Lot Sizing at the Operational Planning and Shop Floor Scheduling Levels of the Decision Hierarchy of Various Production Systems

Ming Chen

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
Kimberly P. Ellis
G. Q. Lu
Yaesmin Merzifonluoglu

October 19, 2007

Blacksburg, Virginia

Key words: lot sizing, production planning, lot streaming, shop floor control, operational planning, scheduling

Lot Sizing at the Operational Planning and Shop Floor Scheduling Levels of the Decision Hierarchy of Various Production Systems

Ming Chen

ABSTRACT

The research work presented in this dissertation relates to lot sizing and its applications in the areas of operational planning and shop floor scheduling and control. Lot sizing enables a proper loading of requisite number of jobs on the machines in order to optimize the performance of an underlying production system. We address lot sizing problems that are encountered at the order entry level as well as those that are faced at the time of distributing the jobs from one machine to another and those that arise before shipping the jobs (orders) to customers. There are different issues and performance measures involved during each of these scenarios, which make the lot sizing problems encountered in these scenarios different from one another. We present algorithms and relevant theoretical analyses for each of the lot sizing problems considered, and also, present results of numerical experimentation to depict their effectiveness

We first study the lot sizing problem encountered while transferring jobs from one machine to another. A lot of the jobs is to be split into smaller lots (called sublots) such that the lot is processed on multiple machines in an overlapping manner, a process which is known in the literature as lot streaming. Two lot streaming problems, *FL2/n/C* and *FLm/1/C*, are investigated in Chapter 2.

FL2/n/C involves a two-machine flow shop in which multiple lots are to be processed. The objective is to minimize the combined cost of makespan and material handling (the latter is proportional to the number of sublots). A dynamic programming-based methodology is developed to determine the optimal sublot sizes and the number of sublots for each lot while assuming a known sequence in which to process the lots. We

designate this problem as **LSP-DP**. This methodology is, then, extended to determine an optimal sequence in which to process the lots in conjunction with the number of sublots and sublot sizes for each lot. We designate this problem as **LSSP-DP**. Three multidimensional heuristic search procedures (denoted as **LSSP-Greedy**, **LSSP-Cyclic** and **LSSP-ZP**) are proposed for this problem in order to obtain good-quality solutions in a reasonable amount of computational time. Our experimentation reveals that both lot streaming and lot sequencing generate significant benefits, if used alone. However, for the objective of minimizing total handling and makespan cost, lot streaming is more beneficial than lot sequencing. The combined use of lot streaming and sequencing, expectedly, results in the largest improvement over an initial random solution. **LSP-DP** is found to be very efficient, and so are the three **LSSP** heuristics, all of which are able to generate near-optimal solutions. On the average, **LSSP-Greedy** generates the best solutions among the three, and **LSSP-Cyclic** requires the least time.

FLm/1/C deals with the streaming of a single lot over multiple machines in a flow shop. The objective is a unified cost function that comprises of contributions due to makespan, mean flow time, work-in-process, transfer time and setup time. The distinctive features of our problem pertain to the inclusion of sublot-attached setup time and the fact that idling among the sublots of a lot is permitted. A solution procedure that relies on an approximation equation to determine sublot size is developed for this problem for equal-size sublots. The approximation avoids the need for numerical computations, and enables the procedure to run in polynomial time. Our experimentation shows that this solution procedure performs quite well and frequently generates the optimal solution. Since the objective function involves multiple criteria, we further study the marginal cost ratios of various pairs of the criteria, and propose cost sensitivity indices to help in estimating the impact of marginal cost values on the number of sublots obtained.

The lot sizing problem addressed in Chapter 3 is motivated by a real-life setting associated with semiconductor manufacturing. We first investigate the integration of lot sizing (at the operational planning level) and dispatching (at the scheduling and control level) in this environment. Such an integration is achieved by forming a closed-loop control system between lot sizing and dispatching. It works as follows: lot sizing module determines lot sizes (loading quota) for each processing buffer based on the current buffer

status via a detailed linear programming model. The loading quotas are then used by the dispatching module as a general guideline for dispatching lots on the shop floor. A dispatching rule called "largest-remaining-quota-first" (LRQ) is designed to drive the buffer status to its desired level as prescribed by the lot sizing module. Once the buffer status is changed or a certain amount of time has passed, loading quotas are updated by the lot sizing module. Our experimentation, using the simulation of a real-life wafer fab, reveals that the proposed approach outperforms the existing practice (which is based on "first-in-first-out" (FIFO) model and an ad-hoc lot sizing method). Significant improvements are obtained in both mean values and standard deviations of the performance metrics, which include finished-goods inventory, backlog, throughput and work-in-process.

The integration of lot sizing and dispatching focuses on the design of an overall production system architecture. Another lot sizing problem that we present in Chapter 3 deals with input control (or workload control) that complements this architecture. Input control policies are responsible for feeding the production system with the right amount of work and at the right time, and are usually divided into "push" or "pull" categories. We develop a two-phase input control methodology to improve system throughput and the average cycle time of the lots. In phase 1, appropriate operational lot sizes are determined with regard to weekly demand, so as to keep the lot start rate at the desired level. In phase 2, a "pull" policy, termed CONLOAD, is applied to keep the bottleneck's workload at a target level by releasing new lots into the system whenever the workload level is below the desired level. Since the operators are found to be the bottleneck of the system in our preliminary investigation, the "operator workload" is used as system workload in this study. Using throughput and cycle time as the performance metrics, it is shown that this two-phase CONLOAD methodology achieves significant improvement over the existing CONWIP-like policy. Furthermore, a reference table for the target operator workload is established with varying weekly demand and lot start rate.

The last lot sizing problem that we address has to do with the integration of production and shipping operations of a make-to-order manufacturer. The objective is to minimize the total cost of shipping and inventory (from manufacturer's perspective) as well as the cost of earliness and tardiness of an order (from customer's perspective). An

integer programming (IP) model is developed that captures the key features of this problem, including production and delivery lead times, multiple distinct capacitated machines and arbitrary processing route, among others. By utilizing the generalized upper bound (GUB) structure of this IP model, we are able to generate a simplified first-level RLT (Reformulation Linearization Technique) relaxation that guarantees the integrity of one set of GUB variables when it is solved as a linear programming (LP) problem. This allows us to obtain a tighter lower bound at a node of a branch-and-bound procedure. The GUB-based RLT relaxation is complemented by a GUB identification procedure to identify the set of GUB variables that, once restricted to integer values, would result in the largest increment in the objective value. The tightening procedure described above leads to the development of a RLT-based branch-and-bound algorithm. Our experimentation shows that this algorithm is able to search the branch-and-bound tree more efficiently, and hence, generates better solutions in a given amount of time.

Dedication

This dissertation is dedicated to my parents, Meiyun Huang and Liujiang Chen, and my wife, Yi Cao. Without their support, I could not go this far in the pursuit of Ph.D. degree.

Acknowledgements

I am indebted to my advisor, *Dr. Subhash C. Sarin*, who has guided me with great scholarship and patience over all these years. It is his encouragements and guidance that lightens up my journey, and instills me with motivation and power to reach the destination. I also owe great thanks to my other committee members, including *Dr. Kimberly P. Ellis*, *Dr. G. Q. Lu*, *Dr. Michael P. Deisenroth* and *Dr. Yaesmin Merzifonluoglu*, who have been very supportive, and have provided many beneficial suggestions.

My thanks also go to Grado Department of Industrial and Systems Engineering of Virginia Tech, and the Center for High Performance Manufacturing. Without the consistent support of assistantships, it is impossible for me to dedicate my energy to research and projects. I would also like to thank the professors who have prepared me well for my research with their lectures over my stay in Virginia Tech, especially, *Dr. Hanif D. Sherali*. I really appreciate the efforts of our staff in Industrial and Systems engineering, particularly, Lovedia Cole and Kim Ooms, who have greatly helped me with my academic documents and teaching responsibilities.

I have made quite a few friends during my study in Virginia Tech. Their friendship and exemplar achievements have been a great motivation for my study. Particularly, I would like to thank Liming Yao for helping me with academic forms while I'm away, and Yuqiang Wang and Peifang Tsai for their advices in preparing the final defense.

Finally, I would like to thank my wife, Yi Cao, for all the love and support that provided me strength and reasons to move forward.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

1.1 Background and Motivation

The rapid advancement in technology has created new opportunities as well as competition in the manufacturing environment. Supply chain integration and globalization are the direct results of technological advancements, and are keys for an enterprise to stay competitive. Since the success of an enterprise largely depends on the success of the supply chain that it belongs to, it must consistently strive to improve performance in order to meet the increasingly higher standards that are set for every participant of the supply chain.

The performance of an enterprise can be affected by improving the decision-making process at various levels of its decision hierarchy. The decision-making hierarchy of an enterprise, and in particular, that of a manufacturing enterprise, comprises of the following planning levels:

- Strategic planning
- Tactical planning
- Operational planning
- Scheduling and control

The strategic planning level pertains to long term decision making (generally for 3-10 years). The types of decisions that are considered at this level include selection of the location of a facility, determination of the required capacity and selection of a production technology. The tactical planning level addresses decisions over a shorter time horizon (generally from 6 months to 3 years). These include determination of the workforce levels, process routings and production rates. The operational planning level covers decisions that are made for a planning horizon of 1 week - 6 months. These include allocation of jobs to the machines, determination of lot sizes in which to process the jobs, overtime usage, and amount of subcontracting/outsourcing to use (Sipper and Bulfin [65]). The lowest level of hierarchy relates to the operations on the shop floor that

includes the scheduling and control of work over a period that spans from real time to one week. These also include determination of processing lot sizes, lot release, lot processing sequence as well as lot dispatching to the machines (i.e., their starting times). The work accomplished in this dissertation belongs to the last two levels of this decision hierarchy.

We address the issue pertaining to lot sizing at the operational planning and scheduling and control levels of the decision hierarchy in order to improve the overall system performance. A lot is a set of identical jobs that are processed and/or transported together through various processing stages of a production system. A setup is required before processing the jobs at a stage. Lot sizing requires determination of an appropriate number of jobs to be processed together. The size of a lot impacts the number of setups required, work-in-process as well as the production lead time. Due to the key role that it plays in the management of a production system that involves setups, lot sizing has been an active subject of study since its first emergence as an Economical Order Quantity (EOQ) model proposed by Harris [31]. Although the production practice has changed drastically over the years, lot sizing still remains a key issue in the management of a production system. However, the lot sizing models that have been developed for relatively simple production systems cannot be easily applied to modern complex production systems (such as those found in semiconductor manufacturing). Furthermore, the need for integrating different functions within an enterprise, such as production and distribution, has led to new problems that incorporate not only the lot sizing decisions but also decisions pertaining to the shipping of lots.

The processing of a lot at the shop floor level may involve its splitting into sublots and processing of the sublots in an overlapping fashion over several machines in order to reduce the production flow time. This process of splitting a lot into sublots is known as lot streaming. Lot streaming captures the interactions that are encountered in the processing of a lot (or lots) on the machines, and thus, complements lot sizing at the operational planning level where such interactions are largely ignored or simplified. By integrating lot streaming with lot sizing in our study, we intend to provide a more comprehensive treatment for the sizing of the lots.

1.2 Problem Description

We consider a variety of lot sizing problems and in different machine configurations. First, we consider a lot streaming problem involving two machines. Multiple lots are to be processed on these machines. As a lot is split into sublots, a material handling cost is encountered for the handling of each sublot during its transportation from one machine to another. Our objective is to determine the number and sizes of the sublots of each lot so as to minimize a joint function of the makespan and handling costs. We, then, consider an extension of this problem that also includes determination of the sequence in which to process the lots.

Next, we expand the two-machine flow shop lot streaming problem to a multiple-machine flow shop problem, and also, generalize the objective function to include cost components pertaining to average flow time, work-in-process (WIP), transfer times and setup times. However, this scenario is considered for a single lot in order to keep the problem tractable.

We, then, consider a complex batch production environment to study the impact of using various operational lot sizes as well as the effectiveness of using various input control strategies and an integrated lot sizing and dispatching system in such an environment. Our objective, in this part of our work, is to study the implications of making lot sizing decisions in a real-life manufacturing environment. In particular, we consider the manufacturing environment of a wafer fab. A wafer fab differs from a traditional discrete batch production system in many aspects. These include presence of a complex re-entrant flow, a large number of processing steps, unreliable machines and processes, and a variety of machines and products (see Fowler, et al. [19] for a detailed description of these differences). The lot sizing issues in this environment arise at both scheduling and control, and operational planning levels. For the scheduling and control level, it is essential to determine appropriate operational lot sizes, as well as input control strategies in order to achieve a desired throughput rate while maintaining minimal cycle time. For the operational planning level, we need to determine the amount of wafers of each product family for processing in the wafer fab in each time period so that the

inventory and shortage costs are minimized. Because a modern wafer fabrication system is a highly automated system that requires coordinated decision-making at various levels, an integration of lot sizing decisions with other functional decisions is highly desirable. Therefore, we first focus on the integration of lot sizing with the dispatching of wafers at individual processing areas in order to minimize output variability. We, then, determine effective operational lot sizes, and also, strategies for releasing these lots into the shop floor. In particular, we present a new input control methodology to achieve a better control of operator's workload.

The last problem that we consider has to do with the integration of lot sizing decisions (for production) with the loading decisions (for shipping) in order to minimize the total cost of shipping and inventory incurred by the manufacturer as well as the cost of delivering an order early or late to the customer. The shipping cost is a linear function of the number of trips required on each route, and the inventory cost is proportional to the amount of time an order stays in inventory after having completed production. The earliness and tardiness costs of an order are determined with regard to its due date. We assume a general multi-machine production system. The customer orders are released for processing on the shop floor on a daily basis. The details of the production and dispatching processes are not considered in our analysis since our focus is on the operational planning level. A variety of products are assumed to be produced in the make-to-order environment. Each customer order may contain different product types, and they must be delivered together before a pre-specified due date. The shipping resources consist of a fleet of vehicles, each with a finite capacity. The lot sizing process puts the orders into groups for processing in a time period. The shipping decisions, on the other hand, group the orders for shipping on appropriate vehicles, with each vehicle making shipments to various customers in a region. Only those orders that have been produced can be shipped. Hence, coordination between the production and shipping of orders is required in order to minimize the total cost incurred. Therefore, the two distinctive features of this problem are: 1) an integration of lot sizing decisions pertaining to the production and shipping operations, and 2) consideration of costs pertaining to both the supplier (production and shipping) and the customer (penalty for delivering early or late) in the objective function.

1.3 Research Objectives

Our primary objective of this dissertation is to provide insights and effective solution methodologies for the lot sizing problems encountered at various levels of the decision-making hierarchy of an enterprise. The problems that we consider address issues that are faced in real-life environments. These include consideration of both scheduling of production lots and their handling from one machine to another, the integration of production and distribution functions, and integration of lot sizing and dispatching in complex batch production systems. The specific objectives are to do the following:

- To study the impact of lot streaming on a combined function of makespan and material handling costs, where the lots are processed in a two-machine flow shop, and to develop an efficient solution methodology to determine an optimal number of sublots and the sublot sizes for each lot, and the sequence in which to process the lots.
- To study the impact of lot streaming on a unified cost function consisting of makespan, mean flow time, WIP, and setup and transfer times, where a lot is processed in a multi-machine flow shop, and to develop a methodology to obtain an optimal number of equal-size sublots, and to study trade-offs among various measures constituting the objective function in the presence of lot streaming.
- To study the impact of using various operational lot sizes, a new lot release strategy, and a new integrated lot sizing and dispatching system, in a complex batch production environment.
- To obtain insights into the integration of lot sizing and vehicle loading decisions for a make-to-order production system, and to develop an effective methodology for solving this integrated production and shipping lot sizing problem.

1.4 Contributions of This Dissertation

The contributions of this dissertation are multi-faceted. First of all, we develop

effective methodologies to obtain optimal sublot sizes for two, new lot streaming problems. Our results in this respect can be used as building blocks for solving real-life problems. For example, our methodology for the two-machine, multiple-lot problem can be used for coordinating deliveries between a supplier and a manufacturer in a two-level supply chain (see Li and Xiao [45]). Secondly, our study on the use of operational lot sizes in a complex batching system, and the development of new methodologies for lot release as well as integration of lot sizing and dispatching, are useful for practitioners since they can be easily adapted to real-life applications. Finally, we develop a new integrated production and shipping lot sizing methodology, which is applicable to a myriad of real-life situations.

In spite of the various problems and methodologies that we address, our work revolves around one common theme, namely, lot sizing and its impact on the performance of a system. Our work provides insights for use at both the operational planning and scheduling and control levels. Such insights have contributed in developing efficient solution methodologies for the various problems at hand. Furthermore, our study on the integration of the lot sizing decisions with other functional decisions of an enterprise, addresses supply chain management issues, and hence, offers strategies for achieving significant benefits in real-life applications.

1.5 Organization of This Dissertation

The rest of this dissertation is organized as follows: In Chapter 2, we discuss two lot streaming problems. For each of these problems, we present structural properties as well as develop an efficient solution methodology. Chapter 3 contains our study on the impact of using an integrated lot sizing and dispatching system, and various operational lot sizes with a new input control strategy, in a complex batch production system. In Chapter 4, we focus on the integration of lot sizing required for production and shipping. An integer programming formulation is presented for this problem, which is further tightened to obtain a good lower bound. A branch-and-bound procedure is developed for its solution and the effectiveness of this procedure is demonstrated through numerical examples.

Chapter 2 Lot Streaming Problem

2.1 Introduction

Lot streaming is the process of splitting a lot (of jobs) into sublots for processing in an overlapping manner on the machines. The primary benefits of lot streaming include a decrements in the production lead time and work-in-process (WIP). However, these benefits are achieved at the cost of an increment in non-productive activities such as setups and transportation of sublots. Lot streaming involves determination of the following three decisions: (i) number of sublots; (ii) size of each sublot, and (iii) sequence in which to process the lots, in case multiple lots are involved. In this chapter, we address two lot streaming problems. The first problem addresses the streaming of a given set of lots over a two machine flow shop, and involves all of these decisions. The second problem addresses streaming of a lot over a multi-machine flow shop but considers equal-size sublots, and hence, only requires determination of the number of sublots to use. The objective of the first lot streaming problem is to minimize the total cost incurred due to the makespan and sublot handling. In the second lot streaming problem, we expand the objective function to include more cost components such as WIP, mean flow time and setup time. For the ease of discussion, we designate the first lot streaming problem as *FL2/n/C* and the second problem as *FLm/1/C.* In this terminology, the first field refers to the machine configuration (flow shop, *F,* in our case and *L* for lot streaming); the second field refers to the number of machines (2 and m in our case); the third field refers to the number of lots (n and 1 in our case); and the last field refers to the objective function (*C* for cost-based in our case). We discuss the *FL2/n/C* problem in section 2.3 and the *FLm/1/C* problem in section 2.4.

2.2 Notation and Terminology

The following notation is used in this chapter:

Variables:

x	number of sublots
L_{kj}	sublot size of the k^{th} sublot of lot j
C_{ij}	completion time of lot j on machine i
μ	makespan
ν	mean flow time
π	work-in-process
θ	total setup time
τ	total transfer time

Parameters:

m	number of machines
n	number of lots
U_j	number of items in lot j
p_{ij}	unit processing time of lot j on machine i,
d	transfer time between two machines
η_i	material handling cost per sublot for lot j
λ	unit makespan cost
$\boldsymbol{s_i}$	setup time per sublot on machine i

Any occasional notation that is not shown above, but used in the sequel, is defined when used. Note that, if the problem only involves one lot, the lot subscript is omitted.

2.3 *FL2/n/C* - Multiple Batch, Two Machine, Flow Shop Lot Streaming to Minimize Makespan and Handling Costs

There are two key features of *FL2/n/C* that have not been addressed in the literature. These are: determination of an optimal number of sublots in the presence of sublot handling cost, and the simultaneous consideration of the three decisions mentioned above in section 2.1. The work reported in the literature, typically, assumes a given number of sublots. Consequently, the problem reduces to the determination of sublot sizes. Also, the cost involved in the handling of the sublots has been considered indirectly through a budget constraint. Here, we optimize the sublot handling cost by directly including it in the objective function.

For a lot streaming problem involving a single lot, two-machine flow shop and makespan objective, optimal sublot sizes have been shown to be geometric in nature (see Trietsch [71] and Potts and Baker [50]), for a given number of sublots. The geometric sublot sizes can take real numbers, and hence, are continuous in nature. In the case of discrete sublot sizes, Chen and Steiner [13] provide structural properties of the optimal solution, and Trietsch and Baker [73] present a polynomial time procedure to obtain discrete sublot sizes. Sriskandarajah and Wagneur [68] also address the determination of optimal sublot sizes (both continuous and discrete) in a no-wait flow shop. Sen, et al. [58] and Bukchin, et al. [9] have considered minimization of flow time for the same problem.

The multiple-lot problem involves the additional issue of determining the sequence in which to process these lots. This issue has been addressed in the literature for a simplified version of the underlying problem by assuming sublots of unit size (see Vickson and Alfredsson [78], Cetinkaya and Kayaligil [11], Baker [3]). Cetinkaya [10] and Vickson [77] relax the assumption of unit sublot sizes and also address the issue of determining sublot sizes for a given number of sublots along with the sequence in which to process the lots. The former study considers lot-detached setup and removal times while the latter considers lot-detached and attached setup times. Assuming equal sublot sizes, Kalir and Sarin [37] address the sequencing and sublot sizing problem with sublot-attached setups. The simultaneous determination of the number of sublots, the sublot

sizes and the sequence in which to process the lots has been addressed by Sriskandarajah and Wagneur [68] and Kumar, et al. [39] for a no-wait flow shop. The former addresses a two machine problem while the latter addresses a problem involving multiple machines. In this dissertation, we also address these three issues simultaneously but for the general two machine flow shop problem.

The presence of a significant cost for the handling of the sublots impacts determination of the optimal number of sublots of a lot. Truscott [74] has addressed this issue by directly modeling a transporter in the model. The optimal number of sublots is determined by using a branch-and-bound procedure for a two-machine flow shop. Trietsch and Baker [73] provide a polynomial time algorithm for a given number of transporters; however, this approach is difficult to extend to the cases involving more than two machines, and it only addresses single lot problems. An alternative approach is to associate a handling cost to each transfer, as in Trietsch [71] and Trietsch [72]. In Trietsch [71], a given budget for total handling cost is used to derive an appropriate number of sublots for a single lot. For multiple lots, a mathematical programming-based approach is provided to determine an optimal number of sublots for a given budget amount. However, this approach is not efficient for the solution of large-size problems if the integrity of the number of sublots is assumed. As alluded to earlier, in this dissertation, we include the cost involved in the handling of the sublots in the objective function and optimize it along with the makespan cost.

The remainder of this section for the *FL2/n/C* problem is organized as follows. In section 2.3.1, a mathematical programming formulation for the problem is presented and a dynamic programming-based methodology is developed for its solution. Then, the problem of simultaneously determining the number of sublots, sublot sizes and the sequence in which to process the lots is addressed in section 2.3.2. Numerical experimentation is conducted to show the effectiveness of our approach for this problem in section 2.3.3. Finally, concluding remarks are made in section 2.3.4.

2.3.1 Determination of the Optimal Number of Sublots and Sublot Sizes

First, we address the problem of determining an optimal number of sublots and sublot sizes for each of the *n* lots, assuming the sequence in which to process these lots is given. We relax this assumption later in Section 2.3.2. Our objective is to minimize a joint function of the makespan and sublot handling costs. We designate this problem **LSP** (Lot Streaming Problem) henceforth.

We make the following assumptions:

a) all the lots are available at time zero,

b) no setup time is required between the processing of different lots on a machine,

c) preemption of a lot is not allowed, i.e., there is no intermingling among the sublots belonging to different lots, and,

d) there is an unlimited buffer space between the machines. In addition, we assume that the sublot sizes of a lot are consistent on both machines.

Let η_j be the cost per transfer of a sublot of lot *j* (we assume a different handling cost for each lot), and λ be the unit cost of makespan ($ per unit time).

The problem can be mathematically formulated as follows:

$$\textbf{LSP}: \text{minimize } Z(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} \eta_j \cdot x_j + \lambda \cdot C_{2n} \tag{2.1}$$

Subject to:

$$C_{2n} = \max_{\substack{1 \le k \le x_j \\ 1 \le j \le n}} \left(\sum_{i=1}^{j-1} p_{1i} U_i + \sum_{i=1}^{k} p_{1j} L_{ij} + d + \sum_{i=k}^{x_j} p_{2j} L_{ij} + \sum_{i=j+1}^{n} p_{2i} U_i \right), \tag{2.2}$$

$$\sum_{k=1}^{x_j} L_{kj} = U_j, \qquad j=1,...,n;$$

$$1 \le x_j \le U_j, \text{ integer}, \qquad j=1,...,n.$$

The objective function in (2.1) is a linear combination of the handling and makespan costs. Expression (2.2) is based on the notion of a critical path, which crosses the machines at a sublot k of lot j.

Clearly,

$$C_{2n} = \max_{\substack{1\le k\le x_j \\ 1\le j\le n}} \left(\sum_{i=1}^{j-1} p_{1i}U_i + \sum_{i=1}^{k} p_{1j}L_{ij} + \sum_{i=k}^{x_j} p_{2j}L_{ij} + \sum_{i=j+1}^{n} p_{2i}U_i \right) + d \,,$$

and let $C_{2n} = C'_{2n} + d$ where C'_{2n} is the makespan without the constant transfer time. Obviously, the optimality of the solution is not affected by the value of d. Hence, in the sequel, we only consider the case when d=0, i.e., the case without constant transfer time.

Note that for a fixed number of sublots of each lot, the objective function reduces to that of minimizing the makespan. Consequently, we can use the result of Cetinkaya [10] and Vickson [77], which states that the optimal sublot sizes that minimize the makespan of a single lot, two-machine flow shop problem are also optimal for each lot of the n-lot problem. In other words, geometric sublot sizes that have been shown to be optimal for the case of continuous, 2-machine flow shops for the objective of minimizing the makespan (see Trietsch [71] and Potts and Baker [50]), are also optimal for each of the n lots. Thus, our problem boils down to determining the optimal number of sublots for each of the n lots. The geometric sublot sizes are calculated as follows:

$$q_j = p_{2j} / p_{1j} \,, \qquad j=1,...,n, \tag{2.3}$$

$$L_{1j} = \begin{cases} \dfrac{(1-q_j)\cdot U_j}{1-q_j^{x_j}}, q_j \neq 1, & j=1,...,n, \quad (2.4) \\ \dfrac{U_j}{x_j}, q_j = 1, & j=1,...,n; \quad (2.5) \end{cases}$$

$$L_{k+1,j} = q_j \cdot L_{kj} \,, \qquad k=1,...,x_j\text{-}1;\ j=1,...,n; \tag{2.6}$$

Note that, such geometric sublot sizes are continuous.

LSP-DP model:

Clearly, **LSP** is an integer and nonlinear program and, as such, is difficult to solve. However, note that the objective function of **LSP** is additive, and hence, separable, as follows:

$$Z(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} f_j(C_{2,j-1}, x_j) = \sum_{j=1}^{n} \left[\eta_j \cdot x_j + \lambda \cdot (C_{2j} - C_{2,j-1})\right].$$

Therefore, we can use dynamic programming for its solution. A stage of this multi-stage decision process is a lot j, with the corresponding decision, x_j. The single stage return is given by $f_j(C_{2,j-1}, x_j)$ and the input and output variables are $C_{2,j-1}$ and C_{2j}, respectively. For the sake of convenience, we represent $f_j(C_{2,j-1}, x_j)$ by $f_j(x_j)$ for a given $C_{2,j-1}$. Consequently, the state transformation can be derived as follows:

$$C_{2j}=max(C_{2,j-1}, C_{1j})+U_j \cdot p_{2j} = max(C_{2,j-1}, \sum_{i=1}^{j-1} U_i \cdot p_{1i} + L_{1j} \cdot p_{1j}) + U_j \cdot p_{2j}, \qquad (2.7)$$

where

$$C_{1j} = \sum_{i=1}^{j-1} U_i \cdot p_{1i} + L_{1j} \cdot p_{1j}, j=1,...,n. \qquad (2.8)$$

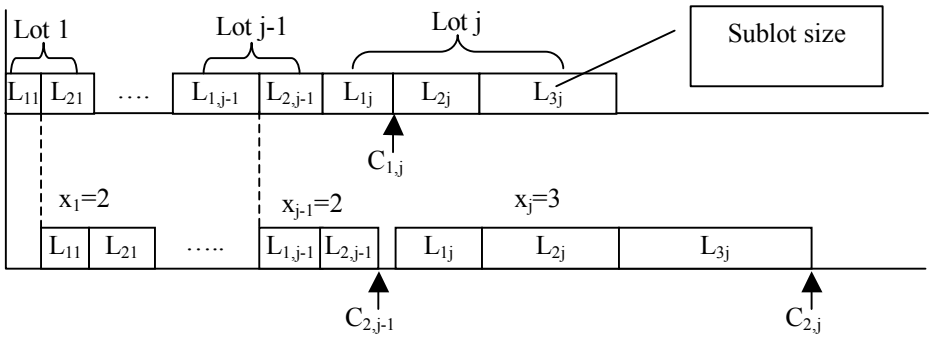


Figure 2.1 Illustrative Schedule of *FL2/n/C*

Expression (2.7) follows by the fact that there is no idle time between the sublots of the same lot on machine 2, which is a characteristic of the geometric sublot sizes.

We use backward recursion for which the recursive equation is given by,

$$G_j(C_{2,j-1}) = \min_{1 \le n_j \le U_j} \left(f_j(C_{2,j-1}, x_j) + G_{j+1}(C_{2j}) \right), \; j = 1,...,n, \qquad (2.9)$$

where $G_{n+1}(C_{2n})=0$. We denote the above DP approach by **LSP-DP**.

Some Structural Properties of **LSP-DP**

Next, we develop some structural properties of **LSP-DP** that help in curtailing computations at every stage of the DP recursive process.

Proposition 2.1 *C_{1j}, j=1,...,n is a monotone decreasing and strictly convex function of x_j.*

Proof: The proof follows since C_{1j} is a linear function of L_{1j}, which, in turn, is monotone decreasing and strictly convex in x_j j=1,...,n (see Expressions (2.4) and (2.5)). □

By Proposition 2.1, $f_j(0, x_j)$ and $Z(x_1, x_2, ..., x_n)$ are also convex functions of x_j since the convexity of C_{1j} w.r.t. x_j also implies that max($C_{2,j-1}$, C_{1j}) is convex for j=1,...,n. Hence, we can obtain an upper bound on the number of sublots for each j over the range [1, U_j]; x_j is increased starting from 1 until $f_j(0, x_j)$ starts to increase. Define this upper bound by $\hat{x}_j$. By Proposition 2.1, C_{1j} achieves its lower bound (LB_j) when $x_j = \hat{x}_j$, and its upper bound (UB_j) when x_j = 1. With this in view, we can effectively determine the optimal value of x_j for a given $C_{2,j-1}$ by analyzing the following three cases.

Case 1*:* $C_{2,j-1} \le LB_j$

In this case, $C_{2,j} = C_{1j} + U_j \cdot p_{2j}$, which implies that $C_{2,j-1}$ does not affect the makespan, i.e., the optimal number of sublots, x_k^* ($j \le k \le N$), is independent of $C_{2,j-1}$. Hence, we have the following property.

Property 2.1 *At stage j, the optimal partial solution for x_k^*, ($j \le k \le N$), when $C_{2,j-1}$ = LB_j, is also optimal for any $C_{2,j-1} < LB_j$, and $x_j^* \in [1, \hat{x}_j]$.*

Consequently, we have $f_j(x_j) = g_{1j}(x_j) = \eta_j \cdot x_j + \lambda \cdot (C_{1j} - C_{2,j-1}) + \lambda \cdot U_j \cdot p_{2j}$.

Case 2*:* $LB_j < C_{2,j-1} < UB_j$

Let $x_j^{'}$ be the number of sublots of lot j for which $C_{1j}^{'} = C_{2,j-1}$. For any $x_j^{''} > x_j^{'}$, we have $C_{1j}^{''} < C_{1j}^{'} = C_{2,j-1}$, and $C_{2j}^{''} = C_{2j}^{'} = C_{2,j-1} + U_j \cdot p_{2j}$. This implies that $x_j^{''}$ can not be optimal since it leads to the same makespan cost as $x_j^{'}$ does ($C_{2j}^{''} = C_{2j}^{'}$), but it incurs a higher material handling cost (since $x_j^{''} > x_j^{'}$). Thus, $x_j^* \in [1, x_j^{'}]$. This leads to the following property.

