a contradiction. □

As an example of this result, consider a single lot, two-machine, flow shop with no lot streaming, where one machine is influenced by learning and the other has constant processing times. Changing the size of the first sublot impacts the ratio of the times required to

process it (relative to the total processing time of the lot) differently on the two machines. There does not exist any two-machine flow shop with constant processing times where this occurs. Consequently, there does not exist a transformation of the original problem to a constant-processing-time two-machine problem such that the lots and sublots have identical completion times in both cases.

Next, we present some classes of problems where the transformation of Proposition 3.1 can be applied.

Corollary 3.1. In the presence of homogeneous and independent learning effects, the scheduling problems with a regular objective functions, and/or a function of the number of sublots, and the following machine environments are in $\mathcal{L}$, and they can be solved using the transformation of Proposition 3.1 if the sublot sizes are consistent:

- Multiple lot, flow shop problems.
- Multiple lot, hybrid (flexible) flow shop problems where all the jobs of a lot are processed on the same machine at each stage.
- Job shop problems.
- Assembly shop problems.

Proof. Since learning effects are homogeneous and independent, conditions (e) and (c) are satisfied, respectively. Conditions (d) and (a) are satisfied by assumption. We consider condition (b) for each case. By defintion, all the lots follow the same machine sequence in a flow shop. In a hybrid flow shop, parallel machines exist at each stage. Different lots may be processed on different machines at each stage. However, if all the jobs of a lot are processed on the same machine at each stage, condition (b) will hold. Each lot may follow a different machine sequence in a job shop; however, since learning is independent for different lots, and all the jobs of a lot follow the same sequence, (b) holds true. Multiple lots combine to form a single lot in assembly shops. All the jobs of a lot, however, follow the same machine sequence, thereby satisfying condition (b). □

3.3.1 Illustration of the use of the transformation of Proposition 3.1

We illustrate the use of Proposition 3.1 by the following examples.

Example 3.2. Two-machine flowshop, single-lot problem, with sublot-attached setup times.
Consider the two-machine, single-lot, makespan minimization problem in the presence of sublot-attached setup times. Since this is a single lot problem, we do not use the subscript j

to identify lot number. The lot is assumed to be continuously divisible and sublots are not restricted to be equal. For the case with constant processing times, this problem has been addressed by Baker and Trietsch (2009) (see research notes for Chapter 12) and Alfieri *et al.* (2012), and optimal sublot sizes are "geometric" (see (Potts and Baker, 1989a; Trietsch, 1987)) in nature. We use τ_i to denote the sublot-attached setup time on machine i, and Wright-Patterson's learning curve.

Consider the following data: $p_1 = 3$, $p_2 = 6$, $\tau_1 = 4$, $\tau_2 = 19$, $d = 0.312$ and $U = 80$. The maximum number of sublots permitted is given by $N = 80$.

The transformed lot size for the equivalent problem without learning is given by:

$$\begin{aligned} U' &= \frac{U^{1-d}}{1-d} \\ &= 29.63. \end{aligned}$$

The unit processing times and setup times for the transformed problem are the same as their initial values in the original problem. For the transformed problem, we obtain the optimal makespan to be 242.91, and the optimal number of sublots, $n = 3$. The optimal sublot sizes are: $s_1 = 1.38,\ s_2 = 7.75$ and $s_3 = 20.5$.

For the Wright-Patterson's learning curve, we have $F^{-1}(X) = ((1-d)X)^{\frac{1}{1-d}}$. We can, now, obtain cumulative sublot sizes for the original problem as follows: $U_1 = ((1-d)s_1)^{\frac{1}{1-d}} = 0.923$; $U_2 = ((1-d)(s_1+s_2))^{\frac{1}{1-d}} = 14.45$; and $U_3 = ((1-d)(s_1+s_2+s_3))^{\frac{1}{1-d}} = 80$. Consequently, the individual sublot sizes are given by: $u_1 = U_1 = 0.923$; $u_2 = U_2 - U_1 = 13.524$; and $u_3 = U_3 - U_2 = 65.553$ with the makespan value the same as that obtained for the transformed problem.

The importance of incorporating the influence of learning in processing times for scheduling models can be gauged by the improvement achieved in the quality of the schedules. If the phenomenon of learning is ignored when solving Example 3.2, an optimal schedule for the constant processing time model would be: $n = 4$, $s_1 = 1.67$, $s_2 = 8.33$, $s_3 = 21.67$, $s_4 = 48.33$, and optimal makespan value = 565. However, because of learning, the actual makespan resulting from the use of these sublot sizes would be 264. Such a large difference between the planned and actual makespan values of a schedule would lead to implementation difficulties. Also, the makespan value of 264 is larger than the optimal makespan value of 242.9 obtained for the learning case. For illustration, Table 3.1 depicts the suboptimal makespan values ($M_{suboptimal}$) achieved when the optimal sublot values for the constant-processing-time model are implemented in the presence of learning, the optimal makespan values obtained when learning effects are incorporated a priori (M^*_{learn}), and the percentage improvement in makespan value, $PI = \frac{M_{suboptimal} - M^*_{learn}}{M^*_{learn}} \times 100$, obtained with increment in d for the scenario of Example 2. As may be expected, the importance of including the impact of learning a-priori increases with increment in the value of d.

d	$M^*_{suboptimal}$	M^*_{learn}	PI
0	565	565	0
0.15	378	366	3.27
0.312	264	243	8.64
0.6	176	138	27.54

Table 3.1: Importance of Considering Learning

Example 3.3. Three-stage, single lot, job shop problem.

Consider the three-stage job shop problem for a single lot studied by Glass *et al.* (1994a) for the case when there exist two machines and the lot must be processed on the first machine, followed by the second machine, and then again, on the first machine. Let the influence of learning be independent for each operation, and be defined by an S shaped curve with

$$f(X) = \begin{cases} 1 & 0 \leq X \leq 10 \\ 1.1 - 0.01X & 10 < X \leq 90 \\ 0.2 & 90 < X. \end{cases}$$

Note that we have constant processing times for the first 10 jobs and for all jobs after 90. Again, we drop the subscript for job number.

Let $p_1 = 3, p_2 = 5, p_3 = 10$, $n = 2$ and $U = 100$.

First, we obtain

$$F(X) = \begin{cases} X & 0 \leq X \leq 10 \\ 1.1X - 0.005X^2 - 0.5 & 10 < X \leq 90 \\ 40 + 0.2X & 90 < X. \end{cases}$$

Therefore, $F(U = 100) = 60$. Next, we use the method outlined by Glass *et al.* (1994a) to obtain the optimal sublot sizes when the lot size is 60. We have the case $25 = p_2^2 \leq p_1 p_3 = 30$, and hence, $q = \dfrac{p_2 + p_3}{p_2 + p_1} = \dfrac{15}{8}$, and $s_2 = qs_1$. The optimal sublot sizes are given by: $s_1 = 20.87$, and $s_2 = 39.13$. To obtain optimal sublot sizes for the original problem, first we find $U_1 = F^{-1}(20.87)$. We have $1.1U_1 - 0.005U_1^2 - 0.5 = 20.87$, which gives, $U_1 = 21.53$. Also, $U_2 = 78.47$. The optimal makespan value remains unchanged.

3.4 Parallel Machine Environments

While lot streaming problems are inherently multi-stage in nature, lot splitting problems for parallel machine environments involving single stage processing are similar. Note that Proposition 3.1 does not apply to parallel machine environments, particularly when different

jobs of a lot may be processed on different machines, which violates condition (b). This is due to the fact that the limits of integration in (3.3) are no longer necessarily identical for all the machines. As a result, the total time required to process a lot, i.e., the sum of the times spent on the sublots of the lot on each machine varies with changes in the size of the sublots. This is not true for any constant-processing-time problem. However, the structure of the solution to the corresponding constant-processing-time case can still be utilized to obtain solutions for the case of learning effects. Consider the following example.

Example 3.4. Unrelated parallel machines.
A single lot, of size U, needs to be processed on m parallel unrelated machines such that a minimum makespan is achieved. Let the initial processing time and learning functions for machine i be given by p_i and $f_i(X)$, respectively, $i = 1, \cdots, m$.

Note that for the constant-processing-time case, this problem has a simple solution of dividing the work in the inverse ratio of p_i, so that the total time for which any pair of machines are busy is equal. It is easy to see that this criteria is also true for an optimal solution even when learning is present. Hence, we must have $p_1 f_1(u_1) = p_i f_i(u_i) \forall i = 2, \cdots, m$. We must also have $\sum_{k=1}^{m} u_k = U$. Solving these m equations yields an optimal division of the lot.

We also investigate this problem further, to obtain an optimal division of work among workers who have different levels of experience, but are otherwise identical, when the learning effect follows the Wright-Patterson curve. The time required by a worker i with Δ_i experience and U_i work load, is given by $F_i(U_i + \Delta_i) - F_i(\Delta_i)$ (note the modification of U_i from the cumulative sublot size to the share of work performed by worker i, and permission of different learning functions on different machines). Using this result, for any two workers i and j, we have, $p_i \frac{(U_i+\Delta_i)^{1-d}-(\Delta_i)^{1-d}}{1-d} = p_i \frac{(U_j+\Delta_j)^{1-d}-(\Delta_j)^{1-d}}{1-d}$, where U_i is the amount of work assigned to worker i, and, Δ_i is the amount of experience the worker had prior to the beginning of work assignment. Sublot sizes can then be obtained by solving these expressions, along with the constraint, $\sum_{i=1}^{m} U_i = U$.

For the case when $\Delta_i \gg U \ \forall i$, we have

$$(U_i + \Delta_i)^{1-d} - (\Delta_i)^{1-d} = (U_j + \Delta_j)^{1-d} - (\Delta_j)^{1-d}.$$

Using Taylor's expansion, and ignoring higher order terms in $\frac{U_i}{\Delta_i}$, for $i = 1, \cdots, m$, we have $\frac{U_i}{U_j} = \left(\frac{\Delta_i}{\Delta_j}\right)^{\frac{d}{1-d}}$. This is an approximate method of apportioning work to workers with different levels of experience. If work needs to be distributed between a new worker, i, and an experienced worker, j, i.e., $\Delta_i = 0$ for some i, and $\frac{\Delta_j}{U} \ll 1$, we have, $U_i^{1-d} = (1-d)\Delta_j^{-d} U_j$. Note that if U is large, the above expressions can still be used by splitting U into appropriately small parts, and then, repeatedly evaluating each workers' workload. Since, we ignore higher order terms, these values are approximate; however, the simplicity of the expressions make such a solution attractive.

3.5 Concluding Remarks

In this chapter, we have demonstrated use of a transformation to convert a general lot streaming problem, in which the processing times are influenced by a general learning curve, to a problem where the processing times are constant. This permits use of the solution methods available in the existing literature to solve scheduling problems related to learning, with the addition of a few simple pre- and post-processing steps. We have presented sufficient conditions under which this transformation can be applied, and we also discuss the necessity of these conditions. Both of these aspects highlight the scope of the problems to which the transformation can be applied. We have illustrated the use of this transformation to diverse scheduling environments. Also, we have solved a new problem of assigning work to workers with different experience levels. In particular, the transformation presented here is applicable to single lot, two-machine flow shop problems, and hence, enables the solution of the sub-problems associated with the ALSP when the effect of learning is consistent (see Chapter 5).

Chapter 4

Lot Streaming vs Dual Sourcing When Processing Times are Stochastic

Dual sourcing is a strategy of ordering the material required to process a lot (order) from two suppliers, instead of a single supplier. This strategy has been well-studied in the literature and has been shown to reduce lead time. However, the additional ordering cost incurred by dual sourcing makes this strategy unattractive. In this chapter, we compare it to an alternative strategy of sourcing the material required for processing a lot from a single supplier, but by permitting its delivery in two partial shipments (referred to here as single-sourcing with lot streaming). We consider the case when the lead time of suppliers consists of the time required to process a lot. We assume that the processing times of the supplier and the manufacturer are stochastic, and that, the density functions of their distributions are dependent on the size of the lot; in particular, the expected time to process a lot is assumed to be proportional to the lot size. We show that, besides offering a lower ordering cost, single-sourcing (with lot streaming) incurs lower inventory and affords a better stockout risk behavior over dual-sourcing. We also relate our work to similar problems studied in the deterministic scheduling area pertaining to economic lot sizing and lot streaming.

4.1 Introduction

The use of an appropriate sourcing strategy plays an important role in improving the performance of a supply chain. When a single supplier supplies an inventory item to a manufacturer, it is referred to as single-sourcing. Instead, if two or more suppliers supply an item, the sourcing strategy is referred to as dual-sourcing or multiple-sourcing, respectively. While dual-sourcing protects a manufacturer from large losses in revenue if a supplier is unable to

supply parts for an extended period of time (possibly, because of catastrophic failure, see for example Latour (2001)), in this chapter, we explore the merits of this sourcing strategy when no such extended delay is encountered. A large body of literature has studied the use of dual-sourcing because of its apparent advantage in reducing the effective lead time when the lead time of individual suppliers is stochastic. Most of the existing literature assumes that the lead time of a supplier is independent of the quantity ordered. The majority of this literature either studies an optimal dual-sourcing strategy or compares the performance of dual-sourcing with that of single-sourcing. Sculli and Wu (1981) have shown that the first order statistic of two random variables is smaller than the mean of either, and therefore, ordering from two suppliers simultaneously would lead to a shortening of the lead time. This reduces the safety stock requirements or improves service levels for a given safety stock (see Sculli and Wu, 1981; Sculli and Shum, 1990; Kelle and Silver, 1990; Kelle and Miller, 2001; Pan *et al.*, 1991). Ramasesh *et al.* (1991) showed that dual-sourcing decreases the total expected cost when the ordering cost is sufficiently low. Lau and Lau (1994) have shown that, in addition to reducing safety stock, dual sourcing also reduces the total cycle inventory when the expected lead time for each supplier is different. Inventory costs may be further lowered if the order from the second supplier arrives just when the order from the first supplier is exhausted. In order to delay the arrival from the second supplier suitably, one may choose non-identical suppliers or offset the order times (Lau and Lau, 1994; Lau and Zhao, 1993; Hon-Shiang and Zhao, 1994).

Another sourcing strategy that has been studied in the literature is that of single sourcing with order splitting. In such a strategy, material is sourced from a single supplier, but each order is delivered in two or more shipments. In case the lead times are independent of the size of the shipment, the performance of such a strategy can be expected to be similar to that of dual sourcing with identical suppliers, when orders are not placed simultaneously, i.e., when the expected arrival time of the order from each supplier is different. Chiang and Chiang (1996) have shown that such a strategy decreases cycle inventory, in comparison to single sourcing with no order splitting, for the case when the demand follows a normal distribution and the lead times are deterministic. Hill (1996) also showed such a reduction in cycle stock when both lead times and demand are stochastic, and a single supplier supplies an order in multiple, equal, parts. More recently, Mishra and Tadikamalla (2005) compared single sourcing with order splitting with dual sourcing and showed that most of the benefits of dual-sourcing are because of lot splitting, and that, single-sourcing with order splitting can result in better performance than dual-sourcing. To the best of our knowledge, all studies related to dual sourcing and single sourcing with order splitting assume that the mean and variance of lead times are independent of the quantity ordered from the supplier. In this chapter, we relax this assumption. Instead, we assume that the lead time is due to the processing time incurred by the supplier. The deterministic version of this problem is a SLFS-LSP with two sublots, with a cost-base objective function. Lot streaming has been studied for the objective of minimizing inventory and ordering costs in a two-stage system when there exists a continuous demand at the second stage. This literature extends the models related to Economic-Order-Quantity (EOQ) by allowing replenishment of inventory

in several sublots.

The problem of minimizing the joint costs of the supplier and buyer for shipping a supplier's production lot in several sublots, when lead times, demand rates and production rates are deterministic is referred to as the joint economic lot sizing problem (JELS). Note that the objective of this problem is different from the objective for sourcing problems, where any cost incurred by suppliers is ignored. Another difference between these problems is that, in JELS, the time required by a supplier to produce a sublot is proportional to the size of the sublot (sometimes transportation times are also considered), while in problems related to sourcing strategy, the time required by a supplier to produce an item is ignored. The JELS problem was first studied by Goyal (1977) under the assumption of an infinite production rate for the supplier and without lot splitting. Banerjee (1986) extended this problem to the case when the production rates are finite. Then, Goyal (1988) introduced the idea of splitting the supplier's production lot into equal-sized sublots and permitting the transfer of a sublot to the buyer while the subsequent sublots are still being processed on the current machine. Goyal (1995) solved the problem for unequal sublot sizes, with the restriction that the sublot sizes form a geometric sequence whose common ratio is the production rate divided by the demand rate. The problem was solved for a general common ratio by Hill (1997). Finally, Hill (1999) obtained an optimal schedule when no assumptions regarding sublot sizes are made. Sajadieh *et al.* (2009) solved the JELS problem for the case of equal sublot sizes, when the supplier's lead time is the sum of a deterministic component, as used in Goyal (1988), and a stochastic component that follows an exponential distribution. The parameter for this distribution is independent of the sublot sizes used. Since they assume a deterministic demand, adding a deterministic value to the lead time merely changes the re-order point by a fixed value and does not affect the optimization model. Ben-Daya and Hariga (2004) consider stochastic demand, equal sublots, and, assume that the deterministic lead time increases linearly with order size (with an additional constant delay). A large amount of other research in the area of JELS exists that is not relevant to our discussion here. The reader is referred to Ben-Daya *et al.* (2008) for a review of literature in this field.

As noted above, the stochastic element considered in the literature has been limited to the situations where the lead time is assumed to be independent of the size of the sublot processed. Such a situation may arise when suppliers carry large inventories, from which orders are satisfied, and lead times exist only because of transportation times. In contrast, in this chapter, we consider the case when the lead time of the supplier consists solely of the time required to process (manufacture) a sublot. We assume that the processing time of the supplier and the manufacturer are stochastic, and that the parameters of their density functions are dependent on the sublot size used. Such a size-dependent lead time would exist when the time required for transportation is negligible compared to the time required for manufacturing. Considering the case when transportation times are negligible enables us to draw a clearer contrast between the two cases of lead time being independent or dependent of sublot size. Also, the literature related to dual-sourcing and JELS addresses the case when the demand for a product at the second stage is continuous and indefinite, i.e., each lot is

followed (and preceded) by another identical lot. In contrast, we consider the case when a fixed number of units are produced in a lot in response to a customer order. Hence, orders are never satisfied from inventory.

When an order is delivered in two separate shipments, there exists a risk of stockout before the arrival of the second shipment. In this chapter, we study the relationship of this stockout risk with order splitting decisions. We also compare the stockout risk, lead time and inventory cost resulting from the use of single-sourcing (with lot streaming) with those obtained when dual-sourcing is used. While we mathematically analyse differences between these sourcing strategies, we also use numerical results whenever the analytical approach is too complex to provide any insight. Such a use of numerical results to explore the consequences of employing a sourcing strategy is frequently used in the literature (see for example (Ramasesh *et al.*, 1991, 1993; Lau and Lau, 1994; Mishra and Tadikamalla, 2005)).

We perform this analysis in two steps. In the first step, we consider the case when the manufacturer's processing rate is constant. For this case, first, we analytically show that single-sourcing (with lot streaming) is better than dual sourcing when both result in similar probabilities of stockout risk. Then, we consider the case when the manufacturer's processing rate is also stochastic and use numerical experimentation to show that single-sourcing results in large improvements for this case as well.

As noted earlier, the problem of single sourcing with order splitting has been studied in the literature related to the JELS problem for deterministic environments and continuous (incessant) demand. We use the results of Hill (1999) to find optimal sublot sizes for a deterministic equivalent system (with deterministic processing times equivalent to the expected values of processing times) where the lot size is given. We examine the probability of stockout when such a solution is implemented in a stochastic environment. Note that, in case processing times are constant, it is possible to split an order (of some predetermined size) into two unequal parts such that no stockout occurs. The deterministic version of the problem of minimizing the lead time, under the constraint that stockouts do not occur, is identical to a makespan minimization SLFS-LSP.

In the remainder of the chapter, we proceed as follows. In Section 2, we present an overview of the problem studied, describe the production environment considered, and also introduce the notation used. In Section 3, we compare single sourcing (with lot streaming) and dual sourcing when the manufacturer's processing time is constant, while the suppliers experience stochastic processing times. We continue with this comparison in Section 4 where stochastic processing times are encountered by both the suppliers and the manufacturer. In Section 5, we use numerical experimentation to compare single- and dual-sourcing strategies. The processing times are assumed to be uniformly distributed. We also consider the case of gamma distributed processing times. In Section 6, we investigate the effect of using solutions to some related deterministic (constant processing time) problems in the stochastic environment studied here. Finally, we make concluding remarks and discuss avenues of future research in Section 7.

4.2 Problem Description, Assumptions, and Notation

In this chapter, we compare the performances of single-sourcing (with lot streaming) and dual-sourcing used to acquire material from suppliers when the processing times of both the supplier and manufacturer are stochastic. We mathematically express the lead time and stockout probability for each case when an order consists of U identical items, which are processed in two stages, first by one or more suppliers at stage 1 followed by a manufacturer at stage 2. For the single-supplier case, material is transferred in two sublots of sizes denoted by s_1 and $s_2\,(= U - s_1)$. In the two-supplier case, we assume that the suppliers are identical and that the first supplier supplies a sublot of size s_1 while the second supplier supplies a sublot of sizes s_2. For this case, we allow the manufacturer to place an order with the second supplier δ time units after ordering from the first supplier. The single supplier always manufactures an order as a single lot, but may transfer a sublot of finished goods to the manufacturer before completing the entire lot.

Similar to Ramasesh *et al.* (1991), we define the expected effective lead time, denoted by L, to be the expected time required by the first sublot of raw material to be received by the manufacturer (for the sake of brevity, we refer to the "expected value of lead time" simply as "lead time"). The stockout probability considered in this chapter is the probability that the manufacturer exhausts the first sublot before the second sublot arrives. We evaluate it by assuming that the first sublot is processed immediately upon arrival.

Consider the following notation:

Parameters:

δ := Time interval between the placing of orders with two suppliers.
p_a := Mean unit processing time for the supplier (it is the same for all the suppliers for both single-supplier and two-supplier cases).
p_b := Mean unit processing time for the manufacturer.
U := Lot size

Random Variables:

X_1, X_2 := Processing times of the first and second sublots for the single-supplier system.
Y_1 := Processing time for the sublot, of size s_1, processed by the first supplier when dual-sourcing is used.
Y_2 := Processing time for the second sublot, of size s_2, processed by the second supplier when dual-sourcing is used. Note that the completion time for the second supplier is $\delta + Y_2$.
Z_1, Z_2 := Processing times of the first and second sublots for the manufacturer.

Dependent Random Variables:

L := Effective lead time, which represents the time until the arrival of the first sublot from a supplier.

In the general case studied in Section 4.4, the processing times X_1, X_2, Y_1, Y_2, Z_1 and Z_2 are all assumed to be random. However, we begin our analysis with the case when Z_1 and Z_2 are deterministic. We assume that the probability density functions of these random processing times depend solely on the size, s, of the sublot being produced and the mean unit processing time, p, of the stage. The probability density function (pdf) and cumulative distribution function (cdf) are denoted by $f(x; s, p)$ and $F(x; s, p)$, respectively, where x represents a value of the variable being considered. For brevity, we write $f(x; s, p)$ as f_X when the parameters of the distribution of the random variable X are apparent in the context that they are used. We consider both uniformly distributed and gamma distributed processing times. The mean and variance of the processing time for both distributions are assumed to be ps and ps^2 respectively, where $p \in \{p_a, p_b\}$. This implies that, for the case of uniformly distributed processing times, the support of the random variables extends over $\left[ps - p\sqrt{3s},\, ps + p\sqrt{3s}\right]$, and for the gamma distribution, its shape factor is given by s, and the scale factor by p. In order to have feasible limits for the uniform distribution, we assume $s \geq 3$ and $U \geq 6$. We also assume that transportation times are negligible.

4.3 Constant Manufacturer Processing Time

In this section, we consider the case when the manufacturer's processing time is a deterministic constant, i.e., Z_1 and Z_2 are deterministic, and compare the relative costs of single sourcing (with lot streaming) and dual sourcing. In both cases, the material is transferred from the first stage to the second in two sublots of sizes s_1 and s_2. The expected lead time of the first sublot, $L = p_a s_1$ and the expected arrival time of the second sublot is $p_a(s_1 + s_2) = p_a U$ when single sourcing (with lot streaming) is used, while for dual sourcing, it is $\delta + p_a s_2$. In order to have a fair comparison of the two strategies, we consider the case when the sublot sizes and the expected lead times of the second sublot are the same in both cases. To that end, we set $\delta = p_a s_1$. Note that, for dual sourcing, it is possible that the support of Y_1 and Y_2 intersect. In such cases, the orders may cross, i.e., the second sublot of size s_2 arrives first. However, since the value of δ is large enough to be comparable to the lead time of a sublot, the probability of order-crossing can be expected to be negligibly small (Pan *et al.*, 1991). So, we ignore order-crossing for the analysis presented in this section (however, we consider it in the next section). We also assume the reorder level R to be the same in both cases.

Next, we compare the expected values of the cost incurred for both sourcing strategies. Note that, by assumption, the first sublot (of size s_1) arrives first, and, the distribution of its arrival time, L, is identical for both cases; hence, we can compare the two strategies

conditioned on L. Also, for the same realization of L, the inventory level immediately after the arrival of the first sublot will also be the same in both cases; and as in Ramasesh *et al.* (1991), we denote it by H.

In order to obtain $E(cost|L)$, we need to consider the following two possible cases: Case 1. The second sublot arrives before the first sublot is exhausted; and, Case 2. The second sublot does not arrive before the first sublot is exhausted, and a backordering cost is incurred. Figure 4.3.1 depicts the manufacturer's inventory level for these two cases. For either case, the cost can be expressed as a function of the time interval, designated by T, between the arrival of the sublots. In the case of single sourcing (with lot streaming), the length of this interval is simply the time required by the supplier to manufacture the second sublot, i.e., $T_{SS} = X_2$. When dual sourcing is used, the lead time for the second sublot is independent of the first sublot, and is given by $\delta + Y_2$, and consequently,

$$T_{DS} = \delta + Y_2 - Y_1,$$

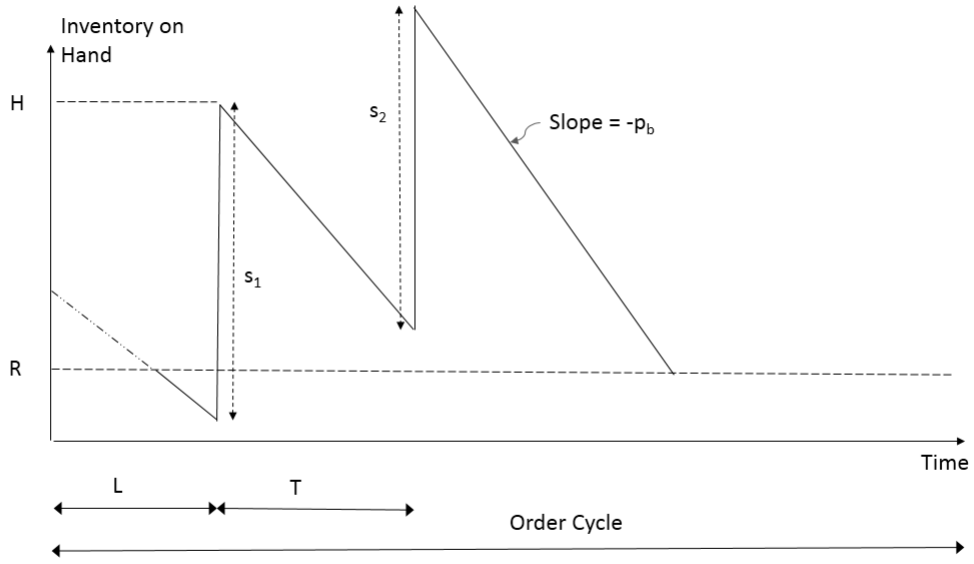
where $\delta = p_a s_1$ as chosen earlier. Note that $E[T_{SS}] = E[T_{DS}]$ since we have set $\delta = p_a s_1$, and $E[X_1] = E[Y_1] = p_a s_1$ and $E[X_2] = E[Y_2] = p_a s_2$.

For Case 1, the manufacturer's inventory level immediately after the arrival of the first sublot is given by H. The second sublot arrives after time T, and immediately prior to the arrival of the second sublot, the inventory level is $H - \dfrac{T}{p_b}$. Since the length of the time interval between these periods is T, the inventory cost incurred during this interval is $\tilde{h}\left(H + H - \dfrac{T}{p_b}\right)\dfrac{T}{2}$, where $\tilde{h}$ is the unit inventory cost per time period. The inventory level immediately after the arrival of the second sublot is $H - \dfrac{T}{p_b} + s_2$, which is reduced to the reorder level R in time interval $p_b\left(H - {}^{T}/_{p_b} + s_2 - R\right)$. Therefore, the inventory holding cost incurred after the arrival of the second sublot is given by $\dfrac{\tilde{h}p_b}{2}\left(H - \dfrac{T}{p_b} + s_2 + R\right)\left(H - \dfrac{T}{p_b} + s_2 - R\right)$. Hence, the total inventory cost incurred is obtained by adding these costs, and after simplification, it reduces to

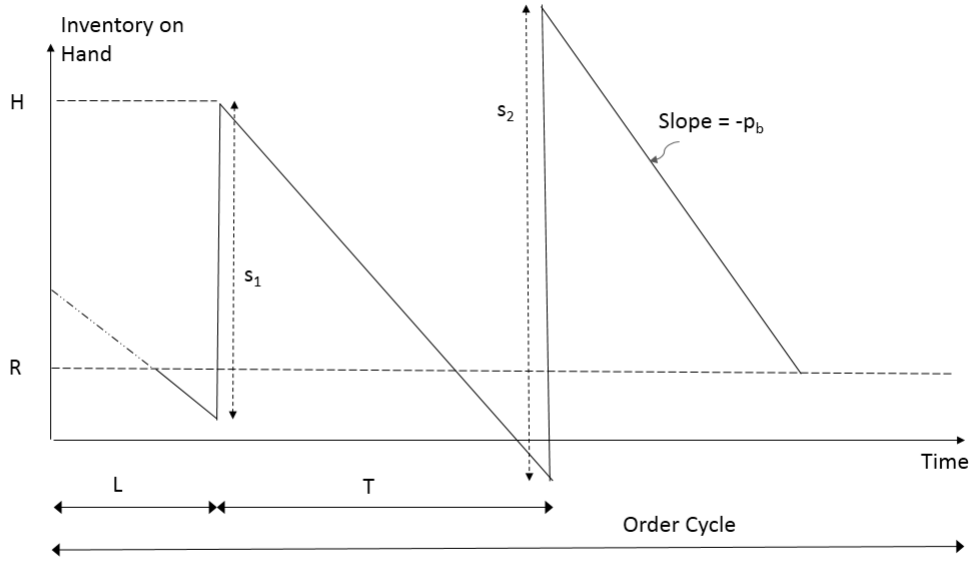
$$\tilde{h}p_b\frac{(H + s_2)^2 - R^2}{2} - s_2\tilde{h}T. \tag{4.1}$$

Similarly, for Case 2, the inventory is completely exhausted prior to the arrival of the second sublot, and the inventory cost is given by $\dfrac{\tilde{h}H^2p_b}{2}$, while the cost after its arrival is again given by $\dfrac{\tilde{h}p_b}{2}\left(H - \dfrac{T}{p_b} + s_2 + R\right)\left(H - \dfrac{T}{p_b} + s_2 - R\right)$; note that $\left(H - \dfrac{T}{p_b}\right)$ is negative because of backordering. The time interval over which backordering occurs is $T - p_bH$, during which the number of units backordered is $\dfrac{T - p_bH}{p_b}$, resulting in a backordering cost of $\dfrac{\tilde{b}\left(T - p_bH\right)^2}{2p_b}$, where $\tilde{b}$ is the unit backordering cost per unit time period. After simplification, we have the

Figure 4.3.1: Manufacturer's inventory for deterministic processing times for (a) Case 1: Second sublot arrives before the first sublot is exhausted, and for (b) Case 2: Second sublot does not arrive before the first sublot is exhausted.



(a)



(b)

total inventory and backordering cost for Case 2 given by

$$\left(\tilde{h}+\tilde{b}\right)\frac{T^2}{2p_b}-\tilde{h}\left(H+s_2\right)T-\tilde{b}HT+\frac{\tilde{b}p_bH^2}{2}+\tilde{h}p_b\frac{\left(H+s_2\right)^2+H^2-R^2}{2}. \tag{4.2}$$

Next, we show that

$$E[T_{SS}^2]<E[T_{DS}^2]. \tag{4.3}$$

Recall that $T_{DS}=\delta+Y_2-Y_1$. As Y_1 and Y_2 are independent, $Var(T_{DS})=Var(Y_2)+Var(Y_1)$. Also, Y_2 is identical to T_{SS} in distribution. Therefore, $Var(T_{DS})-Var(T_{SS})=Var(Y_1)$. Noting that $Var(T_{DS})-Var(T_{SS})=E[T_{DS}^2]-E[T_{SS}^2]$ (since $E[T_{DS}]=E[T_{SS}]$])$=Var(Y_1)>0$, we have the desired result.

Note that the expected cost for Case 1, given by (4.1), is a function of $E[T]$, and since $E[T_{DS}]=E[T_{SS}]$, Case 1 results in an identical expected cost for both sourcing strategies. Also, from (4.1), the variance of inventory cost for Case 1 is given by $\left(s_2\tilde{h}\right)^2 Var(T)$, which is higher for dual sourcing. Hence, for Case 1, single sourcing (with lot streaming) gives a more predictable performance, and hence, is better.

For Case 2, the expected total inventory and backordering cost for single sourcing and dual sourcing are different. By (4.2), the difference between their expected costs is given by $\left(\tilde{h}+\tilde{b}\right)\frac{E[T_{SS}^2]-E[T_{DS}^2]}{2p_b}$ (all other terms are either constants or functions of $E[T]$, and hence, cancel out).

The overall expected cost for either sourcing strategy, however, depends on the probability of the occurence of either Case 1 or Case 2, and is given by

$$P(Case\,1|L)E[cost|Case1,L]+P(Case\,2|L)E[cost|Case2,L_1].$$

The probability with which Case 1 or Case 2 occurs depends on the sourcing strategy used as well as the values of the parameters, R, s_1, s_2, p_a, and p_b as follows.

Note that we only consider the case when the orders do not cross. Therefore, $P(Case\,1|L)=1-P(Case\,2|L)$. For single sourcing (withlot streaming), Case 1 occurs when the second sublot completes processing at the first stage before the inventory that is available immediately after arrival of the first sublot (given by $H+s_1$) is exhausted, Since the second stage is assumed to use inventory at a constant rate of $\frac{1}{p_b}$, we have $P(Case\,1|L)=P(X_2\leq p_b(H+s_1))$, and similarly for dual sourcing, $P(Case\,1|L)=P(Y_2\leq L+p_b(H+s_1)-\delta)$. Note that these are dependent on the distribution of the processing time and the values of the parameters

Although single sourcing (with lot streaming) is at least as good as dual sourcing for both cases, the overall expected cost of single sourcing may be worse if one of the cases has a significantly higher cost, and that case occurs more frequently when single sourcing is used.

However, single sourcing results in a better co-ordination between the arrival of the sublots, since any delay in the arrival of the first sublot delays the second sublot as well. In order to better understand the behavior of each sourcing strategy, we next study the fully stochastic case for which the manufacturer's processing times are also stochastic.

4.4 Stochastic Manufacturer and supplier Processing Times

We now consider the case when Z_1 and Z_2 are also stochastic. First, we consider the case when a single supplier supplies all material in the form of two sublots of sizes s_1 and $s_2\,(= U - s_1)$. Since the transportation times are assumed to be negligible, the expected effective lead time is simply the expectation of the time required by the supplier to process the first sublot. Therefore,

$$E[L] = E[X_1] = \int_0^\infty x_1 f(x_1; s_1, p_a) dx_1. \tag{4.4}$$

The manufacturer begins processing the first sublot immediately upon it's arrival, and at the same time, the supplier begins processing the second sublot. A stock-out occurs when the manufacturer completes processing the first sublot before the supplier delivers the second sublot, i.e., when $X_2 > Z_1$. Therefore, the stock-out risk is given by

$$\begin{aligned} P(X_2 > Z_1) &= \int_0^\infty \int_0^{x_2} f(z_1; s_1, p_b)\, f(x_2; s_2, p_a) dz_1 dx_2, \\ &= \int_0^\infty F(x_2; s_1, p_b)\, f(x_2; s_2, p_a) dx_2. \end{aligned} \tag{4.5}$$

Next, consider the case when dual-sourcing is used and an interval of δ time units exists between the times at which the orders are placed with the two suppliers. In this case, either of the two suppliers may deliver the first sublot. Therefore, the expected effective lead time

is given by, $E[min(Y_1,\ \delta + Y_2)]$. We have,

$$\begin{aligned} E[L] &= \int_0^{\delta}\int_0^{\infty} y_1\, f_{Y_2}\, f_{Y_1} dy_2 dy_1 \\ &\quad + \int_{\delta}^{\infty}\int_{y_1-\delta}^{\infty} y_1\, f_{Y_2}\, f_{Y_1}\, dy_2\, dy_1 \\ &\quad + \int_0^{\infty}\int_{y_2+\delta}^{\infty} (y_2+\delta)\, f_{Y_1}\, f_{Y_2}\, dy_1\, dy_2 \\ &= \int_0^{\delta} y_1\, f_{Y_1} dy_1 \\ &\quad + \int_{\delta}^{\infty} y_1\, f_{Y_1}\, (1 - F(y_1-\delta; s_2, p_a))\, dy_1 \\ &\quad + \int_0^{\infty} (y_2+\delta)\, (1 - F\,(y_2+\delta; s_1, p_a))\, f_{Y_2} dy_2. \end{aligned} \tag{4.6}$$

If the first supplier supplies a sublot before it is delivered by the second supplier, i.e. $Y_1 \leq \delta + Y_2$, the manufacturer experiences a stock-out when $Y_1 + Z_1 < \delta + Y_2$. In case the second supplier supplies raw material first, i.e. $Y_1 > \delta + Y_2$, a stock-out occurs when $\delta + Y_2 + Z_1 < Y_1$. Note that $Y_1 + Z_1 < \delta + Y_2 \Rightarrow Y_1 \leq \delta + Y_2$ and $\delta + Y_2 + Z_1 < Y_1 \Rightarrow Y_1 > \delta + Y_2$ since $Z_1 \geq 0$. Also, $max(\delta + Y_2 - Y_1 - Z_1,\, 0)$ and $max(Y_1 - \delta - Y_2 - Z_1, 0)$ cannot take positive values simultaneously. So, the probability of a stock-out is given by $P(\delta + Y_2 - Y_1 - Z_1 > 0) + P(Y_1 - \delta - Y_2 - Z_1 > 0)$. Using $\omega = max(y_2 + \delta - z_1, 0)$, we have,

$$\begin{aligned} &P(\delta + Y_2 - Y_1 - Z_1 > 0) \\ &= \int_{z_1=0}^{\infty}\int_{y_2=0}^{\infty}\int_{y_1=0}^{\omega} f_{Y_1} f_{Y_2} f(z_1; s_1, p_b) dy_1 dy_2 dz_1 \\ &= \int_{z_1=0}^{\infty}\int_{y_2=0}^{\infty} F(\omega; s_1, p_a) f_{Y_2} f(z_1; s_1, p_b) dy_2 dz_1 \end{aligned} \tag{4.7}$$

and

$$\begin{aligned} &P(Y_1 - \delta - Y_2 - Z_1 > 0) \\ &= \int_{z_1=0}^{\infty}\int_{y_2=0}^{\infty}\int_{y_1=\delta+y_2+z_1}^{\infty} f_{Y_1} f_{Y_2} f(z_1; s_2, p_b) dy_1 dy_2 dz_1 \\ &= \int_{z_1=0}^{\infty}\int_{y_2=0}^{\infty} (1 - F(\delta + y_2 + z_1; s_1, p_a))\, f_{Y_2} f(z_1; s_2, p_b) dy_2 dz_1. \end{aligned} \tag{4.8}$$

Next, we illustrate the use of these general expressions to obtain closed-form expressions for the respective measures using uniformly distributed processing times.

4.5 Analysis for Specific Distributions

Consider the case when a single supplier delivers material in two shipments. The processing times are uniformly distributed with, $X_1 \sim U\left(p_a s_1 - p_a\sqrt{3s_1},\ \ p_a s_1 + p_a\sqrt{3s_1}\right)$, $X_2 \sim U\left(p_a s_2 - p_a\sqrt{3s_2},\ \ p_a s_2 + p_a\sqrt{3s_2}\right)$, $Z_1 \sim U\left(p_b s_1 - \ p_b\sqrt{3s_1},\ \ p_b s_1 + p_b\sqrt{3s_1}\right)$ and $Z_2 \sim U\left(p_b s_2\ \ - p_b\sqrt{3s_2},\ \ p_b s_2 + p_b\sqrt{3s_2}\right)$. For simplicity of expressions, we shall denote the lower limit and upper limit of a uniformly distributed random number X by l_X and u_X, respectively. The length of the support will be denoted by Δ_x. For example, $l_{Z_1} = p_b s_1 - p_b\sqrt{3s_1}$ and $\Delta_{Z_1} = 2p_b\sqrt{3s_1}$.