Property 2.2 *At stage j, if* $LB_j < C_{2,j-1} < UB_j$ *,and* $x_j^{'}$ *is the number of sublots for which* $C_{1j}^{'} = C_{2,j-1}$*, then* $x_j^* \in [1, x_j^{'}]$.

Consequently,

$$f_j(x_j) = \begin{cases} g_{1j}(x_j) = \eta_j \cdot x_j + \lambda \cdot (C_{1j} - C_{2,j-1}) + \lambda \cdot U_j \cdot p_{2j}, \ x_j \le x_j^{'}, \\ \\ g_{2j}(x_j) = \eta_j \cdot x_j + \lambda \cdot U_j \cdot p_{2j}, \ x_j > x_j^{'}. \end{cases}$$

Case 3*:* $C_{2,j-1} \ge UB_j$

In this case, $C_{2,j} = C_{2,j-1} + U_j \cdot p_{2j}$, which implies that x_j doesn't affect the makespan. Therefore, $x_j^* = 1$, and we have the following property.

Property 2.3 *At stage j, if* $C_{2,j-1} \ge UB_j$*, then* $x_j^* = 1$.

Consequently, $f_j(x_j) = g_{2j}(x_j) = \eta_j \cdot x_j + \lambda \cdot U_j \cdot p_{2j}$.

Consider the functions $g_{1j}(x_j)$ and $g_{2j}(x_j)$ as defined above. Clearly, $g_{2j}(x_j)$ is a linear function of x_j. Furthermore, $g_{1j}(x_j)$ is a strictly convex function of x_j since C_{1j} is a strictly convex function of x_j by Proposition 2.1. Therefore, $f_j(x_j)$ is a strictly convex function of x_j as well. Figure 2.2 illustrates the nature of the cost function $f_j(x_j)$ in these three cases. Note that $g_{1j}(x_j)$ dictates the determination of x_j^* in Cases 1 and 2. For the last lot, $G_n(C_{2,n-1}) = f_n(C_{2,n-1}, x_n)$.

The optimal number of sublots, x_n^* is easy to obtain as follows: examine $f_n(x_n)$ starting from $x_n = 1$. Once the function value starts to increase, the previous x_n value must be optimum.

Properties 2.1-2.3 are illustrated in Figure 2.2, and they help in significantly reducing the computational effort at every stage of the DP recursive process.

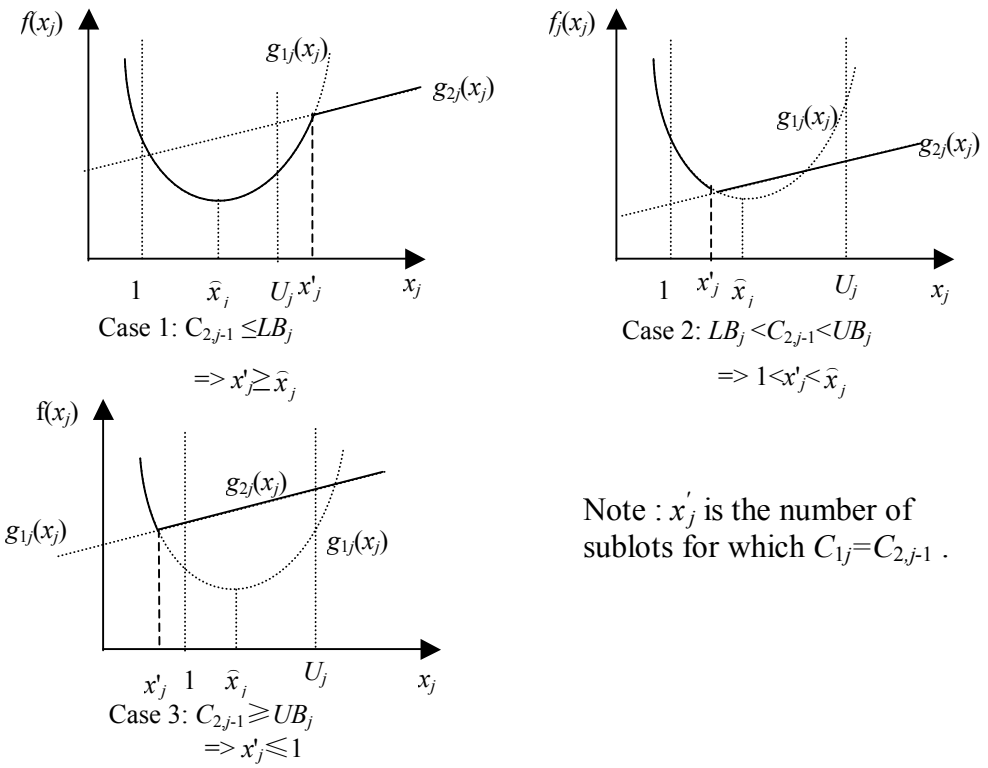


Figure 2.2 LSP-DP structural properties

2.3.2 Determination of the Optimal Number of Sublots, Sublot Sizes and the Sequence in Which to Process *n* Lots

We designate this problem by **LSSP** (the lot streaming and sequencing problem). Our objective, once again, is to minimize a joint function of the makespan and sublot handling costs. The sublots of a lot are assumed to be processed consecutively, i.e., intermingling among the sublots belonging to different lots is not allowed. Our methodology for this problem is a modification of **LSP-DP,** and we designate it as **LSSP-DP**.

First, divide the lots into two sets, *I* and *II* as follows: $I = \{j : p_{1j} \le p_{2j}, j = 1,...,n\}$ and $II = \{j : p_{1j} > p_{2j}, j = 1,...,n\}$. Also, define $h_j = p_{1j} \cdot L_{1j}$ to be the head of lot *j,* and $t_j = p_{2j} \cdot L_{x_j j}$ to be the tail of lot *j*. Figure 2.3 illustrates the heads and tails in a schedule with geometric sublot sizes.

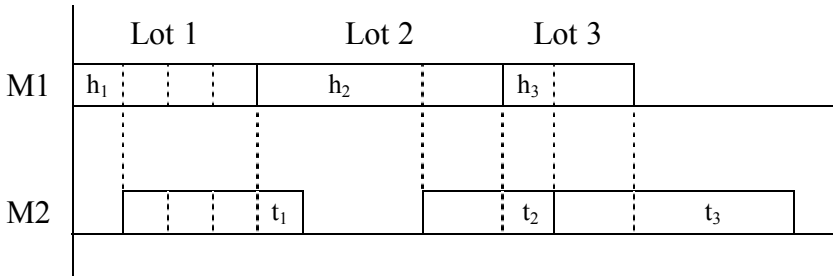


Figure 2.3 Heads and tails of lots with geometric sublot sizes

The following proposition is based on the sequencing rule proposed by Cetinkaya [10] and Vickson [77] for a given number of sublots for each lot and geometric sublot sizes.

Proposition 2.2 *There exists an optimal sequence of lots in which the lots in set I are arranged in the non-decreasing order of their heads, and are followed by those in set II that are arranged in the non-increasing order of their tails.*

By definition, h_j and t_j are monotone decreasing and strictly convex functions of x_j, j=1,...,n. Consequently, $h_j \in [\breve{h}_j, \hat{h}_j]$ and $t_j \in [\breve{t}_j, \hat{t}_j]$, where they reach their lower bounds when $x_j = \hat{x}_j$ and upper bounds when x_j=1. Therefore, in view of Proposition 2.2, we have the following dominance property:

Dominance Property: *Lot A dominates lot B* (*written as* $A \to B$) *if any of the following is true: (1) A,B∈ I and* $\hat{h}_A \le \check{h}_B$*; (2) A∈ I, B∈ II; or (3) A,B∈ II and* $\check{t}_A \ge \hat{t}_B$.

We make use of this dominance property in our LSSP-DP model.

LSSP-DP model:

In this DP model, a stage i, i = 1,...,n, corresponds to a position in the sequence of the lots. Let the $(i\text{-}1)^{th}$ position be occupied by lot k. Then, as input to stage i, we have C_{2k}, the completion time of lot k, lot k, and the leftover set of lots, $J_{i\text{-}1}$. The decisions at a stage are: (1) selection of a lot, j, out of $J_{i\text{-}1}$, to process at position i, and (2) determination of the number of sublots, x_j, to be used for processing this lot.

Define $D_{i\text{-}1}$ to be the set of dominant lots within $J_{i\text{-}1}$. The DP recursion proceeds as follows:

Step 1. If $\exists j \in J_{i-1} : j \to k$, then $G_i(k, C_{2k}, J_{i-1}) = +\infty$; else, go to Step 2. This follows because by Proposition 2.2, j must precede k.

Step 2. $\forall j \in D_{i-1}$, if $j \in I$ and $k \in I$ (by Proposition 2.2, we must satisfy $h_j \ge h_k$, which provides a new upper bound for x, namely, $\hat{x}_j'$, corresponding to the maximum x_j that meets this condition.), then go to step 2 (a); else, go to Step 3.

(a) If $\hat{h}_j < h_k$, then $G_i(k, C_{2k}, J_{i-1}) = +\infty$, else go to Step 2 (b).

(b) We have

$$G_i(k, C_{2k}, J_{i-1}) = \min_{\substack{x_j \in [1, \hat{x}_j'] \\ j \in D_{i-1}}} \left(f_j(C_{2k}, x_j) + G_{i+1}(j, C_{2j}, J_{i-1} - \{j\}) \right).$$

Recalculate LB_j for C_{1j} using $\hat{x}_j'$.

Step 3. $\forall j \in D_{i-1}$, if $j \in II$ and $k \in II$, (by Proposition 2.2, we must satisfy $t_j \le t_k$ which provides a new lower bound for x_j, namely, $\breve{x}_j'$, corresponding to the minimum x_j that meets this condition.), then go to Step 3 (a); else go to Step 4.

(a) If $\breve{t}_j > t_k$, then $G_i(k, C_{2k}, J_{i-1}) = +\infty$; else, go to Step 3 (b).

(b) We have

$$G_i(k, C_{2k}, J_{i-1}) = \min_{\substack{x_j \in [\breve{x}_j, \bar{x}_j] \\ j \in D_{i-1}}} \left(f_j(C_{2k}, x_j) + G_{i+1}(j, C_{2j}, J_{i-1} - \{j\}) \right).$$

Recalculate UB_j for C_{1j} using $\breve{x}_j'$.

Step 4. In this case $j \in II$ and $k \in I$, we have

$$G_i(k, C_{2k}, J_{i-1}) = \min_{\substack{x_j \in [1, \bar{x}_j] \\ j \in D_{i-1}}} \left(f_j(C_{2k}, x_j) + G_{i+1}(j, C_{2j}, J_{i-1} - \{j\}) \right).$$

Note that all the structural properties of **LSP-DP** also apply here.

Obviously, the **LSSP-DP** model is expected to require more computational effort for the solution of the LSSP than that required by the **LSP-DP** model for the solution of the LSP due to the combinatorial nature of the sequencing of the lots. The **LSSP** can be viewed as a non-linear, non-differentiable optimization problem with independent variables representing the number of sublots for each lot. Let these variables be designated by a vector $\boldsymbol{x}$. Also, for a given $\boldsymbol{x}$, we can determine an optimal sequence by using the result of Proposition 2.2. We denote the objective function value of this sequence, for a given $\boldsymbol{x}$, by $Y(\boldsymbol{x})$. Then, it is sufficient to search over the values of $\boldsymbol{x}$ in order to determine an optimal solution. To that end, we propose three multidimensional heuristic search procedures that obtain good solutions efficiently. These methods are: a greedy method, designated **LSSP-Greedy,** a cyclic coordinate method, denoted **LSSP-Cyclic,** and a method based on the procedures proposed by Zangwill [81] and Powell [51], designated **LSSP-ZP**.

Designate the coordinate direction vectors by Δ_j, j=1,...,n, in which the j^{th} element is 1 and all other elements are 0. Let α, β, γ be the vectors of the number of sublots.

LSSP-Greedy Heuristic:

Initialization: Set $\boldsymbol{\alpha}$=(1,1,...,1).

Step 1. Let $k = \arg\max_{1\le j\le n}\left(Y(\boldsymbol{\alpha})-Y(\boldsymbol{\alpha}+\boldsymbol{\Delta}_j),Y(\boldsymbol{\alpha})-Y(\boldsymbol{\alpha}-\boldsymbol{\Delta}_j)\right)$, $\theta = Y(\boldsymbol{\alpha})-Y(\boldsymbol{\alpha}+\boldsymbol{\Delta}_k)$,

$\tau = Y(\boldsymbol{\alpha})-Y(\boldsymbol{\alpha}-\boldsymbol{\Delta}_k)$.

If $\theta > \tau$, and θ>0, then $\boldsymbol{\alpha} = \boldsymbol{\alpha}+\boldsymbol{\Delta}_k$, repeat Step 1;

else, if $\theta > \tau$, and θ<0 or τ>θ, and τ<0, go to Step 2;

else, if $\tau > \theta$ and τ>0, then $\boldsymbol{\alpha} = \boldsymbol{\alpha}-\boldsymbol{\Delta}_k$, repeat Step 1;

Step 2. For j,l =1,..., n, j>l, let $\boldsymbol{\beta}_1^{j,l} = \boldsymbol{\alpha}+\boldsymbol{\Delta}_j+\boldsymbol{\Delta}_l$, $\boldsymbol{\beta}_2^{j,l} = \boldsymbol{\alpha}-\boldsymbol{\Delta}_j+\boldsymbol{\Delta}_l$, $\boldsymbol{\beta}_3^{j,l} = \boldsymbol{\alpha}+\boldsymbol{\Delta}_j-\boldsymbol{\Delta}_l$,

$\boldsymbol{\beta}_4^{j,l} = \boldsymbol{\alpha}-\boldsymbol{\Delta}_j-\boldsymbol{\Delta}_l$. Also, let

$$\theta_2 = \max_{\substack{1\le j,l\le n\\ j>l}}(Y(\boldsymbol{\alpha})-Y(\boldsymbol{\beta}_1^{j,l}),Y(\boldsymbol{\alpha})-Y(\boldsymbol{\beta}_2^{j,l}),Y(\boldsymbol{\alpha})-Y(\boldsymbol{\beta}_3^{j,l}),Y(\boldsymbol{\alpha})-Y(\boldsymbol{\beta}_4^{j,l})),$$

and (i,k) and $\boldsymbol{\beta}^{i,k}$ be the values associated with θ_2. If θ_2>0, then let $\boldsymbol{\alpha} = \boldsymbol{\beta}^{i,k}$, go to Step 1; otherwise stop.

At each iteration, this heuristic increases or decreases the number of sublots by 1 for the lot that results in the maximum improvement. If no improvement can be made for a single lot, the **LSSP-Greedy** heuristic increases the number of sublots for each possible pair of lots and checks if an improvement can be made. The heuristic stops when no more improvement is possible.

LSSP-Cyclic Heuristic:

Initialization: Set $\boldsymbol{\alpha}_1$=(1,1,...,1). Let $\boldsymbol{\beta}_1$=$\boldsymbol{\alpha}_1$, i=j=1 and k=2.

Step 1. If $Y(\boldsymbol{\alpha}_j+\boldsymbol{\Delta}_j)\le Y(\boldsymbol{\alpha}_j)$, then let $\boldsymbol{\alpha}_{j+1}$=$\boldsymbol{\alpha}_j$+$\boldsymbol{\Delta}_j$, go to Step 2; otherwise, if $Y(\boldsymbol{\alpha}_j - \boldsymbol{\Delta}_j)\le$ $Y(\boldsymbol{\alpha}_j)$, let $\boldsymbol{\alpha}_{j+1}$=$\boldsymbol{\alpha}_j$ - $\boldsymbol{\Delta}_j$; else, let $\boldsymbol{\alpha}_{j+1}$=$\boldsymbol{\alpha}_j$. Go to Step 2.

Step 2. If $j<n$, let $j=j+1$, and repeat Step 1. Otherwise, let $\boldsymbol{\beta}_{i+1}=\boldsymbol{\alpha}_{n+1}$; if $\boldsymbol{\beta}_{i+1}=\boldsymbol{\beta}_i$, stop.

Otherwise, $\boldsymbol{\alpha}_1=\boldsymbol{\alpha}_{n+1}$, $j=1$, $i=i+1$, and repeat Step 1.

This heuristic starts at the basic non-split solution, and searches along directions of x_j, $j=1,...,n$, one-at-a-time. After n steps (having searched in all directions), if improvement is found, it restarts from the best solution found so far; otherwise, it stops.

LSSP-ZP Heuristic:

Initialization: Set $\boldsymbol{\alpha}_1=(1,1,...,1)$, $\boldsymbol{\gamma}_1=\boldsymbol{\beta}_1=\boldsymbol{\alpha}_1$. Let $i=j=k=1$. Define $\boldsymbol{e}_j=\boldsymbol{\Delta}_j$, $j=1,...,n$.

Step 1. If $Y(\boldsymbol{\gamma}_i+\boldsymbol{e}_i) \le Y(\boldsymbol{\gamma}_i)$, then let $\boldsymbol{\gamma}_{i+1}=\boldsymbol{\gamma}_i+\boldsymbol{e}_i$; else $\boldsymbol{\gamma}_{i+1}=\boldsymbol{\gamma}_i$. If $i<n$, replace i by $i+1$, and repeat Step 1; otherwise, go to Step 2.

Step 2. Let $\boldsymbol{e}=\boldsymbol{\gamma}_{n+1}-\boldsymbol{\gamma}_n$, If $Y(\boldsymbol{\gamma}_{n+1}+\boldsymbol{e}) \le Y(\boldsymbol{\gamma}_{n+1})$, then let $\boldsymbol{\beta}_{j+1}=\boldsymbol{\gamma}_{n+1}+\boldsymbol{e}$; else $\boldsymbol{\beta}_{j+1}=\boldsymbol{\gamma}_{n+1}$. If $j<n$, then replace $\boldsymbol{e}_l$ by $\boldsymbol{e}_{l+1}$ for $l=1,2,...,n-1$, let $\boldsymbol{e}_n=\boldsymbol{e}$, and let $\boldsymbol{\gamma}_1=\boldsymbol{\beta}_{j+1}$, $i=1$， $j=j+1$, repeat Step 1; otherwise, $j=n$, go to Step 3.

Step 3. Let $\boldsymbol{\alpha}_{k+1}=\boldsymbol{\beta}_{n+1}$. If $\boldsymbol{\alpha}_{k+1}=\boldsymbol{\alpha}_k$, stop; otherwise, let $i=j=1$, $\boldsymbol{\gamma}_1=\boldsymbol{\alpha}_{k+1}$, go to Step 4.

Step 4. If $Y(\boldsymbol{\gamma}_i+\boldsymbol{\Delta}_i) \le Y(\boldsymbol{\gamma}_i)$, then let $\boldsymbol{\gamma}_{i+1}=\boldsymbol{\gamma}_i+\boldsymbol{\Delta}_i$; else $\boldsymbol{\gamma}_{i+1}=\boldsymbol{\gamma}_i$. If $i<n$, replace i by $i+1$, and repeat Step 1; otherwise, $i=1$, $\boldsymbol{\gamma}_1=\boldsymbol{\beta}_1=\boldsymbol{\gamma}_{n+1}$. Let $k=k+1$, repeat Step 1.

This last heuristic, **LSSP-ZP**, has been shown to be very efficient for solving multidimensional non-linear problems without using derivatives. It also allows discrete steps in searching directions which is an important feature to obtain an integer solution of the **LSSP**.

2.3.3. Numerical Experimentation

In this section, we present the results of our experimentation to depict the effectiveness of the solution procedures developed above. A problem setting is composed of four factors that impact the problem size and structure, namely, the number of lots (n), lot size (U), processing times and the makespan vs. handling cost ratio. By varying these factors at low and high levels, we intend to cover a wide range of problem instances. Obviously, the number of lots has the most significant impact on problem size. Therefore,

we vary this factor over five levels: 5, 10, 20, 100, and 200. Note that the problem size also depends on the upper bound specified on the number of sublots, $\hat{x}_j$. We determine an upper bound by comparing the makespan savings achieved with the increment in handling cost. The remaining factors are lot size, unit processing times on the machines and unit makespan to handling cost ratio. The lot size is randomly sampled at two levels: 1 to 10 and 10 to 100. The processing times are similarly sampled at two levels: 1 to 5 and 10 to 100. For the factor of unit makespan to handling cost ratio, we set the unit makespan cost at a reference value of 10, and randomly sample the handling cost from 1 to 0.1 and 10 to 100, respectively. Using the test runs, it was found that, when the handling cost is high, the LSSP heuristic procedures create identical solutions in which lot splitting is at a minimum level. The above factor levels result in 40 different combinations, and for each combination, five data sets were randomly generated, which lead to 200 data sets in total.

Our first experiment was focused on determining the performance of **LSP-DP**, which is measured by the run-time required to find an optimal solution for the LSP. It was found that **LSP-DP** takes less than a second for every data set used. This shows that **LSP-DP** is a very efficient procedure.

Next, we performed experimentation to compare the solution qualities of **LSP-DP, LSSP-DP**, **LSSP-Greedy, LSSP-Cyclic** and **LSSP-ZP** procedures, as well as two other solutions, designated as **Initial** and **SP**. **Initial** solution is the solution obtained for the initial, randomly generated sequence and with no lot streaming (i.e., the number of sublots for each lot is equal to 1), while **SP** is a sequencing procedure (based on Proposition 2.2) but involves no lot streaming. In Table 2.1, we present the improvements achieved by these methods over the **Initial** solution, expressed as % of the **Initial** solution objective value. This improvement measure is obtained for the data sets involving up to 20 lots for which the **LSSP-DP** is still efficient to use. For this case, there are 120 data sets in total. For the cases of 100 or 200 lots, we summarize, in Table 2.2, the improvement values obtained by all procedures except for **LSSP-DP**. Both Table 2.1 and Table 2.2 are set up in a format in which the values depicted in a column belongs to a solution procedure, and those in a row share the same factor setting indicated by the first

two columns of the table. Note that the number in a cell corresponding to a setting of a factor indicates the average value taken over settings of all other factors. To give the reader an appreciation of the range for each of these values, they are presented in Table A.1 and A.2 included in Appendix A, which depict their minimum and maximum values.

By comparing the columns of **LSP-DP** and **SP** in the tables above, it can be observed that both lot streaming and lot sequencing by themselves generate significant improvements over the initial solution for the objective function on hand (12.68% for **LSP-DP**; 10.14% for **SP**), and lot streaming is more beneficial than lot sequencing, even when the benefits of lot streaming are dampened by a large number of lots (3.12% for **LSP-DP** and 3.09% for **SP**). However, the highest quality solutions are obtained when lot streaming and lot sequencing are used simultaneously (the **LSSP**).

Table 2.1 Average improvement over initial solution with n≤20

Factor	Level	**SP**	**LSP-DP**	**LSSP-Greedy**	**LSSP-Cyclic**	**LSSP-ZP**	**LSSP-DP**
Number of Lots	5	9.65%	16.42%	16.89%	16.82%	16.72%	16.95%
	10	11.95%	13.76%	13.99%	13.91%	13.90%	14.12%
	20	8.82%	7.86%	9.40%	9.39%	9.40%	9.48%
Processing Time	1-5	8.96%	11.32%	11.91%	11.86%	11.89%	12.06%
	10-100	11.32%	14.05%	14.95%	14.88%	14.79%	14.97%
Lot Size	1-10	9.54%	12.29%	12.76%	12.67%	12.61%	12.92%
	10-100	10.74%	13.07%	14.09%	14.07%	14.07%	14.12%
Handling Cost	0.1-1	10.22%	13.09%	13.86%	13.82%	13.75%	13.99%
	10-100	10.06%	12.28%	12.99%	12.93%	12.93%	13.05%
Average		**10.14%**	**12.68%**	**13.43%**	**13.37%**	**13.34%**	**13.52%**

Table 2.2 Average improvement over initial solution with n≥100

Factor	Level	**SP**	**LSP-DP**	**LSSP-Greedy**	**LSSP-Cyclic**	**LSSP-ZP**
Number of Lots	100	3.78%	3.82%	3.83%	3.83%	3.83%
	200	2.41%	2.42%	2.43%	2.43%	2.43%
Processing Time	1-5	2.79%	2.81%	2.83%	2.82%	2.82%
	10-100	3.40%	3.43%	3.44%	3.44%	3.44%
Lot Size	1-10	2.70%	2.71%	2.73%	2.73%	2.73%
	10-100	3.48%	3.53%	3.53%	3.53%	3.53%

Handling Cost	0.1-1	3.47%	3.50%	3.52%	3.52%	3.52%
	10-100	2.71%	2.74%	2.74%	2.74%	2.74%
Average		**3.09%**	**3.12%**	**3.13%**	**3.13%**	**3.13%**

With regard to the performance of the proposed heuristic procedures for the **LSSP**, all three heuristics are, on the average, within 0.2% of the improvement obtained by **LSSP-DP**. Although Table 2.2 doesn't provide **LSSP-DP** solutions, the differences between the heuristic procedures get smaller due to the dampening effect of large number of lots, which is evident from significant differences between the values shown in Tables 2.1 and 2.2. It is clear from these results that the **LSSP** heuristics generate near-optimal solutions.

Among the three **LSSP** heuristics, **LSSP-Greedy** generates slightly better solutions on average and it is followed in performance by **LSSP-Cyclic**. The differences are small, however, even when the number of lots is no more than 20, and they begin to disappear for large number of lots (again, due to the dampening effect of a large number of lots on lot streaming).

Regarding the impacts of different factors, as noted earlier, with an increment in the number of lots, the impact of lot streaming tends to decrease. The same is true for the handling cost as well. However, a large size of a lot and processing time values, on the other hand, tend to promote lot streaming.

Regarding the run-times of these algorithms, **LSP-DP** always takes less than one second, and the **LSSP** heuristics' run-times only become recognizable when the number of lots exceeds 20. **LSSP-DP,** as expected, takes much more time to finish. For the cases in which the number of lots is up to 20, the maximum time it takes is 5 hours, and the average is close to 10 minutes. The run-time of **LSSP-DP** is prohibitively long for 100 and 200-lot problems, therefore it is excluded from such runs. Table 2.3 summarizes the run-times of **LSSP** heuristics for the case of 100 and 200 lots. Note that the run-times required by these heuristics are quite reasonable. However, **LSSP-Cyclic** is the fastest and a more appropriate one to use for the applications for which the run-time is critical. **LSSP-Greedy** is slower than **LSSP-ZP**, but it generates slightly better solutions.

Table 2.3 Average LSSP heuristics run-times (in seconds)

Number of Lots	LSSP-Greedy	LSSP-Cyclic	LSSP-ZP
100	12.2	<1	11.9
200	144.7	1.3	141.5

2.3.4. Concluding Remarks

In this section, we have addressed a multiple-lot, 2-machine, flow shop lot streaming problem. Our focus has been to determine an optimal number of sublots for each lot so as to minimize a joint function of the makespan and sublot handling costs. A dynamic programming-based methodology (**LSP-DP**) was developed to solve this problem. Several structural properties were used to accelerate the DP solution procedure. We have also addressed the problem of simultaneously determining the sequence in which to process the lots, the number of sublots for each lot and sublot sizes in order to minimize a joint function of the makespan and sublot handling costs. We propose a new dynamic programming model (**LSSP-DP**) and three heuristic procedures (**LSSP-Greedy**, **LSSP-Cyclic**, and **LSSP-ZP**) for this problem. Our experimentation has shown that the **LSP-DP** procedure solves large LSP problems within one second of CPU time. Significant improvements are obtained over initial randomly generated solutions as a result of both lot sequencing and lot streaming. However, lot streaming affords larger improvements over lot sequencing in view of our objective of minimizing a joint function of the makespan and sublot handling costs. The highest quality solutions are obtained by combining both lot streaming and lot sequencing. The heuristic procedures that we have proposed for the **LSSP** generate near-optimal solutions (within 0.2% of the optimal solution) and are quite efficient (run-time within 2.5 min for a 200-lot problem). Among the three heuristics, **LSSP-Greedy** generates slightly better solutions, but it takes a relatively longer run-time than that required by the **LSSP-Cyclic** heuristic. **LSSP-Cyclic** heuristic dominates **LSSP-ZP** as it generates solutions of the same quality but requires much shorter run-time.

2.4 *FLm/1/C* -- A Single Lot, Unified Cost-Based Flow Shop Lot Streaming Problem

2.4.1 Introduction

In this section, we study a single lot, multiple machine flow shop lot streaming problem (*FLm/1/C*) of determining optimal number of sublots so as to minimize a unified cost-based function that includes cost components pertaining to the various criteria of makespan, work-in-process (WIP), mean flow time of sublots, number of setups as well as the time required to transfer a sublot from one machine to another. This unified cost function captures a more realistic and a comprehensive view of the practical environment. An algorithm is presented that generates near optimal solutions, when not optimal.

In the lot streaming literature, a vast majority of research focuses on the optimization of one criterion in order to determine the sublot sizes while assuming the number of sublots to be given. However, several variants have also been presented. For the work pertaining to two machine flow shop studies, see Trietsch [71], Potts and Baker [50], Trietsch and Baker [73], Glass, et al. [22], Sen, et al. [58] and Bukchin, et al. [9]. In the case of multiple machine flow shop problems, simplifying assumptions, such as consistent sublots or equal sublot sizes, are made to keep the problem tractable. For example, see Baker [2], Trietsch [72] , Baker and Pyke [4], Kropp and Smunt [38], , Glass and Potts [23] and Chen and Steiner [12]. Kalir and Sarin [36] consider equal sublot sizes and incorporate the sublot-attached setup and transfer times in their methodology, and determine the optimal number of sublots and sublot sizes for the minimization of makespan. This study extends the work of Kalir and Sarin [36] in that we consider multiple conflicting objectives. In view of multiple objectives, Steiner and Truscott [69] deal with the minimization of makespan, flow time and processing cost (that includes inventory carrying cost and transportation cost) in a single lot m-machine open shop. However, makespan is considered separately from flow time. Bukchin and Masin [8] consider these two objectives simultaneously in a flow shop, and propose a methodology to determine an efficiency frontier in order to achieve good trade-offs

between the two objectives.

Because of the consideration of costs due to WIP and setup, one can view our lot streaming problem as a multi-stage lot sizing problem, or the economic lot sizing problem (see Eilon [17]) in the presence of lot streaming. In such problems, it is often assumed that a constant demand over an infinite horizon is to be satisfied by producing a single lot repeatedly. A typical objective involves the minimization of total cost consisting of inventory holding, setup and transportation. The last two cost components are usually proportional to the number of sublots, while the first cost component is associated with the total flow time (sometimes called "time-weighted inventory"), and hence, can be expressed as a function of the lot size and the number of sublots (see Szendrovits [70], Goyal [27], Ramasesh, et al. [55], Li and Xiao [46], Bogaschewsky, et al. [6], Drezner, et al. [16], Goyal and Szendrovits [28], Hoque and Kingsman [34], and Hoque and Goyal [33]). However, these models are based on an important assumption that the sublots must be processed consecutively on a machine without incurring idle time among them. The setup time, if considered, is encountered only once at the start of each machine. In this study, we allow sublot-attached setup and also permit idling among the sublots. Sublot-attached setup can be used to model loading and cleaning of machines, or pre-heating step in oven operations. The general version of our sublot-attached setup problem is difficult to handle; thus, we assume equal and consistent sublot sizes.

In the sequel, we first present the notation that is specific to this section. This is followed by a discussion on the properties of mean flow time (MFT) and average WIP as a function of the number of sublots. A unified cost-based model consisting of the criteria mentioned above is then developed and analyzed to generate useful insights. A polynomial-time algorithm is presented that generates near optimal number of sublots, if not optimal. A numerical experimentation is performed to determine the effectiveness of the proposed algorithm. Finally, we present results that depict the relative impact that the weights (marginal costs), used in the unified cost function (corresponding to different criteria), have on the number of sublots into which a lot is split.

Let s_j denote the sublot-attached setup time on machine j; $\mu(n)$ represent the makespan and $\mu_j(x)$ the workload on machine j for x sublots. Because of analytical

tractability, we assume equal and consistent sublot sizes. This condition, however, is not that restraining as the sublot sizes that are employed in practice are generally equal and consistent due to the convenience of using standard-size containers (e.g., the use of cassettes in semiconductor manufacturing). Moreover, we assume that the transfer time is smaller than the maximum of the summation of the unit processing time and setup time over all machines, i.e. the transfer time does not become bottleneck for any number of sublots.

For the makespan criterion, when n equal sublots (of a lot containing U items) are streamed over m machines of a flow shop, the resulting makespan is given as follows:

$$\mu(x) = \left(\frac{U}{x} \sum_{k=1}^{m} p_k + \sum_{k=1}^{m} s_k \right) + (x-1) \max_{1 \le j \le m} \left\{ \frac{U}{x} p_j + s_j \right\} + (m-1)d \,. \tag{2.10}$$

All the sublots on the bottleneck machine lie on the critical path that determines the makespan. Therefore, the first term in Exp. (2.10) consists of the time for the first sublot to be processed up to the bottleneck machine and the time for the last sublot to be completed after leaving the bottleneck machine. The second term in (2.10) is simply the processing time of the remaining (x-1) sublots on the bottleneck machine. The last term in (2.10) is the transfer time encountered between the m machines. Figure 2.4 illustrates an example consisting of 6 machines.

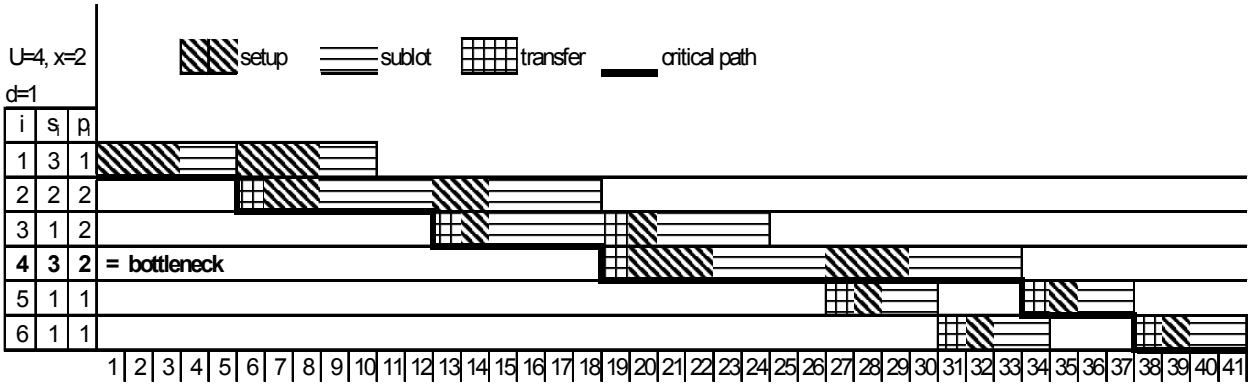


Figure 2.4 Illustration of the makespan function $\mu(x)$

Note that, we can rewrite $\mu(x)$ as follows:

$$\mu(x) = \max_{1\le j\le m}\left[\left(\frac{U}{x}\sum_{k=1}^{m} p_k + \sum_{k=1}^{m} s_k\right) + (x-1)\left(\frac{U}{x}p_j + s_j\right) + (m-1)d\right]$$

$$= \max_{1\le j\le m}\left[\mu_j(x)\right]$$

where, $\mu_j(x) = \left(\frac{U}{x}\sum_{k=1}^{m} p_k + \sum_{k=1}^{m} s_k\right) + (x-1)\left(\frac{U}{x}p_j + s_j\right) + (m-1)d$.

We have the following proposition (see Kalir and Sarin [36]):

Proposition 2.3 $\mu(x)$ *is a strictly convex function of x.*

Proof : For the sake of simplicity, let $S = \sum_{k=1}^{m} s_k$, $P = \sum_{k=1}^{m} p_k$, $a = (m-1)d$. We have,

$$\mu_j(x) = \left(\frac{UP}{x} + S\right) + (x-1)\left(\frac{U}{x}p_j + s_j\right) + a .$$

The second derivative of $\mu_j(x)$,

$$\frac{d^2\mu_j(x)}{dx^2} = \frac{2U(P-p_j)}{x^3} > 0 .$$

Hence, $\mu_j(x)$ is a strictly convex function of x. Since, $\mu(x) = \max_{1 \le j \le m}\left(\mu_j(x)\right)$, $\mu(x)$ is also a strictly function of x.

It is easy to see that, under sublot-attached setups, the mean flow time function is as follows:

$$\nu(x) = \left(\frac{U}{x}P + S\right) + \left(\frac{x-1}{2}\right)\max_{1 \le j \le m}\left\{\frac{U}{x}p_j + s_j\right\} + a \,. \tag{2.11}$$

Exp. (2.11) can be derived by applying Exp. (2.10) to each sublot in order to obtain its flow time, and then, averaging the flow times of all sublots. As before, let $\nu(x) = \max_{1 \le j \le m}\left(\nu_j(x)\right)$, where, $\nu_j(x) = \left(\frac{U}{x}P + S\right) + \left(\frac{x-1}{2}\right)\left(\frac{U}{x}p_j + s_j\right) + a$. Similar to the proof of Proposition 2.3, it can be easily shown that the second derivative of $\nu_j(x)$ is greater than 0, thereby implying the strict convexity of $\nu_j(x)$. By setting the first derivative of $\nu_j(x)$ to 0 and solving the resulting equation, we have the optimal number of sublots, x_j^*, corresponding to machine j, to be as follows

$$x_j^* = \sqrt{\frac{U\left(2P - p_j\right)}{s_j}} \,. \tag{2.12}$$

We define WIP as follows,

$$WIP = \frac{\text{total flow time}}{\text{cycle time}} \,.$$

The numerator is given by $\nu(x)$ multiplied by the number of units, U, while the cycle time is equivalent to the makespan $\mu(x)$. Consequently, the WIP function can be written as

$$\pi(x) = U \frac{\left(\frac{U}{x}P + S\right) + (\frac{x-1}{2}) \max\limits_{1\le j\le m}\left\{\frac{U}{x}p_j + s_j\right\} + a}{\left(\frac{U}{x}P + S\right) + (x-1) \max\limits_{1\le j\le m}\left\{\frac{U}{x}p_j + s_j\right\} + a}. \tag{2.13}$$

For machine j, we have:

$$\pi_j(x) = U \frac{\left(\frac{U}{x}P + S\right) + (\frac{x-1}{2})\left\{\frac{U}{x}p_j + s_j\right\} + a}{\left(\frac{U}{x}P + S\right) + (x-1)\left\{\frac{U}{x}p_j + s_j\right\} + a}.$$

Next, we develop some useful properties of the WIP functions $\pi(x)$ and $\pi_j(x)$.

Proposition 2.4. *For a given x,* $\pi(x) \le \pi_j(x)$, $\forall j, 1 \le j \le m$.

Proof: Expression (2.13) can be reduced to the following form:

$$\pi(x) = \frac{U}{2}(1 + \frac{1}{1+R(x)}), \tag{2.14}$$

where

$$R(x) = (x-1)\frac{\max\limits_{1\le j\le m}\{s_j x + Up_j\}}{(S+a)x + UP}.$$

Let $R_j(x) = (x-1)\frac{s_j x + Up_j}{(S+a)x + UP}$, and $\pi_j(x) = \frac{U}{2}(1 + \frac{1}{1+R_j(x)})$, for machine *j*, $1 \le j \le m$.

Then, we have

$$R(x) = (x-1)\frac{\max\limits_{1\le j\le m}\{s_j x + Up_j\}}{(S+a)x + UP} = \max_{1\le j\le m}\{R_j(x)\}.$$

Therefore, the result follows by Expression (2.14).

Proposition 2.5**.** $\pi_j(x)$ *is a strictly decreasing function of x, and it is also a strictly convex function of x if*

$$s_j UP < (S+a)Up_j + (s_j + Up_j)^2.$$

Proof: We can re-write $\pi_j(x)$ as follows:

$$\pi_j(x) = U\left[1 - \frac{1}{2}G_j(x)\right],$$

where

$$G_j(x) = \frac{(x-1)\left(s_j x + Up_j\right)}{(S+a)x + UP + (x-1)\left(s_j x + Up_j\right)}.$$

In order to show that $\pi_j(x)$ is a strictly decreasing function of x, it is sufficient to show that $\frac{dG_j(x)}{dx} > 0$. For simplicity, further define:

$k_1 = s_j$, $k_2 = S + a$, $k_3 = Up_j$, $k_4 = UP$ and $k_{ij} = k_i \cdot k_j$.

Substituting these in $G_j(x)$ and by taking the first order derivative of $G_j(x)$, we have

$$\frac{dG_j(x)}{dx} = \frac{k_{12}x^2 + k_{14}(2x-1) + k_3(k_2 + k_4)}{\left[k_1 x(x-1) + (k_3 + k_2)x + k_4 - k_3\right]^2},$$

which is indeed positive since $x \geq 1$.

Now, we show the second part of this proposition. After some mathematical manipulations, the second order derivative of $G_j(x)$ reduces to

$$\frac{d^2G_j(x)}{dx^2} = 2\frac{F_j(x)}{\left[k_1 x(x-1) + (k_3 + k_2)x + k_4 - k_3\right]^3}$$

where

$$F_j(x) = -k_1^2 k_2 x^3 - 3k_1^2 k_4 x(x-1) - 3k_{13}(k_2 + k_4)x + k_1 k_4^2 - \left[k_{134} + k_4(k_3 - k_1)^2 + k_2(k_3 - k_1)(k_3 + k_4) + k_2^2 k_3\right].$$

Since $x \geq 1$, it is obvious that $F_j(x)$ is a strictly decreasing function of x. As a result, $F_j(x)$ achieves its smallest upper bound when $x=1$. We designate this upper bound by $\overline{F_j}$, where

$$\overline{F_j} = (k_2 + k_4)\left(k_{14} - k_{23} - (k_1 + k_3)^2\right).$$

Clearly, when

$$k_{14} < k_{23} + (k_1 + k_3)^2, \text{ i.e. } s_j UP < (S + a)Up_j + (s_j + Up_j)^2, \tag{2.15}$$

we have $\frac{d^2 G_j(x)}{dx} < 0$, or $\frac{d^2 \pi_j(x)}{dx} > 0$, thereby implying $\pi_j(x)$ to be a strictly convex function of x.

Note that the condition in (2.15) can be rewritten as follows:

$$\frac{s_j}{(S+a)} < \frac{p_j}{P} + \frac{(s_j + Up_j)}{(S+a)} \times \frac{(s_j + Up_j)}{UP}.$$

This form maybe a little easier to decipher from a practical stand point. Moreover, note that in $F_j(x)$, the coefficients appearing in the first, second and third order terms of x are all negative. Therefore, even if (2.15) is violated, $\pi_j(x)$ would quickly turn into a convex function if not already convex, with increment in the value of x. Hence, it is safe to assume that $\pi_j(x)$ remains strictly convex in a vast majority of real situations. In the sequel, this assumption is implied in our discussion.

Corollary 2.1 $\pi(x)$ *is a segmental strictly convex function.*

Since $\pi(x)$ is the minimum over all $\pi_j(x)$ for a given x, we can not generalize the convexity of $\pi_j(x)$ to $\pi(x)$. Nonetheless, $\pi(x)$ remains convex and decreasing on each segment (as each $\pi_j(x)$ function is decreasing, by Proposition 2.5) that is associated with the dominant machine j, $j = \underset{1 \le k \le m}{\arg\max} \left\{ \frac{U}{x} p_k + s_k \right\}$, for a given x. Figure 2.5 demonstrates the nature of the $\pi(x)$ function.

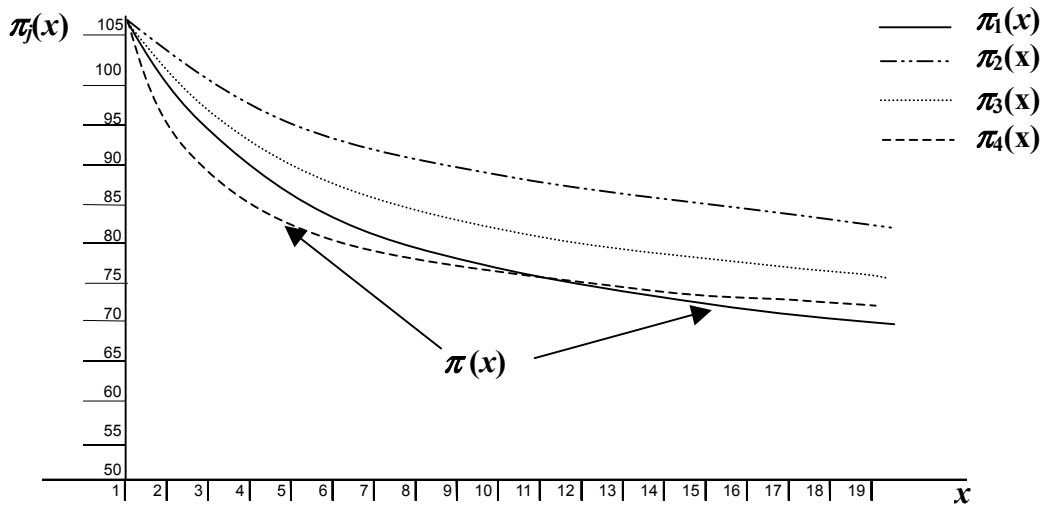


Figure 2.5 Plots of functions π (x), and $\pi_j(x)$,1≤ j ≤ 4

Obviously, the nature of our WIP function tends to increase the optimal number of sublots, if minimization of the WIP is included in the overall cost function. This is illustrated in Figure 2.6, where the optimal number of sublots is 4 when the objective is only to minimize the makespan, and it increases to 6 when a joint function of the WIP and makespan is minimized. Of course, the degree of such an impact depends on the value of the weights associated with each criterion. We discuss this issue later in this dissertation.

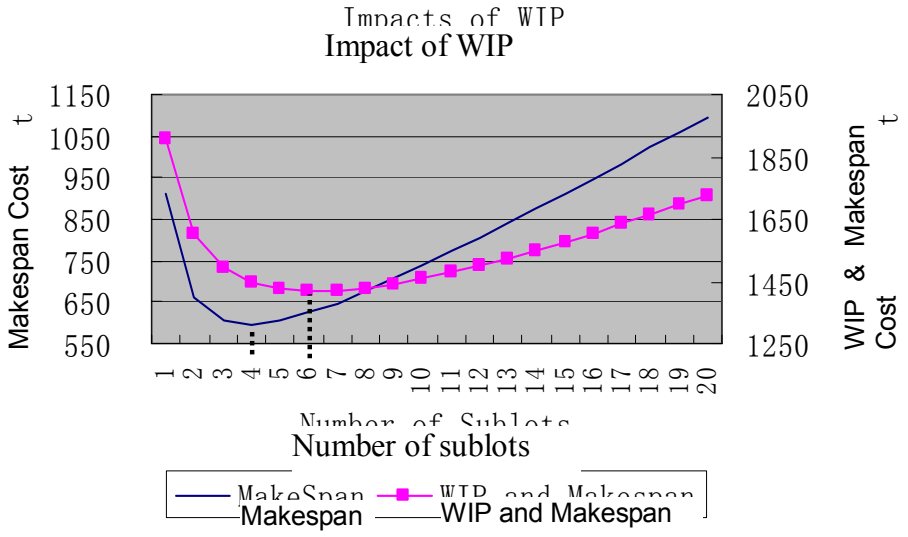


Figure 2.6 Impact of WIP objective

2.4.2 A Unified Cost-Based Model

Our unified cost-based model comprises of criteria pertaining to makespan (μ), MFT (ν), WIP (π), setup (θ) and transfer time (τ). Let c_1, c_2 , c_3 , c_4 , c_5 be the weights (marginal costs) associated, respectively, with the above criteria, which also reflect the degree of importance of these criteria. Then, we have:

Minimize $Z(x) = c_1\mu(x) + c_2\nu(x) + c_3\pi(x) + c_4\theta(x) + c_5\tau(x)$ (2.16)

Subject to

$1 \le x \le \text{U}.$

The expressions for the functions $\mu(x)$, $\nu(x)$ and $\pi(x)$ are presented in (2.10), (2.11) and (2.13), respectively. $\theta(x)$ is the sum of all the setup times performed on the lot. That is,

$$\theta(x) = S \cdot x\,, \tag{2.17}$$

while the total transfer time,

$$\tau(x) = a \cdot x\,. \tag{2.18}$$

As noted earlier, $\mu(x)$ and $\nu(x)$ are strictly convex functions of x. Since $\theta(x)$ and $\tau(x)$ are linear, and $\pi(x)$ is a segmental convex function (by Corollary 2.1), it follows that $Z(x)$ is a segmental, strictly convex function. This is stated below.

Corollary 2.2. *The objective function, Z(x), is a segmental strictly convex function of x.*

As alluded to in the discussion following Corollary 2.1, the segmental nature of $Z(x)$ is caused by the bottleneck expression $\max\limits_{1\le k\le m}\left\{\frac{U}{x}p_k + s_k\right\}$. Obviously, the machines that

never achieve this maximum value need not be considered in our search for the segments. This leads to the following "dominance property" among the machines.

Dominance Property. If, for any machine pair (k,l), the following holds:

$$(s_k \geq s_l) \text{ and } (p_k \geq p_l),$$

then machine l is not a bottleneck.

Denote by $\overline{S}$ the set of bottleneck machines obtained after applying the above property. The segments are defined by the intersection points of the cost function belonging to different bottleneck machines. These intersection points can be found by equating the bottleneck expressions of two consecutive bottleneck machines, j and l. This gives the intersection point, n_{jl}, as follows

$$x_{jl} = \frac{U(p_j - p_l)}{s_l - s_j}. \tag{2.19}$$

Note that, when n=1, the bottleneck machine is the one with the minimum s value among the dominating machines. As n increases from 1 to Q, we find the next bottleneck machine l by using the following expression

$$l = \arg\min_{k \in \overline{S}} \left\{ \frac{p_k - p_j}{s_j - s_k} \right\},$$

where j is the previous bottleneck machine. Proceeding in this fashion, we can find the sequence of bottleneck machines and the associated intersection points that are encountered with the increment of x.

Before we introduce the algorithm, we discuss the issue of the first derivative test which is utilized by the algorithm. For the objective function, we have (for each segment in which j is the maximizing index):

$$\frac{dZ_j(x)}{dx} = c_1 \frac{d\mu_j(x)}{dx} + c_2 \frac{d\nu_j(x)}{dx} + c_3 \frac{d\pi_j(x)}{dx} + c_4 \frac{d\theta(x)}{dx} + c_5 \frac{d\tau(x)}{dx}$$

which leads to:

$$\frac{dZ_j(x)}{dx} = \frac{-U}{x^2}\left(c_1\left(P - p_j\right) + c_2\left(P - \frac{p_j}{2}\right)\right)$$

$$+\left(s_j\left(c_1 + c_2/2\right) + c_4 S + c_5 a\right) + c_3 \frac{d\pi_j(x)}{dx}. \tag{2.20}$$

Note that

$$\frac{1}{Q}\frac{d\pi_j(n)}{dn} = -\frac{k_{12}n^2 + 2k_{14}n + (k_3 - k_1)k_4 + k_{23}}{2\left[k_1 n^2 + \left(k_3 - k_1 + k_2\right)n + k_4 - k_3\right]^2}. \tag{2.21}$$

Substituting Exp. (2.21) in Exp. (2.20), and setting it to zero, leads to an equation in the fourth power of n for which we can obtain a solution only by applying numerical techniques. Instead, we propose a quick approximation for Exp. (2.21) that will enable us to obtain a closed-form solution when Exp. (2.20) is set to zero. In the approximation, we only consider the highest power terms of Exp. (2.21). The approximation is given by Exp. (2.22).

$$\frac{d\pi_j(x)}{dx} \approx -Q\frac{x^2 k_{12}}{x^4 2k_1^2} = -\frac{1}{x^2}U\frac{S + a}{2s_j}. \tag{2.22}$$

Substituting (2.22) in (2.20) and setting it equal to zero leads to a closed-form solution for the desired value of x:

$$x_j^* = \sqrt{\frac{c_1 U\left(P - p_j\right) + c_2 U\left(P - p_j/2\right) + c_3 U\dfrac{S + a}{2s_j}}{s_j\left(c_1 + c_2/2\right) + c_4 S + c_5 a}}. \tag{2.23}$$

Clearly, this approximation is more accurate for large values of x. For small values of x, we can simply enumerate over all possible values of x and choose the best. In other words, this approximation is more accurate and useful when the optimal x^* is potentially large and enumeration is not an option. Note that the above expression of x_j^* implies the following:

1. x_j^* is proportional to the square root of U.

2. An Increment in $p_k, k \neq j$, will also increase x_j^*.

3. If p_j or s_j is increased while keeping the total processing/setup time the same, x_j^* will decrease, i.e., evenly distributed processing/setup times promote lot streaming.

4. The sequence in which the operations are performed in the flow shop does not affect the optimal x^*.

We are now ready to introduce the solution procedure. This procedure works as follows. It searches for the minimizing solution over each segment of the function $\pi(x)$, keeping the 'best solution so far', as it moves from one segment to the next. Within each segment, it first checks the end-point of the segment (i.e., the intersection point), and then, checks for an optimal solution using the first derivative test. We use "check" to denote "compare with the current best solution and update it if necessary". This procedure is repeated until all the segments have been searched. At this point, the procedure stops.

We note that the solution procedure finds the optimal solution if it occurs at an intersection point. If, however, it does not occur at an intersection point, the solution procedure finds a quick approximation to the optimal solution, based on the above analysis (Exp. (2.23)). The approximation is more accurate when the marginal cost c_3 is very small in comparison with the values of other marginal costs, or the optimum occurs for a large value of *x*. To solve the problem to optimality, one can replace the equation given in the second step of the procedure with a numerical technique for the solution of the equation obtained by equating Exp. (2.20) to zero.

Solution Procedure

Step 1. Set segment start point $x_s = 1$; $Z_s = Z(1)$. Also, set the current best solution $x^* = 1$; $Z^* = Z(1)$.

Apply the Dominance Property to eliminate some of the candidate machines, and define set $\overline{S}$.

Locate the first bottleneck machine: $j = \arg\min_{k \in \overline{S}} \{s_j\}$.

For the sake of convenience, we designate $Z(x_j)$ as Z_j in the sequel.

Step 2. Compute x_j^* using expression (2.23).

Case 1: $|\overline{S}|$=1 (we are at the last segment)

Set segment end point $x_e = U$; $Z_e = Z(U)$. Compare Z_e and Z^* . If $Z^* > Z_e$, then set $Z^* = Z_e$ and $x^* = x_e^*$.

If $x_s \le x_j^* \le x_e$, compare Z_j^* and Z^* . If $Z^* > Z_j^*$, then set $Z^* = Z_j^*$ and $x^* = x_j^*$. Otherwise, the current best solution is optimal. Stop.

Case 2: $|\overline{S}|$>1 (we have more than one segment to search)

Let $l = \arg\min_{k \in \overline{S}} \left\{ \frac{p_k - p_j}{s_j - s_k} \right\}$,

To find the next intersection point, calculate $x_{jl} = \frac{U(p_j - p_l)}{s_l - s_j}$.

If $x_{jl} > x_s$, then continue. Otherwise go to step 3.

Set $x_e = x_{jl}$; $Z_e = Z_{jl}$.

If $Z^* > Z_e$, then set $Z^* = Z_e$, and $x^* = x_e$.

Compare x_j^* with both x_s and x_e.There are three possible cases:

a) $x_j^* < x_s$: the current x^* is still the best.

b) $x_s \le x_j^* < x_e$: if $Z^* > Z_j^*$, then set $Z^* = Z_j^*$, and $x^* = x_j^*$. Otherwise, the current x^* is still the best.

c) $x_j^* \geq x_e$: The end-point is already checked, and hence, the current x^* is still the best.

After cases a), b) or c) whichever is encountered, set $x_s = x_e$.

Step 3. Remove j from $\overline{S}$.

Set $j \leftarrow l$.

Go to step 2.

The complexity of this solution procedure is $O(m)$computations and $O(m^2)$ comparisons.

2.4.3 A Study on the impact of assigning weights to various criteria

Having analyzed the unified cost function, we can determine optimal number of sublots for any combination of the criteria by setting the weights of the other criteria as zero. Figure 2.7 depicts the nature of each criterion when considered by itself. For instance, the WIP decreases with an increment in the number of sublots while the number of setups and transfer time increase. The makespan and MFT first decrease and then increase with an increment in the number of sublots. The impact of considering the WIP in conjunction with the makespan was shown in Figure 2.7.

Next, we study the impact that the assignments of weights (marginal costs) have on the optimal number of sublots. We consider two sets of performance measures: an absolute measure (as defined previously) and a relative measure (to be defined below). We use a relative measure to determine the relative impact of a change in the value of a weight on x^* since the absolute values of the criteria can be quite different, and therefore, may not reveal the true impact of the marginal costs used. For example, if the makespan value is much larger than that of the WIP, a large cost will have to be associated with WIP in order for this criterion to be effective. However, it is not clear as to what this value ought to be. Moreover, the optimal solution may not be sensitive enough to a change in the marginal cost value, depending on the problem instance. We will designate this behavior as "problem sensitive". Assigning an appropriate marginal cost value to a criterion becomes difficult due to this problem sensitive nature. Alternatively, instead of

estimating the worth of a job in WIP or that of a unit time in makespan, we may prefer to use the cost of one percent of increment over the optimal individual value of a criterion, or the cost of one percent of the benefit obtained over the maximum possible range. By using these relative measures, the optimal solution may become more sensitive to cost assignments, which we will designate as "cost sensitive" in the sequel.

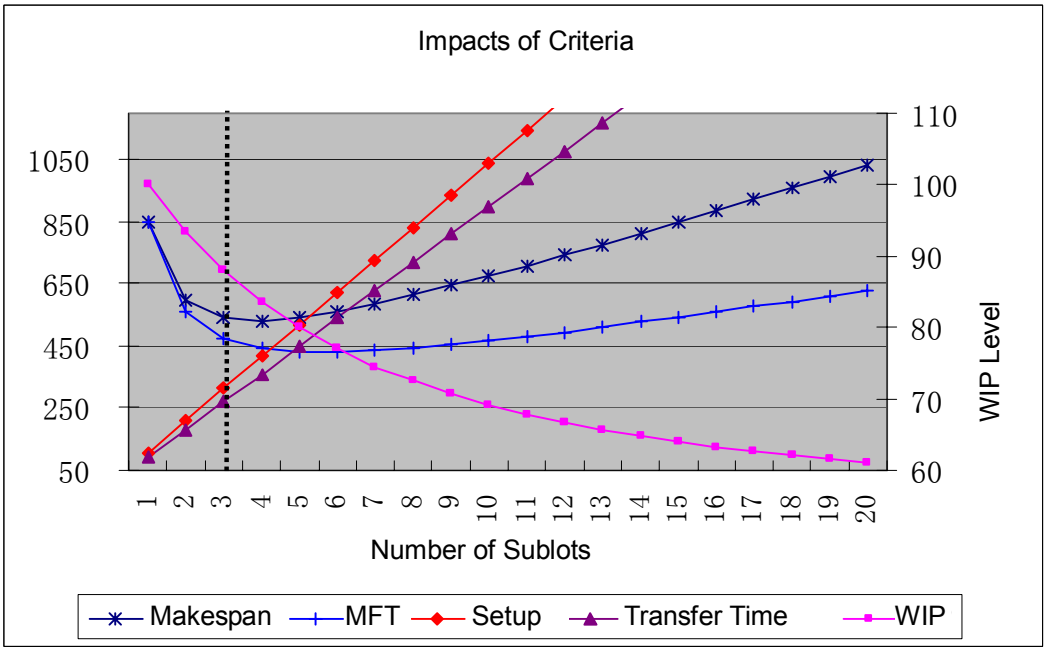


Figure 2.7 Shapes of the cost components

To that end, one way to define a relative measure is to use the relative increment over the optimal solution. Let $Y(x)$ be the absolute value of criterion Y for a given x and Y^* be the optimal value with respect to that criterion alone under lot streaming. We define a relative measure as follows:

$$Y_1\%(n) = \frac{Y(x) - Y^*}{Y^*} \times 100\% .$$

Note that $Y_1\%(n)$ ranges from 0 to $\frac{\overline{Y} - Y^*}{Y^*}$ where $\overline{Y}$ is the smallest upper bound corresponding to objective Y (for makespan and mean flow time, we take $\overline{Y}$ as the

objective value when no lot streaming is allowed or x=1). This range is much smaller than that of the absolute measure which is from Y^* to $\overline{Y}$, and makes it more sensitive to marginal cost values. We call this measure Relative 1.

Alternatively, we can define another relative measure for objective Y as

$$Y_2\%(n) = \frac{Y(x) - Y^*}{\overline{Y} - Y^*} \times 100\ \% \ .$$

In this case, $Y_2\%(x) \in [0,1]$. We designate this measure as Relative 2.

We can replace the absolute objective functions in $Z(x)$ with these two sets of relative objectives to obtain two relative total cost functions, designated $Z_1\%(x)$ and $Z_2\%(x)$, respectively. Note that the nature of these two objective functions is still identical to that of the original $Z(x)$, which enables us to apply the proposed algorithm for their solution. The cost values used in Expression (2.23) are now scaled down by Y^* and $\overline{Y} - Y^*$, respectively.

In order to reveal the impact of marginal costs on the optimal number of sublots, we consider only pairs of criteria since the case with more than two criteria quickly becomes intractable. For average WIP and makespan, we let c_2=c_4=c_5=0, $r = \frac{c_3}{c_1}$, $a_j = \frac{U(S+a)}{2s_j^2}$ and $b_j = \frac{U(P-p_j)}{s_j}$ in Expression (2.23), and obtain

$$x_j^* = \sqrt{a_j r + b_j} \ . \qquad (2.24)$$

First note that the dominant intervals of segments are invariant to a change in the marginal cost value (see Expression 2.19). Consequently, x_j^* may move from one dominant interval to another with a variation in r, and so will x^*.

Considering the relative measures, the costs associated with the makespan and WIP, namely, c_1 and c_3, become $\frac{c_1}{\mu^*}$ and $\frac{c_1}{\overline{\mu} - \mu^*}$, and $\frac{c_3}{\pi^*}$ and $\frac{c_3}{U - \pi^*}$, respectively, for the

relative measures 1 and 2. Expression (2.24) now becomes $x_j^* = \sqrt{a_j \sigma_i r + b_j}$ for relative measure i, i=1,2, where $\sigma_1 = \frac{\mu^*}{\pi^*}$ and $\sigma_2 = \frac{\bar{\mu} - \mu^*}{Q - \pi^*}$, respectively. Therefore, the values of the product of a_j and σ_1 or a_j and σ_2 reflect the sensitivity of the two relative measures to the marginal cost assignment. Similarly, we can obtain corresponding expressions and coefficients for other pairs of criteria. In Table 2.4, we present such expressions for marginal cost ratios of c_3 and c_1, c_2 and c_1, c_3 and c_4, and c_1 and c_4. Cost functions pertaining to the mean flow time and transfer time are similar to those for the makespan and setup time, respectively. Therefore, the expressions and coefficients associated with the corresponding comparisons are omitted from Table 2.4, for the sake of brevity.

Table 2.4 Relationships between marginal cost ratios and x_j^*

Marginal Cost Ratios (r)	Expression under Absolute Measure	a_j	b_j	σ_1	σ_2
c_3/c_1	$x_j^* = \sqrt{a_j r + b_j}$	$\frac{U(S+a)}{2s_j^2}$	$\frac{U(P-p_j)}{s_j}$	$\frac{\mu^*}{\pi^*}$	$\frac{\bar{\mu} - \mu^*}{U - \pi^*}$
c_2/c_1	$x_j^* = \sqrt{a_j - \frac{b_j}{r+2}}$	$\frac{U(2P-p_j)}{s_j}$	$\frac{2UP}{s_j}$	$\frac{\mu^*}{\nu^*}$	$\frac{\bar{\mu} - \mu^*}{\bar{\nu} - \nu^*}$
c_3/c_4	$x_j^* = \sqrt{a_j r}$	$\frac{U(S+a)}{2s_j S}$	N/A	$\frac{S}{\pi^*}$	$\frac{(U-1)S}{U - \pi^*}$
c_1/c_4	$x_j^* = \sqrt{a_j - \frac{a_j b_j}{r+b_j}}$	$\frac{U(P-p_j)}{s_j}$	$\frac{S}{s_j}$	$\frac{S}{\mu^*}$	$\frac{(U-1)S}{\bar{\mu} - \mu^*}$

It is clear from Table 2.4 that x_j^* and the listed marginal cost ratios are positively correlated. However, these expressions demonstrate two different types of relationships. The expressions associated with the average WIP objective (coefficient c_3) are much more cost sensitive than the others, especially under relative measures, due to the product

form of the coefficients. For such expressions, the product of coefficients a_j and μ_1 or a_j and μ_2 can be used as the cost sensitivity index (CSI). The larger the value of this cost sensitivity index, the greater the care one must execute in making marginal cost assignments. For the other expressions, which do not involve the average WIP objective, their corresponding μ_1 and μ_2 can be directly used as the cost sensitivity index.