Evaluating Expression (4.4), we get $E[L] = E[X_1] = p_a s_1$. Clearly, this implies a smaller lead time for small values of s_1. In particular, the lead time is smaller than that for the case when the production lot is not split into two parts, since $s_1 < U$, by assumption. Using Expression (4.5) and the pdf and cdf of a uniform distribution, we obtain the stock-out probability,

$$P(X_2 > Z_1) \ = \ \begin{cases} 1 & \text{for } l_{X_2} > u_{Z_1} \\ 0 & \text{for } u_{X_1} < l_{Z_1}, \\ \frac{\left(w_2 - l_{Z_1}\right)^2 - \left(w_1 - l_{Z_1}\right)^2}{2\Delta_{Z_1}\Delta_{X_2}} & \\ + \frac{u_{X_2} - w_2}{\Delta_{X_2}}, \text{ otherwise,} & \end{cases} \tag{4.9}$$

where $w_1 = max\left(l_{X_2},\, l_{Z_1}\right)$, $w_2 = min\left(u_{X_2}, u_{Z_1}\right)$.

Next, consider the case when two identical suppliers supply material, but the order to the second supplier is placed δ time units after placing an order to the first supplier. Evaluating Expression (4.6) for expected lead time, we have $E[L] = \delta + p_a s_2$ if $l_{Y_1} - \delta > u_{Y_2}$, and $E[L] = p_a s_1$ if $u_{Y_1} - \delta < l_{Y_2}$.

Otherwise,

$$\begin{aligned} E[L] &= \frac{y_1^2}{2\Delta_{y_1}}\Big|_{min\left(\delta,l_{Y_1}\right)}^{min\left(\delta,u_{Y_1}\right)} + \frac{y_1^2}{2\Delta_{y_1}}\Big|_{max\left(\delta,l_{Y_1}\right)}^{max(\delta,w_1)} \\ &+ \left(\frac{y_1^2}{2\Delta_{Y_1}} + \frac{(\delta+l_{y_2})\,y_1^2}{2\Delta_{Y_2}\Delta_{Y_1}} - \frac{y_1^3}{3\Delta_{Y_2}\Delta_{Y_1}}\right)\Big|_{max(\delta,w_1)}^{max(\delta,w_2)} \\ &+ \frac{(y_2^2+2\delta y_2)}{2\Delta_{Y_2}}\Big|_{l_{Y_2}}^{max(l_{Y_1}-\delta,l_{Y_2})} \\ &+ \left(\frac{(y_2+\delta)^2}{2\Delta_{Y_2}} + \frac{l_{Y_1}(y_2+\delta)^2}{2\Delta_{Y_2}\Delta_{Y_1}}\right. \\ &\left. -\frac{(y_2+\delta)^3}{3\Delta_{Y_2}\Delta_{Y_1}}\right)\Big|_{max\left(l_{Y_1}-\delta,\,l_{Y_2}\right),}^{min\left(u_{Y_1}-\delta,\,u_{Y_2}\right)} \end{aligned} \tag{4.10}$$

where $w_1 = max\,(\delta + l_{Y_2},\, l_{Y_1})$ and $w_2 = min\,(\delta + u_{Y_2},\, u_{Y_1})$.

Next, we evaluate stockout risk. It is given by the sum of Expressions (4.7) and (4.8). We illustrate computation of Expression (4.8) next. A similar development can be used to evaluate Expression (4.7). Recall that $P(Y_1 - \delta - Y_2 - Z_1 > 0) > 0$ implies that $Y_2 < Y_1$. Therefore, the size of the first sublot processed by the manufacturer is s_2 and $Z_1 \sim U\left(p_b\left(s_2 - \sqrt{3s_2}\right),\, p_b\left(s_2 + \sqrt{3s_2}\right)\right)$. Using Expression (4.8), we have $P(Y_1 - \delta - Y_2 - Z_1 > 0) = 1$ when $z < l_{Y_1} - \delta - u_{Y_2}$ and $P(Y_1 - \delta - Y_2 - Z_1 > 0) = 0$ when $z > u_{Y_1} - \delta - l_{Y_2}$. Otherwise, we have

$$\begin{aligned} &P(Y_1 - \delta - Y_2 - Z_1 > 0) \\ =&\frac{max(l_{Y_1} - \delta - u_{Y_2} - l_{Z_1}, 0)}{\Delta_{Z_1}} + \int_{l_{Y_1}-\delta-u_{Y_2}}^{u_{Y_1}-\delta-l_{Y_2}} \left(\frac{w_1' - l_{Y_2}}{\Delta_{Y_2}} + \right. \\ &\left.\frac{w_2' - w_1'}{\Delta_{Y_2}}\left(1 - \frac{\delta + z_1 - l_{Y_1}}{\Delta_{Y_1}}\right) - \frac{w_2'^2 - w_1'^2}{2\Delta_{Y_2}\Delta_{Y_1}}\right) f(z_1; s_1, p_b)\, dz_1 \end{aligned} \tag{4.11}$$

where $w_1' = max\,(l_{Y_2},\, l_{Y_1} - \delta - z_1)$ and $w_2' = min\,(u_{Y_2},\, u_{Y_1} - \delta - z_1)$.

Expression (4.11) can be evaluated separately for the cases $s_1 \leq s_2$ and $s_1 > s_2$. For $s_1 \leq s_2$, the interval over which z is integrated can be divided into three smaller intervals over which w_1' and w_2' remain constant. The first such interval is given by $[l_{Y_1} - \delta - u_{Y_2},\, u_{Y_1} - \delta - u_{Y_2}]$, and in this part, we have $w_1' = l_{Y_1} - \delta - z$ and $w_2' = u_{Y_2}$. Using these values and substituting in the first two terms of the integral, we have $\left(\frac{w_1' - l_{Y_2}}{\Delta_{Y_2}} + \frac{w_2' - w_1'}{\Delta_{Y_2}}\right) f(z_1; s_1, p_b) = \frac{w_2' - l_{Y_2}}{\Delta_{Y_2}} f(z_1; s_1, p_b) = \frac{1}{\Delta_{Z_1}}$, since $w_2' = u_{Y_2}$ and $\Delta_{Y_2} = u_{Y_2} - l_{Y_2}$. Consider the next term, $-\frac{w_2' - w_1'}{\Delta_{Y_2}}\left(\frac{\delta + z_1 - l_{Y_1}}{\Delta_{Y_1}}\right) f(z_1; s_1, p_b)$. Substituting w_1', w_2' and $f(z_1; s_1, p_b)$, we obtain

$\frac{-\left(u_{Y_2}-l_{Y_1}+\delta+z\right)\left(\delta+z-l_{Y_1}\right)}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} = \frac{-\left(u_{Y_2}\right)\left(\delta+z-l_{Y_1}\right)}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{\left(\delta+z-l_{Y_1}\right)^2}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}}$. Similarly, the last term reduces to $-\frac{u_{Y_2}^2-\left(l_{Y_1}-\delta-z\right)^2}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}}$. Adding these terms, we get $\frac{1}{\Delta_{Z_1}} - \frac{\left(u_{Y_2}\right)\left(\delta+z-l_{Y_1}\right)}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{\left(\delta+z-l_{Y_1}\right)^2}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{u_{Y_2}^2}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} + \frac{\left(l_{Y_1}-\delta-z\right)^2}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} = \frac{1}{\Delta_Z} - \frac{\left(u_{Y_2}\right)\left(\delta+z-l_{Y_1}\right)}{\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{\left(\delta+z-l_{Y_1}\right)^2}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{u_{Y_2}^2}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}}$. By integrating this expression, we have $\frac{\Delta_{Y_1}+\Delta_{Y_2}}{\Delta_{Z_1}} - \frac{\left(2u_{Y_2}(\delta-l_{Y_1})+u_{Y_2}^2\right)\left(\Delta_{Y_1}+\Delta_{Y_2}\right)}{2\Delta_Y\Delta_{Z_1}} - \frac{u_{Y_2}\left(\left(u_{Y_1}-\delta-l_{Y_2}\right)^2-\left(l_{Y_1}-\delta-u_{Y_2}\right)^2\right)}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} - \frac{\left(u_{Y_1}-l_{Y_2}-l_{Y_1}\right)^3+\left(u_{Y_2}\right)^3}{6\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}}$. Therefore,

$$
\begin{aligned}
& P(Y_1-\delta-Y_2-Z_1>0) \\
=& \frac{max(l_{Y_1}-\delta-u_{Y_2}-l_{Z_1},0)}{\Delta_{Z_1}} \\
& + \frac{\Delta_{Y_1}+\Delta_{Y_2}}{\Delta_{Z_1}} \\
& - \frac{\left(2u_{Y_2}(\delta-l_{Y_1})+u_{Y_2}^2\right)\left(\Delta_{Y_1}+\Delta_{Y_2}\right)}{2\Delta_{Y_2}\Delta_{Z_1}} \\
& - \frac{u_{Y_2}\left(\left(u_{Y_1}-\delta-l_{Y_2}\right)^2-\left(l_{Y_1}-\delta-u_{Y_2}\right)^2\right)}{2\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}} \\
& - \frac{\left(u_{Y_1}-l_{Y_2}-l_{Y_1}\right)^3+\left(u_{Y_2}\right)^3}{6\Delta_{Y_2}\Delta_{Y_1}\Delta_{Z_1}}.
\end{aligned}
$$

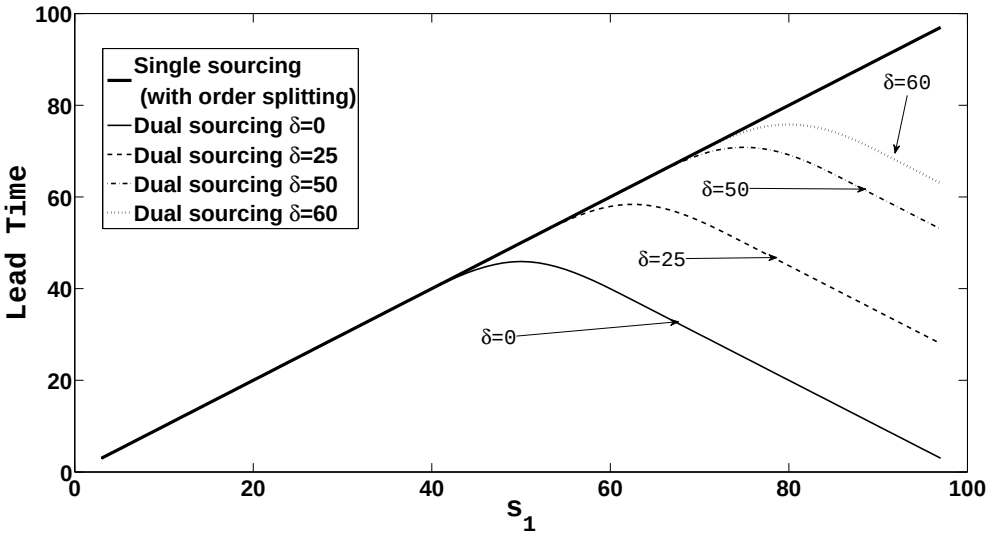
The integral can similarly be evaluated over the other intervals to obtain closed-form expressions. The next interval is given by $(u_{Y_1}-\delta-u_{Y_2},\ l_{Y_1}-l_{Y_2}-\delta]$, and we have $w_1'=l_{Y_1}-\delta-z$ and $w_2'=u_{Y_1}-\delta-z$ over this interval. The last interval is $(l_{Y_1}-l_{Y_2}-\delta,\ u_{Y_1}-\delta-l_{Y_2}]$, and we have $w_2'=u_{Y_1}-\delta-z$, $w_1'=l_{Y_2}$ over this interval.

Next, we employ the above expressions to empirically investigate the performances of dual-sourcing and single-sourcing (with lot streaming) when the processing times are uniformly distributed. Later, we also present the results of our analysis for gamma distributed processing times. We restrict sublot sizes to be integer valued in this analysis, unless otherwise mentioned. The parameter values are set to take the following default values: $p_a=1$, $p_b=1$, $U=100$; and maximum permissible stockout risk $p_{max}=0.001$.

4.5.1 Variation of lead time with s_1

For the case of uniformly distributed processing times, the variation of lead time with s_1 is shown in Figure 4.5.1. Note that when single-sourcing (with lot streaming) is used, the lead time, given by Expression (4.4), increases linearly with s_1 as shown. However, for dual-sourcing, when s_1 is small, the first supplier is likely to deliver the first sublot to the

Figure 4.5.1: Expected lead time as a function of s_1.



manufacturer, thereby causing the lead time to be close to the lead time for the single supplier case, i.e., $p_a s_1$. For sufficiently small δ and $s_1 \gg s_2$, the second supplier is likely to supply the first sublot to the manufacturer, and the lead time approaches $p_a s_2 = p_a (U - s_1)$. As δ increases, the first supplier is likely to supply the first sublot (until larger values of s_1), thereby causing the lead time to be close to $p_a s_1$.

4.5.2 Variation of stockout risk with s_1

The probability of stockout for the single-sourcing (with lot streaming) case is given by Expression (4.5) and for the dual-sourcing case by the sum of Expressions (4.7) and (4.8). As can be seen in Figure 4.5.2, the stockout risk is close to 1 for small s_1 when single-sourcing (with lot streaming) is used. But, it decreases quickly over a small interval of s_1 values, and then, remains close to 0. As may be expected, the stock-out risk is high when $E[X_1 + Z_1] < E[X_2]$, i.e., $p_a s_1 + p_b s_1 < p_a s_2$. On the other hand, when dual-sourcing is used, the stock-out risk also decreases from 1 to 0 over a small interval of s_1, remains close to 0 for a short range, but then, increases again to 1. This behavior of stockout risk for dual-sourcing is very different from that for single-sourcing (with lot streaming) and is prone to taking higher values for a greater number of values of s_1. In this sense, the stockout behavior of single-sourcing (with lot streaming) is better than that for dual-sourcing. Note that, the stockout risk is high when $p_a s_1 + p_b s_1 < \delta + p_b s_2$ or $\delta + p_a s_2 + p_b s_2 < p_a s_1$. The interval over which the stockout risk is close to 0 begins at larger values of s_1 as δ increases.

Figure 4.5.2: Stockout risk as a function of s_1.

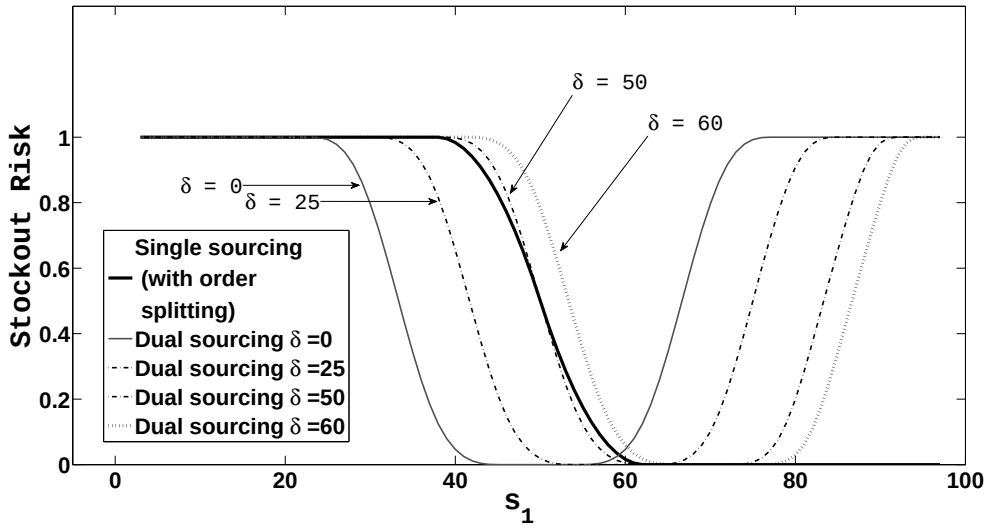


Figure 4.5.3: Minimum lead time possible with variation in p_{max}.

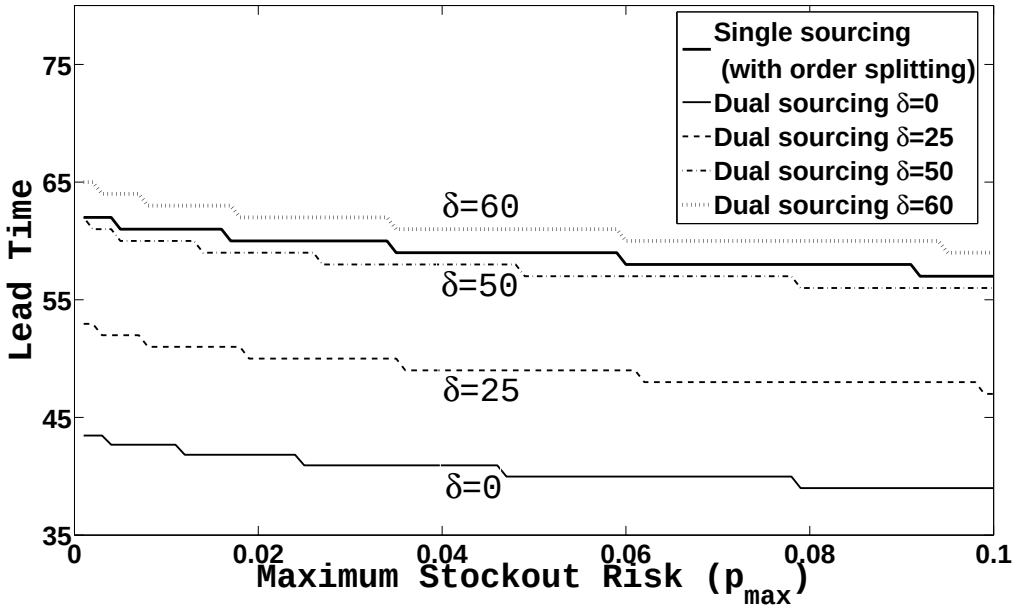


Table 4.1: Optimal s_1 values vs δ for dual-sourcing when $p_{max} = 0.001$, and optimal s_1 for single-sourcing (with lot streaming).

δ	s_1 for dual-sourcing	
	Uniform Distribution	Gamma Distribution
0	44	45
10	48	49
20	51	53
30	55	56
40	58	59
50	62	63
60	65	66
s_1 for single-sourcing (with lot streaming)		
62		65

4.5.3 Minimum lead time for a given stockout risk

We consider values of s_1 that result in a stockout risk that is less than or equal to a given value, p_{max}. Figure 4.5.3 shows the smallest lead times possible for $p_{max} \in [0.001, 0.01]$ under single-sourcing (with lot streaming) and under dual-sourcing with $\delta = 0$, 25, 50, and 60. As can be expected, the minimum feasible lead time decreases with increase in p_{max} for all these cases. As δ increases, the minimum lead time for dual-sourcing approaches the minimum lead time value for single-sourcing. As a result, the performance of dual-sourcing approaches that of single-sourcing as δ increases. In fact, for sufficiently large δ, the performance of dual-sourcing becomes worse than that for single-sourcing (with lot streaming). The value of s_1 that yields the minimum feasible lead time for dual-sourcing also approaches, and then, exceeds the optimal s_1 for single-sourcing. Table 4.1 shows optimal s_1 values for dual-sourcing when $p_{max} = 0.001$ and δ is increased. The optimal value for single-sourcing is $s_1 = 62$, with an expected lead time of 62.