In order to illustrate the relationships observed above, consider a problem instance with ten machines. The lot size (U) is 100 and the transfer time (d) is 10. The other problem data is listed in Table 2.5.Under the three performance measures, namely, Absolute, Relative 1 and Relative 2, we vary the marginal cost ratio of WIP and makespan from 1 to 10, and obtain x^* using our algorithm. The results are shown in Figure 2.8.

Table 2.5 Problem data for experiment 1

Machines	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	Total
Setup Time	12	34	29	5	40	**29**	23	35	14	12	233
Proc. Time	7	9	4	9	6	**8**	9	3	5	1	61

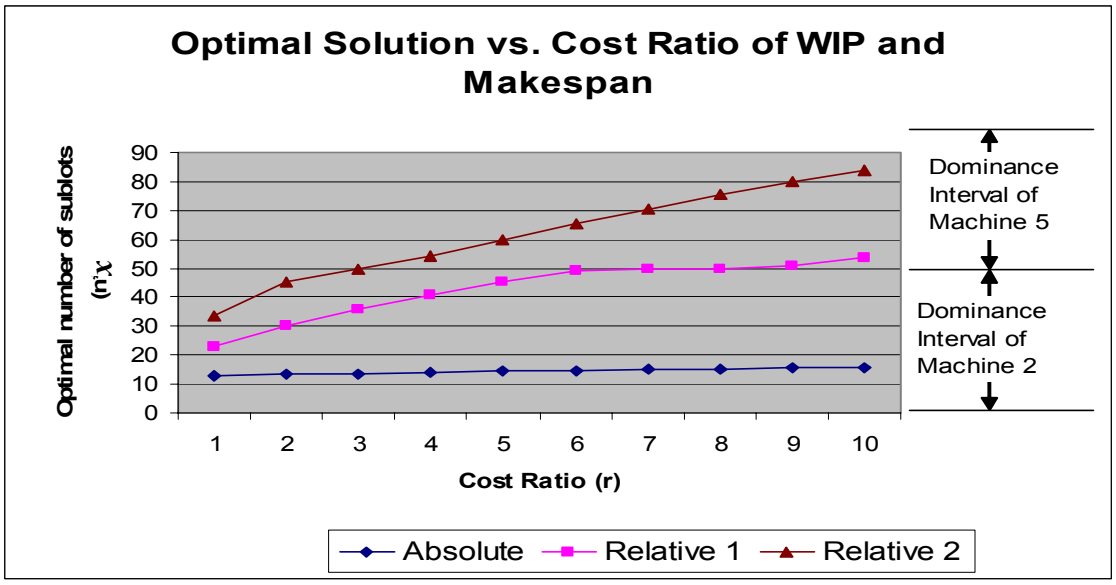


Figure 2.8 Optimal number of sublots under various marginal cost ratios of average WIP and makespan

In this problem instance, the dominant machines are machines 2 and 5. The dominance interval of machine 2 is [1,50] and that of machine 5 is [50, 100]. The *CSI*'s for all three performance measures for machine 2 are as follows: Absolute: $CSI = a_2 = 10.08$; Relative 1: $CSI = a_2\sigma_1 = 379.62$; and Relative 2: $CSI = a_2\sigma_2 = 960.11$. Note that these cost sensitivity indexes are only valid in machine 2's dominant interval [1,50]. The *CSI* values of Relative 1 and Relative 2 measures are much larger than that of the Absolute measure, and hence, they capture the sensitivity of x^* to cost ratios as shown in Figure 2.8. Moreover, note that the nature of the x^* values changes after $x^* = 50$. This is due to a shift in the bottleneck machine from machine 2 to 5 in accordance with their dominance interval.

To illustrate the other class of relationships that does not involve average WIP, we plot x^* against the cost ratios of mean flow time and makespan in Figure 2.9.

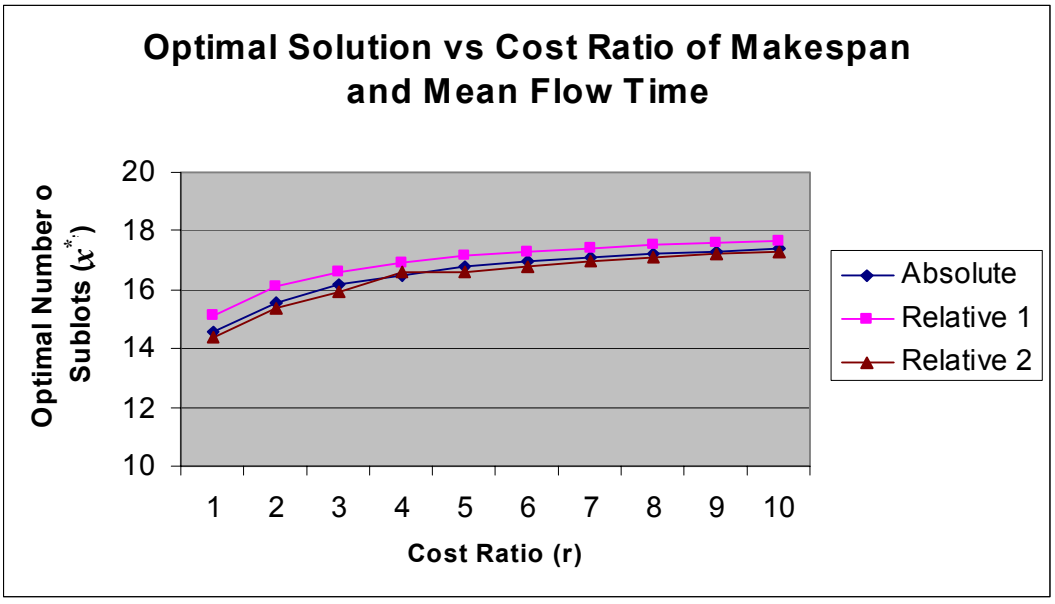


Figure 2.9 Optimal number of sublots under various cost ratios of Mean Flow time and Makespan

It is clear that, for this case, the optimal solution is not as sensitive as in the previous example. Moreover, σ_1 and σ_2 for this instance are found to be 1.77 and 0.91, respectively. Since they are close to 1, Absolute, Relative 1 and Relative 2 measures are

in the same range of marginal cost sensitivity. Note that the optimal makespan is always greater than the optimal mean flow time (see Expression (2.10) and (2.11)), but is less than twice of the mean flow time (again by Expression (2.10) and (2.11)). Moreover, the upper bounds of makespan and mean flow time, $\overline{\mu}$ and $\overline{\nu}$, defined above for x=1, are equal. Therefore, we have $\sigma_2 < 1 < \sigma_1 < 2$, which indicates that the sensitivity to marginal cost values of absolute and relative measures are not as significant as far as the makespan and mean flow time are concerned.

To summarize, our results show that the optimal solution is more sensitive to the marginal cost ratios involving the average WIP criterion than others, especially under the relative measures. Consequently, one ought to be careful in assigning marginal cost values based on the criteria involved. The use of cost sensitivity indexes, presented above, can help in determining appropriate values of these marginal costs. Extra care should be taken while adjusting the cost coefficients of those criteria that have large cost sensitivity indices.

The purpose of this experimentation is to show the effectiveness of using Expression (2.23). Consider a six-machine flow shop. The lot size, U = 2500. Setup and processing times are given in Table 2.6. The coefficients for the various contents of the unified cost function (i.e., for makespan, MFT, setup, and transfer) are equal to 1 except that for WIP, which is 2. The transfer time per sublot, d=10 time units. The solution found by our procedure is $x = 6.25$ with $Z = 19{,}084$. The optimal solution found via numerical analysis is $x^* = 5.14$ with $Z^* = 18{,}867$. Thus, the objective function value of the sub-optimal solution obtained using Expression (2.23) is only 1.15% above the objective function value of the optimal solution. As alluded to earlier, the quality of the solution obtained by using Expression (2.23) improves for higher values of x. Thus, this gap may even be smaller for large values of x. However, note that, the discrete solution, under the approximation scheme as well as the optimal scheme, is attained for $x^* = 5$ with $Z^* = 18{,}932$.

Table 2.6 Problem data for experiment 2

Machines	M1	M2	M3	M4	M5	M6	Total
Setup Time	10	100	180	210	100	200	800
Proc. Time	1.20	1.10	0.90	0.80	0.70	0.50	5.2

Given the sensitivity of the optimal solution to the value of c_3 used, based on our analysis presented in section 4, next, we numerically demonstrate the impact of c_3 on the quality of solution obtained by using Expression (2.23). To that end, we varied both c_3 and U. The solutions obtained by our procedure are compared with the optimal solution. The results are summarized in Table 2.7.

It is evident from Table 2.7 that, even with the variation of the most sensitive coefficient (c_3), our solution procedure performs extremely well. For the instances when our solution procedure does not find the optimal solution, the difference of its objective function value from that of the optimal solution is mostly within 1%. Moreover, our solution procedure finds the optimal solution in many cases, because it occurs at an intersection point. It is also interesting to note that the optimal solution does frequently occur at an intersection point.

Table 2.7 Quality of solutions obtained by our procedure with variation in the values of the WIP cost coefficient, c_3

		Our Procedure		Optimal Solution		Difference
U	C_3	x	Z	x^*	Z^*	in Z from Z^* (%)
2,500	0.00	4.99	14,950	4.99	14,950	0.00
2,500	0.50	5.65	16,054	5.15	15,919	0.85
2,500	1.00	5.47	17,002	4.75	16,901	0.60
2,500	2.00	6.25	19,084	5.14	18,867	1.15
5,000	0.00	6.61	22,928	6.61	22,928	0.00
5,000	0.50	7.68	24,869	6.80	24,767	0.41
5,000	1.00	7.76	26,676	7.15	26,602	0.28
5,000	2.00	8.95	30,512	7.75	30,254	0.85
7,500	0.00	8.33	30,033	8.33	30,033	0.00
7,500	0.50	8.33	32,671	8.33	32,671	0.00
7,500	1.00	8.33	35,309	8.71	35,301	0.02
7,500	2.00	8.33	40,584	9.12	40,517	0.17
10,000	0.00	9.97	36,835	9.97	36,835	0.00
10,000	0.50	11.11	40,266	10.27	40,215	0.13
10,000	1.00	11.11	43,596	11.11	43,596	0.00
10,000	2.00	11.11	50,257	11.11	50,257	0.00

2.4.4 Concluding Remarks

In this section, we have studied a single-lot, multiple-machine flow shop lot streaming problem for the objective of minimizing a unified cost function. An algorithm is developed to determine the optimal number of same-size sublots. For faster results, a quick approximation equation has been developed and incorporated in the proposed

solution procedure. This equation avoids the need for numerical analysis. The complexity of this solution procedure is $O(m)$ computations and $O\left(m^2\right)$ comparisons, where m is the number of machines. Experiments with our solution procedure reveal that it frequently finds the optimal solution. When it does not find an optimal solution, the difference of its objective function value from that of the optimal solution is very small (i.e., the solution is near-optimal). We have also demonstrated the relationships between the solution generated by our solution procedure and the marginal cost ratios of various pairs of criteria. Cost sensitivity indices are proposed, which can help in estimating the impact that the adjustments in marginal cost values have on the number of sublots into which a lot is split.

Chapter 3 Lot Sizing in a Complex Batch Production System

3.1 Introduction

Shop floor control (SFC) and operational planning are two levels in the decision hierarchy of a manufacturing system. Together, they cover most of the activities that are encountered in its day-to-day operation. Hence, the effectiveness of decision making at these levels impact the overall system performance. In a complex batch production system such as wafer fabrication, these two levels of decision making are particularly important and challenging due to the inherent dynamics and complexity of the operations involved. As indicated by Uzsoy, et al. [75, 76] , issues related to operational planning and shop floor control have been studied extensively but separately for such systems in the literature. This has restricted the potential of achieving a high-level performance of these systems. Therefore, it is essential to develop methodologies that consider interaction of decisions at these two levels of the decision hierarchy. Another reason for this integration is the prevalence of computer integrated manufacturing (CIM) in wafer fabrication. CIM not only makes it possible to extract necessary shop floor information for use in operational planning, but also requires such integration for accurate control of the manufacturing system.

To study the impact of lot sizing in a complex batch production system, we consider a wafer fabrication facility. An important component of operational planning in a wafer fab is lot sizing, which specifies the amount of material to release for each customer order and the time at which to release it. It links planning decisions with shop floor control via the generation of input to the shop floor (that is, the release of wafers). Each wafer may contain a large number of chips to be fabricated by using hundreds of chemical-mechanical processing steps. Therefore, the cost associated with the processing of a wafer is very high. An appropriate lot sizing method, thus, plays an important role in controlling the cost of overproduction or shortage. We define such deviations from the

desired customer demand volume as output variability.

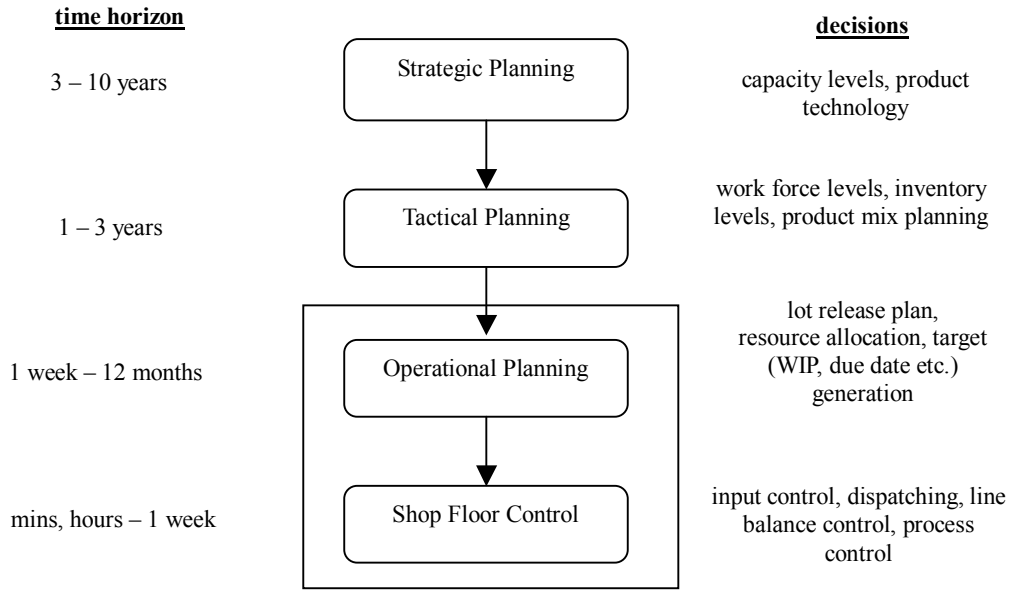


Figure 3.1 Hierarchy of decision making in a manufacturing system

Output variability is one of the major causes that impair the effectiveness of a wafer fab in meeting on-time delivery requirements. It can be handled by utilizing the following three types of buffers: the buffer of inventory (safety stock), the buffer of capacity, and the buffer of time (safety lead time). In a highly dynamic manufacturing environment, like that of a wafer fab, maintaining a fair amount of safety stock to absorb the fluctuations of demand is not always an option. This is due to a short product life cycle, which, essentially, amounts to a make-to-order system, and high cost of products. Therefore, redundant capacity and safety lead time are the potential alternatives for dealing with output variability in a wafer fab. However, even in the presence of these two types of buffers, significant surges in demand and uncertainties in process yield and cycle time still lead to a high degree of output variability. The use of appropriate lot sizes can, thus, help in further controlling output variability, as output variability can be reduced by reducing input variability. To that end, a typical lot sizing approach is to treat the production system as a "black box" and inflate the quantity to be released by the expected yield or to add a safety time-buffer upon the release of new material, hoping that the final output will meet the target. Since the production lead time in wafer fabrication is

relatively long (two to four weeks), this "aim-and-fire" strategy does not effectively utilize the available capacity and time buffers (if there is any), and it responds poorly to the dynamic shop floor status. Therefore, a better lot sizing strategy is needed to effectively utilize the available capacity and lead time buffers in the face of shop floor dynamics that are caused by demand fluctuations, machine breakdowns and random process yields in order to minimize output variability.

Input control and dispatching strategies form two basic functions of shop floor control. Input control regulates the release of materials (lots) to the production system in a controlled manner in order to maintain the balance of the workflow. Dispatching policy determines which lot to process next upon the availability of a processing machine, and thus, it directs the workflow within the wafer fab. The relationships among lot sizing, input control and dispatching are depicted in Figure 3.2.

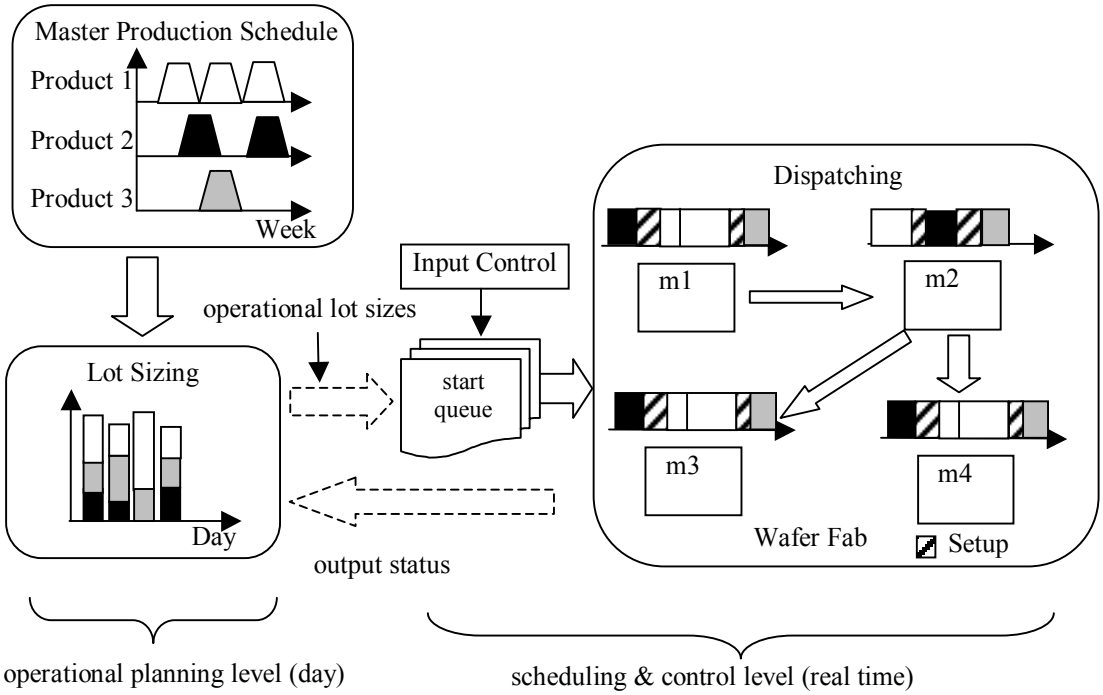


Figure 3.2 Relationships among lot sizing, input control and dispatching

The rest of this chapter is organized as follows: We first describe the wafer fab that

is considered in this study in section 3.3. We, then, introduce an integrated lot sizing and dispatching methodology in section 3.4. This is followed by our study on the input control strategy in section 3.5. The input control issue is studied separately from the lot sizing and dispatching issues because of the complexities involved. Moreover, the integrated framework for lot sizing and dispatching can easily accommodate an input control strategy, once developed. But, first, we introduce the notation used in this chapter in Section 3.2.

3.2 Notation

We use the following notation in this chapter.

Indices:

i	index of part type, i=1,...,N;
j	index of a step in the processing route of a certain part ;
t	index of period, t= 1,...,T ;
m	index of station family, m=1,...,M;

Variables:

x_{ijt}	quantity of lots to be processed at processing step j of part i in period t;
q_{ijt}	number of lots in the buffer following the processing step j of part i at the end of period t;
I_{it}	finished-goods inventory of part i at the end of period t;
S_{it}	backlog of finished-goods of part i at the end of period t;

Parameters:

a_{it}	finished-goods inventory cost (per unit / per period) of part i in period t;
b_{it}	finished-goods backlog cost (per unit / per period) of part i in period t;
c_{it}	unfinished-goods holding cost (per unit per period) of part i in period t;
f_{it}	raw material cost (per unit) of part i in period t;
r_{it}	revenue (per unit) of part i in period t;
u_{ij}	expected yield at processing step j of part i;
d_{it}	demand of part i in period t;
p_{ij}	time required to process step j of part i;
ρ_{mt}	efficiency factor ($0 \le \rho_{mt} \le 1$) of station family m in period t;
h_m	number of parallel stations in station family m;
λ_m	average batch size of station family m;
Q_m	physical queue space available for station family m;
L_{mt}	total available time of for station family m in period t;
e_{ijm}	a binary parameter indicating whether station family m is required to process step j of part i;

q_{ij0}	initial number of lots in the buffer of processing step *j* of part *i*;
I_{i0} /S_{i0}	initial finished-goods inventory/backlog of part *i;*
ls	number of pieces in the lot;
LT	total operator time to complete a lot;
S	total number of steps (operations);
LA	total lot-attached time;
PA	total piece-attached time;
y_k	yield for step k;
PA^y	expected total piece-attached time.

3.3 Description of the Wafer Fab under Study

The wafer fab that we used as a test bed for our methodology is a small wafer fab that consists of a series of different functional areas such as Photolithography, Etch I, Etch II, Material and Implant. A wafer goes through these areas repeatedly in order to build the desired circuits. Figure 4 shows these functional areas as well as a sample process flow. Note that at steps 5 and 9, a wafer revisits Etch I, and similarly it revisits the Photolithography processing area at steps 7 and 10. Each area contains a set of stations (machines). Most stations are highly automated so that an operator's tasks are mainly to load/unload the wafers and to setup the machines. This enables operators to work on multiple tasks simultaneously. The operators work 12 hours per shift and the plant operates on 2 shifts per day. The workers are cross-trained in order to enable them to work in multiple areas. However, an operator is usually certified in one skill group and works, primarily, in one processing area.

The production system operates under the make-to-order policy and is assumed to have enough capacity to meet the demand. Once an order is accepted and the order entry process is completed, it is divided into one or more production lots of sizes 4, 8, or 20. The determination of which of these standard lot sizes to use is largely based on experience. Each lot has a priority that is dictated by its due date, the order type and some other relevant factors. This priority is subject to change at any time before its completion.

The facility consists of about 115 pieces of processing tools and 28 operators who can work in these processing areas. The operators are classified into 7 skill groups. Around 40 different product types are produced at this facility, and they can be

categorized into 8 product families. The number of steps required to process these products ranges from 240 to 640, which may require two to four weeks to complete. The processing steps are grouped into operations in the wafer fab. An operation is defined as a set of consecutive processing steps that usually occur in the same processing area to accomplish a specific task on a wafer. The average throughput rate of the fab is 250 wafers per week.

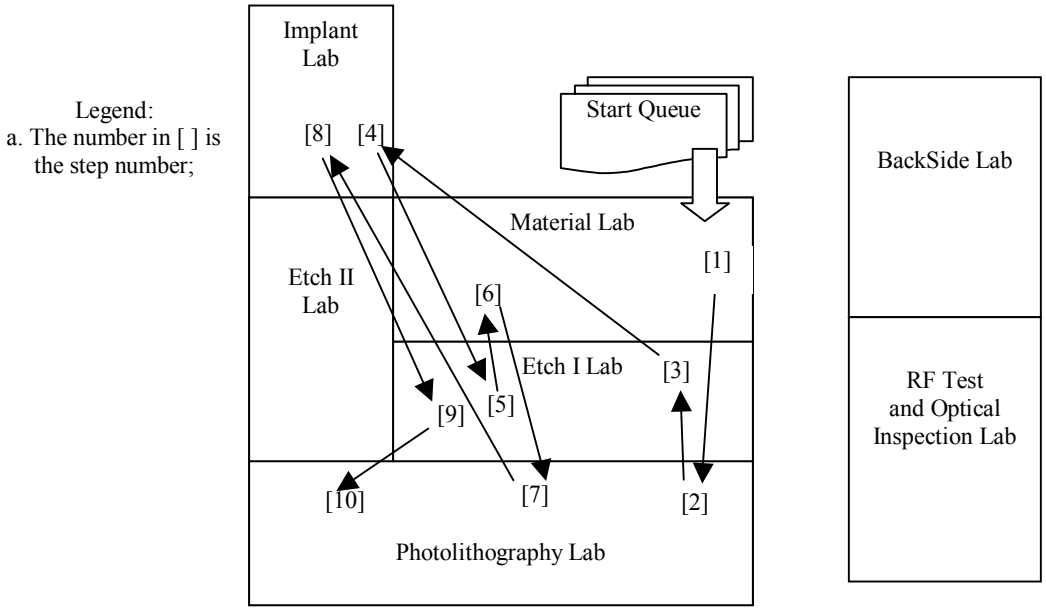


Figure 3.3 Processing areas and process flow of the wafer fab used for experimentation

3.4 Integrated Lot Sizing and Dispatching

3.4.1 Literature Review

Golovin [26] propose a hierarchical framework for the planning and scheduling of the operations involved in a wafer fabrication. Table 3.1 shows this framework, which is identical to that presented in Figure 3.1.

This framework conforms to the actual organizational structure, and hence, is claimed to be organizational effective in the sense that the process or decision-making

model is controlled by people who are responsible for the consequences of the decisions. The hierarchical framework also reduces the difficulty in wafer fab planning and scheduling by decomposing the whole problem into smaller problems at different levels. According to this framework, it is obvious that the key for the integration of shop floor control with upper level planning is the coupling of lot sizing and dispatching.

Table 3.1 Decision framework proposed by Golovin [26]

	User	Horizon	Level of Detail
Capacity Planning	Production Planner	3-12 Months	Production Line or Product Family or Process
Release Planning	Production Scheduler	1-4 Weeks	Product
Dispatching/Set Up	Supervisor/ Expeditor	1-2 Days	Lot/Product/ Equipment

There are two common approaches that can be used for lot sizing. One is to use the MRP approach that assumes a fixed production lead time. This assumption may not be that critical for wafer fabrication, if a safety lead time is included. However, the MRP calculates the release date of an order by deducting the fixed production lead time from its due date. Thus, it does not handle the capacity violations properly, which may be problematic in the face of demand surges. The second approach is to use an aggregate linear programming model. It typically assumes that the released orders are completed in the time period that they are released in, i.e., the time period is longer than the production lead time. This type of LP model is called a "big bucket" model. In this model, the production capacity is simply modeled in an aggregate fashion. Due to the long production lead time encountered in wafer fabrication, the "big bucket" model is not a suitable one. A refined capacity model is also required to capture the reentrant production line capacity.

Compared to the "big bucket" LP model, some more refined LP models, i.e. "small-bucket" models, have been successfully implemented for production planning purposes in wafer fabrication. Leachman [41] addresses the mid-term production planning problem of

a wafer fabrication facility using linear programming. This LP model prescribes capacity-feasible starts and outs for each facility of the company. Compared to lot sizing, which belongs to short term planning, the model presented by Leachman [41] addresses the capacity planning problem for both frontend and backend facilities. It is also shown to enable modeling of various physical constraints encountered, such as machine capacity, demand priorities, and classification and substitution in product structures, among others. Leachman and Carmon [42] propose a LP formulation technique, called the capacity set generation, to efficiently model the capacity of alternative machines. An extension of this technique is presented by Li and Xiao [45] for solving the bin allocation planning problem in semiconductor manufacturing systems. Their formulation technique greatly reduces the size of the LP model used. Li and Xiao [45] further improve the technique of Leachman and Carmon [42] and propose a hybrid formulation technique called capacity participation generation procedure, which overcomes the difficulty caused by a uniform assumption of Leachman and Carmon [42]. Li and Xiao [45] utilize simulation to obtain accurate flow time estimates that are used in their LP model to generate a production plan, which in turn, is fed into the simulation model to obtain new estimates of flow time. This iterative production planning approach has been shown to be effective by applying it on a real-life facility.

The studies mentioned above focus on the mid-term capacity planning problem and determine the weekly starts and outputs of a production system. They do not address the integration of lot sizing and shop floor control. In this regard, Hwang and Chang [35] develop a hierarchical approach for lot sizing and production scheduling for a wafer fabrication facility. Their approach is based on a two-level hierarchy. The lot sizing problem is solved at the upper level of the hierarchy. This solution prescribes a target number of wafer moves (a wafer move is the completion of one operation per wafer) and the WIP distribution per operation of a part type. An operation refers to a series of processing steps that are mostly completed in one processing area and belong to a certain processing function. For example, a photolithography operation may form a circuit pattern on the wafer surface. With the daily targets generated from the lot sizing, the production scheduling, which is a lower level decision, gives a detailed schedule for each day that meets the target level of wafer moves and WIP distribution. The time horizon at

this lower level is one day, as compared to that of lot sizing, which is on a monthly basis. Two similar mixed integer programming (MIP) models are developed for lot sizing and production scheduling, respectively. A Lagrangian relaxation-based solution methodology is employed to obtain solutions of these two MIP models. The proposed two-level hierarchical approach is tested using data extracted from a real wafer fab, and the results show that this approach has the potential to improve resource utilization, reduce WIP and increase throughput. Moreover, a significant reduction in output variation is obtained.

The two-level approach proposed by Hwang and Chang [35] utilizes a MIP to solve the production scheduling problem which may involve hundreds of processing tools and thousands of production lots for a large-scale wafer fab. Considering a highly dynamical wafer fab operation, it is likely that the production schedule will soon become obsolete. In this regard, we propose to use dispatching rules instead of a globally optimized schedule to meet the targets generated from lot sizing. As in most hierarchical approaches, the information flow in the model proposed by Hwang and Chang [35] is in a single direction, that is, from the planning level to the scheduling level. We believe that information flow between production scheduling and lot sizing should be in both directions in order to achieve a higher degree of integration. Therefore, our lot sizing model is designed such that the shop floor status is captured and the model can be solved efficiently. These two features allow us to determine the lot sizes very frequently and in a rolling horizon fashion so as to effectively capture the shop floor dynamics.

3.4.2 Solution Methodology

The approach proposed in this section focuses on the integration of lot sizing with shop floor control, and hence, involves two major components, namely, the lot sizing module and the dispatching module. Our goal is to minimize output variability.

3.4.2.1 Lot Sizing Module

Suppose that part i ($i = 1,...,N$) goes through K_i steps before exiting the production system. There are M station families in the system. Let the station family used at step j of part i be denoted by $\sigma(i,j)$. There is a buffer after step j of part i (B_{ij}) and its level is denoted by q_{ij}. This buffer is physically located in the storage area of the station family

$\sigma(i,j+1)$. For station family m (m=1,…, M), its storage area has a finite capacity denoted by Q_m. There is a stochastic yield associated with each processing step j of part i, denoted by u_{ij}. The station family m is available for ρ_m percent of time due to shop floor disruptions. It may contain h_m parallel stations and each station requires the same lot size, λ_m. An example of such a manufacturing system is shown in Figure 3.4.

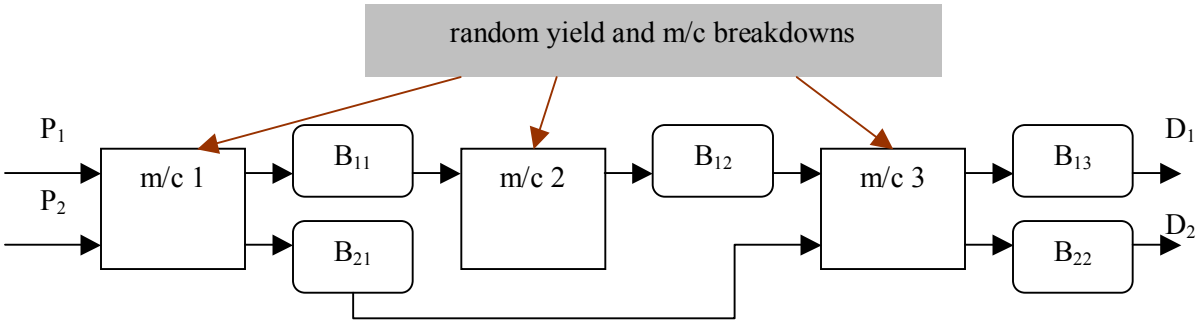


Figure 3.4 Multi-stage, multi-products system with homogeneous step buffers

The problem that we address can be stated as follows: given the demand of part type i, i=1,…,N, in period t, determine the number of lots of part type i to be processed at its processing step j in order to minimize the output variability, i.e., to minimize the deviations (including inventory and shortage) between output and demand over a given planning horizon without accumulating excess inventory of unfinished goods.

We make the following assumptions:

- The processing time required at each step is less than the length of the period.
- Supply of raw materials is not constraining.
- The last step is never blocked.
- Backlog is allowed in each period.
- Demand is satisfied at the end of each period.

We designate this lot sizing problem as DynaLRP (dynamic lot release planning). Its

LP formulation is as follows:

DynaLRP:

$$\text{minimize} \sum_{t=1}^{T}\sum_{i=1}^{n}[a_{it}I_{it} + b_{it}S_{it} + c_{it}\sum_{j=1}^{K_i-1} q_{ijt} + f_{it}x_{i1t} - r_{it}(d_{it} + S_{i,t-1} - S_{it})] \tag{3.1}$$

Subject to:

$$q_{ijt} = u_{ij}x_{ijt} + q_{i,j,t-1} - x_{i,j+1,t} \qquad \forall i = 1,...,N; j = 1,...,K_i - 1; t = 1,...,T \tag{3.2}$$

$$I_{it} - S_{it} = u_{iK_i}x_{iK_it} + I_{i,t-1} - S_{i,t-1} - d_{it} \quad \forall i = 1,...,N; t = 1,...,T \tag{3.3}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{K_i} e_{ijm}p_{ij}x_{ijt} \le \rho_{mt}\lambda_m h_m L_{mt} \qquad \forall t = 1,...,T; m = 1,...,M \tag{3.4}$$

$$\sum_{i=1}^{N}\sum_{j=0}^{K_i-1} e_{i,j+1,m}q_{ijt} \le Q_m \qquad \forall t = 1,...,T; m = 1,...,M \tag{3.5}$$

$$x_{ijt} \le \sum_{k=1}^{z_{ij}} q_{i,j-k,t-1} \qquad \forall i = 1,...,N; j = 1,...,K_i; t = 1,...,T \quad z_{ij} > 0 \tag{3.6}$$

$$x_{ijt}, q_{ijt}, I_{it}, S_{it} \ge 0 \qquad \forall i = 1,...,N; j = 1,...,K_i; t = 1,...,T$$

Our objective is to minimize the total cost of finished goods inventory and the shortage incurred, which are represented by the first two terms of (3.1). The third term in (3.1) pertains to the holding cost of unfinished goods, and the fourth term captures the raw material cost. These costs are used to avoid those plans that result in a large amount of unfinished goods. The last term in (3.1) refers to the revenue generated by the shipping of finished goods. The constraint set (3.2) maintains the conservation of material flow at intermediate stages, that is, the current buffer level of step *j* of part *i* should be equal to its original level plus the difference of input and output. Notice that the buffer levels are nonnegative. Hence, the output is bounded by the sum of its original level and input. The constraint set (3.3) enforces inventory balance at the last stage. At this stage, backlog can occur and will be carried over to later stages. The constraint set (3.4) states that the capacity of a station family should not be exceeded. The constraint set (3.5) requires that the physical capacity of the buffer at each station family should not be exceeded. The

constraint set (3.6) limits the movement of materials along their routes. Since the previous constraints do not consider the cycle time and the delays encountered by the lots, the resulting production plan may assume a faster movement rate than it is actually possible. In order to overcome this drawback, we assume that only the upstream material which is within z_{ij} steps from step j of part i can reach this step in one period. Naturally, the release quantity x_{ijt} must be less than or equal to the total available WIP in this interval. The parameter z_{ij} can be regarded as the rate at which the wafers move into step j of part i. This type of constraint is also used in Hwang and Chang [35] to model the workflow more accurately.

The above formulation addresses the random yield and machine breakdowns in a deterministic fashion. The effects of station family breakdowns have been captured by assigning appropriate efficiency values. Expected yield values are used to discount the output at each step (see constraint sets (3.2) and (3.3)). The inclusion of yield factors at every stage helps in detecting potential shortages in a timely manner and in prescribing appropriate release amount (subject to station family capacity) to compensate for yield loss.

Note that our formulation takes into account the real-time status of the system, namely, the current WIP distributions, the inventory and backlog, as well as the station family efficiencies. Thus, it can be used in a dynamic fashion to react to the random disruptions encountered at the shop floor.

We designate this lot sizing approach as DynaLRP (dynamic lot release planning) which has been implemented in an Excel-based software tool. Appendix B provides a detailed description of this software tool.

3.3.2.2 Dispatching Module

In the previous section, we presented a mathematical formulation for lot sizing, which determines the output requirement for part i at processing step j in period t (x_{ijt}) as well as the WIP level of part i at step j in period t (q_{ijt}) , over a certain number of time periods (days). We use a dispatching scheme to meet the target wafer moves specified for each day. The dispatching rule that we use in this regard is termed the Largest-Remaining-Quota-First (LRQ) rule, that gives higher priority to the buffer with the

largest unsatisfied quota. The unsatisfied quota is calculated as the difference between the target wafer moves (x_{ijt}) and the current real wafer moves for part i at processing step j. The LRQ rule can also be used in conjunction with other dispatching rules to develop a more complex dispatching scheme.

An integrated lot sizing and shop floor dispatching system is depicted in Figure 3.5. The input of this system constitutes of the weekly releases provided by capacity planning. The information about WIP status on the shop floor and current inventory/backlog is then gathered. With these inputs, lot sizing module (DynaLRP) is executed to generate the daily target wafer moves for all the buffers in the system, including input/output and internal buffers. Since a linear programming model can be solved very efficiently, DynaLRP can conveniently handle a realistic problem size. New lots are released according to the target wafer moves of the input buffer. For internal buffers, LRQ dispatching rule is used to meet the target wafer moves as closely as possible. After a certain number of time periods has elapsed, or if a certain condition (e.g. a specified value of the differences in the target WIP distribution and current WIP distribution has reached a specified limit) is met, the input of DynaLRP is then updated and a new set of target wafer moves is obtained by solving the linear programming model.

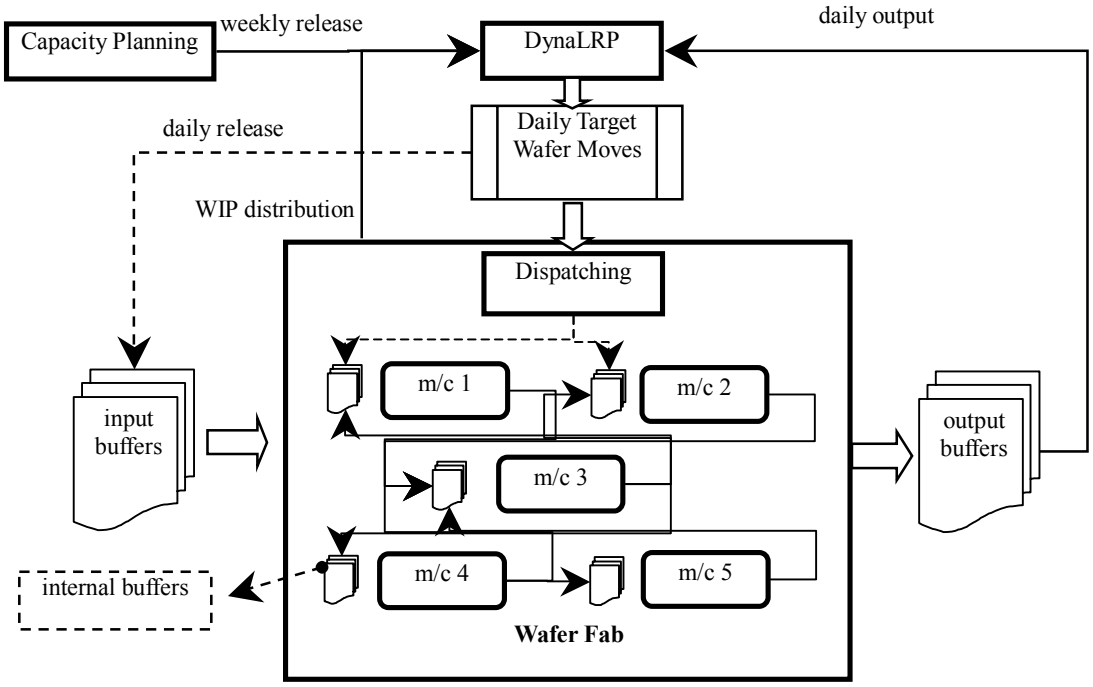


Figure 3.5 Integration scheme of DynaLRP and shop floor dispatching

The closed-loop control described above allows it to detect potential overproduction or shortages in advance, and make judicious adjustments automatically in order to minimize the output variability. The ability to accurately model the workflow and predict future output is particularly important for better resource utilization in a wafer fab. In reality, overproduction and shortage can occur simultaneously for different part types. It is not uncommon to find a great amount of effort to have been spent on wrong part types before the problem becomes serious enough to be detected by a production scheduler. In practice, hot lots (high priority lots) are used to correct the shortage problem. However, it is well known that hot lots are expensive in the sense that they disturb the work flow and cause additional setups and require additional effort. With the closed-loop control, DynaLRP can predict the future outputs with relative accuracy, which helps in preventing a potential shortage at the early stage of its occurrence. This will lead to an improvement in the overall efficiency of the system, which is the main motivation of integrating the lot sizing and shop floor control decisions.

3.4.3 Experimentation

A simulation model of the wafer fab was developed using AutoSched AP. Each processing area is modeled as an individual machine in our simulation model because an operation/stage typically occurs in one processing lab, and hence, it can be treated as a single step. Furthermore, shop floor dispatching also occurs at this level. This level of detail helps in avoiding consideration of unnecessary processing details in the lot sizing model while maintaining enough details to represent the workflow. This simulation model accommodates seven active part types as well as their process routings across the seven processing labs. The capacity of a processing lab is not as straight forward to model since the processing areas are composed of a group of different processing tools. A processing lab can operate on multiple lots simultaneously whether they are on the same processing tools or not. Thus, each processing lab is modeled as a batching machine with average batch size estimated to be the average WIP within that lab. We also collected, from the wafer fab, the average cycle time required for each operation. Since the

differences between cycle times of the same operation for different part types are negligible, we use the same cycle time for an operation across different part types. The number of operations in the processing route of a part type ranges from 30 to 63.

Next, we tested the effectiveness of our DynaLRP approach in conjunction with the LRQ dispatching rule. For this purpose, the real lot sizing data (in conjunction with the FIFO (first-in-first-out) dispatching rule which is the approach used in practice) is compared with the proposed DynaLRP and dispatching approach over a planning horizon of 60 days.

We used exponential distribution to model the random processing time of the operations involved in the processing route of the part types. Ten simulation runs were performed independently using the standard random seeds provided in AutoSched AP. The average performance measures over these ten simulation runs were used to compare the approach used in practice with our proposed approach. The performance measures used include inventory ($\sum_{t=1}^{60}\sum_{i=1}^{7} I_{it}$), backlog ($\sum_{t=1}^{60}\sum_{i=1}^{7} B_{it}$)and total WIP ($\sum_{t=1}^{60}\sum_{i=1}^{7}\sum_{j=1}^{K_i} q_{ijt}$).

Figure 3.6 depicts the actual customer demand of all part types over the planning horizon by their due dates. Note that the daily demand is quite variable with high surges. This demonstrates the dynamic nature of customer demand that the wafer fab faces. Figure 6 shows the release plans of new lots (start plan) for both the real-life and our lot sizing approaches. Clearly, there are fewer variations in the DynaLRP start plan which is in agreement with its goal. Plots of inventory, shortages and WIP for the release plan used in practice in conjunction with the FIFO dispatching rule (Real_FIFO) and the DynaLRP release plan in conjunction with the LRQ dispatching rule (DynaLRP_LRQ) are shown in Figures 3.7-3.9. Figure 3.10 summarizes the total output of wafers for both approaches.

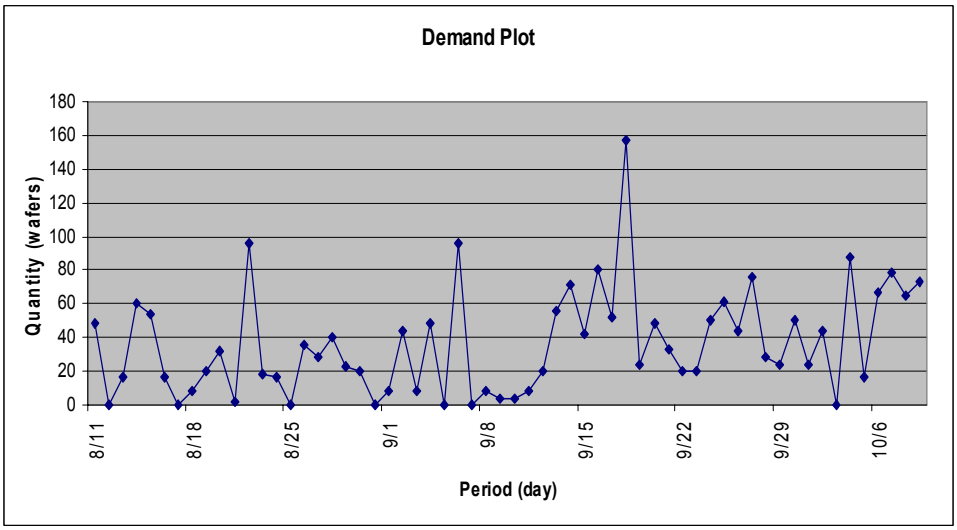


Figure 3.6 Customer demand distribution

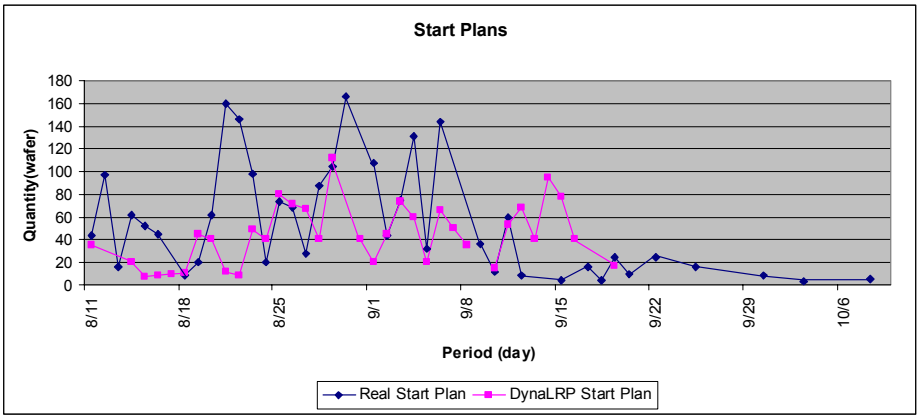


Figure 3.7 Start plans of new lots

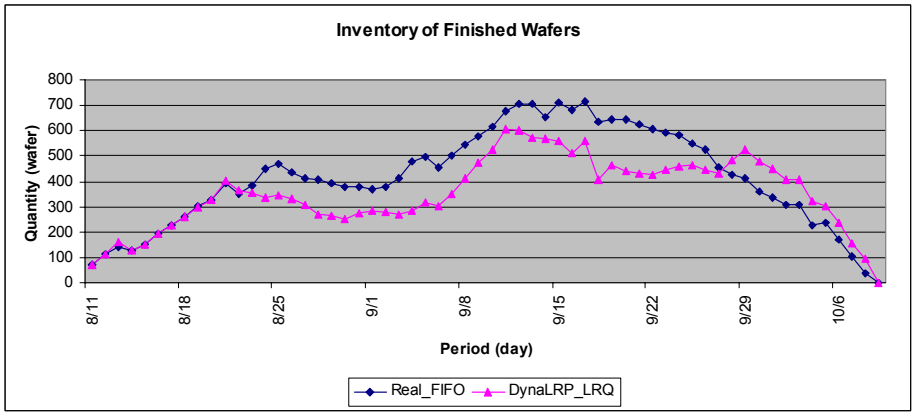


Figure 3.8 Inventory of finished wafers

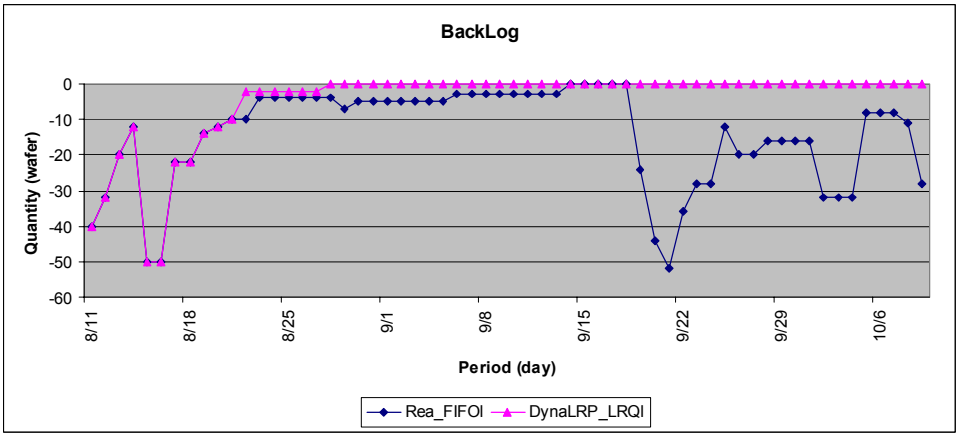


Figure 3.9 Shortage of finished wafers (backlog)

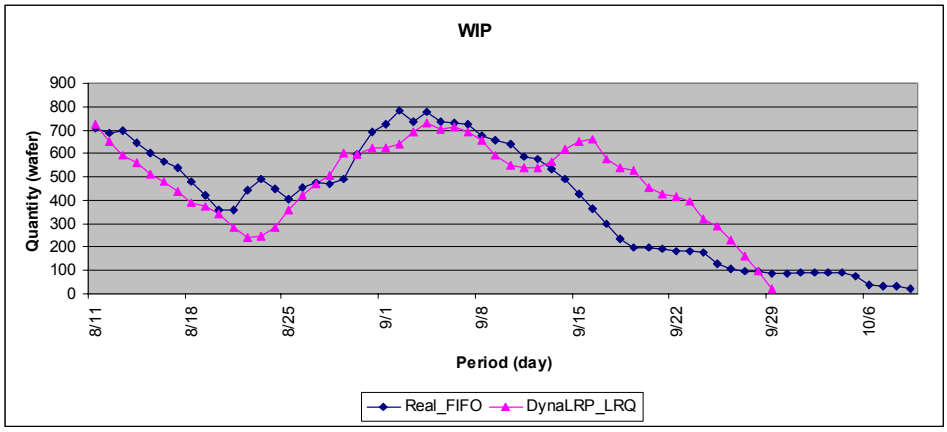


Figure 3.10 WIP plots

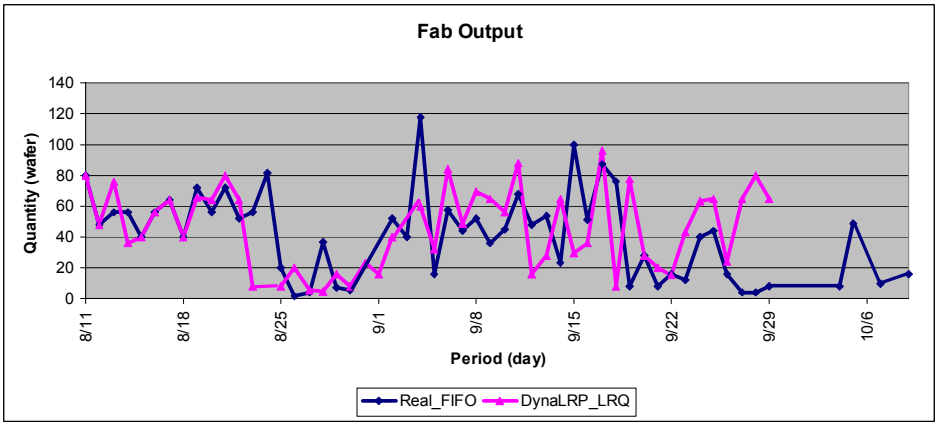


Figure 3.11 Output of wafers

Note that both inventory and backlog exist in some periods. This is because of the presence of multiple part types. Some part types may be in inventory and some may be backlogged. Such occurrences also indicate improper utilization of resources since the capacity used to produce the part types in inventory should, in reality, be used to eliminate backlog. It is clear that DynaLRP_LRQ approach results in less inventory and shortages when compared to Real_FIFO, i.e. the output variability is reduced. We also calculated the standard deviations of the performance measures over the 60 days. This is shown in Table 3.2. Table 3.3 lists the summation of the performance measures throughout the planning horizon. All values in these two tables are in units of wafers.

Table 3.2 Standard deviations of performance measures

	Start	Inventory	Backlog	Throughput	WIP
Real_FIFO	48.10	189.39	13.05	27.66	243.63
DynaLRP_LRQ	27.00	140.74	11.75	23.11	172.74
Improvement	44%	26%	10%	16%	29%

The standard deviations of DynaLRP_LRQ are smaller than those of the Real_FIFO over all the performance measures used, which clearly demonstrate the effectiveness of the proposed approach. Table 3.3 further confirms the superiority of the proposed approach as higher throughput is achieved with lower finished wafers' inventory and backlog.

Table 3.3 Summation of performance measures

	Inventory*	Backlog	Throughput	WIP
Real_FIFO	25088.8	913.9	2139.3	23883.73
DynaLRP_LRQ	21313.5	313.3	2193	24152.9
Improvement	15%	66%	3%	-1%

3.4.4 Concluding Remarks

Due to the complexity and uncertainty involved in wafer fabrication, the integration of operational planning and shop floor control has not been well addressed in the

literature. Lot sizing bridges the gap between operational planning and shop floor control. Hence, it is key to the integration of operational planning and shop floor control. We propose an effective approach to address this issue. To address the lot sizing problem, the proposed approach employs a linear programming model, termed dynamic lot release planning (DynaLRP) model, to determine the release quantities (wafer moves) at all buffers/stages of the production system based on the current status of the shop floor. At the shop floor control level, a dispatching rule, called Largest-Remaining-Quota-First (LRQ) rule is used in order to meet the target wafer moves given by the DynaLRP model. Once the shop floor status has changed, the new shop floor status can be updated in the DynaLRP model to obtain a new set of target wafer moves, which, in turn, will drive the workflow in the desired direction. The effectiveness of this approach is tested using simulation by comparing it with the lot release plan combined with First-In-First-Out (FIFO) dispatching rule used in practice. Based on the data used, the results show that the proposed approach has the potential of reducing output variability by 65.8% in backlog and 15.0% in finished wafers' inventory.

3.5 A New Input Control Policy Based on Operator Workload

3.5.1 Introduction

In this section, we study the issue of input control encountered in the complex batch production system of wafer fabrication. A semiconductor manufacturing system consists of several processing stages. These include silicon substrate preparation, wafer fabrication, chip probing and singulation, and product packaging and testing. In the sequel, we focus our attention on wafer fabrication, which transforms silicon wafers into integrated circuits (called ICs or chips). Wafer fabrication is, essentially, a batch production system in which several identical wafers are transported from one work center to another. A wafer carrier can contain only a certain number of wafers, based on its maximum capacity. The number of wafers that are transported together constitutes an operational lot size. Depending on the given demand, appropriate operational lot sizes of different products need to be determined to maintain desired customer service level. Once the operational lot sizes of different customer orders have been determined, the

next issue pertains to the release of these lots into production. The determination of operational lot sizes of products and their release into a wafer fab constitute the input control as it regulates the flow of material into the fab. Note that the lot size determined in section 3.4 consists of a number of these operational lot sizes.

Wafer fabrication is the most capital intensive stage from among the processing stages of a semiconductor manufacturing system. It is commonly regarded as one of the most complex and dynamic manufacturing environments today. The difficulty involved in making effective input control decisions for a wafer fab is contributed by the complexity of its operation. This complexity arises because of several external and internal factors. Among others, the external factors include a rapid technological development in semiconductor industry, thereby, leading to new products and production processes as well as fluctuations and diversities in the markets of these products and customer demands. The internal factors pertain to the complex product routings that are reentrant in nature, unreliable processes and equipment, and a mixture of single and batch processing modes in which wafers are processed in different processing areas. Furthermore, with advancement in technology, multi-tasking tools are now available, which enable flexible routings, and hence, add to dynamics on the shop floor.

In the face of demand variation, an inappropriate lot size can lead to either too many lots in the system causing a large number of setups, or too few lots, thereby, leading to ineffective utilization of processing tools. In other words, if the lot size is invariable, the input rate of new lots will vary with dynamic customer demand. Such a variation in system input, consequently, leads to workload variation on the processors, which prevents them from operating at an ideal workload level. Through variable lot sizing, the input variation can be reduced, which makes it easier to achieve effective workload control. Moreover, products at different stages of their life cycle, may require different lot sizes. For example, it is a common practice to use small lot sizes for newly-developed products (during the ramp-up production stage) in order to detect problems early, and hence, improve on process yield at an early stage of production. In this study, we focus on the determination of lot sizes so as to reduce variation in input workload.

Lot release control plays an important role in regulating the arrivals of lots at various stages of the system. In the presence of reentrant flow and inherent dynamics of a wafer fabrication system, production lots at various processing steps compete for the same processing tools. If new lots are released inappropriately, they may cause congestion or create artificial bottleneck(s) in a processing area while starving processing tools in other areas. An appropriate loading of a production system, thus, helps in balancing the production line and achieving effective utilizations of resources which are extremely expensive.

The highly dynamic and complex nature of a wafer fab, especially due to the reentrant nature of process routings, makes it difficult to analyze such a system. As alluded to earlier in Section 3.4, linear programming models have been proposed for planning production of such as system over long planning horizon (e.g., see Leachman [41]). However, the dynamics of a system is best studied through the use of computer simulation, especially if the underlying system is complex in nature. In this study, we propose a simulation-based input control strategy which incorporates both operational lot sizing and lot release control in a wafer fab. We integrate operational lot sizing with lot release control because operational lot sizing has a direct impact on the workload of a wafer fab, and an appropriate lot release control can facilitate in manipulating this impact.

The joint impact of operational lot sizing and lot release decisions has been largely overlooked in most existing input control studies, which may largely be due to the fact that many wafer fabs use uniform lot size for the ease of planning and control. However, there is no one lot size that is always the best to use in a dynamic environment. The importance of using an appropriate lot size is further reinforced by increment in wafer size. A 12-inch wafer now holds up to 2.5 times as many chips as does an 8-inch wafer. As a result, it takes longer to process this wafer than it takes to process an 8-inch wafer, even though processing time per chip has been greatly reduced. Also, the size of a lot impacts waiting time of wafers. Finally, a low but a greater variety of customer demand promotes the use of flexible lot sizes, and this trend is becoming more of a norm now-a-days.

3.5.2 Literature Review

Input control can be divided into the following three levels, namely, order entry, order release and priority dispatching (see Breithaupt, et al. [7]). A key issue pertaining to order entry is the size of a lot to use, while order release involves determination of when to release a lot. We address these two aspects of input control in this section. Lot sizing has been an active field of interest for both researchers and practitioners since the emergence of management science (e.g. see Harris [31]). The main objective of lot sizing is to achieve a good trade-off between minimizing number of setups and inventory in meeting customer demand. To put our discussion in perspective, lot sizing also arises at the production planning level, where families of products are grouped in order to minimize production and inventory costs. However, here, our focus is at the shop floor level, where work is released for processing at the machines (processing centers). Winz, et al. [80] study this issue in the backend of a wafer fabrication facility, where individual chips and other products are packaged. In their study, operating curves for different operational lot sizes are obtained by using computer simulation of the facility. They contend that throughput can be enhanced by studying the performance of a facility through such operating curves (in their case by 14%) while still meeting the target cycle time. In general, there are two broad categories of methods used for releasing the lots for processing at the shop floor. These are push and pull methods (Spearman and Zazanis [67]). Various studies have been conducted to study their performance. Lee, et al. [43] conduct an experimental study in order to compare different scheduling and input control rules for bottleneck machines. Their study shows that the pull-based scheduling and input control rules outperform the push-based rules for most of the performance measures. Wein [79] has conducted a comprehensive simulation study using a variety of lot release and sequencing rules. Four lot release rules are used in his study, including Poisson arrivals, uniform release, closed loop and workload regulating. His study reveals that, although both the sequencing and lot release rules have significant impacts on the wafer fab performance, greater benefits can be realized by improving the way the lots are released. Lawton, et al. [40] use workload

regulating (WR), which attempts to control the workload at the bottleneck machine(s). In case there are multiple bottlenecks, WR can be implemented in two ways: 1) release a new lot if the current total workload at these bottlenecks combined is below a target level; 2) release a new lot if the current workload of each individual bottleneck is below a target level. Once a lot is processed at the bottleneck machine, the current workload is updated. Their study shows that WR has the potential of reducing the cycle time by 40-60%.

Spearman, et al. [66] introduce another lot release method called CONWIP, which attempts to keep the WIP in the system at a constant level. Under the CONWIP policy, a new lot is released only if the current WIP level drops below a target level. Compared to WR, CONWIP is simpler, and hence, easier to implement. It has also been proved to be robust (Hopp and Spearman [32]). Framinan, et al. [20] have presented a review of studies in this field. The difficulty of implementing CONWIP in wafer fabrication lies in its inflexibility to adapt to dynamic situations such as machine breakdowns and demand fluctuations. For a review of input control methods, see Uzsoy, et al. [76] and Fowler, et al. [19].

In this section, instead of studying the simultaneous impact of operational lot sizing and lot release decisions in a hypothetical system, we study their performances in the real-life environment of a wafer fab. Our objective is not only to come up with strategies that are ready for implementation in a real-life environment, but also to demonstrate the extent of their usefulness in practice. To that end, we developed a simulation model of a wafer fabrication facility using AutoSched AP (also introduced in Section 3.4.3), validated the model using the relevant data from the fab, and then, used it to implement our proposed strategies. Specifically, there are two aspects that we studied: 1) the effect of different operational lot sizes on cycle time under various start rates, and 2) the joint impact of various operational lot sizes and lot release decisions on cycle time. We develop a new strategy to release the lots into the system. This strategy is an adaptation of workload regulating and CONWIP, and we show its superiority over a CONWP-based policy that is used in practice.

3.5.3 Simulation-based Approach for Input Control

In order to conduct our analysis on input control strategies, a comprehensive simulation model of a fabrication facility was developed using AutoSched AP simulation software. Figure 3.12 illustrates the functional areas and sizes of entities involved. This simulation model was validated using a real-life dataset that covers one year of its operation.

Please refer to Appendix C for more detailed information about the simulation model of the wafer fab considered in this study.

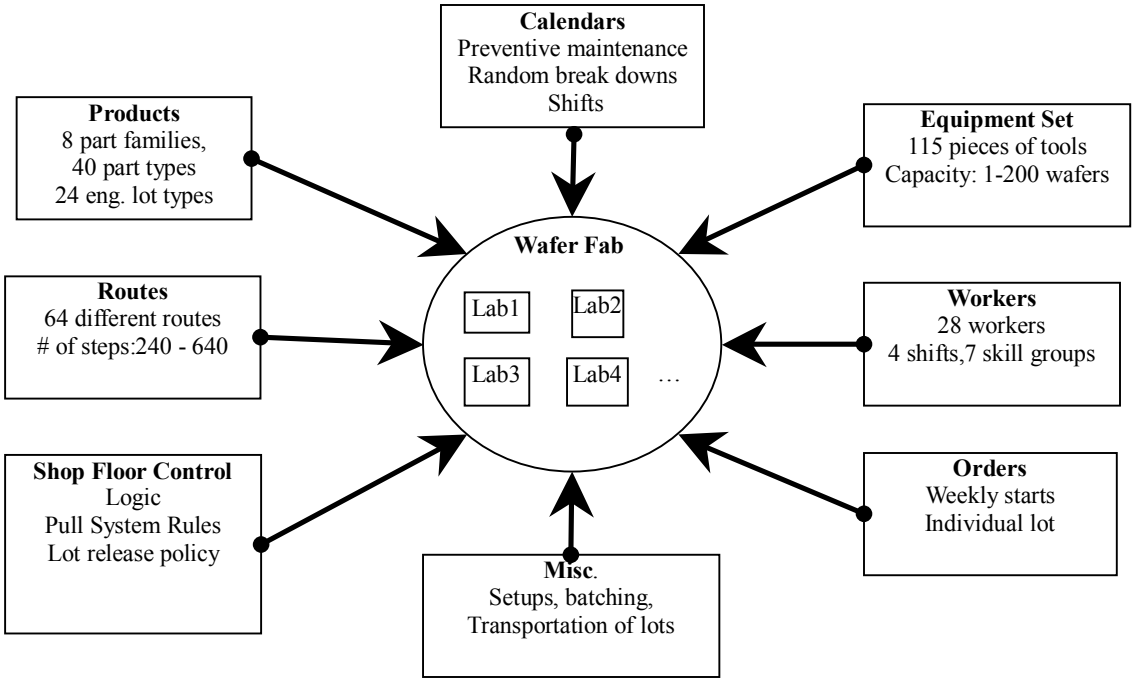


Figure 3.12 Key components for the simulation model of the wafer fab in consideration

3.5.3.1 Impact of Operational Lot Sizes on Cycle Time

When processing a lot in a processing area, there are two types of operations. One type of operation involves only lot-attached work, i.e., the work required depends only

on a lot and not on its size, and the other type involves piece-attached work, i.e., the work required depends on every piece that is processed. The total amount of work required is called the workload of a machine. A typical example in this regard that of a stepper in the photolithography processing area, where wafers are processed piece-by-piece. The stepper, first, needs to be setup before processing a lot. Thus, the overall workload of a processing cycle is composed of total lot-attached time and total piece-attached time. The former is directly determined by the number of lots while the latter by the demand. Of course, the product mix can be another factor that affects workload since different products may require different processing times. In our study, we assume the product mix to be fixed since the changes in the product mix are relatively slow and less frequent.