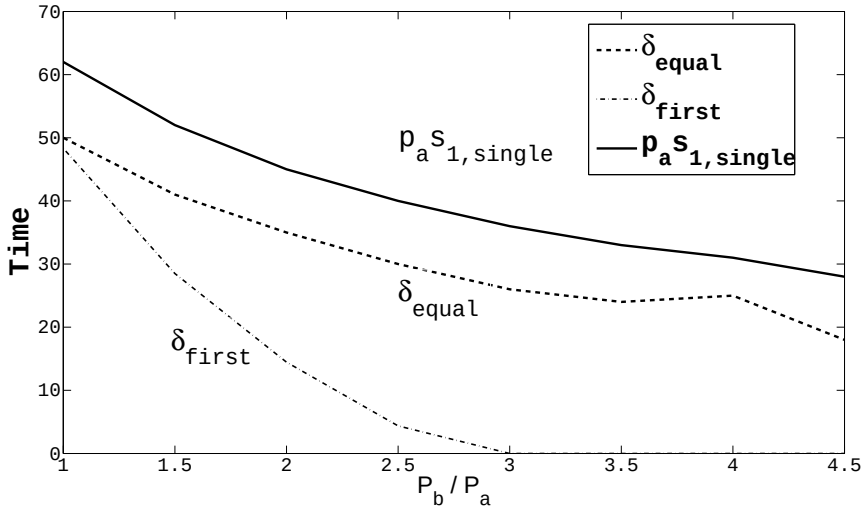
4.5.4 Impact of p_b/p_a and δ

Next, we investigate the effects of relative production speeds of the supplier(s) and manufacturer. To that end, the minimum lead time possible with $p_{max} = 0.001$ was evaluated for different values of processing time ratios. Table 4.2 shows lead times for single-sourcing (with lot streaming) and dual-sourcing for different values of δ. The magnitude of possible lead time improvement achieved by using dual-sourcing over single-sourcing decreases as p_b/p_a increases. The lead time of dual-sourcing increases with increment in δ, and, for sufficiently large values, single-sourcing performs better than dual-sourcing. We denote the value of δ (in time units) at which the lead time for dual-sourcing is equal to that for single-sourcing by δ_{equal}, and the value of δ for which the first supplier delivers the first sublot to the manu-

Table 4.2: Lead time values for single-sourcing (with lot streaming) and dual-sourcing for different values of p_b/p_a, ($p_a = 1$, $p_{max} = 0.001$).

p_b/p_a	L:single-sourcing	L:dual-sourcing			
		$\delta =$ 0	$\delta =$ 25	$\delta =$ 50	$\delta =$ 60
1	62	44	53	62	65
1.5	52	39	47	54	57
2	45	35	42	49	51

Figure 4.5.4: Variation of δ_{equal}, δ_{first} and $p_a s_{1,single}$ vs p_b/p_a.



facturer with probability 1 by δ_{first}. Here, δ_{first} is determined by finding the smallest values of δ for which $u_{X_1} < l_{X_2} + \delta$.

The values of δ_{equal} and $max(\delta_{first}, 0)$ for different p_b/p_a values are shown in Figure 4.5.4. We also evaluated the optimal size of the first sublot, s_1, and have plotted the value of $p_a s_1$ as p_b/p_a is varied. Note that δ_{equal} decreases with increase in p_b/p_a. Therefore, the range of δ over which the lead time performance of dual-sourcing is better than that of single-sourcing (with lot streaming) decreases with increase in p_b/p_a.

A larger value of δ tends to decrease the manufacturer's inventory level since it prevents the second sublot from arriving too soon, i.e., when the first sublot has barely been consumed. In view of Table 4.2, for a larger value of δ, the lead time for dual-sourcing can also exceed that for single-sourcing (with lot streaming). We further discuss the impact of both dual-sourcing and single-sourcing (with lot streaming) on inventory levels in the next section. Also, a supplier often has much higher processing speed (smaller p_a) than the manufacturer (i.e., $p_b/p_a \gg 1$), which suggests that single-sourcing is a viable alternative to dual-sourcing even when lead time performance is important and stockouts are not acceptable.

4.5.5 Amount of inventory held

Consider the inventory levels of the manufacturer for the single-sourcing (with lot streaming) and dual-sourcing systems for the case when the expected lead time performances of the two systems are identical. The minimum single-sourcing lead times possible for given values of p_a and p_b, and a given maximum stockout risk, $p_{max} = 0.001$, are shown in Table 4.3. We also show the value of δ_{equal}, δ_{first} and the optimal size of the first sublot for single-sourcing (with lot streaming), denoted by $s_{1,single}$, and dual-sourcing, denoted by $s_{1,dual}$ (all for $p_{max} = 0.001$). Note that $\delta_{equal} > \delta_{first}$ for the range of values of p_b/p_a considered in Table 4.3, which implies that, when dual-sourcing is used, the first supplier delivers the first sublot to the manufacturer. Also note that, for the parameters considered in Table 4.3, optimal sizes of the first sublot, s_1, for both cases are equal. This may be expected since we are considering the cases when the two systems have equal lead times and the expected lead time of dual-sourcing is exactly equal to $p_a s_{1,dual}$ when the first sublot arrives first at the second stage. Since the optimal sizes of the first sublot are equal for dual-sourcing and single-sourcing, the sizes of the second sublot $s_2 = U - s_1$ must also be equal. Now, consider the expected time at which the second sublot arrives at the second stage. For single- and dual-sourcing, these are given by $p_a s_1 + p_a s_2$ and $\delta_{equal} + p_a s_2$, respectively. Figure 4.5.4 shows that

$$\delta_{equal} < p_a s_{1,single} \tag{4.12}$$

for the range of parameters considered. Therefore, when dual-sourcing is used, the second sublot arrives at the second stage earlier than when single-sourcing is used. Since the sizes of the second sublot are equal in both the cases, dual-sourcing results in a higher amount of inventory, given the same lead time performance and stockout risk p_{max}.

Next, we study the variation of δ_{equal} and the optimal value of $p_a s_{1,single}$ for the stochastic single-sourcing (with lot streaming) over different values of p_{max}. The results are presented in Table 4.4. Notice that δ_{equal} stays nearly constant over different values of p_{max} while $p_a s_{1,single}$ decreases. This suggests that Expression (4.12) holds true as long as $p_{max} \leq 0.50$. This further implies that dual-sourcing results in higher amounts of inventory for $p_{max} \leq 0.50$.

Table 4.3: Values of lead times, δ_{equal}, δ_{first}, $s_{1,single}$, and $s_{1,dual}$ for single-sourcing (with lot streaming).

p_b/p_a	L	δ_{equal}	δ_{first}	$s_{1,single}$	$s_{1,dual}$
1	62	50	48	62	62
1.5	52	41	28	52	52
2	45	35	14	45	45
2.5	40	30	4	40	40
3	36	26	0	36	36
3.5	33	24	0	33	33
4	31	25	0	31	31
4.5	28	18	0	28	28

4.5.6 Impact of processing time distribution

All the analysis performed until now assumes a uniform processing time distribution. Next, we consider the case when processing times follow a gamma distribution. The shape factor of this distribution is given by sublot size, s, and size factor by the processing time of the stage. The lead time and expected stockout risk were evaluated using numerical quadrature. The expected lead times obtained are depicted in Figure 4.5.5, stockout risk in Figure 4.5.6 and minimum lead time for a given stockout risk in Figure 4.5.7. Note that the results are nearly identical to those for the uniform distribution. The optimal value of $s_1 = 65$ when single-sourcing is used. Table 4.1 shows that the optimal sublot size for dual-sourcing approaches and exceeds that for single-sourcing for both the distributions. The influence of parameters δ and p_b/p_a on minimum lead time for $p_{max} = 0.001$ is shown in Table 4.5. Since the results for the uniform distribution and gamma distribution are very similar, our conclusion that the lead time performance of single-sourcing is competitive with dual-sourcing is not unique to a particular processing time distribution.

Figure 4.5.5: Expected lead time as a function of s_1 when processing times follow gamma distribution.

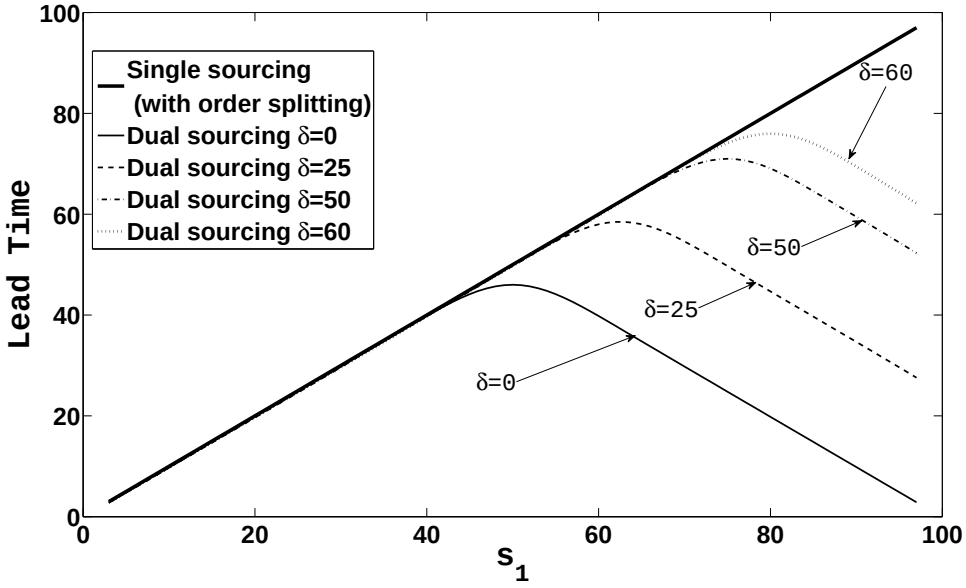


Figure 4.5.6: Stockout risk as a function of s_1 when processing times follow gamma distribution.

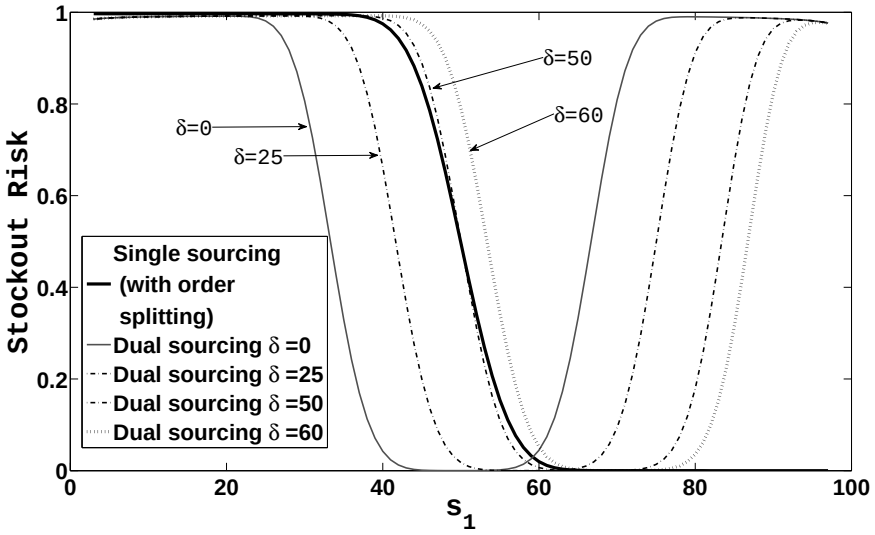


Figure 4.5.7: Minimum lead time possible with variation in p_{max} when processing times follow gamma distribution.

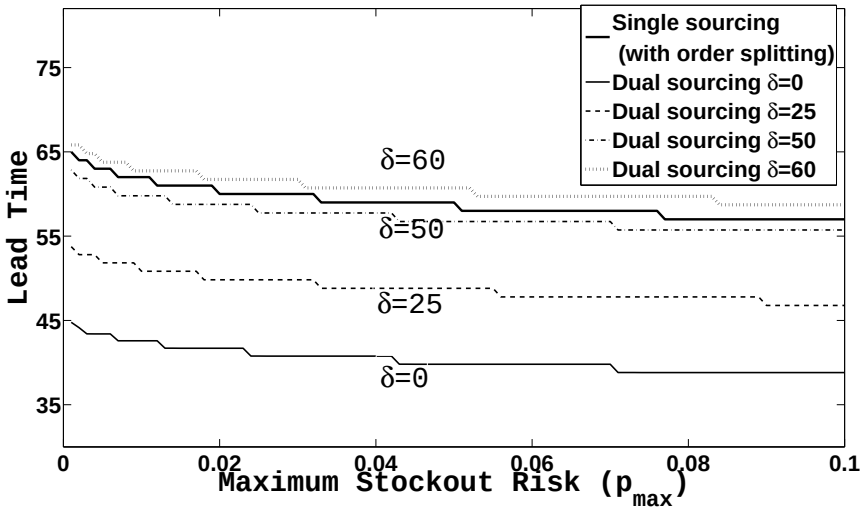


Table 4.4: Variation of $p_a s_1$ and δ_{equal} with p_{max}.

p_{max}	δ_{equal}	$p_a s_{1,single}$
0.001	50	62
0.1	52	57
0.2	52	55
0.3	50	53
0.4	50	52
0.5	50	50
0.6	50	49

Table 4.5: Lead times for single- and dual-sourcing under different values of p_b/p_a, ($p_a = 1$, $p_{max} = 0.001$) when processing times follow gamma distribution.

p_b/p_a	L: single-sourcing	L: dual-sourcing			
		$\delta = 0$	$\delta = 25$	$\delta = 50$	$\delta = 60$
1	65	45	54	63	66
1.5	55	41	48	56	59
2	49	37	43	50	53

4.6 Comparison of Stochastic and Deterministic Solutions

In this section, we consider two problems that have been studied in the literature for deterministic environments. We investigate the stockout risk that results from using the optimal solutions for the deterministic case in a stochastic environment (when processing times are stochastic). First, we consider the problem of splitting an order into two, not necessarily equal-sized, shipments such that the lead time is minimized, but stockout does not occur. As mentioned earlier, when single sourcing is used, this problem is identical to the problem of minimizing the makespan in a two-machine flow shop with lot streaming. Hence, we obtain optimal shipment sizes for the deterministic case by using geometric shipment sizes (see Potts and Baker (1989b)) for the case of single-sourcing. These are given by, $s_1 = \frac{U}{1+p_b/p_a}$ and $s_2 = U - s_1$. There does not exist such a closed-form expression for optimal sublot sizes for the case of dual sourcing, and we obtain these next. When dual-sourcing is used, we have two possibilities: either the first or the second supplier may deliver the first sublot to the manufacturer. In case, the first supplier delivers the first sublot, then the lead time is given by $p_a s_1$, and it decreases with decrease in s_1. In order to prevent a stockout, we must have $p_a s_1 + p_b s_1 \geq \delta + p_a s_2 = \delta + p_a(U - s_1)$, or, $s_1 \geq \frac{\delta + p_a U}{2p_a + p_b}$. Since s_1 should be as small as possible to reduce lead time, the optimal value of $s_1 = \frac{\delta + p_a U}{2p_a + p_b}$, and the minimum lead time $L = \frac{p_a \delta + p_a^2 U}{2p_a + p_b}$. Similarly, when the second supplier supplies the first sublot, the optimal value of $s_2 = \frac{p_a U - \delta}{2p_a + p_b}$ and the minimum lead time $L' = \frac{p_a^2 U - p_a \delta}{2p_a + p_b} + \delta$. We have, $L' - L = \frac{\delta p_b}{p_a(2p_a + p_b)} > 0$. Therefore the minimum lead time is obtained when the first supplier supplies the first sublot, and the optimal sublot size is given by $s_1 = \frac{\delta + p_a U}{2p_a + p_b}$.

Next, we evaluate the stockout probability that results from using single-sourcing (with lot streaming) and $s_1 = \frac{U}{1+p_b/p_a}$ (and $s_2 = U - \frac{U}{1+p_b/p_a} = \frac{U\, p_b/p_a}{1+p_b/p_a}$), when processing times follow a uniform distribution. Note that $E[X_2] = p_a s_2 = \frac{p_a p_b U}{p_a + p_b} = p_b s_1 = E[Z_1]$ for this value of s_1. We can rewrite Expression (4.5) as $P(X_2 > Z_1) = \int_0^{E[X_2]} F(x_2; s_1, p_b)\, f(x_2; s_2, p_a) dx_2 - \int_{E[X_2]}^{\infty} (1 - F(x_2; s_1, p_b))\, f(x_2; s_2, p_a) dx_2 + \int_{E[X_2]}^{\infty} f(x_2; s_2, p_a) dx_2$. If the processing time distribution has zero skewness, we have $\int_{E[X_2]}^{\infty} (1 - F(x_2; s_1, p_b))\ \ f(x_2; s_2, p_a) dx_2 = \int_0^{E[X_2]} F(x_2; s_1, p_b)\ f(x_2; s_2, p_a) dx_2$. Therefore,

$$P(X_2 > Z) = \int_{E[X_2]}^{\infty} f(x_2; s_2, p_a) dx_2 = 0.50, \tag{4.13}$$

since the mean and median are equal when the skewness is zero, as is true for the uniform distribution.

We also find the stockout probability when dual-sourcing is used with $s_1 = \frac{\delta + p_a U}{2p_a + p_b}$ for this stochastic system. Table 4.6 shows the stockout risks and lead times values when the solution to the deterministic, single-sourcing (with lot streaming) model is used in a stochastic

Table 4.6: Stockout risks (P) and lead times (L) when the optimal solution of the deterministic single-sourcing (with lot streaming) model, $s_{1,deter}$, is used in a stochastic system ($p_a = 1$).

p_b/p_a	$s_{1,deter}$	P	L
1	50	0.50	50
1.5	40	0.50	40
2	33	0.50	33

Table 4.7: Stockout risks (P) and lead times (L) when the optimal solution of the deterministic dual-sourcing model, $s_{1,deter}$, is used in a stochastic system ($p_a = 1$, $\delta = 0$).

p_b/p_a	$s_{1,deter}$	P	L
1	33	0.53	33
1.5	29	0.46	29
2	25	0.5	25

system.The optimal solution for the deterministic case is denoted by $s_{1,deter}$ in the table. L gives the expected lead time when $s_{1,deter}$ is used in a stochastic system and P is the resulting stockout risk. Table 4.7 shows the corresponding values for the optimal solution to the deterministic, dual-sourcing problem when $\delta = 0$. Referring to the third column in both the tables, note that the deterministic solution results in a rather large stockout risk value for both single-sourcing (with lot streaming) and dual-sourcing solutions. Note that, in view of Figure 4.5.2, a high stockout risk can be avoided by choosing an appropriately larger s_1 value.

Next, we investigate the performance of the the optimal solution for the JELS problem in a stochastic environment when the lot size is given. In particular, we consider the solution obtained by Hill (1999) using a deterministic production rate of $P = \frac{1}{p_a}$, and demand rate $D = \frac{1}{p_b}$. In addition to the production and demand rate, Hill (1999) also assumes stock holding costs at the first and second stages denoted by h_1 and h_2, respectively, with $h_2 > h_1$. Let $c = \frac{h_1}{h_2 - h_1}$. The optimal solution is given by $s_1 = \frac{U}{2}\left(1 - c\frac{p_a}{p_b}\right)$ when $c \leq \frac{p_b}{p_a} + \frac{2(p_b/p_a)}{1+p_b/p_a}$, and $s_1 = \frac{U}{1+p_b/p_a}$, otherwise. Note that the optimal sublot sizes for the case $c > \frac{p_b}{p_a} + \frac{2(p_b/p_a)}{1+p_b/p_a}$ are the same as the optimal sublot sizes obtained for the makespan minimization SLFS-LSP, i.e., they are geometric (Potts and Baker (1989b)).

Consider the case when $c = 3 > \frac{p_b}{p_a} + \frac{2(p_b/p_a)}{1+p_b/p_a}$, with $h_1 = 1$. The optimal sublot size, $s_1 = 50$. However, the resulting stockout risk for this solution is 0.5, which is quite high. An increment of s_1 to 62 decreases the stockout risk below $p_{max} = 0.001$. Similarly, if we consider the case when $c = 0.5 \leq \frac{p_b}{p_a} + \frac{2(p_b/p_a)}{1+p_b/p_a}$, we have $s_1 = 25$, with a stockout risk close to 1, again quite high. Therefore, using solutions to a deterministic model when processing times are stochastic can result in a high stockout risk. Hence, it is important to incorporate

the influence of stochasticity when analyzing a problem.

4.7 Concluding remarks and future research

In this chapter, we have studied two strategies for ordering material for a manufacturer: dual sourcing and single sourcing (with lot streaming). In dual sourcing, the material is supplied by two suppliers in shipments (sublots) that are δ-time units apart, while in single sourcing (with lot streaming), the material is supplied by one supplier in two shipments (sublots). We assume that the lead time of suppliers is composed solely of the time required by the supplier to produce the sublot. The sublot supplied by the supplier is then processed by the manufacturer. We consider the case when the processing time of the supplier and the manufacturer are stochastic, and the parameters of their density functions are dependent on the sublot size processed. In this aspect, our work is different from that presented on a similar problem in the literature. There is a risk of stockout between the arrival of two sublots when dual sourcing, or single-sourcing (with lot streaming) is used. In this chapter, we have studied the occurrence of stockout risks as well as the lead time and inventory cost incurred for both of these strategies. We have conducted this analysis in two steps. In the first step, we assume the manufacturer's processing rate to be a deterministic constant, and analytically show that single sourcing (with lot streaming) is better than dual sourcing when both result in a similar probability of stockout risk. Then, we consider the manufacturer's processing time to be stochastic as well and show the superiority of single-sourcing (with lot streaming) over dual sourcing. The deterministic equivalent version of our problem has been addressed in the literature, and it is known as lot streaming. We have thus extended the lot streaming concept to a stochastic environment. We have also shown that implementing the solution to deterministic problems in a stochastic environment, results in high stockout risks. In view of a lower ordering cost, lower inventory levels, competitive lead time performance, and a better stockout risk behavior, our investigation has shown that single-sourcing (with lot streaming) is an attractive alternative. Furthermore, the use of single-sourcing (with lot streaming) in stochastic supply chain environments is a promising area for future research.