It is obvious from above that the lot-attached part of workload can be affected by controlling the number of lots via operational lot sizing, for a given wafer start rate (WSR). We studied this phenomenon by trying different operational lot sizes under various start rates, and determining average cycle times over all the products. The results are shown in Figure 3.13. In this experiment, we examined 9 operational lot sizes ranging from 4 to 20 and plotted an operating curve for each of these operational lot sizes. The weekly WSR was varied from 50 to 325 pieces with a step size of 25 pieces. Since we prefer a smaller cycle time and a higher throughput rate, a lower curve for a given throughput rate, or a curve to the right for a given cycle time gives a better performance. Note that, under a stable system, the throughput rate will be the same as the start rate (not considering the yield factor).

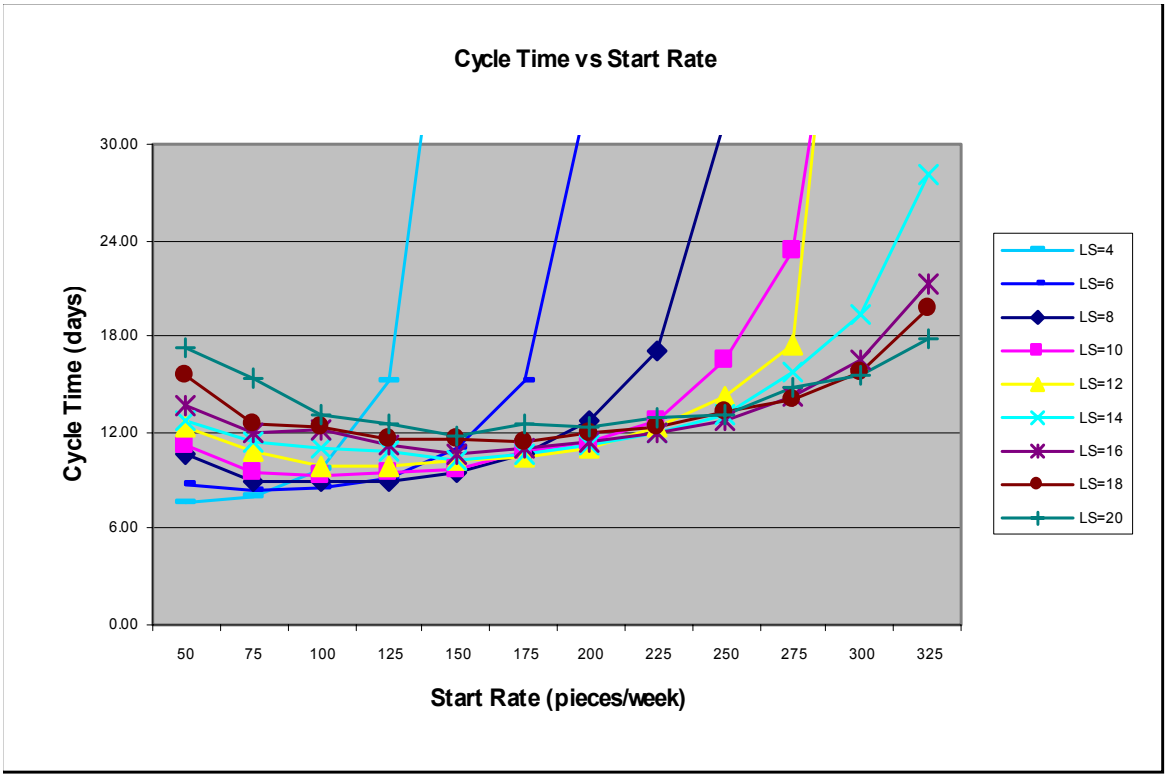


Figure 3.13 Operating curve: cycle time vs wafer start rate for various lot sizes

From Figure 3.13, it can be observed that the best operational lot size varies with the WSR. When the WSR is low, a smaller operational lot size tends to perform better. As the WSR increases, larger operational lot sizes start to dominate. The increased cycle times at 50-pieces-per-week start rate are due to the batching rule at certain machines, which requires a minimum batch of wafers to start. The low start rate delays the formation of batches, and hence, leads to an increment of cycle time. A production system becomes unstable for the WSRs beyond a certain point if a relative smaller operational lot size is used. A smaller operational lot size leads to a high lot start rate (LSR), where the LSR is the WSR divided by operational lot size. The best operational lot sizes and the associated WSR and LSR are shown in Table 3.4.

Table 3.4 Recommended operational lot sizes for various wafer start rates

Wafer Start Rate (WSR) (pieces/week)	Recommended Operational Lot Size	Lot Start Rate (LSR) (lots /week)
⩽50	4	10-13
50 - 100	8	7-13
100 – 150	10	10-15
150 – 200	14	11-15
200 – 250	16	13-16
250 - 300	20	13-15
> 300	20	>15

In Table 3.4, the recommended operational lot size is the one that achieves the lowest cycle time for a given start rate. It is obvious that for the wafer fab that we studied, the most appropriate LSR is within [10, 15]. With an increment in the WSR, the recommended range of the LSRs barely changes except when the WSR goes beyond 300 (capacity limit) and the operational lot size reaches its upper bound. Based on the above observations, we can make the following two conjectures:

Conjecture 1: There exists a range of LSRs that gives optimal cycle times for a given fab.

Conjecture 2: The range of LSRs specified by Conjecture 1 does not change drastically with a change in the WSR as long as the WSR keeps the system stable.

Even though we have not attempted to prove these conjectures analytically, which may be tedious given the myriads of interacting factors involved, yet their justification may simply follow from the following two factors: 1) a compromise between the lot-attached and piece-attached processing times, and 2) maintaining the stability of the system. Nevertheless, the impact of these conjectures is tremendous from production viewpoint because, now, a planner can appropriately adjust percentages of wafers for processing in different operational lot sizes such that the resulting LSR is within the specified range without significantly impacting cycle times. Once the operating curves of a system have been generated, they can also be used to predict the cycle times if the actual LSR shifts away from the desired range.

3.5.3.2 A New Lot Release Strategy

As mentioned in section 3.4.2, various lot release methods have been studied in the literature, and they can be divided into the categories of “pull” or “push” methods or a hybrid of the two. Simple "push" methods, which can be found in many MRP systems, do not work well in the highly dynamic and complex environment of wafer fabrication. By not paying attention to the status of the current workflow, a “push” method cannot mitigate the variability encountered inside a production system. For example, in the case of a machine breakdown, a "push-based" method will cause excessive WIP (WIP bubble) to accumulate in front of that machine. Such a WIP bubble causes an imbalance of the production line and may lead to the so called "system nervousness", which may persist long after the breakdown machine returns to work. On the other hand, the "pull" methods, such as CONWIP and WR (workload regulating), have been embraced by practitioners in this field for their simplicity and robustness. However, both CONWIP and WR methods have their own shortcomings. According to the workload regulating policy, the workload at a bottleneck machine is monitored and kept at a constant level. This workload is updated upon the completion of lots on that machine, which results in a better pull mechanism. However, a WR method becomes difficult to implement when the bottleneck shifts from one machine to the other, which is not an uncommon occurrence in wafer fabrication facilities. The effectiveness of this approach is also impaired by the presence of a number of potential bottlenecks in a wafer fab. The other "pull" method, CONWIP, attempts to maintain a constant WIP level in the system. A constant WIP level helps in keeping appropriate utilizations of resources and leads to relatively less output variability. It maintains a constant WIP level by allowing a new lot to enter the system only when the WIP level drops below a target level. A major difficulty in implementing the CONWIP method is the setting of a constant WIP level. When facing a highly variable demand volume as in wafer fabrication, this task becomes even more challenging. Another difficult issue pertains to the CONWIP-based method's treatment of the production system as a "black box". Such a treatment can be effective for a simple and less dynamic manufacturing system, but is not effective for a complex wafer fabrication system. In the reentrant wafer fabrication systems, WIP bubbles are often created due to irregular workflows and unreliable machines and

processes. The "black box" treatment of a production line prevents a CONWIP method from detecting such WIP bubbles, and hence, makes it less effective in smoothing out the workflow and maintaining a line balance.

In summary, both WR and CONWIP methods attempt to keep the workload at a constant level while controlling lot release. They differ in that, the WR-based method measures the workload by machine hours on bottleneck(s) while the CONWIP-based method measures the workload by the number of lots in the entire system. A WR-based method leads to a better "pull" mechanism than a CONWIP-based method due to its higher frequency of workload updates. However, it suffers from a restricted view of the system since it only considers bottleneck(s). A CONWIP-based method, on the other hand, considers the entire system, as a "black box", which results in less responsiveness and a loose "pull" mechanism. A common issue for both of these "pull" type lot release methods is how to determine the target workload level, regardless of whether it is measured by hours on a bottleneck machine or by the number of WIP-lots in the system.

Next, we propose a new lot release method called CONLOAD, that not only enables an effective "pull" mechanism but also overcomes the shifting bottleneck problem associated with the WR method by broadening the definition of a bottleneck. We also address the issue of setting target workload levels for the proposed CONLOAD method.

Development of a New Lot Release Method

In most bottleneck-oriented lot release control methods, bottlenecks are assumed to be machines, which, consequently, results in their difficulty of implementation since bottlenecks shift. A 200mm wafer fab highly relies on operators for setting up processing tools, transportation of lots, loading/unloading operations and data logging tasks. However, the number of operators has been reduced greatly due to improved processing tools and cost cutting campaigns. As a result, operators have, typically, become a major bottleneck in such a wafer fab. This fact was clearly exhibited by a high utilization of operators and a large proportion of a cycle time attributed to "Wait For Operator" status by completed lots in computer simulation of a 200mm fab (see Appendix C). Therefore, we define the bottleneck as the operators and try to keep the

operator workload at a constant level in the proposed CONLOAD method. The advantage of treating operators as the bottleneck is two fold: 1) we can maintain a better pull mechanism by updating operator workload frequently, and 2) operators are required at each processing tool, and hence, reflect a more global view of the workflow, which overcomes the drawback of a WR-based method. In addition, the number of major operator classes is relatively small, which makes this approach less complicated. For example, if a machine becomes bottleneck due to a breakdown event, the traditional WR-based method may not detect the shifting of bottleneck. However, this breakdown event will be immediately reflected in the change of an operator workload.

In order to study the workload handled by operators, a measure to quantify workload is required. An operators' time is spent on loading/unloading operations, setups, inspections and some manual operations. This time can be divided into the same two categories as mentioned in section 3.5.3.1: lot-attached time (*la*) and piece-attached time (*pa*). The lot-attached time refers to the time that is fixed for each lot and is independent of its lot size. For example, the time required to setup a machine is the lot-attached time. The piece-attached time is the time that an operator spends on every piece. An example of a piece-attached time is the inspection time spent on a wafer, if each wafer in the lot is to be inspected. Consequently, the total operator time required for processing a lot in a processing area is as follows:

$$LT = la + pa * ls$$

By summing *LT* over all operations in the route of a lot, we can obtain the total operator time required for completing a lot. That is,

$$LT = \sum_{s=1}^{S} la_s + ls * \sum_{s=1}^{S} pa_s = LA + PA * ls \tag{3.7}$$

In the presence of a yield factor (*y*), we need to regard *ls* as a random variable. In that case, the expected value of *LT* is calculated as follows:

$$E[LT] = \sum_{s=1}^{S} E[la_s] + \sum_{s=1}^{S} E[pa_s * ls_s] = LA + \sum_{s=1}^{S} (pa_s * \prod_{k=1}^{s} y_k) * ls \tag{3.8}$$
$$= LA + PA^y * ls$$

Note that this formula is identical to that in (3.7) except that, now, the piece-attached parameter is calculated differently.

In our new lot release method, called CONLOAD, we use the total workload on the operators for processing the lots in the system, to determine when to release a new lot into the system. In particular, once a new lot arrives, the current total workload of the operators due to the lots in the system is examined. If it is less than a target workload level, then the lot is released into the system and the current workload is increased by the total workload of this new lot (given by (3.7) or (3.8) as the case may be). Otherwise, the lot is held in the start queue and the above examination is triggered each time the current operator workload is updated. The current operator workload can be updated upon the completion of every processing step of a lot (termed step update), or at the completion of the last step of its processing route (termed exit update). Consequently, several versions of CONLOAD can be proposed.

- CONLOAD with exit update (EU-CONLOAD): this is the simplest version of CONLOAD control where the total operator workload is updated once a lot exits the system.

- CONLOAD with step update (SU-CONLOAD): in this method, the total operator workload is updated once a lot completes an operation in a processing area.

- EU-CONLOAD with front end control (EUF-CONLOAD): A lot is enterable if the target workload level is not exceeded and the number of lots in the front end of a fab does not exceed a predetermined FRONTMAX. The total workload is updated upon the exit of a lot. This method considers FRONT END threshold in addition to the operator workload.

- SU-CONLOAD with Front End control (SUF-CONLOAD): it is similar to EUF-CONLOAD except that the update method is step update.

The step update methods require the lot-attached and piece-attached operator time for each step. The calculations of this data is not an easy task especially when various part types and large number of operations are present. Batching is another complicating

factor for these calculations. Therefore, the operator workload used by these CONLOAD methods is only an approximation of the real operator workload. The effectiveness of these methods is, therefore, affected by the accuracy of the procedures used to estimate the operator workload. For our experimentation, the data used in the simulation model was validated as mentioned in section 3.4.3. Consequently, these workload estimates are fairly accurate.

We compared the performance of the above CONLOAD methods with a CONWIP-based method currently used in the wafer fab under study. This method makes use of two threshold values, namely, FRONTMAX and LINEMAX for lot release. A lot is allowed to enter the production system if the total number of lots in the front end of the production line (operation index $\leqslant$3200) and the total number of lots in the entire production line are less than the above two values, respectively. We denote this lot release method by $CONWIP_2$. The subscript "2" means that it utilizes two threshold values.

Evaluation of CONLOAD Methods

The above CONLOAD lot release methods were coded in C++ and integrated into the simulation model of the fab developed using AutoSched AP. Two realistic input data sets were used to compare the performances of these lot release methods. One is a long-term data set that is composed of one year's real start rates. The other is a mid-term input data set that uses the exact lot-related information (including the initial WIP condition) of nearly three months. The performance measures that were used include average cycle time, standard deviation of cycle time and total number of pieces completed (total throughput).

The evaluation of various lot release methods was divided into two stages. In stage one, we compared different versions of CONLOAD method using the long-term data set. The best CONLOAD version of stage one was, then, used in stage two, where more experiments were conducted to compare this CONLOAD method with $CONWIP_2$. Both long term and mid-term data sets were used in stage two.

Stage One – Selection of a CONLOAD Method

Since the lot size information was not available for the long-term input data set, Table 3.4 was used to determine lot sizes for a given WSR. However, in reality, the WSR does not remain stationary over time. Hence, to avoid sudden changes in lot sizes, we took an average of the WSRs over four weeks and used that to determine the lot size to use. This helps in smoothing out the lot size over time. The original start rates and smoothed start rates are shown in Figure 3.14.

Having determined the lot sizes, various CONLOAD methods were tested using this input data set. For each CONLOAD method, various target workload levels were used and values of the performance measures were collected over the last forty four weeks. The front-end control parameter, FRONTMAX, is set at the same value as was actually used in the real facility, being 35 lots, for all the experiments. The results are shown in Figure 3.15, 3.16 and 3.17.

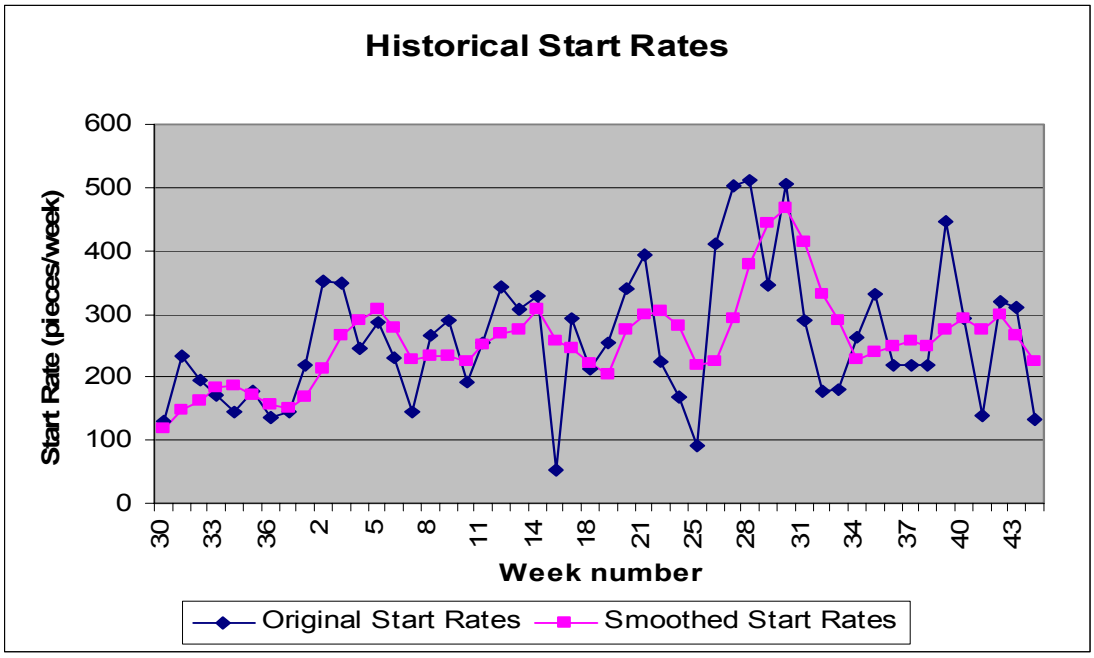


Figure 3.14 Wafer start rates for long-term data

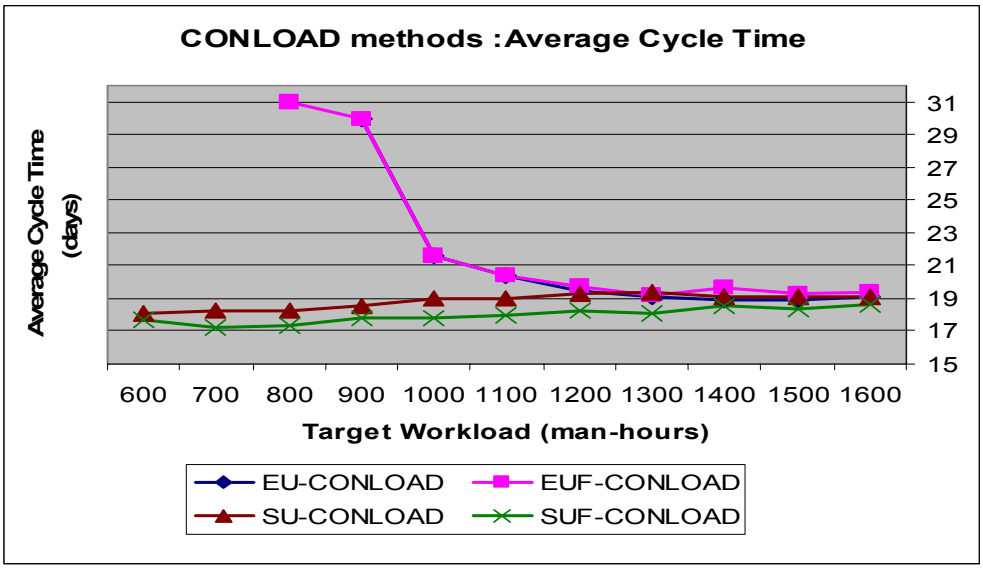


Figure 3.15 Comparison of CONLOAD methods: average cycle times

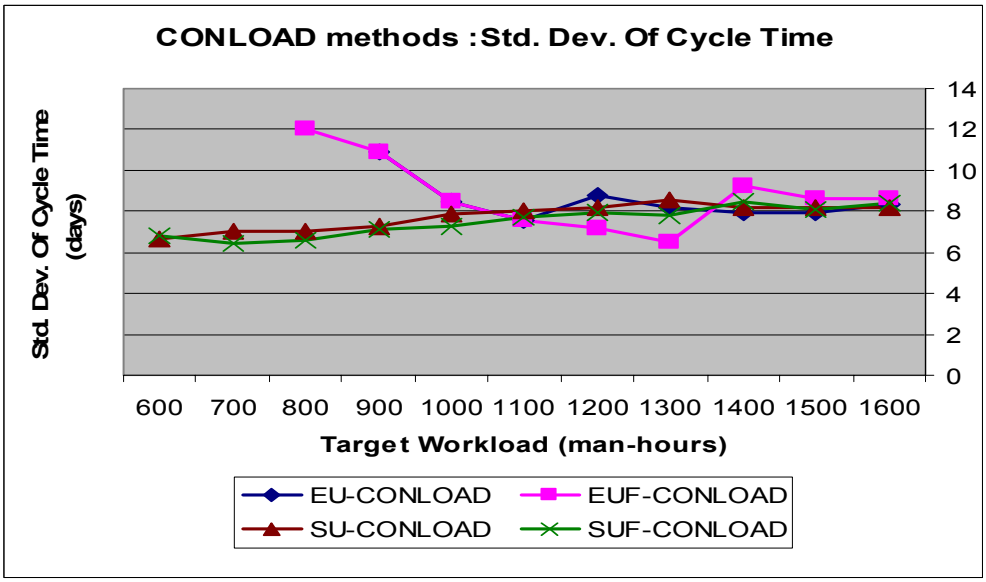


Figure 3.16 Comparison of CONLOAD methods: standard deviation of cycle time

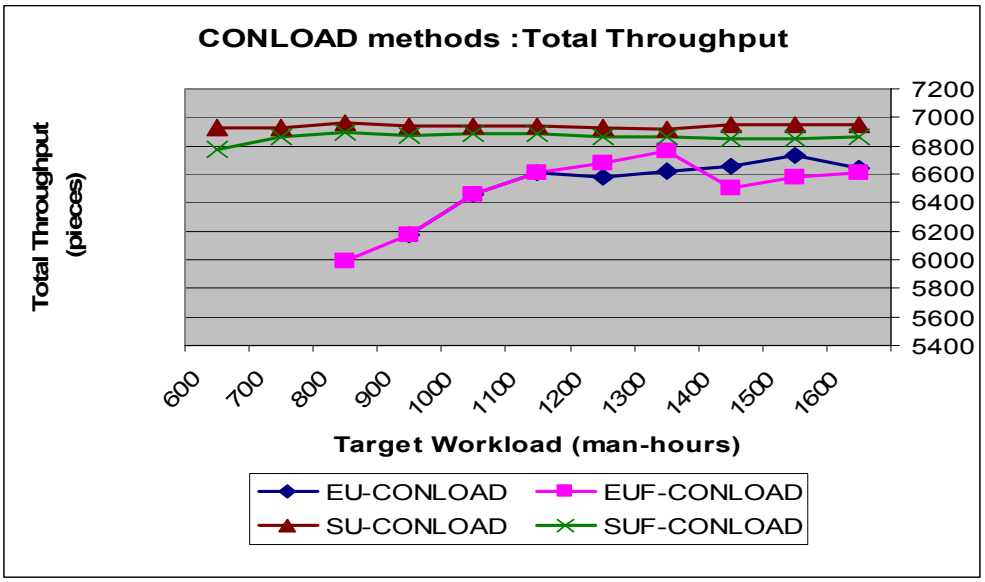


Figure 3.17 Comparison of CONLOAD methods: total throughput

From the above figures, it can be observed that the step update (SU) CONLOAD methods outperform the exit update (EU) methods with respect to all performance measures. The reason for the poor performance of the EU methods lies in the treatment of the whole system as a black box in which a lot at the beginning of its route is the same as a lot at the end of its route. This may not be problematic if the lots are evenly distributed along their routes. However, this is not true in most situations. The workflow frequently goes through equipment breakdown, and processing at batching stations, among others. EU methods can not detect such "pulses", and hence, are not effective in smoothing the workflow, which is one of the important desired features of a lot release method.

Concerning the SU methods, SUF-CONLOAD appears to be the best although it is second to SU-CONLOAD with regard to total throughput. If we consider the best performances of these methods, SU-CONLOAD is worse than SUF-CONLOAD by 4 percent with regard to average and standard deviation of cycle time, while it is only 1 percent better in throughput rate. The advantage of the front end control is highlighted by the difference of these two SU methods. The SU methods monitor a change in operator workload more closely than that done by EU methods. Therefore, the resulting

"pull" mechanism is more effective. This is one of the reasons for their superiority.

SUF-CONLOAD was next compared with $CONWIP_2$.

Stage Two: Comparison of SUF-CONLOAD with $CONWIP_2$

In order to compare SUF-CONLOAD with $CONWIP_2$, the following experiments were conducted. For the long-term data set, uniform lot sizes (20, 16 and 10) were used. Lot size 8 is not used as it does not result in a stable state. Note that the instability appears if only lots of size 8 are used. In practice, lot size 8 is used in conjunction with other lot sizes and together they result in a stable system. Even though lots of different sizes are used in practice, yet we assume a uniform lot size because of the ease in controlling operator workload levels. For the mid-term data set, a combination of lot sizes was used. Since this data set contains the real-life input, the performance of SUF-CONLOAD was compared with that of the existing system. We also recorded the performances of a push-based method in which planned weekly starts were released uniformly over each week. This method was used for reference purpose, and is designated as UNIFORM henceforth. The experimental results are presented in Table 3.5.

The results shown in Table 3.5 clearly indicate that the SUF-CONLOAD method dominates the $CONWIP_2$ method as well as the PUSH method with respect to all performance measures. The difference in performance doesn't seem to be very large though. This may be due to the fab being not heavily loaded in most of the periods as was observed during simulation runs, which reduces the impact of a lot release method. However, even a small reduction in the cycle time can lead to a significant improvement in fab performance. Referring to Table 3.5, the minimum improvement in average cycle time is about 0.7 days, and the maximum improvement is 2.23 days. If the fab's output is 40 lots per month, this leads to a reduction in cycle time of 28 to 89 days for all of these 40 lots.

Table 3.5 Comparison results: SUF-CONLOAD vs. CONWIP$_2$ vs. UNIFORM

Data Set	Lot Size	Lot Release Method	Target Workload (man-hrs)	Avg. Cycle Time (days)	Cycle Time Std. Dev. (days)	Throughput (pieces)
Long Term Data Set	Optimized*	SUF-CONLOAD	700	17.27	6.41	6860
		CONWIP$_2$	N/A	18.72	8.26	6866
		UNIFORM	N/A	18.60	8.39	6858
	20	SUF-CONLOAD	600	17.35	6.02	6970
		CONWIP$_2$	N/A	19.58	8.57	6948
		UNIFORM	N/A	18.62	7.3	6970
	16	SUF-CONLOAD	700	20.88	8.53	6729
		CONWIP$_2$	N/A	21.60	9.24	6730
		UNIFORM	N/A	21.72	9.68	6704
	10	SUF-CONLOAD	1100	28.59	12.15	6212
		CONWIP$_2$	N/A	29.56	14.14	6096
		UNIFORM	N/A	29.17	14.99	6069
Mid Term Data Set	Mixed	SUF-CONLOAD	800	18.52	4.10	1725
		CONWIP2	N/A	19.51	4.19	1669
		UNIFORM	N/A	19.22	4.68	1694

Note: * the lot sizes are determined using Table 3.4.

As alluded to earlier, the reasons for the superiority of SUF-CONLOAD are accurate estimation of bottleneck workload and the use of a more effective "pull" mechanism. The drawback of SUF-CONLOAD method is that it requires more operating overhead. First, the operator workload data for each operation must be estimated accurately. This amounts to estimating an operator's average operating time (including lot-attached time and piece-attached time) at each machine. In order to account for batching factor, the operator's time can be divided by the average batch size. Second, and may be the most important factor for the success of the SUF-CONLOAD method, is to appropriately set the target workload levels. If the target workload level is set too high, the effectiveness of the lot release will be hampered; if it is set too low, a bottleneck may be underutilized and jobs may encounter unnecessary delays in the start queue. Therefore, determination of effective target workload levels is essential for a successful implementation of the SUF-CONLOAD method. We discuss this next.

Determination of Reference Table for Target Workload Levels

Referring to (3.8), it can be seen that operator workload is affected by both the WSR and LSR. Usually, the WSR is predetermined by customer demand. The LSR is affected by the lot size used. In our previous study of lot sizing, it has been shown that for the same WSR, LSR can be very different (for different lot sizes) and the associated performance can be significantly different as well. A combination of the WSR and LSR defines a workload scenario. There is an optimal target workload level associated with this workload scenario. The reason that a LSR is selected in the design of a reference table is that an appropriate LSR can handle a real-life situation when different lot sizes are used. We proceeded as follows to obtain an optimal target workload level for each workload scenario, that is, a combination of WSR and LSR.

We use the same performance measures as mentioned before. The WSR is chosen from 150 to 300 pieces per week. The LSR ranges from 15 to 35 lots per week. For a given WSR, we use three standard lot sizes, 4, 8 and 20, to achieve a desired LSR. This is done by adjusting the percentage of pieces belongs to each of these lot sizes. For example, if given a WSR of 300, 10% of the 300 pieces will be released with a lot size of 4, 20% with a lot size of 8 and the rest with a lot size of 20, then, the resulted LSR is calculated as follows:

$$LSR = \left\lceil \frac{0.1*300}{4} \right\rceil + \left\lceil \frac{0.2*300}{8} \right\rceil + \left\lceil \frac{0.7*300}{20} \right\rceil = 27,$$

where $\lceil x \rceil$ is the smallest integer larger than x. The released lots are broken into seven part types that represent the seven part families based on a real-life product mix. The simulation horizon is thirty weeks and the performance measures are collected over the last twenty weeks. For a given WSR and LSR, a range of target workload levels are tested and the one with the best performance is selected. The experimental results are shown in Table 3.6.

The reason that we list performance measures associated with each target workload in Table 3.6 is to demonstrate how LSR affects the performance. The best LSR is 15 lots per week among the five LSRs tested for every wafer start rate, which agrees with our

findings in Section 3.4.3.1. The maximum LSR is near 35 lots per week when WSR is less than 300 pieces per week. For a WSR larger or equal to 300 pieces per week, the LSR should be kept below 30 lots per week to avoid overloading the system.

Table 3.6 Reference table for target operator workload and its associated performance

WSR (pieces/week)	Performances and Target Workload	LSR (lots/week)				
		15	20	25	30	35
300	Target Workload (man-hrs)	500-600	600-700	1000	1300	N/A
	Avg. Cycle Time (days)	14.42	17.98	21.93	32.67	N/A
	Cycle Time Std. Dev. (days)	1.26	2.04	1.94	4.51	N/A
	Throughput (pieces)	3607	3607	3574	3346	N/A
250	Target Workload (man-hrs)	400-500	500-600	600-700	900	1500
	Avg. Cycle Time (days)	12.21	14.60	17.29	24.83	36.70
	Cycle Time Std. Dev. (days)	1.60	1.99	2.16	2.71	4.70
	Throughput (pieces)	3073	3072	3067	2931	2661
200	Target Workload (man-hrs)	400-500	400-500	500-600	700-800	1200
	Avg. Cycle Time (days)	10.99	11.73	14.36	19.04	31.89
	Cycle Time Std. Dev. (days)	1.97	1.90	1.92	1.81	4.19
	Throughput (pieces)	2445	2432	2430	2377	2193
150	Target Workload (man-hrs)	400-500	400-500	500-600	500-600	700
	Avg. Cycle Time (days)	9.50	9.82	11.20	14.51	25.89
	Cycle Time Std. Dev. (days)	1.80	1.76	1.57	1.34	4.50
	Throughput (pieces)	1836	1827	1827	1823	1692
Note: the highlighted workload scenarios indicate overloading situation.						

Since there are trade-offs between cycle time and throughput, the approximate ranges for the best target workload levels are given in Table 3.6. The differences in fab performance measures among the target workload levels that fall in this range are not significant, and the listed performance measures are representative for the whole range. Concerning the distribution of optimal target workload levels, it appears that the best target workload level tends to increase with both WSR and LSR, which is expected in

accordance with (3.8).

Table 3.6 serves as a good reference for setting target workload for the proposed CONLOAD method. In reality, the WSR may vary from week to week, and this reference table can be helpful in adjusting the target workload level.

3.5.4 Concluding Remarks

In this chapter, an input control approach that incorporates decisions pertaining to lot sizes and lot release for a complex batch production system, like a wafer fab, has been developed. This input control approach involves two phases. The first phase is the lot sizing phase in which a desirable lot start rate (LSR) range is determined for various wafer start rates (WSR). This desirable LSR range is then used to determine the lot sizes to use for satisfying multiple customer orders such that the overall lot start rate (consisting of a mixture of various lot sizes) is as close to this LSR range as possible. The purpose of the lot sizing phase is to reduce the variation in LSR so that the workload of the wafer fab can be maintained around the optimal target workload range. It is this optimal target workload range that maintains the effectiveness of CONWIP-like lot release control methods.

The second phase involves lot release in which the workload of operators is closely monitored and kept below the target workload level. The employment of operator workload for lot release control is based on the fact that the operation of a wafer fab highly relies on operators for processing and material handling, which makes the operators a major driving force for the workflow. By releasing lots based on the consumption of operator workload, our proposed CONLOAD method integrates, to a certain degree, human involvement into lot release control, which is an important feature for a highly dynamic environment because the operators are constantly making dispatching decisions according to the prevailing shop floor status. Such decisions inevitably affect the workflow and the resulting workload consumption which, in turn, affects lot release as captured by the CONLOAD method. In the case of machine breakdowns, operators will automatically slow down in the affected area, or even inform operators in the unaffected areas to slow down as well to avoid building of excessive

WIP at the affected processing area. This is how CONLOAD manages to overcome difficulties resulting from shifting bottlenecks as faced by bottleneck-oriented lot release methods, such as the workload regulating and the starvation avoidance methods. We have examined two major versions of this lot release control strategy. The step update version, including SUF-CONLOAD and SU-CONLOAD, proves to be more effective than the exit update version, including EX-CONLOAD, EXF-CONLOAD and $CONWIP_2$. This demonstrates the superiority of a tighter pull mechanism (as a result of step update) over the loose pull mechanism (as a result of "black box" treatment of a production system) as in CONWIP.

Besides the two phases of the proposed input control approach, we have also addressed the issue of setting target workload level. We define a workload scenario based on both the WSR and LSR, which is in accordance with the calculation of total workload as indicated by (3.8). The optimal range of target workload level is obtained for each workload scenario defined. As a result, a reference table of the optimal target workload levels is obtained, which prescribes the range of optimal target workload level, the expected cycle time and throughput rate, and the highest LSR before overloading a production system. Such a reference table greatly facilitates the setting of target workload levels. It can also be useful for lot sizing purposes since it provides expected performances for various LSRs, which is an important piece of information for a planner in determining the lot sizes and the resulting LSR. We have demonstrated this methodology by simulating the operation of a real-life wafer fab.

We have also demonstrated the effectiveness of the proposed two-phase CONLOAD approach by comparing it with the CONWIP approach. Our methodology is more comprehensive in the sense that it not only addresses the lot release, but also incorporates lot sizing in order to realize the maximum benefits. The effectiveness of the operator-based lot release appears to be quite interesting. First of all, the employment of operator workload control achieves a proper tradeoff between the local perspective (the bottleneck-oriented approach) and the global perspective (the "black box" approach) of the production system, due to the involvement of operators at all processing steps. Secondly, it helps in integrating the dispatching decisions made by operators into input control.

For future research, we propose an investigation on the use of the proposed CONLOAD approach in a highly automated 300 mm wafer fabs where human involvement is minimized. However, the role of the operators is now substituted by an automated material handling system (AMHS). The AMHS plays a central role in material handling and manufacturing execution system (MES) for setups and dispatching. Consequently, the proposed CONLOAD method can be employed with minor modifications. For example, lot release method can now be based on the workload of the AMHS which is the carrier of workflow, and hence, enables the same pull mechanism.

Chapter 4 An Integrated Production and Shipping Planning Problem

4.1 Introduction

In this chapter, we consider the problem of planning the production and shipping operations of a manufacturing enterprise in an integrated fashion to minimize the total costs of shipping, inventory, and job earliness and tardiness. This problem is motivated by a real-world scenario faced by a make-to-order manufacturing company that utilizes its own specialty trucks to ship orders to the customers that are distributed over a geographical area. Due to a large distance covered by these trucks and a relatively small number (in relation to the vehicle capacity) of customer orders, an appropriate batching of customer orders is important in order to minimize the shipping cost incurred, which is one of the major components of the manufacturer's operating cost. The amount that can be shipped also depends on the available production capacity. Therefore, the benefits resulting from an effective planning of the shipping operation can not be fully realized without appropriately coordinating it with production.

The integration of production and shipping operation is an important issue because of its impact on an organization's overall effectiveness. For example, Martin, et al. [47] report annual savings of over $2.0 million due to the implementation of a system called FLAGPOL at the Flat Glass Products Group of the Libbey-Owens-Ford Glass company. FLAGPOL integrates the production, inventory and distribution decisions of a multi-plant system through a single linear programming model that has helped the company in making effective tactical and operational planning decisions and in achieving a globally optimized system rather than a system that optimizes the operation of each plant in isolation. In an another study reported by Ramachandran and Pekny [53], a coordination between production and distribution planning has been shown to achieve from 3% to 20% improvement in setup, inventory and distribution costs over 132 different test cases that they used. Gimenez and Ventura [21] examine the integration of logistics with production and marketing, respectively. They also study its relationship with the

(external) integration of the company with other companies in a supply chain environment. They report that the integration of logistics with production is effective in reducing cost, stock-outs and production lead times.

Sarmiento and Nagi [56] provide a review of the work reported on integrated analysis of production and distribution systems. Their integrated analysis involves the production, inventory and distribution functions within a company. In general, the integration-based studies fall into two major categories, namely, inventory and distribution planning or production and distribution planning. Most of the studies reported in the literature pertain to the integrated inventory and distribution planning problems while very few of these studies have addressed the integrated production and distribution planning problem. Due to an increasing emphasis on reducing inventory, the authors point out the need of integrating production and distribution. The work of Ramachandran and Pekny [53] is regarded as one of the first studies in this area. A review of the solution methodologies that have been proposed for the production and distribution problem is presented by Bilgen and Ozkarahan [5]. Their review addresses decisions at the strategic, tactical and operational levels. The strategic level decisions pertain to facility location, plant capacity and processing technologies. The tactical level decisions address material flow management issues such as production levels, lot sizes and inventory levels, while at the operational level, the decisions involve the scheduling of work for on-time delivery of final products to customers. Studies that cover strategic and tactical levels are reported in Elhedhli and Goffin [18], Dogan and Goetschalckx [15], Goetschalckx, et al. [25], and Schmidt and Wilhelm [57], who also discuss the scheduling issues for multi-echelon assembly networks.

At the operational level, the integration of production and distribution has also received attention, but mostly from the scheduling perspective. Hall and Potts [30] study a variety of scheduling, batching and delivery problems with the objective of minimizing a combined objective of the delivery cost and one of the classical scheduling objectives, in a supply chain environment composed of one supplier, multiple manufacturers and multiple customers. They provide efficient dynamic programming-based algorithms for some of these problems, and the complexity results for the others. Besides the supplier, each manufacturer is modeled as a single machine in their study. This assumption is

further relaxed to include parallel machines in Hall, et al. [29]. The delivery vehicle's capacity and the total number of available vehicles are, however, not considered in their study.

Chiu, et al. [14] also study machine scheduling in combination with job deliveries. They consider one vehicle with limited capacity for delivery, and address the coordination of delivery with job processing on one or two parallel machines in order to minimize the total job arrival time. Only one customer is considered for the problem involving one and two parallel machines, and two customers for the one machine case. It is shown that, even for the most simplified version (single machine, vehicle and customer), the problem is strongly NP hard. Heuristics for these problems are developed and their worst-case performance bounds are provided. Li, et al. [44] continue in this line of research by considering multiple customers. As a result, the vehicle routing becomes a decision variable in their study. The general problem is shown to be strongly NP hard. Polynomial-time algorithms are provided for several special cases of this problem, including fixed number of customers, direct shipment to customers (i.e., one customer per trip) and infinite capacity of the delivery vehicle.

Pundoor and Chen [52] focus on a trade-off between delivery cost and due date performance in coordinating production and delivery. Their study involves multiple customer locations and a single machine with infinite number of capacitated vehicles. The objective is to minimize a joint function of maximum tardiness and delivery cost. The deliveries are assumed to be direct shipments. Since the general problem is shown to be NP hard, they provide a heuristic procedure for the general problem and prove that this heuristic is asymptotically optimal as the number of orders approaches infinity. They use a computational study, and compare the solution value obtained with a lower bound obtained by using a column generation approach to show that this heuristic is able to generate near-optimal solutions.

All the above studies assume a single or parallel machine to model the production function, which is an over-simplification of the production systems encountered in real-life. However, the introduction of more complex shop floor models will inevitably result in an intractable problem that is difficult to analyze. In view of this fact, such an

integration of production and shipping can be addressed at the operational planning level, which allows for a simpler model to capture the production function. Park [49] considers the integration of production and distribution at this level. The problem that is addressed involves multiple plants, multiple products and multiple retail outlets over multiple periods. A mixed integer programming (MIP) model is developed for this problem. However, because of its size, this model soon becomes intractable. Consequently, a heuristic procedure is developed to solve the problem, and it is shown to generate good results.

The problem that we study in this chapter involves a single production facility and multiple customer locations. It can be described as follows: A company owns a production facility, which is used to manufacture products that the company ships to customers by using company-owned trucks. The customer locations are grouped into regions or routes, and consequently, the customer orders belonging to a region can be shipped together on a truck going to that region, which takes a fixed amount of travel time. The problem, then, is to appropriately coordinate production and shipping operations so that the total cost of shipping, inventory, earliness, and tardiness is minimized. In case the customer orders going to a region have not been produced, extra trips may have to be undertaken to satisfy customer due dates. The shipping cost is a linear function of the number of trips taken to each route weighted by the average shipping cost per trip for the route. The production resources consist of multiple distinct machines (or departments) with limited capacities. An order is processed on a subset of these machines and is assumed to be completed within a fixed lead time. The shipping resource is a set of identical trucks with limited capacities as well. It is assumed that a customer order can not be split and must be shipped on one truck. Deliveries are also assumed to take a fixed travel time that is specific to a route/region. We designate this problem as an integrated production and shipping planning problem (IPSPP).

The proposed problem has several features that distinguish it from the existing studies reported in the literature:

1. Multiple distinct capacitated machines in a plant
2. Finite number of capacitated vehicles and arbitrary order sizes

3. Make-to-order type of system
4. Production lead time and delivery lead time
5. Manufacturer's shipping and inventory costs combined with the customer's due date performances

To put the problem in perspective, consider an example consisting of 6 customer orders that need to be produced and delivered over the next 4 weeks. These orders belong to 3 different routes and only orders belonging to the same route can be shipped together in a truck. Suppose the orders have arrived in the sequence of their indices, i.e. O1, O2, O3, O4 and O5 (see Figure 4.1). Figure 4.1 also shows order due dates (in weeks) and the processing time of each order (number in parenthesis). For simplicity, assume that all orders are processed on one machine, which is available 42 hours per week. If the orders are processed in accordance with the earliest due date (EDD) rule, then we obtain a production schedule that is shown in Figure 4.2.

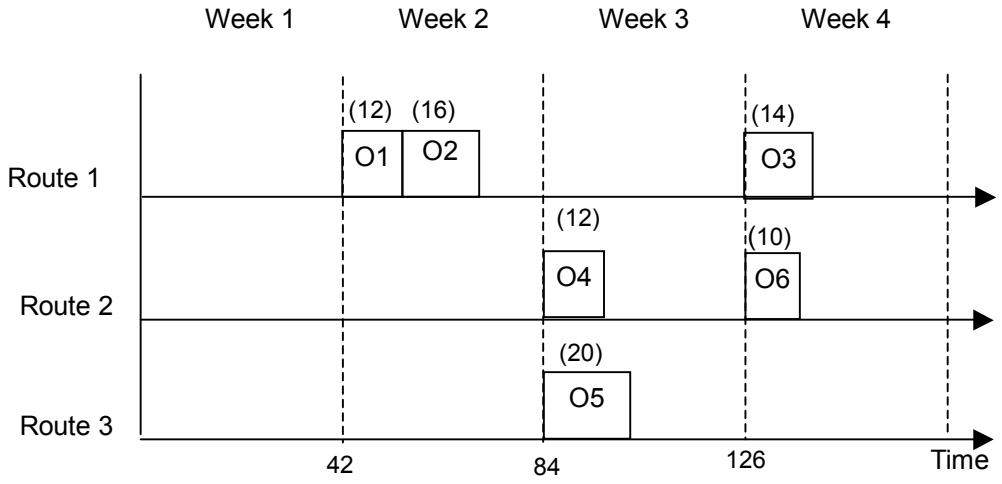


Figure 4.1 Order due time

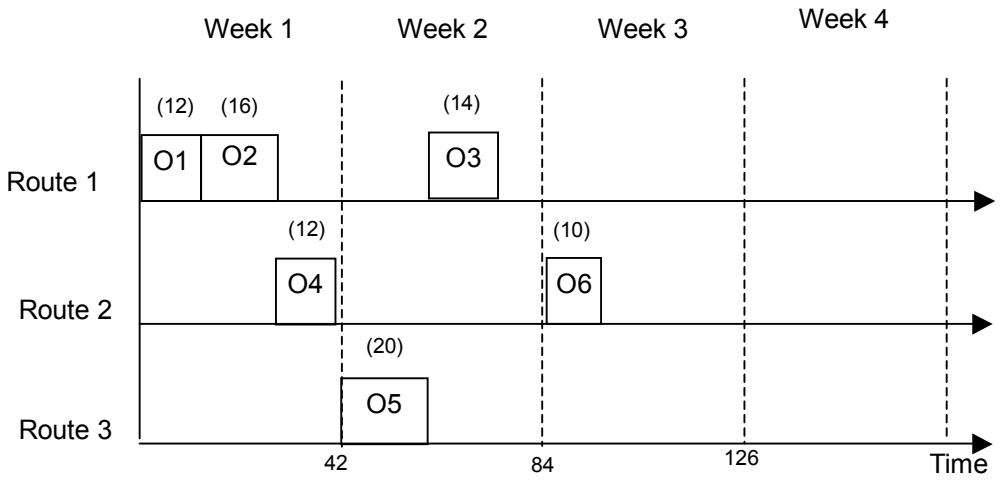


Figure 4.2 EDD production and shipping schedule

Suppose that the earliest an order can be delivered is in the week right after the week in which it is produced. Assuming sufficient truck capacity, this EDD production schedule will require 5 trips for the delivery of these orders. However, an optimal production schedule (shown in Figure 4.3) requires only 3 trips. Thus, it is essential to judiciously group the orders to achieve the desired objective.

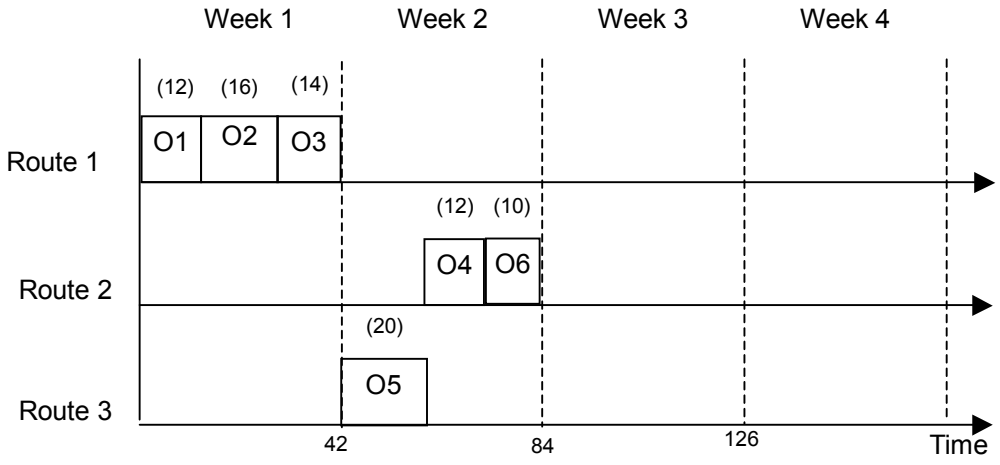


Figure 4.3 Optimal production and shipping schedule

The notation that is used in this chapter is presented in Section 4.2. The rest of this chapter is organized as follows: In section 4.3, we present an integer programming

formulation for the IPSPP on hand. The complexity of this problem is also shown. In section 4.4, a method is presented to obtain a tighter linear programming relaxation for this problem. Also, a branch-and-bound procedure is described for the solution of the IPSPP. The numerical experimentation conducted to test the effectiveness of the proposed branch-and-bound procedure is described in section 4.5. Finally, concluding remarks are made in section 4.6.

4.2 Notation

The notation used in this chapter is as follows:

Indices:

t	time period index, (t = 1,…,T)
o	order index, (o =1,…,O)
r	route index, (r=1,…,R)
m	machine index, (m =1,…,M)

Variables:

x_{ot}	binary variable, =1, if an order o starts shipping in time period t; =0, otherwise
y_{rt}	integer variable, number of vehicles assigned to route r in time period t
z_{ot}	binary variable, =1, if an order o enters the production system (i.e., production is started on it) in time period t; =0 otherwise

Parameters

d_r	cost factor proportional to average distance traveled on route r
t_o	time period in which an order o is due
a_o	cost for delivering an order o early per time period
b_o	backlog cost (lateness penalty) per time period for order o
c_o	inventory cost per time period incurred at the production facility for an order o
k_r	number of time periods required for a single trip on route r
v_o	Production lead time of an order o in number of time periods
U	total vehicle capacity (integer units)
w_o	vehicle capacity units required for an order o,($0 \le w_o \le U$, integer)
S_r	set of orders belonging to route r ; $O = \left\lvert \bigcup_{r=1}^{R} S_r \right\rvert$

K	total number of available vehicles
l_{om}	number of time periods required by an order o to complete all steps up to machine m
C_m	total available capacity of machine m per time period
$\boldsymbol{p_{om}}$	capacity of machine m required by an order o

4.3 Formulation of the Integrated Production and Shipping Problem

The IPSPP can be formulated as follows:

IPSPP: minimize Z=

$$\sum_{r=1}^{R}\sum_{t=1}^{T} d_r y_{rt} + \sum_{r=1}^{R}\sum_{o\in S_r}\left(\sum_{t=v_o}^{t_o-k_r} a_o(t_o-k_r+1-t)x_{ot} + \sum_{t=\max(v_o,t_o-k_r+2)}^{T} b_o(t-t_o+k_r-1)x_{ot}\right) + \sum_{o=1}^{O} c_o I_o \quad (4.1)$$

subject to:

$$Uy_{rt} - \sum_{o\in S_r} w_o x_{ot} \geq 0\,, \qquad \forall r = 1,...,R; \forall t = 1,...,T \quad (4.2)$$

$$\sum_{r=1}^{R}\sum_{i=t-2k_r+1}^{t} y_{ri} \leq K\,, \qquad \forall t = 1,...,T\,, \quad (4.3)$$

$$\sum_{t=v_o}^{T} x_{ot} = 1\,, \qquad \forall o = 1,...,O, \quad (4.4)$$

$$\sum_{t=1}^{T-v_o+1} z_{ot} = 1\,, \qquad \forall o = 1,...,O \quad (4.5)$$

$$I_o = \sum_{t=v_o}^{T} t x_{ot} - \sum_{t=1}^{T-v_o+1}(t+v_o-1)z_{ot} \geq 0\,, \qquad \forall o = 1,...,O\,, \quad (4.6)$$

$$\sum_{o=1}^{O} p_{om} z_{o,t-l_{om}+1} \leq C_m\,, \qquad \forall t = 1,...,T; \forall m = 1,...,M\,, \quad (4.7)$$

$$x_{ot} = 0\,, \qquad \forall t = 1,...,v_o - 1\,; \forall o = 1,...,O\,,$$

$$z_{ot} = 0, \qquad \forall t = T - v_o + 2, ..., T; \forall o = 1, ..., O.$$

$$x_{ot}, z_{ot} \ Binary, \ y_{rt} \geq 0, \text{integer}, \ \forall o = 1, ..., O; t = 1, ..., T; r = 1, ..., R$$

The objective is to minimize the total cost due to traveling (first term), earliness (second term), tardiness (third term) and inventory (last term). The earliness and tardiness costs are assumed to be linear. If an order o is shipped in time period t (x_{ot}=1), then $(t_o - k_r + 1 - t)$, in the second term of (4.1), represents the number of time periods that the order is early, while $(t - t_o + k_r - 1)$ in the third term of (4.1) indicates the number of time periods that the order is late. I_o, as defined in Constraints (4.6), is a variable that represents the number of time periods between production completion and shipping of an order, which captures inventory of an order o at the production facility. By assigning cost factors to each of these individual terms, we obtain a composite objective function that would drive the production and shipping plan toward a desired trade-off among the performance metrics of shipping, customer service and production.

Constraints (4.2) define the truck capacity constraint, and ensure that sufficient number of trucks be provided to deliver the scheduled orders on route r in time period t. Constraints (4.3) are truck availability constraints and indicate that the number of trucks that are out and booked in time period t should not exceed total number of trucks. Note that a truck spends $2k_r$ time periods on a (round) trip.

Constraints (4.4) and (4.5) are called generalized upper bound (GUB) constraints and capture the fact that an order can be shipped and produced, respectively, only once. Note that an order o can not be shipped earlier than its production lead time v_o. Similarly, an order must be produced no later than T-v_o+1.

Constraints (4.6) define the inventory constraints, which determine the inventory of an order o (the number of time periods that an order spends in the manufacturer's inventory after having completed production). Note that the non-negativity of the constraints ensures that the shipping of an order occurs only after production.

Constraints (4.7) enforce the capacity constraints of machines. It is assumed that it

takes a fixed number of time periods (l_{om}) for an order o to reach machine m after it enters the production system (i.e., production is started on it). Different processing routes can be modeled by varying the parameters p_{om} and l_{om}. For example, if machine m_1 precedes m_2 in order o's processing route, then we can let l_{om_1} be less than or equal to l_{om_2}. If machine m is not on order o's processing route, then we need to simply set p_{om} to 0.

The above formulation consists of $2OT + RT$ integer variables and $3O+(R+M+1)T$ constraints. It is relatively of small size, and thus, can efficiently model real-life scenarios. The resulting small size of this formulation is amenable to developing an effective solution methodology. Next, we establish the complexity of the IPSPP.

Lemma 4.1 *The IPSPP is NP hard.*

Proof: We prove this by showing that the bin packing problem, which is known to be NP hard, is a special case of the IPSPP. In a bin packing problem, objects of arbitrary sizes are packed into identical bins with limited capacity so as to minimize the number of bins used. If we replace the objects by customer orders and the bins by trucks, then the bin packing problem is a single period IPSPP where each order takes zero processing time on the machines and the shipping cost per trip for each route is set at 1. €

4.4 Solution Methodology for the IPSPP Problem

Since the IPSPP is NP hard, it is not likely to solve large instances of this problem efficiently. Nevertheless, we exploit some structural properties of this problem to develop a procedure for its solution and demonstrates its effectiveness.

First, we tighten the IPSPP formulation by using the reformulation linearization technique (RLT) of Sherali and Adams [60]. RLT generates the convex hull representation of a zero-one integer problem. It achieves this goal via the use of additional variables resulting from the multiplications of the lower level variables. Reformulation refers to the process of multiplication while linearization refers to the definition of new variables in order to keep the linearity of the reformulated problem. It has been shown that the convex hull representation as a result of this relaxation can be

obtained after n levels of reformulation and linearization iterations, where n is the number of zero-one variables. With the resulting convex hull representation, we just need to solve the LP relaxation of the reformulated problem and the solution is guaranteed to be integer. However, the number of variables required for generating the convex hull representation increases exponentially with the number of RLT levels required. Nevertheless, special structures of some integer problems may enable RLT to obtain the integer convex hull with less effort. Sherali, et al. [61] extends the original RLT framework (called RLT0) by proposing a new framework (called RLT1) that is designed to exploit the special structures of zero-one problems. Several special structures, including the GUB constraints, are used to demonstrate the strength of RLT1. The number of RLT1 levels required to construct the convex hull for the GUB-constrained zero-one problems is shown to be equal to the number of the GUB constraints. Although this number could still be large in real-life applications, we can significantly tighten the formulation with only the level-one RLT1 relaxation, since the level-one RLT1 for each GUB set is able to construct a facet of the integer convex hull (see Sherali, et al. [61]), i.e., the selected GUB set of variables are guaranteed to take integer values in the LP solution of the level-one RLT1 relaxation. The use of the first level RLT relaxation (either fully or partially) has been shown to be effective in many applications, such as the location-allocation problem (Sherali and Adams [59]), airline gate assignment problems (Sherali and Brown [62]), quadratic assignment problems (Adams, et al. [1], Ramachandran and Pekny [53] and Ramakrishnan, et al. [54]), and the asymmetric traveling salesman problem (Sherali and Driscoll [63], and Sherali, et al. [64]).

4.4.1 Generation of a Tight Lower Bound of the IPSPP Problem

Our proposed procedure is based on the concepts of the GUB-constrained first level RLT1 relaxation. Let IPSPP denote the original IPSPP formulation. First, define GUB variables as the set of x and z variables associated with a constraint of (4.4) and (4.5). A relaxation of IPSPP, designated as IPSPP′, can be obtained by relaxing the integrality of all variables except one set of GUB variables. Obviously, the objective function value of IPSPP', denoted as Z', is a lower bound of Z, the objective function value of IPSPP. By applying RLT1 relaxation for the GUB constraint, we can easily solve IPSPP′ as a LP

problem and obtain a lower bound which can be used in the branch-and-bound procedure to accelerate the tree-search process.

For the ease of presentation, first denote the GUB constraints associated with x variables as XGUB, and those associated with z variables as ZGUB. Since the formulation procedure for a XGUB constraint is no different from that of a ZGUB constraint, we only consider the XGUB constraint. Let this constraint be associated with order i, and the associated set of XGUB variables be $Q=\{x_{it}: t\in E_i\}$ where E_i is the set of time periods involved in the XGUB constraint.

Note that the GUB variables do not include y variables, and the previously mentioned RLT1 relaxation only applies to binary variables. Therefore, the first level RLT1 relaxation is only applied to Constraints (4.4) to (4.7) while Constraints (4.2) and (4.3) are left unchanged in the resulting new formulation (denoted as IPSPP" henceforth). The generation of IPSPP" is as follows:

1. **Reformulation**: this can be achieved by first multiplying both sides of the XGUB constraint by any other x or z variables of IPSPP that are not in Q:

$$\sum_{k\in E_i} x_{ik}x_{ot} - x_{ot} = 0 \qquad \forall o = 1,...,O, o \neq i; \forall t \in E_o,$$

$$\sum_{k\in E_i} x_{ik}z_{ot} - z_{ot} = 0 \qquad \forall o = 1,...,O; \forall t \in E_o^{'},$$

Then multiply both sides of the constraints (4.4) to (4.7) in IPSPP (not including the GUB constraint) with each XGUB variable in Q. The modified constraints are as follows:

$$\sum_{t\in E_o} x_{ik}x_{ot} - x_{ik} = 0 \qquad \forall o = 1,...,O, o \neq i; \forall k \in E_i,$$

$$\sum_{t\in E_o} x_{ik}z_{ot} - z_{ik} = 0 \qquad \forall o = 1,...,O; \forall k \in E_i^{'},$$

$$\sum_{t\in E_o} t x_{ot}x_{ik} - \sum_{t\in E_o^{'}} (t+v_o-1) z_{ot}x_{ik} \geq 0 \qquad \forall o = 1,...,O; k \in E_i,$$

$$\sum_{o=1}^{O} p_{om} z_{o,t-l_{im}+1} x_{ik} - C_m x_{ik} \le 0 \qquad \forall t = 1,...,T; m = 1,...,M; k \in E_i .$$

2. **Linearization**: we can transform the above constraints into linear constraints by introducing new variables and constraints as follows:

   (a) Replace $x_{ot}x_{ik}$ with W_{otik}, if $o \neq i$; else, replace $x_{ot}x_{ik}$ with 0, if $o=i$ and $t \neq k$, or else with x_{ik}. Replace $x_{ot}z_{ik}$ with V_{otik}. Note that $x_{ot}z_{ok}$ may not be 0, and hence, V_{otok} is a valid variable.

   (b) Add the following constraints for each new variable, e.g. $0 \le W_{otik} \le 1$, $0 \le V_{otik} \le 1$.