Chapter 5

Lot Streaming in Assembly Systems

5.1 Introduction

In recent years, manufacturing strategies have focused on responding quickly to customer orders and providing customized products, while keeping the production costs low. The production of customized products makes it particularly difficult to respond quickly to customer orders since different orders may require very different components, which does not make it economically viable to maintain sufficient inventory of all possible components that may be required. As a result, the components are made-to-order by suppliers. In other cases, component inventory may not be maintained because frequent changes in design specifications render old components obsolete, or, simply, to minimize the amount of inventory held in the system. When an order consists of many identical items, the supplier may produce all items in a single production lot, but finished components may be transferred to the assembly stage in smaller sublots (transfer lots). Such a strategy of transferring finished goods to a downstream machine before the entire lot has completed processing at the current machine is called lot streaming. It results in an increase in the velocity of material flow through the system, and hence, improves time-based objectives. However, frequent transfers of small sublots results in an increment in the handling of items supplied by the suppliers. Therefore, it is essential to consider this increment in the handling of the items, and the resultant increment in material handling cost, while improving time-based system performance. Moreover, it is necessary to coordinate the parallel flow of material from different suppliers since assembly operations cannot be started until all the components have arrived.

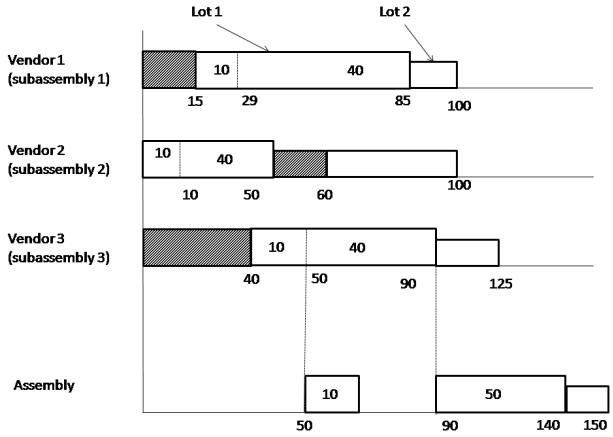


Figure 5.1.1: A two-stage assembly system

The flow of material from several parallel suppliers at the first stage to a single assembly facility at the second stage, has the structure of an assembly flow shop. A gantt chart representation of such a shop is provided in Figure 5.1.1. It depicts processing of two lots. The first lot is split into two sublots while the second lot is processed as is. In this chapter, we study the use of lot streaming in a two-stage assembly system where multiple lots (batches) are manufactured. Each lot consists of many identical items. Our problem involves determination of optimal number of sublots and sublot sizes for each lot, given the maximum number of sublot sizes for each lot that a supplier can use, and, an optimal sequence in which to process the lots. The objective is to minimize the weighted sum of the material handling cost and makespan cost. The makespan of a set of lots is the length of time interval between the start of production (by any supplier) and the completion time of the last sublot at the assembly stage. It represents machine utilization, and also, can be used as a substitute for maximum production lead time for a set of lots. Unit makespan cost can thus represent the opportunity cost for requiring an additional unit of time for finishing production, or incurring additional resource utilization per unit time. The two-stage assembly system considered here consists of a first stage with m distinct suppliers that produce components and a second stage, consisting of a single assembly line, where the components received from the suppliers are assembled into a final product. A supplier begins manufacturing the required components after receiving an order from the assembler, and the second-stage assembly process can begin only after all the components required for a product have been prepared and transported from the first stage. In our problem, several lots (batches) of different products are assembled at the second stage, and each lot consists of identical products.

Two-stage assembly systems are frequently encountered in industry. Such a system for multiple-lots arises in the commercial truck manufacturing industry where customers place orders for trucks in small batches, and customer orders may differ widely from each other. Customers often already have a fleet of trucks and want uniformity in the design of their

trucks in order to make it easier to maintain spare parts inventory and perform maintenance checks or repairs. Therefore, even core elements of a truck's design may have to be adapted to customer requests, and it may not be economically viable to maintain an inventory of all possible components. Consequently, orders are placed with suppliers only after receiving a customer order. Frequently, the truck manufacturer has long-term material purchasing agreements with suppliers, which specify prices of components. Material handling costs, on the other hand, are charged based on actual order sizes and delivery schedules. The number of deliveries plays an important role in determining handling costs, but it is not the only factor. A higher number of transfers, with possibly different sublot sizes for each transfer, increases the complexity of material handling activity for a supplier, and may increase the labor cost incurred, or may even require them to invest in more sophisticated material tracking systems. These additional costs are passed on to the manufacturer. Since truck manufacturers have to quote short lead times in order to remain competitive, an increment in the number of sublots from a supplier may be economically justifiable despite the increase in cost since it improves production lead time. Other examples of two-stage assembly systems can be found in personal computer manufacturing (Potts *et al.*, 1995) , fire engine assembly (Lee *et al.*, 1993) and scheduling queries in distributed database systems (Allahverdi and Al-Anzi, 2006a).

Scheduling multiple lots in an assembly system was first studied by Lee *et al.* (1993) for the objective of minimizing makespan for the case of $m = 2$. Potts *et al.* (1995) considered the problem for an arbitrary m and showed that the problem is NP-hard in the strong sense, even for $m = 2$. They showed that there exists an optimal permutation schedule, and an arbitrary sequence has a worst-case ratio bound of 2. They presented a heuristic with a worst-case bound of $2 - 1/m$. Hariri and Potts (1997) have also studied this problem. They developed a branch-and-bound-based algorithm for its solution and used a lower bound and several dominance properties to improve its performance. Haouari and Daouas (1999) also propose a branch-and-bound based algorithm for the case of $m = 2$. Powerful heuristics for this problem (with $m = 2$) have been proposed by Sun *et al.* (2003). Allahverdi and Al-Anzi (2006a) obtained a dominance relationship for the case when setup times are considered separately from processing times. They also propose evolutionary heuristics to solve this problem. Some other objective functions that have been studied for this system are maximum lateness (Allahverdi and Al-Anzi, 2006b), maximum lateness with setup times (Al-Anzi and Allahverdi, 2007), a bi-criteria of maximum lateness and makespan (Al-Anzi and Allahverdi, 2009), total completion time (Al-Anzi and Allahverdi, 2006), total weighted flow-time (Tozkapan *et al.*, 2003), a bi-criteria of makespan and mean completion time (Allahverdi and Al-Anzi, 2008), and total completion time with setup times (Allahverdi and Al-Anzi, 2009). Recently, Sarin *et al.* (2011) have studied the use of lot streaming when a single lot of identical items is produced in a two-stage assembly system for the objective of minimizing makespan. The number of sublots is determined a-priori in their problem, and optimal integer sublot sizes are obtained for the case when a setup time is incurred at the beginning of production on any machine. They consider the case when sublot sizes are equal across machines, i.e., the first sublot on the first machine is the same size as the first

sublot on any other machine. The multiple-lot case is addressed by Sarin and Yao (2011) in which the authors develop dominance properties and propose a branch-and-bound-based algorithm.

Our solution method for the problem on hand relies on obtaining optimal sublot sizes for the single-lot, two-machine flow shop, makespan minimization, lot streaming problems when the number of sublots is given. This problem for makespan minimization has been well-studied in the literature. Closed-form solutions or polynomial-time algorithms (in number of sublots) are available for various problems. Potts and Baker (1989a) have studied this problem when sublot sizes are continuously divisible, while Trietsch and Baker (1993b) have extended this to several other cases, including the restriction of limited transporter capacity and integer sublot sizes. Sriskandarajah and Wagneur (1999) have considered this problem in the presence of lot-detached setup times while the presence of sublot-attached setup times has been studied by Baker and Trietsch 2009,Alfieri *et al.* (2012), as well as, Mukherjee and Sarin (2014).

In this chapter, we study the use of lot streaming in an assembly system and explicitly consider the minimization of handling cost along with minimization of makespan. In Section 2, we introduce our notation and assumptions, and present some preliminary results. In Section 3, we present a polynomial-time algorithm for the case when a processing sequence for lots is given. In Section 4, we investigate three alternative solution methods for solving the problem when a processing sequence for lots is not given. We compare performances of methods based on two alternative integer programming formulations. Finally, concluding remarks are made in Section 5.

5.2 Notation, Assumptions and Some Preliminaries

In this section, we present the notation that we use, state our assumptions, and also, present some preliminary results. Henceforth, we refer to our problem as the Assembly Lot Streaming Problem (ALSP).

Consider the following notation:

Input Data:

$s :=$ number of suppliers at stage 1,
$l :=$ number of lots,
$p_i^v :=$ total processing and setup time for lot i, $i = 1, \cdots, l$ at supplier v, $v = 1, \cdots, s$,
$p_i :=$ total processing and setup time for lot i, $i = 1, \cdots, l$ at the assembly stage,
$n_i^v :=$ maximum number of sublots that may be used by supplier v, $v = 1, \cdots, s$, for lot i, $i = 1, \cdots, l$ (the supplier may use less than n_i^v sublots),

$s :=$ number of suppliers at stage 1,
$l :=$ number of lots,
$p_i^v :=$ total processing and setup time for lot i, $i = 1, \cdots, l$ at supplier v, $v = 1, \cdots, s$,
$p_i :=$ total processing and setup time for lot i, $i = 1, \cdots, l$ at the assembly stage,
$n_i^v :=$ maximum number of sublots that may be used by supplier v, $v = 1, \cdots, s$, for lot i, $i = 1, \cdots, l$ (the supplier may use less than n_i^v sublots),
$t_{iq}^v :=$ total (positive-valued) cost of transferring lot i from supplier v to the assembly stage when q sublots are used, v, $v = 1, \cdots, s$, $i = 1, \cdots, l$, and $q = 1, \cdots, n_i^v$,
$f_{iq}^v :=$ minimum time window during which supplier v can supply lot i to the assembly stage, and assembly can be completed at stage 2, when q sublots are used, $v = 1, \cdots, s$, $i = 1, \cdots, l$, and $q = 1, \cdots, n_i^v$.

Decision Variables:

$C_{max} :=$ Makespan value.

$$y_{ij} := \begin{cases} 1, & \text{if lot } i \text{ precedes lot } j \text{ (not neccessarily immediately) in the production sequence;} \\ 0, & \text{otherwise, } \forall i, j = 1, 2, \cdots, l,\ i \neq j, \end{cases}$$

$$x_{ij} := \begin{cases} 1, & \text{if lot } i \text{ is assigned to position } j \text{ in the production sequence;} \\ 0, & \text{otherwise,} \forall i, j = 1, 2, \cdots, l, \end{cases}$$

$$\delta_{iq}^v := \begin{cases} 1, \text{if } q \text{ sublots are used for lot } i \text{ by supplier } v; \\ 0, \text{otherwise,} \forall i = 1, \cdots, l,\ v = 1, \cdots, s,\ q = 1, \cdots, n_i^v, \end{cases}$$

Objective Function:

$\kappa C_{max} + \sum_{v=1}^{s} \sum_{i=1}^{l} \sum_{q=1}^{n_i^v} t_{iq}^v \delta_{iq}^v$,

where C_{max} represents the makespan of the assembly schedule, and κ is the unit makespan cost. Without loss of generality, we, henceforth, assume that $\kappa = 1$, since this is equivalent to dividing all handling costs by the constant κ (or multiplying the processing times by κ). Such a modification scales the optimal objective function value, but does not affect optimal decision variable values.

Next, we further explain the parameters t^v_{iq} and f^v_{iq}. We define t^v_{iq} to be the total handling cost incurred when components for lot i are transferred from supplier v to the assembly stage in q sublots. This includes the costs incurred by the supplier due to the added transportation and handling costs as well as the increased material handling costs at the assembly stage. Depending on the amount and complexity of the material handling activity added by increasing the number of sublots, the supplier may charge varying costs. Hence, the increase in handling cost $t^v_{iq+1} - t^v_{iq}$ is not necessarily constant. We assume that increasing the number of sublots increases handling cost, i.e.,

$$t^v_{iq} \;<\; t^v_{iq+1}. \tag{5.1}$$

This is a reasonable assumption since any additional transportation activity will be associated with an increase in handling cost.
Next, we discuss the parameter f^v_{iq}. Consider the length of time interval between the beginning of component production for a lot i by a supplier v and the time lot i completes assembly at the second stage. The parameter f^v_{iq} represents a lower bound on this value. Note that this lower bound is the minimum makespan for a two-machine flow shop, consisting of supplier v at the first stage and the assembly machine at the second stage, when lot i is processed alone and q sublots are used. For example, in case a single sublot is used and transfer times are negligible, we have $f^v_{i1} = p^v_i + p_i$. Note that, while we do not explicitly mention lot-attached/detached setup and transportation times in our parameter set, they are implicitly considered in the parameter f^v_{iq}. As another example, consider the case when a supplier v provides a lot i of 120 components. Let supplier v experience a setup time of 0.5 days before beginning to process lot i, and a processing time of 0.05 days per component. Let the assembly stage require no setup time, and processing time of 0.01 days per item. Also, let each transfer of material (that is, transfer of each sublot) require 1 day of transportation time. When all the material is transferred using a single transfer, the minimum makespan would be 8.7 days. Hence, $f^v_{i1} = 8.7$ days. The schedule resulting from the use of one sublot is shown in Figure 5.2.1(a). In case two sublots are used, the optimal sublot sizes would be $u_1 = 100$ for the first sublot, and $u_2 = 20$ for the second sublot, resulting in a minimum makespan, $f^v_{i2} = 7.7$ days. The schedule resulting from the use of two sublots is shown in Figure 5.2.1(b). The value of f^v_{iq} must be obtained by analyzing the two-machine flow shop, single-lot, makespan minimization problem associated with each lot i processed by each supplier v in q sublots. As noted earlier, these problems have been well-studied in the literature and closed-form solutions or simple methods are available to quickly determine this parameter for a wide variety of production environments. The optimal sublot sizes for a given number of sublots, q, is obtained in time polynomial in n^v_i (which is polynomial in

the length of the input $\left(s + l + \max_{i,v} n_i^v\right)$ for our original assembly problem).

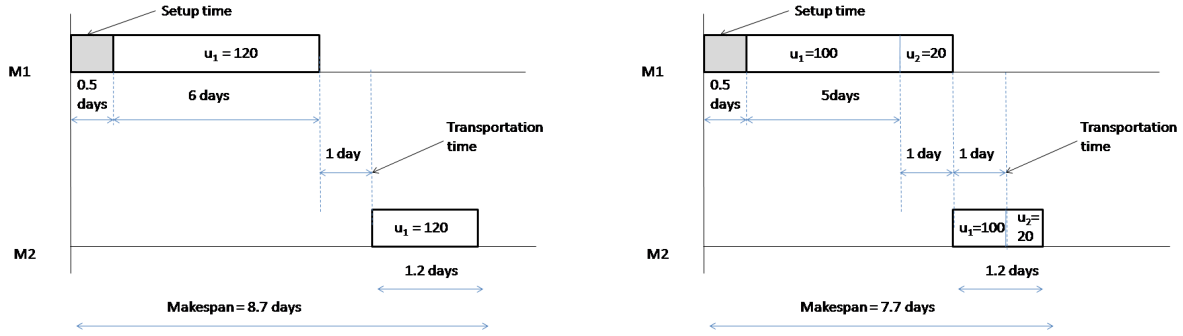


Figure 5.2.1: A schedule representing a two-stage single-lot flow shop problem associated with a lot for (a) one sublot (b) two sublots.

In most cases, the value of f_{iq}^v may be expected to be a monotone decreasing function of q. However, it is possible that certain values of q in $1, \cdots, n_i^v$ are infeasible due to operational reasons, or, multiple shipment methods are available for the same number of sublots, each of which results in a different makespan and handling cost, or, for some q, $f_{iq}^v > f_{iq-1}^v$. By (5.1), we also have $t_{iq}^v > t_{iq-1}^v$ since $q > q - 1$. So, clearly, the use of q sublots is never optimal, and we can ignore such a solution. Therefore, we interpret q to index the alternative lot streaming strategies that are in decreasing order of f_{iq}^v (and are in increasing order of number of sublots, but not necessarily increasing by one sublot at-a-time). They continue to be in increasing order of handling cost. Since values of q that are clearly never optimal, i.e., when the handling cost and associated f_{iq}^v values are both higher than some other alternative q', are excluded, we have $f_{iq+1}^v \leq f_{iq}^v$, $\forall v = 1, \cdots, s$, and $i = 1, \cdots, l$, i.e., the minimum makespan does not increase with an increase in the number of sublots. We also assume that the total production time for a supplier or for the assembly stage is independent of the number of sublots used. This prevents us from considering cases where sublot-attached setup or removal times are present, but does not preclude the cases where sublot transfer times or lot-attached/detached setup or removal times are present. The method discussed here also allows consideration of restrictions such as equal sublot sizes, having an upper bound on the size of sublots, and the like. to be considered. More importantly, each lot, from each vendor, is allowed to have different production constraints (for example, one lot from vendor v may require lot-attached setup times, while another from vendor u may not). This permits the modeling of real-life suppliers who are often very different from each other. The value of f_{iq}^v is affected by the presence of these factors.

Similar to Lee *et al.* (1993) and Potts *et al.* (1995), we assume that a product can begin

assembly only after all its components are available, i.e., we do not consider the case when some components arrive during the assembly process. This is reasonable since the time required for assembling a single unit is usually much smaller than the time required to source components. We consider the case when intermingling of sublots from different lots is not permitted. An infinite buffer is assumed to exist between the suppliers and the manufacturer.

We first note that there exists an optimal schedule that is a permutation schedule for the problem without lot streaming, as shown by Potts *et al.* (1995). Their argument holds true as well for the case when lot streaming is used. Therefore, we restrict our attention to permutation schedules.

In order to develop an optimal schedule, optimal sublot sizes must be evaluated. We next show that there exists an optimal solution for the ALSP where the sublot sizes of each lot corresponds to optimal sublot sizes for a corresponding two-machine flow shop problem. Consider a two-machine flow shop, consisting of a supplier v^* at the first stage and the assembly machine at the second stage, and a single lot i^* is produced and transferred in q^* sublots. For the two-machine problem, q^* is fixed (note that there may be some other value of q^* for which the joint objective function is minimized for the two-machine problem, but that value is not relevant to any results presented in this chapter). Let sublot sizes $u_1,\ u_2, \cdots, u_{q^*}$ minimize the makespan for the two machine problem. Then if, in an optimal schedule for the ALSP, supplier v^* uses q^* sublots to transfer lot i^*, sublot sizes $u_1,\ u_2, \cdots, u_{q*}$ are optimal for that lot. We present it formally below.

Proposition 5.1. *If in an optimal solution for the ALSP $\delta^{q^*}_{i^*v^*} = 1$, then, sublot sizes $u_1,\ u_2, \cdots, u_{q*}$ for lot i^* from supplier v^* are optimal, where $u_1,\ u_2, \cdots, u_{q^*}$ are optimal sublot sizes for two-machine, flow shop, lot streaming problem corresponding to supplier v^* and the assembly machine, obtained using q^* sublots.*

Proof. We prove the result by construction. Consider an optimal schedule for the ALSP in which supplier v^* uses q^* sublots of sizes $u'_1,\ u'_2, \cdots, u'_{q^*}$. In order to simplify notation, assume, without loss of generality, that the optimal sequence in which the lots are processed is $1, \cdots, l$. The makespan for such a production sequence is given by,

$$\underset{\{k,v\}}{max} \left(\sum_{i=1}^{k-1} p_i^v + \sum_{q=1}^{n_k^v} f'^v_{kq} \delta^v_{kq} + \sum_{i=k+1}^{l} p_i \right), \tag{5.2}$$

where f'^v_{kq} is the makespan of the associated two-machine, single-lot flow shop problem when the sublot sizes $u'_1,\ u'_2, \cdots, u'_q$ are used by the supplier for lot k and other terms in (5.2) are independent of the sublot sizes used. By definition $f'^v_{kq} \geq f^v_{kq}$ for all $k = 1, \cdots, l$ and $v = 1, \cdots, s$. Hence, replacing sublot sizes $u'_1,\ u'_2, \cdots, u'_{q^*}$for lot i^* from supplier v^* with $u_1,\ u_2, \cdots, u_{q^*}$, and hence, replacing $f'^{v^*}_{i^*q^*}$with $f^{v^*}_{i^*q^*}$ $\left(\leq f'^{v^*}_{i^*q^*}\right)$ does not increase the makespan. Also note that only the sublot sizes were altered, the number of sublots used remains unaltered as q^* in both the initial and the final schedules. So, the transportation cost is not

altered since it is a function of the number of sublots alone. Hence, the schedule remains optimal. □

Corollary 5.1. There exists an optimal solution for the ALSP where the sublot sizes for each lot correspond to optimal sublot sizes for the associated two-machine, single-lot, problem with an identical number of sublots. The number of sublots used for each lot, from each supplier, may be different.

Proof. In any schedule we have $C_{max} - \sum_{i=1}^{k-1} p_i^v - \sum_{i=k+1}^{l} p_i \geq \sum_{q=1}^{n_k^v} f_{kq}^{\prime v} \delta_{kq}^v$. By definition, $f_{kq}^{\prime v} \geq f_{kq}^v$. So, $C_{max} - \sum_{i=1}^{k-1} p_i^v - \sum_{i=k+1}^{l} p_i \geq \sum_{q=1}^{n_k^v} f_{kq}^v \delta_{kq}^v$ and using optimal sublot sizes for the associated two-machine, single-lot, problem with an identical number of sublots is feasible. Also, the material handling cost remains unchanged since the number of sublots used is not altered.

□

As a consequence of the result presented above, we can solve the ALSP in two steps. In step one, we obtain the optimal sublot sizes, minimum makespan (for the associated two-machine flow shop problem) f_{iq}^v, and handling cost t_{iq}^v for all $i = 1, \cdots, l$, $v = 1, \cdots, s$, and $q = 1, \cdots, n_i^v$. In the second step, we then use the f_{iq}^v and t_{iq}^v values determined in step 1 for all $i = 1, \cdots, l$, $v = 1, \cdots, s$, and $q = 1, \cdots, n_i^v$ to obtain optimal production sequence and number of sublots.

Such a decomposition into two steps is important since step 1 is closely linked to the production and transportation scenario considered. The algorithm used to solve the two-machine flow shop lot streaming problems in this step will be peculiar to the production scenario considered. Step 2, in contrast, is not related to the specifics of the production scenario and remains unchanged for all the ALSP instances. Given the large amount of existing literature that has addressed the problems included in step 1, in this chapter, we will assume that step 1 has been solved a-priori, and therefore, will focus on solving the problem belonging to step 2. Hence, our problem can be stated as:

Assembly Lot Streaming Problem: Given values of parameters f_{iq}^v and t_{iq}^v for $i = 1, \cdots, l$, $v = 1, \cdots, s$, and, $q = 1, \cdots, n_i^v$, for two-machine flow shop problems, find the optimal sequence for processing the lots and the optimal number of sublots used for each lot in order to minimize the sum of the makespan cost (MC), κC_{max}, and material handling cost (MHC), where the MC is a scalar multiple of the makespan

$$C_{max} = \underset{\{k,v\}}{max} \left\{ \sum_{i=1 i \neq k}^{l} p_i^v y_{ik} + \sum_{q=1}^{n_k^v} f_{kq}^v \delta_{kq}^v + \sum_{i=1, i \neq k}^{l} p_i y_{ki} \right\}, \tag{5.3}$$

and the MHC is given by

$$\text{MHC} = \sum_{v=1}^{s} \sum_{i=1}^{l} \sum_{q=1}^{n_k^v} t_{kq}^v \delta_{kq}^v. \tag{5.4}$$

Note that there exists an optimal schedule in which the lots are produced continuously, without intermittent idle time, by both the suppliers and at the assembly stage. This is always feasible since we assume an infinite buffer. Also, delaying production on the second machine does not increase transportation costs. Therefore, given a feasible schedule with a makespan of C_{max}, production of each lot at the assembly stage can be postponed until all the lots and sublots are processed contiguously, with no intermittent idle times, without increasing the transportation cost or the makespan. Similarly, all suppliers can begin production at time zero and process all the lots without any intermittent idle time. This does not change the number of transfers, and clearly, earlier production at the first stage does not increase the makespan. Therefore, the search for optimal schedules can be restricted to those schedules in which the suppliers process material as soon as possible and assembly is completed as late as possible, with the final lot, l, completed at time C_{max}.

Also, we use the term C_{max}^{kqv} to denote the value of

$$C_{max}^{kqv} = \sum_{i=1 i \neq k}^{l} p_i^v y_{ik} + f_{kq}^v + \sum_{i=1, i \neq k}^{l} p_i y_{ki}. \tag{5.5}$$

This is the makespan that would result if lot k from supplier v, transferred in q sublots, were the bottleneck lot.

5.3 A Polynomial-time Algorithm to Obtain Optimal Number of Sublots for a Given Production Sequence

In this section, we develop a polynomial-time algorithm to obtain optimal number of sublots used by all the suppliers for all the lots when the production sequence is given. Without loss of generality, we assume the production sequence to be $1, \cdots, l$. As a first step, we express the minimum MHC for a given production sequence as an upper semi-continuous step function of the makespan.

For a schedule with makespan C_{max} and production sequence be $1, \cdots, l$, the start time of lot k at supplier v in such a schedule is given by $\sum_{i=1}^{k-1} p_i^v$ and completion time on the assembly machine by $C_{max} - \sum_{i=k+1}^{l} p_i$. So, if we consider the schedule of lot k for supplier v and the assembly machine in isolation, we obtain a feasible schedule for a lot k processed as a single lot by supplier v and the manufacturer, i.e., for every feasible schedule of a lot

in the ALSP there exists a feasible schedule for a single-lot two-machine flow shop problem. The makespan for such a single-lot, two-machine flow shop is given by the time-window, $\left[\sum_{i=1}^{k-1} p_i^v, C_{max} - \sum_{i=k+1}^{l} p_i\right]$, over which lot k is processed in the ALSP schedule. For clarity, the time window is shown in Figure 5.3.1 for vendor 2. (Note that the time-window represents the makespan of the lot in the corresponding two-machine problem.) Clearly, if the length of this window is smaller than $p_i^v + p_i$, then lot streaming must be used, but, in order to keep transfer costs low, the number of transfers must be minimal in an optimal schedule. In general, the minimum number of transfers that supplier v needs to use for lot k is given by

$$min\ q:\ f_{kq}^v \leq C_{max} - \sum_{i=k+1}^{l} p_i - \sum_{i=1}^{k-1} p_i^v,\ 1 \leq q \leq n_i^v. \tag{5.6}$$

Hence, the MHC incurred by a lot k from supplier v is given by t_{kq}^v, where q is obtained from (5.6). Note that, in any optimal solution, at least one of the lots k, $k = 1, \cdots, l$, for some supplier $v = 1, \cdots, s$ must satisfy (5.6) at equality, because otherwise, C_{max} could be reduced without increasing the total MHC. This would contradict optimality. The following result states this formally.

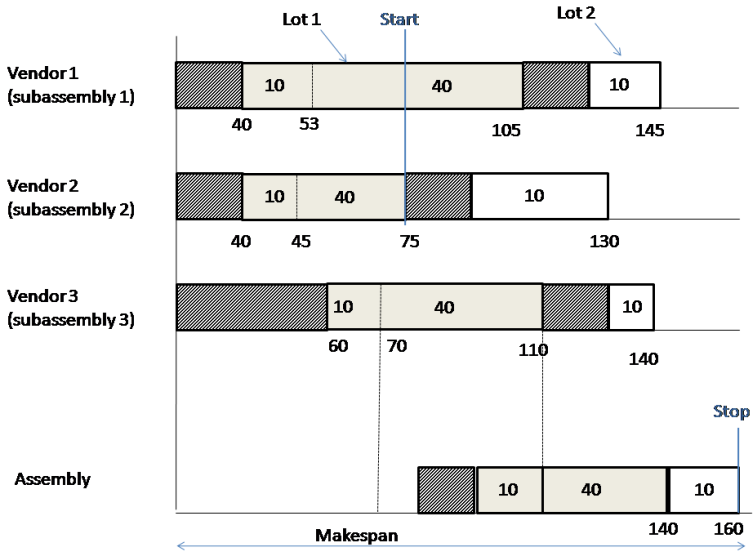


Figure 5.3.1: The time-window available to lot 2 (Start to Stop).

Proposition 5.2. *Given a production sequence* $1, \cdots, l$*, there exists an optimal schedule for the ALSP such that* $C_{max} = \sum_{i=1}^{k^*-1} p_i^v + f_{k^*q^*}^{v^*} + \sum_{i=k^*+1}^{l} p_i = C_{max}^{k^*q^*v^*}$*, for some* $k^* \in \{1, \cdots, l\}, v^* \in 1, \cdots, s$ *and* $q^* \in \{1, \cdots, n_i^v\}$ *and the number of sublots used by all other lots is given by: min* $q:\ f_{iq}^v \leq C_{max} - \sum_{i=k+1}^{l} p_i - \sum_{i=1}^{k-1} p_i^v,\ 1 \leq q \leq n_i^v$.

Proof. Note that in an optimal schedule, there must exist at least one bottleneck lot i and one supplier v such that it is not possible to reduce the makespan any further, i.e., (5.6) holds

at equality, since, if (5.6) were strict inequality for all lots, it would be possible to reduce the makespan without increasing the transportation cost. All other lots use the minimum number of sublots that would minimize the handling cost, i.e., the smallest feasible value of q. Also, using q sublots is feasible only if $f_{iq}^v \leq C_{max} - \sum_{i=k+1}^{l} p_i - \sum_{i=1}^{k-1} p_i^v,\ 1 \leq q \leq n_i^v$. □

Note that rearranging the expression in (5.6), we get $\sum_{i=1}^{k-1} p_i^v + f_{kq}^v + \sum_{i=k+1}^{l} p_i \leq C_{max}$. Using (5.5), the left hand side can be replaced by C_{max}^{kqv} to obtain $C_{max}^{kqv} \leq C_{max}$. Therefore, the MHC for lot k is given by

$$\min_q t_{kv}^q : C_{max}^{kqv} \leq C_{max}. \tag{5.7}$$

Also, t_{kv}^q is strictly increasing in q by assumption. There are $N = \sum_{v=1}^{s} \sum_{k=1}^{l} n_k^v$ possible values of C_{max}^{kqv}, each associated with some value of $k = 1, \cdots, l$, $q = 1, \cdots, n_k$, and $v = 1, \cdots, s$. Let these values be sorted from the largest to the smallest value. We indicate the largest value by $C_{max}^{kqv}[1]$, which denotes that lot k, from supplier v using q sublots results in the highest makespan value when it is the bottleneck, similarly let $f_{kq}^v[w]$ and $t_{kq}^v[w]$ denote the value of f_{kq}^v and t_{kq}^v associated with the k, q, v that result in the w^{th} largest makespan. We next develop an algorithm, designated Algorithm 1 to determine the minimum cost for a given sequence in $O(N \log N + Nl)$ time.

Algorithm 1 Optimal for a given sequence

Step 0 Set $C_{max}^{kqv} = 0, \forall k = 1, \cdots, l,\ v = 1, \cdots, s,\ q = 1, \cdots, n_k^v$, $Cost_{best} = \infty$, $C_{max} = 0$ and $Cost_{temp} = \sum_{k=1}^{l} \sum_{v=1}^{s} t_{k1}^v$,

Step 1 For all $k \in \{1, \cdots, l\}, v \in 1, \cdots, s,\ q \in \{1, \cdots, n_k^v\}$, calculate the makespan C_{max}^{kqv} when sublot q of lot k from supplier v is the 'bottleneck' sublot, i.e., evaluate $C_{max}^{kqv} = \sum_{i=1}^{k-1} p_i^v + f_{kq}^v + \sum_{i=k+1}^{l} p_i$.

Step 2 Sort C_{max}^{kqv} in descending order and index the values by $w = 1, \cdots, N$. Set $Cost_{best} = Cost_{temp} + C_{max}^{kqv}[1]$.

Step 3 For $w = 1, \cdots, N$,

3.1 For the k, q, v values associated with the w^{th} largest makespan, check if $q = n_k^v$. If $q < n_k^v$ it is possible to increment the number of sublots, so proceed to step 3.2. Otherwise, the minimum possible makespan has been reached, complete steps 3.2 and 3.3 and then exit the loop.

3.2 if $w > 1$, set $Cost_{temp} = Cost_{temp} + \left(t_{kq+1}^v[w] - t_{kq}^v[w]\right) + \left(C_{max}^{k'q'v'}[w-1] - C_{max}^{kqv}[w]\right)$.

3.3 If $Cost_{temp} < Cost_{best}$, set $Cost_{best} = Cost_{temp}$.

Step 4 Report C_{best} as the minimum cost possible for that sequence.

In step 0, we initialize the makespan values that would result if lot k, from supplier v, using q sublots were the bottleneck. $Cost_{temp}$ and $Cost_{best}$ are the objective function values for any schedule under consideration, and, for an optimal schedule, respectively. Since the minimum MHC is the cost of using one sublot for all lots, we initialize $Cost_{temp}$ to that value. In step 1, we evaluate C_{max}^{kqv} for all values of k, q, and, v. By Proposition (5.2), one of these C_{max}^{kqv} values is the makespan of an optimal schedule. We next need to obtain the MHC resulting from each possible value of C_{max}^{kqv}. We do so by first ordering C_{max}^{kqv} in descending order in step 2. Let the w^{th} largest value of makespan be C_{max}^{kqv}, followed by $C_{max}^{k'q'v'}$, the next $(w-1^{th})$ largest makespan value. We have $C_{max}^{kqv} > C_{max}^{k'q'v'}$. For now, assume that we have a starting schedule with $C_{max} = C_{max}^{kqv}$ and know the number of sublots used by all lots. Then, reducing C_{max} from C_{max}^{kqv} to $C_{max}^{k'q'v'}$ would require an increment in the number of sublots for lot k from supplier v alone since for all other lots, the inequality (5.7) continues to hold true. As a result of incrementing the number of sublots of lot k from v, increases the MHC by $\left(t_{kq}^{v} - t_{kq+1}^{v}\right)$ and the makespan reduces to $C_{max}^{k'q'v'}$. So, the change in the objective function resulting from this increment in the number of sublots and decrement in makespan is given by $\left(C_{max}^{kqv} - C_{max}^{k'q'v'}\right) + \left(t_{kq+1}^{v} - t_{kq}^{v}\right)$. Note that this may be positive or negative. Also, note that, the maximum makespan (for $w = 1$) is achieved when the number of sublots used by all sublots is one, i.e., no lot uses lot streaming. So, we begin with a known schedule for $w = 1$ and update the number of sublots used by each lot as we increment w. In step 3, the number of sublots used by the current bottleneck lot is incremented and the makespan is reduced until a new bottleneck is reached. The change in the objective function is recorded, so that the minimum value can be found. This value is reported in step 4. Note that ordering the lots in decreasing C_{max}^{kqv} values allows us to increment the number of sublots of exactly one lot at a time since the number of sublots used by a lot increases only at each of the $C_{max}^{kqv}[w]$ values. So, each increment can be done in constant time. We stop incrementing the number of sublots when the bottleneck lot is already using the maximum number of sublots permitted for that lot. The largest possible number of iterations that may be required to reach this condition is $N - ls + 1$ (since each of the ls lots has a last sublot). The computational complexity of the above steps is as follows: step 0, $O(ls)$; step 1, $O(Nl)$; step 2, $O(NlogN)$; step 3, $O(N)$. Clearly, the overall complexity is determined by step 1 or step 2, depending upon the relative size of l and $logN$. Hence, the algorithm has a time complexity of $NlogN + Nl$. It is possible that consecutive makespan values are equal. For example, say two consecutive values C_{max}^{kqv} and C_{max}^{jru} are equal. Let C_{max}^{jru} be lower in the sequence. In order to obtain an optimal solution we require that all makespan values smaller than C_{max}^{kqv} have a MHC higher than that for C_{max}^{kqv} (and hence C_{max}^{jru}) by $\left(t_{kq}^{v} - t_{kq+1}^{v}\right) + \left(t_{jr}^{u} - t_{jr+1}^{u}\right)$, and, at least one of C_{max}^{kqv} and C_{max}^{jru} has the correct MHC value associated with this makespan and the other has a higher value. Both these conditions are satisfied by the algorithm.

Next, we solve the same problem using an LP. Consider a general production sequence defined by order indicators y_{ij}. Note that we must have

$$y_{ij} + y_{ji} = 1 \,\forall i \neq j,\; i, j = 1, \cdots, l. \tag{5.8}$$

We now have $C^{iqv}_{max} = \sum_{k=1}^{l} p^v_k y_{ki} + f^v_{iq} + \sum_{k=1}^{l} p_k y_{ik}$. Using (5.5), we get $C^{iqv}_{max} = \sum_{k=1}^{l} p^v_k y_{ki} + f^v_{iq} + \sum_{k=1}^{l} p_k - \sum_{k=1}^{l} p_k y_{ik}$, rearranging, we get

$$C^{iqv}_{max} = \sum_{k=1}^{l} (p^v_k - p_k) y_{ki} + f^v_{iq} - p_i + \sum_{k=1}^{l} p_k. \tag{5.9}$$

The term $-p_i$ on the right side is due to the fact that $y_{ii} = 0$. The total objective function for the ALSP is obtained by adding the material handling cost to the makespan. Let C^{iqv}_{max} be the makespan of a schedule, then we can determine the MHC incurred by a lot j from vendor u using (5.7), with $C_{max} = C^{iqv}_{max}$ and C^{jru}_{max} in place of C^{kqv}_{max} to indicate makespan values associated with all $j = 1, \cdots, l$, $r = 1, \cdots, s$, $u = 1, \cdots, n^u_j$, and obtain r as $\overset{\min}{r} t^r_{ju} : C^{jru}_{max} \leq C^{iqv}_{max}$. Let $z_{jruiqv} = 1$ indicate that $C^{iqv}_{max} \leq C^{jru}_{max}$. Then, we may write the MHC of lot j from vendor u as $t^u_{j1} + \sum_{r=1}^{n^u_j} \left(t^u_{jr+1} - t^u_{jr}\right) z_{jruiqv}$ (we allow $t^u_{jn^u_j+1}$ to take an arbitrary value. Since $z_{jn^u_j+1uiqv} = 1$ is not feasible, in the proposed LP-Formulation, we use a constraint to prevent the use of this value). As a result, the total objective function when lot i from vendor v, transferred in q sublots, is the bottleneck, is given by $C^{iqv}_{max} + \sum_{j=1}^{l} \sum_{u=1}^{s} \left(t^u_{j1} + \sum_{r=2}^{n^u_j} \left(t^u_{jr+1} - t^u_{jr}\right) z_{jruiqv}\right)$. Here the MHC of all lots has been added. Expanding C^{iqv}_{max} using (5.9), we get the total objective function value when iqv is the critical sublot as:

$$\sum_{k=1}^{l} (p^v_k - p_k) y_{ki} + f^v_{iq} - p_i + \sum_{k=1}^{l} p_k + \sum_{j=1}^{l} \sum_{u=1}^{s} t^u_{j1} + \sum_{j=1}^{l} \sum_{u=1}^{s} \sum_{r=1}^{n^u_j} \left(t^u_{jr+1} - t^u_{jr}\right) z_{jruiqv} \tag{5.10}$$

Since the term $\sum_{k=1}^{l} p_k + \sum_{j=1}^{l} \sum_{u=1}^{s} t^u_{j1}$ is a constant that is added irrespective of which sublot acts as a bottleneck, it can be ignored in (5.10) for the purpose of obtaining an optimal schedule. Let $\lambda_{iqv} = 1$ indicate that $C_{max} = C^{iqv}_{max}$, i.e., lot i from vendor v is shipped in q sublots and is the bottleneck sublot. In addition to these terms, we also use the product of these terms. Such as $z\lambda_{jruiqv} = \lambda_{iqv} \times z_{jruiqv}$. Other product terms are defined similarly. As input data, this formulation uses the production sequence, presented by a set of y_{ij} values. Then multiplying (5.10) with λ_{iqv} and adding over all possible value of i, q, v results in the objective function for the LP shown below.

LP-Formulation 5.3
minimize:

$$\sum_{i=1}^{l} \sum_{v=1}^{s} \sum_{q=1}^{n^v_i} \left(f^v_{iq} - p_i\right) \lambda_{iqv} + \sum_{i=1}^{l} \sum_{v=1}^{s} \sum_{q=1}^{n^v_i} \sum_{k=1,\, k\neq i}^{l} \left(p^v_k - p_k\right) y\lambda_{kiqv}$$

$$+\sum_{i=1}^{l}\sum_{v=1}^{s}\sum_{q=1}^{n_i^v}\sum_{j=1}^{l}\sum_{u=1}^{s}\sum_{r=1}^{n_j^v}\left(t_{jr+1}^{v}-t_{jr}^{v}\right)z\lambda_{jruiqv}$$

subject to:

$$\sum_{i=1}^{l}\sum_{v=1}^{s}\sum_{q=1}^{n_i^v}\lambda_{iqv} = 1 \tag{5.11}$$

$$\begin{aligned}
&\sum_{k=1,k\neq j}^{l}\left(p_k^u-p_k\right)yz_{kjruiqv} && \sum_{k=1,k\neq j}^{l}\left(p_k^u-p_k\right)y_{kj}- \\
&-\sum_{k=1,\,k\neq i}^{l}\left(p_k^v-p_k\right)\bar{y}z_{kjruiqv} && \geq \sum_{k=1,\,k\neq i}^{l}\left(p_k^v-p_k\right)y_{ki} \\
&+\left(f_{jr}^u-p_j+f_{iq}^v-p_i\right)z_{jruiqv} && +\left(f_{jr}^u-p_j+f_{iq}^v-p_i\right) \\
& && \forall i,j=1,\cdots,l,\ u,v=1,\cdots,s, \\
& && q=1,\cdots,n_i^q,\ r=1,\cdots,n_j^u
\end{aligned} \tag{5.12}$$

$$\begin{aligned}
&\sum_{k=1,k\neq j}^{l}\left(p_k^u-p_k\right)yz_{kjruiqv} \\
&-\sum_{k=1,\,k\neq i}^{l}\left(p_k^v-p_k\right)\bar{y}z_{kjruiqv} && \geq 0 \\
&+\left(f_{jr}^u-p_j+f_{iq}^v-p_i\right)z_{jruiqv} && \forall i,j=1,\cdots,l,\ u,v=1,\cdots,s, \\
& && q=1,\cdots,n_i^q,\ r=1,\cdots,n_j^u
\end{aligned} \tag{5.13}$$

$$\begin{aligned}
z\lambda_{jruiqv} &\leq z_{jruiqv} \\
z\lambda_{jruiqv} &\leq \lambda_{iqv} \\
z\lambda_{jruiqv} &\geq z_{jruiqv}+\lambda_{iqv}-1
\end{aligned} \tag{5.14}$$

$$0 \geq z_{jn_j^u uiqv}+\lambda_{iqv}-1 \tag{5.15}$$

$$\begin{aligned}
y\lambda_{kiqv} &\leq y_{ki} \\
y\lambda_{kiqv} &\leq \lambda_{iqv} \\
y\lambda_{kiqv} &\geq y_{ki}+\lambda_{iqv}-1
\end{aligned} \tag{5.16}$$

$$
\begin{aligned}
yz_{kjruiqv} &\leq y_{kj} && (5.17)\\
yz_{kjruiqv} &\leq z_{jruiqv} && (5.18)\\
yz_{kjruiqv} &\geq z_{jruiqv} + y_{kj} - 1 && (5.19)
\end{aligned}
$$

$$
\begin{aligned}
\bar{y}z_{kjruiqv} &\leq y_{ki}\\
\bar{y}z_{kjruiqv} &\leq z_{jruiqv} && (5.20)\\
\bar{y}z_{kjruiqv} &\geq z_{jruiqv} + y_{ki} - 1
\end{aligned}
$$

$\lambda_{iqv}, z_{jruiqv}, y\lambda_{kiqv}, yz_{kjruiqv}, \bar{y}z_{kjruiqv}$ are continuous and lie in the interval $[0,1]$ $\forall i,j,k = 1,\cdots,l$, $r,q = 1,\cdots,n_i^v$, $u,v = 1,\cdots,s$.

Constraint (5.11) restricts the number of critical sublots to one, while constraints (5.12) and (5.13) force the z_{jruiqv} variables to take the correct 0/1 values when $y_{ij} \in \{0,1\}$. The constraints (5.14),(5.16),(5.17),(5.18),(5.19) and (5.20) linearize the product terms. For example, consider $yz_{kjruiqv}$. Constraint (5.17) forces $yz_{kjruiqv} = 0$ when $y_{kj} = 0$, and, constraints (5.18) and (5.19) force $yz_{kjruiqv} = z_{jruiqv}$ when $y_{kj} = 1$. Hence, $yz_{kjruiqv} = y_{kj}z_{jruiqv}$.

Proposition 5.3. *Given a transitivity matrix* $\mathbf{Y}$ *with each of its elements* $y_{ij} = 1$*, if* i *precedes* j*, and* 0 *otherwise, an optimal solution for the LP Formulation 5.3 yields an optimal solution for the ALSP for that sequence.*

Proof. Note that, by inequalities (5.16), $y_{ki} = 0 \implies y\lambda_{kiqv} = 0$, and, $y_{ki} = 1 \implies y\lambda_{kiqv} = \lambda_{iqv}$. Similarly, $z\lambda_{jruiqv} = z_{jruiqv}$, $yz_{kjruiqv} = z_{jruiqv}$, $\bar{y}z_{kjruiqv} = z_{jruiqv}$, when $y_{ki} = 1$, respectively, and they are equal to zero when $y_{ki} = 0$. Using these facts, we can rewrite (5.12) as $\sum_{k=1,k\neq j}^{l}(p_k^u - p_k)\,y_{kj}z_{jruiqv} - \sum_{k=1,\,k\neq i}^{l}(p_k^v - p_k)\,y_{kj}z_{jruiqv} + \left(f_{jr}^u - p_j + f_{iq}^v - p_i\right)z_{jruiqv} \geq \sum_{k=1,k\neq j}^{l}(p_k^u - p_k)\,y_{kj} - \sum_{k=1,\,k\neq i}^{l}(p_k^v - p_k)\,y_{ki} + \left(f_{jr}^u - p_j + f_{iq}^v - p_i\right)$ $\forall i,j = 1,\cdots,l$, $u,v = 1,\cdots,s, q = 1,\cdots,n_i^q$, $r = 1,\cdots,n_j^u$, i.e., $(C_{max}^{jru} - C_{max}^{iqv})\,z_{jruiqv} \geq C_{max}^{jru} - C_{max}^{iqv}$. Similarly, (5.13) represents the constraint $(C_{max}^{jru} - C_{max}^{iqv})\,z_{jruiqv} \geq 0$. Taking these constraints together $C_{max}^{iqv} < C_{max}^{jru} \implies z_{jruiqv} = 1$ and $C_{max}^{iqv} > C_{max}^{jru} \implies z_{jruiqv} = 0$. When $C_{max}^{iqv} = C_{max}^{jru}$ the constraints (5.12) and (5.13) do not affect z_{jruiqv}, however, since the objective function is increasing in z_{jruiqv}, and since this variable does not occur in any other constraint, we have $C_{max}^{iqv} = C_{max}^{jru} \implies z_{jruiqv} = 0$. So, the values of z_{jruiqv}, $yz_{kjruiqv}$, $\bar{y}z_{kjruiqv}$ are fixed and binary for any given set of $y_{ij} \in \{0,1\}$ values. Also, note that $C_{max}^{iqv} = \sum_{k=1,\,k\neq i}^{l}(p_k^v - p_k)\,y_{ki} + f_{iq}^v - p_i + P$, where $P = \sum_{i=1}^{l} p_i$. Since we have $y_{ij} \in \{0,1\}$ and we have shown $z_{jruiqv} \in \{0,1\}$, from (5.14) and (5.16) we have $z\lambda_{jruiqv} = \lambda_{irq}$ and $y\lambda_{kiqv} = \lambda_{irq}$ whenever $z_{jruiqv} = 1$ and $y_{ij} = 1$, respectively. Therefore, the objective function is equal to $\sum_{iqv}\lambda_{iqv}\left(C_{max}^{iqv} + \sum_{j=1}^{l}\sum_{u=1}^{s}\sum_{r=1}^{n_j^v}\left(t_{jr+1}^v - t_{jr}^v\right)z_{jruiqv}\right)$. Note that the term within the parenthesis is simply the objective function that would result from having sublot iqv as the critical sublot. Also note that (5.15) prevents any infeasible λ_{iqv} from being positive. Since at least one iqv is a bottleneck corresponding to minimum objective function value (see Proposition (5.2)), there exists at least one optimal solution with $\lambda_{iqv} = 1$ for some $i = 1,...,l,q,v$. □

5.4 Simultaneously Obtaining an Optimal Sequence and Sublot Sizes

The two-stage assembly scheduling problem, without lot streaming, was shown to be NP-hard by Potts *et al.* (1995). Note that any such problem is a special case of the ALSP with $n_i^v = 1 \forall i = 1, \cdots, l$ and $v = 1, \cdots, s$. Therefore, the ALSP is also an NP-hard problem. In order to solve this problem, we first present an integer programming formulation that uses a Linear Ordering of lots (LO-formulation). Such a formulation is frequently used in scheduling. Then, we present a second formulation that is tighter than the LO-formulation, and it uses an assignment of lots to positions in the sequence. We refer to it as the Assignment-based Formulation (A-formulation). The use of assignment variables, instead of linear ordering, results in non-linear terms, which are linearized in the A-formulation. The resulting formulation is tighter than the LO-formulation. In addition, the sequence is defined by an assignment matrix. Therefore, a linear programming (LP) relaxation of this formulation contains a doubly stochastic assignment matrix. Using Birkhoff-Neuman (Birkhoff, 1946) decomposition, it is possible to obtain candidate permutations from such a matrix. We develop a method to use this fact to obtain valid inequalities for the A-formulation based on it's LP relaxation. Then, we compare the performances of these two formulations.

5.4.1 Linear ordering formulation

Recall that we use $y_{ij} = 1$ if lot i precedes j (not necessarily immediately). Rearranging (5.3), and using y_{ij} to define a sequence (instead of using the sequence $1, \cdots, l$), the makespan is given by $\max\limits_{\{k,v:\, p_k^v > 0\}} \left(\sum_{\substack{i=1 \\ i \neq k}}^{l} (p_i^v - p_i)\, y_{ik} + \sum_{q=1}^{n_i^v} f_{kq}^v \delta_{kq}^v - p_k \right) + \sum_{i=1}^{l} p_i$. Note that we can ignore the constant term $\sum_{i=1}^{l} p_i$ when finding the optimal values of y_{ik} and δ_{kq}^v since it is common to all possible bottleneck iqv values. Consider the following formulation to solve the ALSP:

$$minimize \quad C_{max} + \sum_{v=1}^{s} \sum_{i=1}^{l} \sum_{q=1}^{n_i^v} t_{iq}^v x_{iq}^v \tag{5.21}$$

$$\sum_{\substack{i=1\\ i\neq k}}^{l} \left(p_i^v - p_i\right) y_{ik} + \sum_{q=1}^{n_k^v} f_{kq}^v \delta_{kq}^v - C_{max} \leq 0 \qquad \forall\ \{k, v:\ k = 1, \cdots, l,\ v = 1, \cdots, s\} \quad (5.22)$$

$$y_{ij} + y_{ji} = 1 \qquad \forall i \neq j,\ i, j = 1, \cdots, l \quad (5.23)$$

$$y_{ij} - y_{kj} - y_{ik} \leq 0 \qquad \forall i \neq j \neq k \neq i,\ i, j, k = 1, \cdots, l \quad (5.24)$$

$$\sum_{q=1}^{n_i^v} \delta_{iq}^v = 1 \qquad \forall i = 1, \cdots, l,\ v = 1, \cdots, s. \quad (5.25)$$

$$\delta_{iq}^v,\ y_{ij} \in \{0, 1\} \quad \forall i, j = 1, \cdots, l,\ j = 1, \cdots, n_i^v,\ v = 1, \cdots, s.$$

This is a valid formulation for the ALSP. Constraint (5.23) establishes that each pair of lots have a relative order in the production sequence, while (5.24) enforces a transitive relationship between them. Therefore, the y_{ij} $(i, j = 1, \cdots, l, i \neq j)$ produce a linear ordering of the lots. Variable δ_{ij}^v selects the number of sublots used by suppliers for each lot. Constraints (5.25) ensure that exactly one number between $1, \cdots, n_i^v$ is selected as the number of sublots by each supplier for each lot. The makespan is constrained to be larger than the length of the path through each lot, and, as a result, it is equal to the length of the critical path for the given sequence.

Note that the following additional constraints are also valid. In an optimal solution, if $\delta_{iq}^v = 1$ for any $q \geq 2$, then using $\delta_{iq-1}^v = 1$ would result in a higher makespan. Therefore, if we set $f_{i0}^v = f_{i1}^v + \epsilon$, the following constraint is valid for all optimal solutions.

$$\sum_{\substack{i=1\\ i\neq k}}^{l} \left(p_i^v - p_i\right) y_{ik} + \sum_{q=1}^{n_i^v} f_{i(q-1)}^v \delta_{iq}^v - C_{max} \geq 0\ \forall\ \{k, v:\ p_k^v > 0\ k = 1, \cdots, l,\ v = 1, \cdots, s\}. \quad (5.26)$$

In fact, if all processing times are integers, we can make the inequality strict by using $\epsilon = 1$ and can state it as

$$\sum_{\substack{i=1\\ i\neq k}}^{l} \left(p_i^v - p_i\right) y_{ik} + \sum_{q=1}^{n_i^v} f_{i(q-1)}^v \delta_{iq}^v - C_{max} \geq 1\ \forall\ \{k, v:\ p_k^v > 0\ k = 1, \cdots, l,\ v = 1, \cdots, s\}. \quad (5.27)$$

5.4.2 Assignment-based formulation

Next, we present a tightened formulation that can be solved as an LP-relaxation, and then, a feasible schedule can be extracted from it using the Birkhoff-Neumann decomposition of a doubly stochastic matrix. Consider a formulation based on the assignment of lots to positions

in the sequence. Let $\pi_{ij} = 1$ if lot i is assigned to position j, and $\pi_{ij} = 0$ otherwise. Using this variable, we have the following formulation.

minimize

$$C_{max} + \sum_{v=1}^{s} \sum_{i=1}^{l} \sum_{j=1}^{n_i^v} t_{ij}^v \delta_{ij}^v \tag{5.28}$$

$$\sum_{j=1}^{k} \sum_{r=1}^{l} \left(p_r^v - p_r\right) \pi_{rj} + \sum_{r=1}^{l} \left(\sum_{q=1}^{n_k^v} f_{rq}^v \delta_{rq}^v \right) \pi_{rk} - C_{max} \leq 0 \tag{5.29}$$
$$\forall\, k, v: \; k = 1, \cdots, l, \; v = 1, \cdots, s$$

$$\sum_{i=1}^{l} \pi_{ij} = 1 \, \forall \, j = 1, \cdots, l \tag{5.30}$$

$$\sum_{j=1}^{l} \pi_{ij} = 1 \, \forall \, i = 1, \cdots, l \tag{5.31}$$

$$\sum_{q=1}^{n_i^v} \delta_{iq}^v = 1 \, \forall i = 1, \cdots, l, \; v = 1, \cdots, s. \tag{5.32}$$

$$\delta_{iq}^v, \, \pi_{ij} \in \{0, 1\} \; \forall i, j = 1, \cdots, l, \; q = 1, \cdots, n_i^v, \; v = 1, \cdots, s.$$

Constraints (5.30) and (5.31) assign one and only one lot to each position in the production sequence. Constraint (5.32) is identical to (5.25) and serves the same purpose. Constraint (5.29) restricts the makespan to be larger than the length of a path through the lot assigned to position k in the production sequence, where $k = 1, \cdots, l$. Note that, the term $\sum_{r=1}^{l} \left(\sum_{q=1}^{n_k^v} f_{rq}^v \delta_{rq}^v \right) \pi_{rk}$ is non-linear. In order to linearize it, we introduce the continuous variable $\zeta_{rkq}^v \in [0, 1]$ to replace $\delta_{rq}^v \pi_{rk}$. Thus, constraint (5.29) is replaced with the following constraint:

$$\sum_{j=1}^{k} \sum_{r=1}^{l} \left(p_r^v - p_r\right) \pi_{rj} + \sum_{r=1}^{l} \sum_{q=1}^{n_k^v} f_{rn}^v \zeta_{rkq}^v - C_{max} \leq 0. \tag{5.33}$$

We must have $\zeta_{rkq}^v = 1$ when lot r is assigned to position k in the production sequence, and supplier v uses q sublots to deliver it to the assembly stage; otherwise, $\zeta_{rkq}^v = 0$ for binary values of δ_{rq}^v and π_{rk}. To that end, we add the following linearization constraints:

$$\zeta_{rkq}^v \leq \delta_{rq}^v \tag{5.34}$$

$$\zeta_{rkq}^v \leq \pi_{rk} \tag{5.35}$$

$$\zeta_{rkq}^v \geq \pi_{rk} + \delta_{rq}^v - 1 \tag{5.36}$$

Note that the inclusion of the variable ζ_{rkq}^v allows us to remove δ_{rn}^v. The A-formulation presented above is sufficient to obtain optimal solutions for the ALSP. We next present a method to obtain valid inequalities for the problem based on an LP-relaxation.

Next, we develop a method to generate valid inequalities for the A-formulation in polynomial-time. In order to generate valid cuts, we begin with a solution to the continuous relaxation

of the A-formulation. If all π_{ij} values are binary, then we have an optimal solution and no cuts are generated, otherwise, we consider the square matrix Π with l rows and columns and values of the i^{th} row and j^{th} column given by π_{ij}, $i, j = 1, \cdots, l$. Since we begin with a feasible solution for the A-formualtion, constraints (5.30) and (5.31) are satisfied by π_{ij}, and so, Π is a doubly stochastic matrix. Birkhoff (1946) provided an algorithm that obtains permutation matrices Π_k, $k = 1, \cdots, n$ such that $\Pi = \sum_k a_k \Pi_k$, $\sum_k a_k = 1$, and, $n \leq l^2$, i.e., Π is expressed as a convex combination of discrete permutation matrices. We obtain such a decomposition of Π. Note that each Π_k represents an assignment of lots to positions in the production sequence, and so, they define a complete production sequence. We obtain an optimal number of sublots for each of these Π_k using Algorithm 1 in polynomial time. Now, for each $k = 1, \cdots, n$ we have a production sequence as well as the optimal number of sublots for that sequence. Therefore, we have n feasible solutions (schedules) for the A-formulation, apart from generating cuts, these can also be used for generating upper bounds. Recall that, for each of these feasible schedules, the value of C_{max}^{kqv} can be obtained using 5.5. Also, one of the values is equal to the makespan. Next consider the LP-Formulation 5.3 that was presented earlier to obtain optimal number of sublots for a given production sequence, defined by y_{ij} values. From Algorithm 1 we already know the optimal solution for this LP (for each Π_k), i.e., we know the critical values k^*, q^*, v^* for which $C_{max}^{k^*q^*v^*} = C_{max}$. Therefore, we have $\lambda_{k^*q^*v^*} = 1$ (and all other $\lambda_{kqv} = 0$). Also, since we know the values of all $C_{max}^{kqv}, k = 1, \cdots, l$, $v = 1, \cdots, s$, $q = 1, \cdots, n_k^v$, we know the values of $z_{kqv\,jru}$. As a result we have an optimal solution, and an optimal basis, for the LP. We can find the optimal dual values corresponding to this basis in polynomial-time. The optimal objective for this dual solution can be expressed as $\sum \sum_{i,j=1,\cdots,l} \gamma_{ij} y_{ij} + c$, where c is a constant and γ_{ij} are obtained by adding all dual values that correspond to constraints with y_{ij} in their right hand side. We have a cut

$$C_{max} + \sum_{v=1}^{s} \sum_{i=1}^{l} \sum_{j=1}^{n_i^v} t_{ij}^v \delta_{ij}^v \geq \sum \sum_{i,j=1,\cdots,l} \gamma_{ij} y_{ij} + c \tag{5.37}$$

where we replace the term y_{ij} with $2 - \pi_{jk} - \pi_{ik'}$, $1 \leq k < k' \leq l$ when $\gamma_{ij} < 0$ since $2 - \pi_{jk} - \pi_{ik'} \geq y_{ij}$, when $1 \leq k < k' \leq l$ and similarly $\pi_{ik} + \pi_{jk'} - 1$, $1 \leq k < k' \leq l$ when $\gamma_{ij} > 0$.

5.4.3 Computational Results

The LO-formulation, A-formulation and A-formulation with added cuts (referred to as A^+-Formulation) were implemented for several problem sizes. The results for number of lots, $l = 10, 20, 30, 40$, are presented in Tables 5.1, 5.2, 5.3, 5.4, respectively. For each number of lots l we considered four values of number of suppliers, namely $m = 10, 20, 30, 40$. For each combination of l, m we considered four values of upper bound on maximum number of sublots (each n_i^v value is generated randomly with this as the upper bound). The values used were $3, 6, 9, 12$. The computer used for this experiment was a Dell Precision

Algorithm 2 Solution method using the A-formulation and inequalities.

Step 1 Solve the LP relaxation of the A-formulation.

Step 2 Use Birkhoff-Neuman decomposition to obtain $\Pi_1, \Pi_2, \cdots, \Pi_k$, $k \leq l^2$ permutation matrices such that the fractional solution Π from the LP relaxation is a convex combination of them.

Step 3 Generate valid inequalities using each $\Pi_1, \Pi_2, \cdots, \Pi_k$, Algorithm 1 and 5.3.

Step 4 Add the valid inequalities generated in Step 3 to the A-formulation and solve the IP.

T7600 workstation with two Intel Xeon E5-2687W (3.1GHz) CPUs and 32GB RAM. The best time values are highlighted in bold. Failure to solve a problem within 1800s of clock time (this is different from CPU-time) or 1400 MB of memory usage is denoted by "-". The LO-formulation performs significantly better when $l = 10$, where it gives the lowest CPU times for all instances. However, its performance deteriorates for $l = 20$, where it is unable to generate a solution the specified time limit for 10 out of 16 instances. Also, its performance is worse for the 10 instances for which it generated an optimal solution. For instances with $l \geq 30$ it fails to generate an optimal solution. Note that adding cuts to the A-formulation is useful in reducing CPU-time in several cases (between 18% to 43% for different values of l), but in the majority of cases it requires greater CPU times. Both A-formulations are better than LO-based formulations for medium to large problem sizes. Both A-Formulation and A^+-Formulation were unable to obtain optimal solutions for the largest instances within the specified time limit.

5.5 Concluding Remarks

In this chapter we have presented solution methods for the problem of minimizing the sum of material handling and makespan costs in an assembly system. Our solution methodology is applicable to a wide range of production scenarios and we make very limited assumptions regarding the handling cost parameters. The complete ALSP problem is NP-hard, however we show that for a given sequence the problem can be solved in polynomial-time. We propose two integer programming formulations to solve the complete problem, one based on the standard transitive relations used in scheduling (called the linear-ordering formulation) and a new formulation based on assigning lots to positions in the sequence (called the assignment-based formulation). The latter is shown to have superior computational performance. We also present a fast method to obtain valid inequalities for the Assignment-based Formulation. The computational performance of the formulations based on transitive relations, assignment of lot to positions in the production sequence, and, the assignment-based for-

Table 5.1: CPU times required by A^+-Formulation, A-Formulation, and LO-Formulation for l=10

Number of lots	Number of vendors	max (max number of sublots)	A^+-Formulation	A-Formulation	LO-Formulation	Optimal Objective Value
10	10	3	1.52881	1.38841	**0.343202**	484
10	10	6	8.20565	18.4861	**0.733205**	353
10	10	9	1.01401	1.65361	**0.218401**	279
10	10	12	0.904806	1.96561	**0.452403**	230
10	20	3	0.842405	1.24801	**0.920406**	807
10	20	6	1.77841	1.74721	**0.514803**	529
10	20	9	10.9981	4.43043	**0.639604**	415
10	20	12	7.51925	8.37725	**1.01401**	335
10	30	3	2.48042	3.01082	**0.686404**	1199
10	30	6	10.0465	3.99363	**3.05762**	729
10	30	9	7.30085	5.92804	**0.280802**	620
10	30	12	22.2769	12.1213	**3.86882**	435
10	40	3	4.35243	7.25405	**1.07641**	1626
10	40	6	11.4661	7.67525	**0.374402**	957
10	40	9	24.4298	15.0541	**1.04521**	743
10	40	12	45.0843	34.4294	**0.936006**	541

Table 5.2: CPU times required by A^+-Formulation, A-Formulation, and LO-Formulation for l=20

Number of lots	Number of vendors	max (max number of sublots)	A^+-Formulation	A-Formulation	LO-Formulation	Optimal Objective Value
20	10	3	**3.26042**	4.04043	-	906
20	10	6	**10.6081**	29.0162	822.734	699
20	10	9	22.5421	**16.0993**	-	551
20	10	12	69.1552	**42.3231**	1504.22	439
20	20	3	**23.8838**	25.553	426.304	1576
20	20	6	52.7907	**36.5198**	-	1084
20	20	9	125.019	**71.6513**	-	860
20	20	12	210.835	**97.0794**	-	660
20	30	3	46.8003	**43.4931**	-	2304
20	30	6	131.275	**99.9498**	-	1518
20	30	9	343.639	**194.939**	-	1194
20	30	12	3055.2	**2552.02**	-	863
20	40	3	**101.572**	145.19	112.368	3114
20	40	6	312.548	**300.208**	108.405	1857
20	40	9	700.398	**540.481**	824.153	1466
20	40	12	**7562.34**	10818.3	-	1066

Table 5.3: CPU times required by A^+-Formulation, A-Formulation, and LO-Formulation forl=30

Number of lots	Number of vendors	max (max number of sublots)	A^+-Formulation	A-Formulation	Optimal Objective Value
30	10	3	23.6498	**23.0725**	1387
30	10	6	57.6424	**37.643**	950
30	10	9	122.476	**79.2485**	824
30	10	12	**120.183**	120.339	664
30	20	3	111.619	**107.032**	2507
30	20	6	**352.874**	1433.54	1548
30	20	9	657.045	**448.815**	1288
30	20	12	1174.38	**664.876**	954
30	30	3	301.612	**293.75**	3560
30	30	6	834.995	**738.946**	2098
30	30	9	1756.27	**1216.01**	1732
30	30	12	4536.81	**1966.86**	1259
30	40	3	569.076	**558.359**	4637
30	40	6	1656.98	**1402.62**	2763
30	40	9	3152.53	**2236.34**	2200
30	40	12	**10192.5**	13167	1571

Table 5.4: CPU times required by A^+-Formulation, A-Formulation, and LO-Formulation for l=40

Number of lots	Number of vendors	max (max number of sublots)	A^+-Formulation	A-Formulation	Optimal Objective Value
40	10	3	77.7041	**73.2113**	1901
40	10	6	195.5	**158.591**	1305
40	10	9	439.392	**305.544**	1064
40	10	12	710.741	**439.47**	888
40	20	3	**367.242**	394.963	3287
40	20	6	1027.58	**882.529**	2094
40	20	9	1951.87	**1474.1**	1678
40	20	12	3367	**2084.17**	1279
40	30	3	**931.217**	948.236	4643
40	30	6	2398.72	**2036.36**	2914
40	30	9	5059.36	**3691.58**	2330
40	30	12	9726.48	**6448.63**	1700
40	40	3	**1794.57**	1833.29	6165
40	40	6	4605.98	**3840.14**	3721
40	40	9	-	-	-
40	40	12	-	-	-

mulation with some valid inequalities, are tested experimentally. We also show that adding valid inequalities frequently improves performance, but, in the majority of cases, it is inferior to the assignment-based formulation.

Bibliography

Adams, J., Balas, E., and Zawack, D., 1988. The shifting bottleneck procedure for job shop scheduling. *Management science*, 34 (3), 391–401.

Al-Anzi, F. and Allahverdi, A., 2006. A hybrid tabu search heuristic for the two-stage assembly scheduling problem. *International Journal of Operations Research*, 3 (2), 109–119.

Al-Anzi, F. and Allahverdi, A., 2007. A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times. *European journal of operational research*, 182 (1), 80–94.

Al-Anzi, F. and Allahverdi, A., 2009. Heuristics for a two-stage assembly flowshop with bicriteria of maximum lateness and makespan. *Computers & Operations Research*, 36 (9), 2682–2689.

Alfieri, A., Glass, C., and van de Velde, S., 2012. Two-machine lot streaming with attached setup times. *IIE Transactions*, 44 (8), 695–710.

Allahverdi, A. and Al-Anzi, F., 2006a. Evolutionary heuristics and an algorithm for the two-stage assembly scheduling problem to minimize makespan with setup times. *International journal of production research*, 44 (22), 4713–4735.

Allahverdi, A. and Al-Anzi, F., 2006b. A PSO and a Tabu search heuristics for the assembly scheduling problem of the two-stage distributed database application. *Computers & Operations Research*, 33 (4), 1056–1080.

Allahverdi, A. and Al-Anzi, F., 2008. The two-stage assembly flowshop scheduling problem with bicriteria of makespan and mean completion time. *The International Journal of Advanced Manufacturing Technology*, 37 (1), 166–177.

Allahverdi, A. and Al-Anzi, F., 2009. The two-stage assembly scheduling problem to minimize total completion time with setup times. *Computers & Operations Research*, 36 (10), 2740–2747.

Badiru, A., May. Computational survey of univariate and multivariate learning curve models. *Engineering Management, IEEE Transactions on*, 39 (2), 176–188.

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

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

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

Baker, K.R. and Trietsch, D., 2009. *Principles of sequencing and scheduling*. Wiley.

Baloff, N., 1971. Extension of the Learning Curve–Some Empirical Results. *Operational Research Quarterly*, 22 (4), 329–340.

Banerjee, A., 1986. A joint economic-lot-size model for purchaser and vendor. *Decision sciences*, 17 (3), 292–311.

Ben-Daya, M., Darwish, M., and Ertogral, K., 2008. The joint economic lot sizing problem: Review and extensions. *European Journal of Operational Research*, 185 (2), 726 – 742.

Ben-Daya, M. and Hariga, M., 2004. Integrated single vendor single buyer model with stochastic demand and variable lead time. *International Journal of Production Economics*, 92 (1), 75–80.

Birkhoff, G., 1946. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucumán Rev. Ser. A*, 5, 147–151.

Biskup, D., 1999a. Single-machine scheduling with learning considerations. *European Journal of Operational Research*, 115 (1), 173 – 178.

Biskup, D., 1999b. Single-machine scheduling with learning considerations. *European Journal of Operational Research*, 115 (1), 173–178.

Biskup, D., 2008a. A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research*, 188 (2), 315 – 329.

Biskup, D., 2008b. A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research*, 188 (2), 315–329.

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

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

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

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

Carr, G.W., 1946. Peacetime cost estimating requires new learning curves. *Aviation*, 45 (4), 220–228.

Cetinkaya, F.C., 1994. 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.

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

Çetinkaya, F. and Gupta, J., 1994. Flowshop lot streaming to minimize total weighted flow time. *Research Memorandum*, 94, 24.

Cetinkaya, F.C., 2006. 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.

Chang, J.H. and Chiu, H.N., 2005. A comprehensive review of lot streaming. *International Journal of Production Research*, 43 (8), 1515–1536.

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

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

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

Cheng, M., Mukherjee, N.J., and Sarin, S.C., 2013. A review of lot streaming. *International Journal of Production Research*, 51, 7023–7046.

Chiang, C. and Chiang, W.C., 1996. Reducing inventory costs by order splitting in the sole sourcing environment. *Journal of the Operational Research Society*, 446–456.

Dar-El, E., 2000. *Human learning: From learning curves to learning organizations*. Springer Netherlands.

Dar-El, E., Ayas, K., and Gilad, I., 1995a. A dual-phase model for the individual learning process in industrial tasks. *IIE transactions*, 27 (3), 265–271.

Dar-El, E., Ayas, K., and Gilad, I., 1995b. Predicting performance times for long cycle time tasks. *IIE transactions*, 27 (3), 272–281.

Dar-El, E. and Rabinovitch, M., 1988. Optimal planning and scheduling of assembly lines. *International journal of production research*, 26 (9), 1433–1450.

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

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

Defersha, F. and Chen, M., 2010. A hybrid genetic algorithm for flowshop lot streaming with setups and variable sublots. *International Journal of Production Research*, 48 (6), 1705–1726.

Defersha, F. and Chen, M., 2012. Jobshop lot streaming with routing flexibility, sequence-dependent setups, machine release dates and lag time. *International Journal of Production Research*, 50 (8), 2331–2352.

Feldmann, M. and Biskup, D., 2008. Lot streaming in a multiple product permutation flow shop with intermingling. *International Journal of Production Research*, 46 (1), 197–216.

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

Glass, C.A. and Potts, C.N., 1998. Structural Properties of Lot Streaming in a Flow Shop. *Mathematics of Operations Research*, 23 (3), 624–639.

Glass, C., Gupta, J., and Potts, C., 1994b. Lot streaming in three-stage production processes. *European journal of operational research*, 75 (2), 378–394.

Glass, C. and Possani, E., 2011. Lot streaming multiple jobs in a flow shop. *International Journal of Production Research*, 49 (9), 2669–2681.

Goyal, S.K., 1988. "A joint economic-lot-size model for purchaser and vendor": A comment. *Decision Sciences*, 19 (1), 236–241.

Goyal, S., 1977. An integrated inventory model for a single supplier-single customer problem. *The International Journal of Production Research*, 15 (1), 107–111.

Goyal, S., 1995. A one-vendor multi-buyer integrated inventory model: A comment. *European Journal of Operational Research*, 82 (1), 209–210.

Hall, N.G., *et al.*, 2005. Scheduling and lot streaming in two-machine open shops with no-wait in process. *Naval Research Logistics (NRL)*, 52 (3), 261–275.

Hall, N.G., *et al.*, 2003. Scheduling and Lot Streaming in Flowshops with No-Wait in Process. *Journal of Scheduling*, 6 (4), 339–354.

Haouari, M. and Daouas, T., 1999. Optimal scheduling of the 3-machine assembly-type flow shop. *RAIRO-Operations Research*, 33 (04), 439–445.

Hariri, A. and Potts, C., 1997. A branch and bound algorithm for the two-stage assembly scheduling problem. *European Journal of Operational Research*, 103 (3), 547–556.

Hill, R.M., 1997. The single-vendor single-buyer integrated production-inventory model with a generalised policy. *European Journal of Operational Research*, 97 (3), 493–499.

Hill, R., 1999. The optimal production and shipment policy for the single-vendor singlebuyer integrated production-inventory problem. *International Journal of Production Research*, 37 (11), 2463–2475.

Hill, R.M., 1996. Order splitting in continuous review (Q,r) inventory models. *European Journal of Operational Research*, 95 (1), 53–61.

Hon-Shiang, L. and Zhao, L., 1994. Dual sourcing cost-optimization with unrestricted lead-time distributions and order-split proportions. *IIE transactions*, 26 (5), 66–75.

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

Janiak, A. and Rudek, R., 2009. Experience-based approach to scheduling problems with the learning effect. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39 (2), 344–357.

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

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

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

Kalir, A. and Sarin, S., 2001a. Optimal solutions for the single batch, flow shop, lot-streaming problem with equal sublots. *Decision Sciences*, 32 (2), 387–397.

Kalir, A. and Sarin, S., 2003. Constructing near optimal schedules for the flow-shop lot streaming problem with sublot-attached setups. *Journal of combinatorial optimization*, 7 (1), 23–44.

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

Kalir, A.A. and Sarin, S.C., 2001b. Constructing Near Optimal Schedules for the Flow-Shop Lot Streaming Problem with Sublot-Attached Setups. *Journal of Combinatorial Optimization*, 7 (1), 23–44.

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

Kalir, A.A. and Sarin, S.C., 2001d. Optimal Solutions for the Single Batch, Flow Shop, Lot-streaming Problem with Equal Sublots. *Decision Sciences*, 32 (2), 387–398.

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

Kelle, P. and Miller, P., 2001. Stockout risk and order splitting. *International Journal of Production Economics*, 71 (1-3), 407–415.

Kelle, P. and Silver, E., 1990. Safety stock reduction by order splitting. *Naval Research Logistics (NRL)*, 37 (5), 725–743.

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

Kim, K. and Jeong, I., 2009. 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.

Kim, Y.D., *et al.*, 2004. Parallel machine scheduling considering a job-splitting property.. *International Journal of Production Research*, 42 (21), 4531–4546.

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

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

Latour, A., 2001. Trial by fire: A blaze in Albuquerque sets off major crisis for cell-phone giants. *Wall Street Journal*, 1 (29), 2001.

Lau, H. and Lau, A., 1994. Coordinating two suppliers with offsetting lead time and price performance. *Journal of Operations Management*, 11 (4), 327–337.

Lau, H. and Zhao, L., 1993. Optimal ordering policies with two suppliers when lead times and demands are all stochastic. *European Journal of Operational Research*, 68 (1), 120–133.

Lee, C., Cheng, T., and Lin, B., 1993. Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem. *Management Science*, 39 (5), 616–625.

Lee, W. and Wu, C., 2004. Minimizing total completion time in a two-machine flowshop with a learning effect. *International Journal of Production Economics*, 88 (1), 85–93.

Lee, W. and Wu, C., 2009. Some single-machine and m-machine flowshop scheduling problems with learning considerations. *Information Sciences*, 179 (22), 3885–3892.

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

Liu, C., Chen, L., and Lin, P., 2012. Lot streaming multiple jobs with values exponentially deteriorating over time in a job-shop environment. .

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

Logendran, R. and Subur, F., 2004. Unrelated Parallel Machine Scheduling with Job Splitting.. *IIE Transactions*, 36 (4), 359–372.

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

Mishra, A. and Tadikamalla, P., 2005. Order splitting in single sourcing with scheduled-release orders. *Journal of the Operational Research Society*, 57 (2), 177–189.

Mosheiov, G., 2001. Scheduling problems with a learning effect. *European Journal of Operational Research*, 132 (3), 687 – 693.

Mukherjee, N. and Sarin, S., 2014. Makespan minimization for two-machine flow shops in the presence of learning and sublot-attached setup timel. *Research Report*.

Pan, A., *et al.*, 1991. Multiple sourcing: the determination of lead times. *Operations research letters*, 10 (1), 1–7.

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

Potts, C. and Baker, K., 1989b. Flow shop scheduling with lot streaming. *Operations research letters*, 8 (6), 297–303.

Potts, C., *et al.*, 1995. The two-stage assembly scheduling problem: complexity and approximation. *Operations Research*, 346–355.

Ramasesh, R., Ord, J., and Hayya, J., 1993. Note: Dual sourcing with nonidentical suppliers. *Naval Research Logistics (NRL)*, 40 (2), 279–288.

Ramasesh, R., *et al.*, 1991. Sole versus dual sourcing in stochastic lead-time (s, Q) inventory models. *Management Science*, 428–443.

Rudek, R., 2011. Computational complexity and solution algorithms for flowshop scheduling problems with the learning effect. *Computers & Industrial Engineering*, 61 (1), 20–31.

Sajadieh, M.S., Jokar, M.R.A., and Modarres, M., 2009. Developing a coordinated vendor-buyer model in two-stage supply chains with stochastic lead-times. *Comput. Oper. Res.*, 36, 2484–2489.

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

Sarin, S.C. and Yao, L., 2011. Multiple-lot, lot streaming in a two-stage assembly system. *Essays in Production, Project Planning and Scheduling*.

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

Sculli, D. and Shum, Y., 1990. Analysis of a continuous review stock-control model with multiple suppliers. *Journal of the Operational Research Society*, 873–877.

Sculli, D. and Wu, S., 1981. Stock control with two suppliers and normal lead times. *Journal of the Operational Research Society*, 1003–1009.

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

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

Shallcross, D., 1992. A polynomial algorithm for a one machine batching problem. *Operations Research Letters*, 11 (4), 213–218.

Smith, M., Panwalkar, S., and Dudek, R., 1975. Flowshop sequencing problem with ordered processing time matrices. *Management Science*, 21 (5), 544–549.

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

Smunt, T., 1999. Log-linear and non-log-linear learning curve models for production research and cost estimation. *International journal of production research*, 37 (17), 3901–3911.

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

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

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

Sun, X., Morizawa, K., and Nagasawa, H., 2003. Powerful heuristics to minimize makespan in fixed, 3-machine, assembly-type flowshop scheduling. *European Journal of Operational Research*, 146 (3), 498–516.

Topaloglu, E., Sen, A., and Benli, Ö., 1994. Optimal Streaming of a Single Job in an m Stage Flow Shop with Two Sublots. .

Tozkapan, A., Kirca, Ö., and Chung, C., 2003. A branch and bound algorithm to minimize the total weighted flowtime for the two-stage assembly scheduling problem. *Computers & Operations Research*, 30 (2), 309–320.

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

Trietsch, D. and Baker, K., 1993a. Basic techniques for lot streaming. *Operations Research*, 1065–1076.

Trietsch, D. and Baker, K.R., 1993b. Basic Techniques for Lot Streaming. *Operations Research*, 41 (6), 1065–1076.

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

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

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

Tsubone, H., Ohba, M., and Uetake, T., 1996. 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.

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

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

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

Williams, E.F., Tufekcli, S., and Akansel, M., 1997. O(m2) Algorithms for the two and three sublot lot streaming problem. *Production and Operations Management*, 6 (1), 74–96.

Womack, J.P. and Jones, D.T., 2003. *Lean Thinking : Banish Waste and Create Wealth in Your Corporation, Revised and Updated*. New York: Free Press.

Wright-Patterson, T., 1936. Factors Affecting the Cost of Airplanes. *Journal of Aeronautical Sciences*, 3 (4), 122–128.

Yalaoui, F. and Chu, C., 2006. New exact method to solve the Pm/Rj/Cj schedule problem. *International Journal of Production Economics*, 100 (1), 168–179.

Yelle, L., 1979. The learning curve: Historical review and comprehensive survey. *Decision Sciences*, 10 (2), 302–328.

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

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

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

Zhang, W., *et al.*, 2005. Multi-job lot streaming to minimize the mean completion time in m-1 hybrid flowshops. *International Journal of Production Economics*, 96 (2), 189–200.