The above first GUB-based RLT1 procedure is a simplified RLT procedure that is used in Glover and Sherali [24]. With the new variables, the IPSPP' formulation for the XGUB constraint associated with order *i* becomes

IPSPP": minimize *Z"*=

$$\sum_{r=1}^{R}\sum_{t=1}^{T} d_r y_{rt} + \sum_{r=1}^{R}\sum_{o\in S_r}\left(\sum_{t=v_o}^{t_o-k_r} a_o(t_o - k_r + 1 - t)x_{ot} + \sum_{t=\max(v_o, t_o-k_r+2)}^{T} b_o(t - t_o + k_r - 1)x_{ot}\right) + \sum_{o=1}^{O} c_o I_o \quad (4.1)$$

subject to:

$$Uy_{rt} - \sum_{o\in S_r} w_o x_{ot} \ge 0 \qquad \forall r = 1,...,R, t = 1,...,T \quad (4.2)$$

$$\sum_{r=1}^{R}\sum_{j=t-2k_r+1}^{t} y_{rj} \le K \qquad \forall t = 1,...,T, \quad (4.3)$$

$$\sum_{k\in E_i} x_{ik} = 1$$

$$\sum_{k\in E_i} W_{ikot} - x_{ot} = 0 \qquad \forall o = 1,...,O, o \neq i; t \in E_o,$$

$$\sum_{k\in E_i} V_{ikot} - z_{ot} = 0 \qquad \forall o = 1,...,O; t \in E_o^{'},$$

$$\sum_{t\in E_o} W_{ikot} - x_{ik} = 0 \qquad \forall o = 1,...,O, o \neq i; k \in E_i ,$$

$$\sum_{t\in E_o} V_{ikot} - x_{ik} = 0 \qquad \forall o = 1,...,O; k \in E_i^{'},$$

$$\sum_{t\in E_o} tW_{ot\,ik} - \sum_{t\in E_o^{'}} (t + v_o - 1)V_{ikot} \geq 0 \qquad \forall o = 1,...,O; k \in E_i ,$$

$$\sum_{o=1}^{O} p_{om} V_{iko,t-l_{im}+1} x_{ik} - C_m x_{ik} \leq 0 \qquad \forall t = 1,...,T; m = 1,...,M; k \in E_i .$$

$$x_{ot} = 0 \qquad \forall o = 1,...,O;\ \forall t \notin E_o ,$$

$$z_{ot} = 0 \qquad \forall o = 1,...,O;\ \forall t \notin E_o^{'} .$$

$$0 \leq x_{ot}, z_{ot}, W_{otik}, V_{ikot} \leq 1,\ y_{rt} \geq 0,\ \forall o = 1,...,O; \forall k \in E_i, t = 1,...,T; r = 1,...,R .$$

The optimal solution of IPSPP*"* will automatically have integer values for any variables in Q due to the RLT1 relaxation. The number of variables in IPSPP*"* is $(2O+R+2OT)T$ and the number of constraints is at most $R(T+1)+5OT+MT^2+1$. The size of this LP formulation for a real-life problem is not large, which makes it appropriate to be used as a tightening procedure in a branch-and-bound procedure. The case for the ZGUB constraint is similar to the above procedure for the XGUB constraint, and the resulting ZGUB-based IPSPP" formulation is given below:

IPSPP": minimize $Z"$=

$$\sum_{r=1}^{R}\sum_{t=1}^{T} d_r y_{rt} + \sum_{r=1}^{R}\sum_{o\in S_r}\left(\sum_{t=v_o}^{t_o-k_r} a_o(t_o - k_r + 1 - t)x_{ot} + \sum_{t=\max(v_o, t_o-k_r+2)}^{T} b_o(t - t_o + k_r - 1)x_{ot}\right) + \sum_{o=1}^{O} c_o I_o \quad (4.1)$$

subject to:

$$Uy_{rt} - \sum_{o\in S_r} w_o x_{ot} \geq 0 \qquad \forall r = 1,...,R, t = 1,...,T \quad (4.2)$$

$$\sum_{r=1}^{R}\sum_{j=t-2k_r+1}^{t} y_{rj} \leq K \qquad \forall t = 1,...,T , \quad (4.3)$$

$$\sum_{k\in E_i} x_{ik} = 1$$

$$\sum_{k\in E_i} W_{otik} - x_{ot} = 0 \qquad \forall o = 1,...,O, o \neq i; t \in E_o,$$

$$\sum_{k\in E_i} V_{ikot} - z_{ot} = 0 \qquad \forall o = 1,...,O; t \in E_o^{'},$$

$$\sum_{t\in E_o} W_{otik} - z_{ik} = 0 \qquad \forall o = 1,...,O, o \neq i; k \in E_i,$$

$$\sum_{t\in E_o} V_{otik} - z_{ik} = 0 \qquad \forall o = 1,...,O; k \in E_i^{'},$$

$$\sum_{t\in E_o} tW_{ot\,ik} - \sum_{t\in E_o^{'}} (t + v_o - 1)V_{ikot} \geq 0 \qquad \forall o = 1,...,O; k \in E_i,$$

$$\sum_{o=1}^{O} p_{om} V_{iko,t-l_{im}+1} x_{ik} - C_m x_{ik} \leq 0 \qquad \forall t = 1,...,T; m = 1,...,M; k \in E_i.$$

$$x_{ot} = 0 \qquad \forall o = 1,...,O;\ \forall t \notin E_o,$$

$$z_{ot} = 0 \qquad \forall o = 1,...,O;\ \forall t \notin E_o^{'}.$$

$$0 \leq x_{ot}, z_{ot}, W_{otik}, V_{ikot} \leq 1,\ y_{rt} \geq 0,\ \forall o = 1,...,O; \forall k \in E_i, t = 1,...,T; r = 1,...,R.$$

The GUB-constrained RLT1 formulation, IPSPP*"*, can be utilized to tighten the lower bound at a node of a branch-and-bound procedure as follows: 1) after solving the regular LP relaxation at each node, we examine the LP solution to identify a GUB constraint that is associated with non-integer variables, and then 2) formulate and solve IPSPP*"* corresponding to that GUB constraint to obtain a tightened lower bound *Z"*. It is obvious that there are two key factors that lead to the high quality of the tightened lower bound: (1) the enforcement of integrity of the GUB variables; (2) the selection of the GUB constraint to formulate *Z*". We have addressed (1) previously, and next, turn our attention to (2).

There are two resource constraints in IPSPP, namely, the truck availability constraints (4.3) and the machine capacity constraints (4.7). The concept of the GUB identification heuristic is to identify the non-integer GUB variable(s) that, if rounded up, will lead to the violation of the above two sets of constraints, which, as a result, would give a larger lower bound of *Z*. Given a regular LP relaxation solution at a node, this GUB identification procedure is listed below:

Step 1. For each constraint in (4.3) and (4.7), identify the non-integer *x* and *z* variables that causes the violation of the constraint if it is rounded up. If such a variable is found, add the variable to variable set *G*, where *G* is the set of non-integer *x* and *z* variables.

Step 2. For any variable in *G*, if it is an *x* variable, calculate the change in objective function value as a result of rounding up this variable, and designate it as Δ, where $\Delta = (\frac{w_o}{U} d_r + \chi_{ot})(1 - x_{ot})$, and χ_{ot} is the coefficient of this *x* variable in the objective function.

If it is a *z* variable, then the change in objective function value, as a result of rounding up this variable is $\Delta = \delta_{ot}(1 - z_{ot})$, where δ_{ot} is the coefficient of the *z* variable in the objective function.

Step 3. Sort the variables in *G* in the non-increasing order of Δ values. Select the GUB constraint that contains the variable on top of the list.

4.4.2 Implementation of the Branch-and-bound Algorithm

In order to test the effectiveness of the proposed tightening procedure, a branch-and-bound algorithm is developed. The branch-and-bound tree-search algorithm is composed of three major components, namely, search strategy, lower bound calculation and branching. We have already described determination of lower bound above. The search and branching strategies are described next.

4.4.2.1 Search Strategy

Search strategy addresses the path along which to explore the branch-and-bound tree. Two strategies are generally used, namely, the depth-first strategy and the breadth-first strategy. The depth-first strategy explores the child node first and backtracks only when no more child nodes are available. It keeps a single active node at any time, and hence, requires less storage space. The breadth-first strategy, on the other hand, keeps multiple active nodes, and selects the "best" node to explore according to a certain criterion, such as the "greatest lower bound first" rule. The breadth-first strategy is more storage-intensive, but generally results in a smaller branch-and-bound tree, i.e., explores less number of branch-and-bound nodes. Depth-first strategy is employed in solving the IPSPP due to its ease of implementation and less storage requirement.

4.4.2.2 Branching Strategy

Note that the IPSPP involves both general integer variables (y) and binary variables (x and z), together with GUB constraints for binary variables. Therefore, branching at a given branch-and-bound node also depends on the nature of the variable. If the variable at a node is y_{rt}, then the standard branching scheme for integer variables is used. That is, if y_{rt}=1.3, then the child nodes are $0 \le y_{rt} \le 1$ and $y_{rt} \ge 2$, respectively. For x or z variables, the GUB set is used as the branching variable. For example, suppose that at a given branch-and-bound node, we have $\sum_{t=3}^{8} x_{1t} = 1$, and the LP solution gives $x_{15} = 0.6$, $x_{17} = 0.4$. Since the first non-integer variable is x_{15}, the child nodes will be $\sum_{t=3}^{5} x_{1t} = 1$ and $\sum_{t=6}^{8} x_{1t} = 1$, respectively. The reason for branching over a GUB set of variables is that it helps in reducing the size of the branch-and-bound tree (see Nemhauser [48]).

Given the LP relaxation solution at a node, the question now is: which y variable or GUB set shall we use for branching? The possible options are: 1) branch at the x, y and z variables in a given sequence, or 2) randomly select a non-integer variable or GUB set. For the IPSPP, the number of trucks used (y variable) impacts the set of customer orders selected and also contributes significantly to the total cost incurred. Therefore, it should be given a higher priority than other variables. Preliminary tests have shown that the

sequence in which the branching variables are selected plays an important role in the size of the branch-and-bound tree. On the average, a sequence with y variable in the first position gives a better performance. Therefore, during the branching stage, the y variables are always checked first for non-integer values. This is followed by the identification of a GUB constraint. The GUB constraint that is selected has a larger impact on the objective function value. As a result, this GUB constraint not only helps in tightening the lower bound in the RLT1 relaxation procedure, but also indicates the key area of the solution space that should be explored first.

4.4.2.3 Logical Test

Before presenting the RLT1-based branch-and-bound algorithm, first define the feasible shipping interval (FSI_o) and feasible production interval (FPI_o) of an order o as follows:

$$FSI^o = [t_0, t_1] \text{ where } \sum_{t=t_0}^{t_1} x_{ot} = 1, \text{ and } FPI^o = [t_0, t_1] \text{ where } \sum_{t=t_0}^{t_1} z_{ot} = 1.$$

We can associate a FSI or FPI with each GUB constraint in the IPSPP formulation at a given node. Note that the variable is fixed when t_0=t_1. During the implementation of the branch-and-bound algorithm, we can perform some logical tests on these feasible intervals to reduce the computation effort.

Let a lower bound and upper bound of y_{rt} at an active branch-and-bound node be $LB(r, t)$ and $UB(r, t)$, respectively. Also, for the ease of presentation, let FSI_0^o or FPI_0^o denote t_0, and FSI_1^o or FPI_1^o denote t_1. We can calculate a new lower bound of y_{rt} as follows:

$$LB'(r,t) = \left\lceil \sum_{o \in S_r} \frac{w_o}{U} : t = FSI_0^o = FSI_1^o \right\rceil. \tag{4.8}$$

Proposition 4.1 *At an active branch-and-bound node, for any route r and time period t, if* $LB'(r,t) > UB(r,t)$, *then the node is infeasible and fathomed; otherwise, a new lower bound of* y_{rt}, $LB''(r,t) = \max\left(LB'(r,t), LB(r,t)\right)$.

Another logical test is based on the precedence relationship between shipping and production operations, that is, the shipping time period must be equal to or greater than the production time period plus the production lead time. We have the following proposition:

Proposition 4.2 *At an active branch-and-bound node, for any order o, let* $t_1^p = \min(FPI_1^o, FSI_1^o - v_o + 1)$, $t_0^s = \max(FSI_0^o, FPI_0^o + v_o - 1)$. *If* $t_1^p < FPI_0^o$, *or* $t_0^s > FSI_1^o$, *then the node is infeasible and fathomed; otherwise, new upper and lower bounds are as follows*: $FPI_1^o = t_1^p$, *and* $FSI_0^o = t_0^s$.

4.4.2.4 RLT-based branch-and-bound Algorithm

The steps of the RLT-based branch-and-bound algorithm are as follows:

1. Initialization: define a Last-In-First-Out list L to store the nodes that are not explored (The list ensures that the search is a depth-first search). Add the root node to L. Set the incumbent solution, Z^* to a very big number. Let $LB(r,t) = 0$, and $UB(r,t) = \min\left(\left\lceil \sum_{o \in S_r} \frac{w_o}{U} \right\rceil, K\right), \forall r = 1,...,R; \forall t = 1,...,T$. Also, set the *FSI* and *FPI* values for each order based on Constraints (4.4) and (4.5).
2. If L is empty, stop. Otherwise, select the top node from L, and denote this node as n. Perform logical tests according to Propositions 4.1 and 4.2 on node n. If node n is fathomed, repeat step 2; otherwise go to step 3.
3. Formulate and solve the LP relaxation of IPSPP at node n. Let Z be the objective function value. If the problem is infeasible or $Z \geq Z^*$, then node n is fathomed. Go to step 2. Otherwise, round the solution and check its feasibility. If the rounded solution is feasible, then denote its objective value by Z'. If $Z' < Z^*$, set $Z^* = Z'$. Go to step 4.
4. Obtain the GUB constraint using the GUB identification procedure. Formulate and solve the IPSPP" to obtain a tightened lower bound, Z''. If $Z'' > Z^*$, then node n is fathomed, go to step 2; else, obtain the rounded solution Z'. If it is feasible, then update the incumbent Z^* in case Z' is better. Go to step 5.

5. Check the solution Z'' and use the first non-integer y variable as the branching variable. If no such variable is found, apply the GUB identification procedure based on Z'' to obtain a new GUB set. Branch over the y variable or the new GUB set and create two child nodes. Add the child nodes to stack L and go to step 2.

4.5 Numerical Experimentation

The RLT-based branch-and-bound algorithm is designed with RLT-based lower bound and two logical tests as two options. Hence, the experiment involves four different settings based on whether these two options are selected or not. Since there are many factors in this problem, we chose to fix the number of time periods, truck capacity and machine capacity. The number of orders is varied at three levels: 6, 10 and 20. Associated with each level of number of orders we have fixed number of shipping routes, processing routes and machines. Each order is associated with randomly generated data, including its shipping route, weight, due date, processing route, production lead time (depending on the process route), and cost factors. Each processing route has randomly generated processing times as well. Table 4.1 lists the settings for the randomly generated data sets. Appendix D lists a complete data set in the table format that facilitates its storage in a database. Five data sets are randomly generated for each level of number of orders. For each data set, the four combinations of the two options are used in order to observe the effects of each option. As a result, the total number of runs amounts to 60. Since the branch-and-bound tree search process is time-consuming for larger problem sizes, a time limit of 3 hours (10800 seconds) is used.

Table 4.1 Data settings

Data	Comment
vehicle capacity	100 unit loads
machine capacity	200 hour
weight of orders	1 to 100 unit loads
due date	day 1 to 10
processing time	1 to 100 hour
production lead time	1 to 3 day
traveling cost	$2000 to $6500
earliness cost	$1 to $20
tardiness cost	$1 to $100
inventory cost	$1 to $10

For each run, three measures are recorded, including the objective function value of the incumbent solution, run time and number of branch-and-bound nodes explored. Table 4.2 shows the average values when RLT-based lower bound is utilized. Note that the optimal solutions were obtained when there are 6 orders, and time limit was reached for all other data sets. The detailed results of all observations are provided in Appendix D.

Table 4.2 Average impact of RLT-based lower bound

Number of orders	RLT Bound	Objective	Run time (sec)	Number of nodes
6	No	10016	57	1502
	Yes	10016	64	1469
10	No	29571	10801	154541
	Yes	29563	10801	166430
20	No	65764	10801	104113
	Yes	64118	10801	120629

The table should be read differently for the case in which optimal solution was obtained (shaded area) and the case in which the time limit was reached. In the former case, a smaller run time and number of nodes indicate a better performance, while in the latter case, a smaller objective value and a larger number of nodes are preferred. This is due to the fact that, when an optimal solution has been obtained, the number of nodes is actually the size of the branch-and-bound tree, and hence, a smaller tree reflects the strength of the lower bound. However, the number of nodes becomes an indicator of tree search efficiency when the same amount of time is used. In this regard, the RLT-based lower bound performs better as shown in the table above. Its average objective values are lower than that for the case with standard LP relaxation. When the number of orders is 6, the RLT-based algorithm takes slightly more time to find the optimal solution. In this case, the branch-and-bound tree is rather small, and the benefits of a tighter lower bound are probably not worth the extra effort for calculating the RLT-based lower bounds. Overall, the RLT-based lower bound is shown to be capable of improving the performance of the branch-and-bound algorithm by generating a tighter lower bound.

The average impact of the logical tests is summarized in Table 4.3. Similar to that of RLT-based lower bound, logical tests took more time but resulted in smaller branch-and-bound trees when the number of orders is 6. Inconsistent performances of logical tests

can be found in both the objective values and the number of nodes when problem becomes larger. This is probably due to the non-deterministic nature of the problem, and the fact that the impact of the logical tests is not as significant.

Table 4.3 Average impact of logical tests

Number of orders	Logical Tests	Objective	Run time (sec)	Number of nodes
6	No	10016	58	1496
	Yes	10016	63	1475
10	No	29576	10801	163563
	Yes	29557	10801	157409
20	No	61345	10801	106465
	Yes	68537	10801	118278

4.6 Concluding Remarks

This chapter addresses an integrated shipping and production planning problem. An integer programming formulation is presented that captures various realistic features of the problem. The objective function involves the minimization of not only the shipping and inventory costs incurred by the manufacturer, but also the penalties encountered because of the earliness and tardiness of the orders. The modeling of the shipping operations incorporates the shipping delays and allows multiple time periods to be spent on a trip. The modeling of the production operations enables the orders to take different processing routes that may require several time periods to complete. In order to facilitate the solution of the IPSPP, a procedure to tighten the lower bound at a node of the branch-and-bound procedure is proposed. It achieves this goal by constructing one facet of the IPSPP convex hull using the first-level RLT1 relaxation (see Sherali, et al. [61]). With the help of the special problem structure of the IPSPP, namely, the GUB constraint, the RLT relaxation is greatly simplified, which leads to a moderate increment in problem size. A GUB identification procedure is proposed to find a desirable GUB constraint upon which the GUB-based RLT1 relaxation is formulated. A branch-and-bound algorithm is then developed to incorporate the proposed tightening procedure. The effectiveness of this RLT-based lower bound approach is successfully demonstrated in the subsequent numerical experimentation.

Chapter 5 Concluding Remarks and Future Research

In this dissertation, we have addressed lot sizing problems at the operational planning and scheduling and control levels in the decision-making hierarchy of a manufacturing enterprise. Multiple problems that range from shop floor scheduling and dispatching to production planning are studied, with a significant amount of effort dedicated to addressing the integration of decision making at their respective levels in all of these problems. Chapter 2 addresses the integration of multiple objectives (such as makespan, mean flow time, material handling and WIP etc.) and decisions (such as the sublot sizes, the number of sublots and the sequence in which to process the lots) at the scheduling and control level. The first problem in Chapter 3 addresses the integration of lot sizing at the operational planning level and shop floor dispatching at the scheduling and control level, and the second problem in Chapter 3 addresses the integration of lot sizing with input control strategy at the scheduling and control level. Chapter 4 considers the integration of lot sizing decisions for both production and shipping operations at the operational planning level in order to minimize the total cost of inventory and shipping incurred by a manufacturer as well as customer's service level as measured by due date performance. The methodologies proposed and the insights presented in this research work constitute building blocks for achieving an integrated decision-making system that, eventually, will lead to the optimization of a manufacturing enterprise's overall performance.

Regarding the future work of lot streaming, one extension of *FL2/n/C* is the consideration of discrete sublot sizes. The problem can also be expanded to include other realistic features such as setup times (lot-attached and sublot-attached) and limited transporter availability. For *FLm/1/C* problem, future work may involve the consideration of multiple lots in the multi-machine flow shop, which is a challenging problem due to the difficulty of analytical tractability. Suitable models of such problems may have to be

developed to help in this tractability issue. The other future direction of work in lot streaming pertains to consideration of other production line configurations, such as parallel machines, or the intermingling of batching operations where different lots can be processed simultaneously with lot streaming operations.

For the future direction of work regarding lot sizing in complex batch production systems such as wafer fabs, more work is needed in extending and testing the proposed approaches in an advanced large-scale 300mm wafer fab where automated material handling system (AMHS) is used throughout the production line, and production volume is much larger (thousands of wafers per week). The lot sizing model for generating the target buffer levels at the operational planning level need further refinement in order to accommodate various types of constraints encountered in this environment, e.g., batching machines, long (multi-time period) processing times, sampling at inspection tools and alternative process routings, among others. The proposed CONLOAD input control policy is based on the workload of operators, which are largely replaced by AMHS in the modern 300 mm wafer fabs. An extension of our proposed input control policy to the new manufacturing environment requires more research effort.

Future work can also be directed to develop an efficient solution procedure for the integrated production and shipping planning problem. The work presented in this dissertation explores the special GUB structure of IPSPP and shows how it can be utilized to obtain a tighter lower bound than the regular LP relaxation of IPSPP. Future work can be focused on identifying and utilizing more special structures of IPSPP so that more efficient solution procedures can be developed. Another direction of future direction of research is to consider the routing of trucks, which is a traveling salesman problem (TSP) in conjunction with the production and shipping constraints. The integration of the truck routing problem will bring more realism to the IPSPP, although at the cost of increased difficulty.

Appendices

Appendix A. Experimental Results for the *FL2/n/C* Problem

Table A.1 Improvement range over initial solution with $n \leq 20$

Factor	Level	**SP**	**LSP-DP**	**LSSP-Greedy**	**LSSP-Cyclic**	**LSSP-ZP**	**LSSP-DP**
Number of Lots	5	[0%,26.41%]	[3.8%,33%]	[3.8%,33%]	[3.8%,33%]	[3.8%,33%]	[3.8%,33%]
	10	[2.57%,27.04%]	[5%,28.1%]	[4.99%,28.23%]	[4.99%,28.03%]	[4.99%,28.03%]	[5%,28.26%]
	20	[2.15%,24.42%]	[1.32%,21.27%]	[2.7%,24.88%]	[2.71%,24.88%]	[2.71%,24.88%]	[2.71%,24.89%]
Processing Time	1-5	[0%,20.42%]	[2.36%,23.85%]	[2.7%,26.06%]	[2.71%,26.25%]	[2.71%,26.25%]	[2.71%,26.25%]
	10-100	[0%,27.04%]	[1.32%,33%]	[3.18%,33%]	[3.19%,33%]	[3.19%,33%]	[3.19%,33%]
Lot Size	1-10	[0%,27.04%]	[4.1%,28.52%]	[4.57%,28.65%]	[4.57%,28.63%]	[4.57%,28.63%]	[4.63%,28.65%]
	10-100	[0%,26.41%]	[1.32%,33%]	[2.7%,33%]	[2.71%,33%]	[2.71%,33%]	[2.71%,33%]
Handling Cost	0.1-1	[0%,22.03%]	[2.36%,31.96%]	[2.7%,31.97%]	[2.71%,31.97%]	[2.71%,31.97%]	[2.71%,31.97%]
	10-100	[0%,27.04%]	[1.32%,33%]	[3.18%,33%]	[3.19%,33%]	[3.19%,33%]	[3.19%,33%]
Average		**[0.52%,25.32%]**	**[2.54%,29.52%]**	**[3.39%,30.2%]**	**[3.4%,30.2%]**	**[3.4%,30.2%]**	**[3.4%,30.2%]**

Table A.2 Improvement range over initial solution with $n \geq 100$

Factor	Level	SP	LSP-DP	LSSP-Greedy	LSSP-Cyclic	LSSP-ZP
Number of Lots	100	[1.06%,10.42%]	[1.11%,10.48%]	[1.11%,10.48%]	[1.11%,10.48%]	[1.11%,10.48%]
	200	[0.41%,5.88%]	[0.41%,5.89%]	[0.44%,5.91%]	[0.43%,5.91%]	[0.43%,5.91%]
Processing Time	1-5	[0.41%,7.45%]	[0.41%,7.53%]	[0.44%,7.53%]	[0.43%,7.53%]	[0.43%,7.53%]
	10-100	[0.48%,10.42%]	[0.48%,10.48%]	[0.48%,10.48%]	[0.48%,10.48%]	[0.48%,10.48%]
Lot Size	1-10	[0.41%,7.89%]	[0.41%,7.94%]	[0.44%,7.95%]	[0.43%,7.95%]	[0.43%,7.95%]
	10-100	[0.54%,10.42%]	[0.55%,10.48%]	[0.57%,10.48%]	[0.57%,10.48%]	[0.57%,10.48%]
Handling Cost	0.1-1	[0.41%,10.42%]	[0.41%,10.48%]	[0.44%,10.48%]	[0.43%,10.48%]	[0.43%,10.48%]
	10-100	[0.48%,7.89%]	[0.48%,7.94%]	[0.48%,7.95%]	[0.48%,7.95%]	[0.48%,7.95%]
Average		**[0.53%,8.85%]**	**[0.53%,8.9%]**	**[0.55%,8.91%]**	**[0.55%,8.91%]**	**[0.55%,8.91%]**

Appendix B. Description of DynaLRP Software Tool

About DynaLRP

Output variability is one of the major factors that affect the on-time-delivery performance (OTD) of a manufacturing system, especially that of Hi-Tech manufacturing systems due to the enormous complexity of the production process involved. *DynaLRP* is an operational planning tool that helps in minimizing output variability through a dynamic planning framework which incorporates in it the sources of output variability (due to resource availabilities and random yield). Its planning framework is based on the integration of the release of new lots and the movement of work-in-process (WIP), as well as the use of real-time system status (such as WIP levels and machine status) in a rolling-horizon fashion. With *DynaLRP*, the management can determine the release plan of new materials, set target production quota for each processing area or resource group, identify potential bottlenecks, and predict inventory (finished goods) or WIP levels on a period-by-period basis, in order to minimize output variability.

Features of *DynaLRP*

DynaLRP is tailored to complex manufacturing systems with unreliable processes and long production lead times. However, it can be easily adapted to any manufacturing system. In a manufacturing system, WIP movement decisions are just as important as the release of new lots. *DynaLRP* consists of the following technical features:

- Flexible System Modeling Capability: DynaLRP can easily model different types of production systems including flow shops, job shops, and reentrant lines.
- Use of a Mature Optimization Method: DynaLRP uses large scale Linear Programming (LP) techniques, which have been successful in solving various planning problems, to obtain detailed production plans.

- User Friendliness: DynaLRP is an Excel-based tool that uses input data stored in Microsoft Access. It only requires a working knowledge of these common software. Once the input data is appropriately setup in Microsoft Access, the optimal solution is just a few clicks away.

- Real-time Input: DynaLRP's solution is based on the shop floor status information, including WIP distribution and resource availabilities, which can be extracted from the manufacturing execution system (MES) in real-time via efficient database operations.

- Multi-faceted Solution: The solution provided by DynaLRP can be used for variety of different types of decision making. For example, it can be used to derive production guidelines for a worker operating in a processing area, or can be used to compare different lot release policies.

The major outputs of *DynaLRP* include:

- Lot Release Decisions: these include the release of new lots and the release of WIP at intermediate processing stages.

- Resource Utilization Information: this can aid the user in identifying bottlenecks dynamically.

- Projected Buffer Levels: these pertain both to finished and unfinished goods buffer levels.

- Nominal Costs: these can include multiple components, such as inventory and shortage costs of finished goods, holding cost of unfinished goods or revenue.

Software Requirements for *DynaLRP*

The software requirements for *DynaLRP* are:

- Windows 2000 or higher

- Microsoft Excel 2000 or higher
- Microsoft Access 2000 or higher
- Premium Solver Platform v3.5 (or higher) for Excel
- XPRESS Solver Engine v3.5 (or higher) for Excel

Note that the last two programs are developed and distributed by Frontline Systems Inc. and trial versions are available for download from www.solver.com or www.frontsys.com. The trial versions can be used for 15 days free of charge and have all the features of the full versions.

How to Use *DynaLRP*

The implementation of *DynaLRP* involves two major tasks: 1) Setup of the input data in an Access database; 2)Use of *DynaLRP* to load the input, formulate and solve the planning model, and view solution reports. Next, we briefly explain how to perform each of these tasks step by step and also present the data involved.

Task 1. Access Database Setup

Step 1. Setup the planning horizon and each period of the horizon using the following database tables:

Table B.1 Planning periods

Field Name	Data Type	Description
Period	Number	Index of Period
Name	Text	Name of Period

Table B.2 Time setting

Field Name	Data Type	Description
Period Length	Number	The Length of a Period
Proc Time Ratio	Number	The unit ratio of period length and processing time, e.g. hours/min=60

Step 2. Setup product or part related data tables. These include the list of part types, demand of each part type in each period, the processing route of each part, and the fixed new release plan. Since the release of new lots can be a part of the decisions to be made or it can be a given input, the purpose of the last data table is to compare different release policies.

Table B.3 Parts

Field Name	Data Type	Description
Part	Text	Part Name
Period	Number	Index of Period
Inventory	Number	Initial amount of finished goods inventory
Shortage	Number	Initial amount of backlog

Table B.4 Demands

Field Name	Data Type	Description
Part	Text	Part Name
Period	Number	Index of Period
Demand	Number	The number of units required (a negative number means return)

Table B.5 Routes

Field Name	Data Type	Description
Part	Text	Part Name
Step	Number	The step number
Station Family	Text	Name of the Station Family needed at this step
Proc. Time	Number	The number of minutes required for this step
MoveLimit	Number	The maximum number of previous steps that jobs can complete in one period counting backward from the current step (not including the current step)
Initial WIP	Number	Current WIP level at this step.
Yield	Number	The average proportional yield at this step

Table B.6 New release

Field Name	Data Type	Description
Part	Text	Part Name
Period	Number	Index of Period
Start	Number	Quantity of new lots to release

Step 3. Setup resource related data tables which include the list of station families and the estimation of their availabilities.

Table B.7: Stations

Field Name	Data Type	Description
Station Family	Text	Name of the station family
Station Quantity	Number	Number of stations in this family, assuming they are all the same type.
Batch Size	Number	The average batch size that can be processed at this station family.
Buffer Size	Number	The physical queue capacity of a station family

Table B.8: Station status

Field Name	Data Type	Description
Station Family	Text	Name of the station family
Period	Number	index of the period
Availability	Number	Percent of available capacity of the station family.

Step 4. Setup the unit costs of various cost components.

Table B.9 Costs

Field Name	Data Type	Description
Part	Text	Part Name
Period	Number	Index of Period
Revenue	Number	Unit revenue of this part type
Raw Material Cost	Number	The cost of raw material per unit
Inventory Cost	Number	The standard inventory cost of finished goods per day (can change over time).
Shortage Cost	Number	The standard shortage cost of this product per day (can change over time).
WIP Cost	Number	The average cost of unfinished goods per day (can change over time).

Task 2. *DynaLRP* operations

Step 1. Load and update input data

The input data stored in the Access database file (*.mdb) can be loaded by clicking the "Load System Data" button and selecting the file in the standard open-file window. Note that the database file must contain the data tables described under Task 1 following the same format.

Once the new system status data (such as resource availability and the WIP distribution) is available in the Access database, it can be updated in *DynaLRP* by clicking the "Update Current Status" button.

The following figure shows a screen after a successful loading operation.

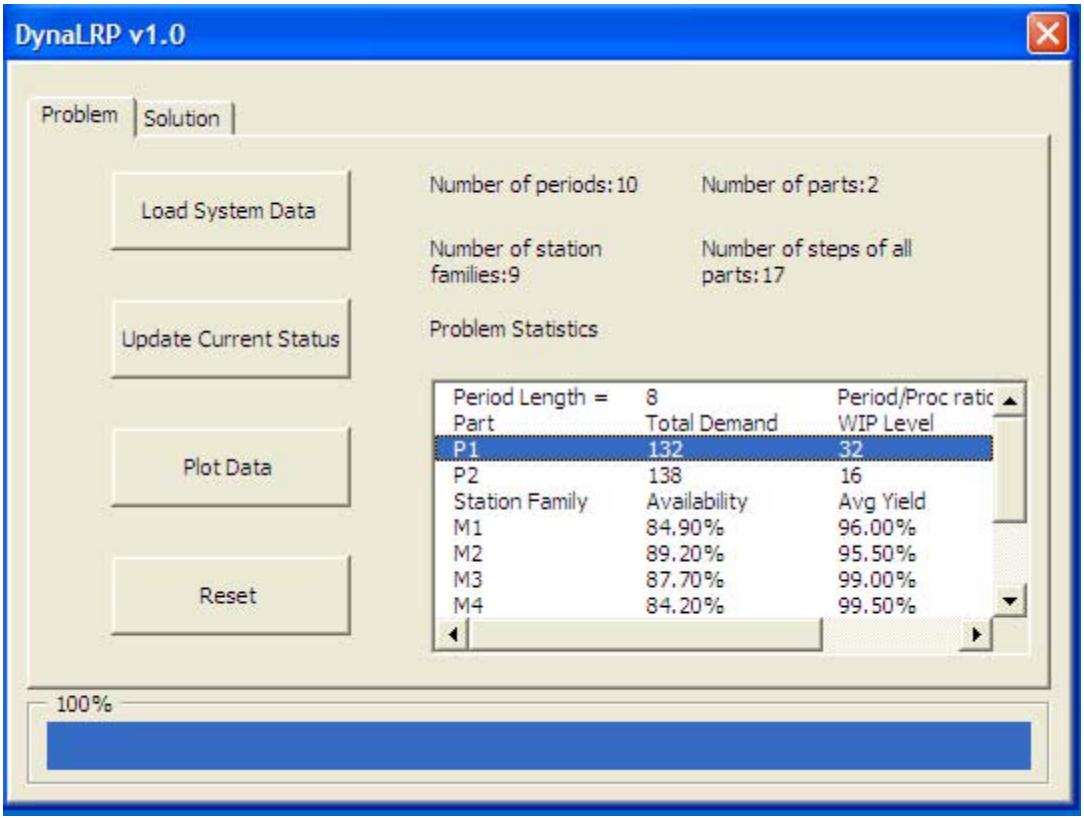


Figure B.1 A screen showing statistics derived from the input data

Step 2. View and reset input data

In order to view the product related data (demand and routing information) and resource availability data, first select a row in the part or station family section of the list box on the bottom right (shown in Figure B.1), and then click "Plot Data" button. A graph of the relevant data will be generated. The user can click the buttons on the graph to obtain a customized view. An example is shown in Figure B.2.

All the data and graphs in *DynaLRP* can be cleared by clicking the "Reset" button. However, if the user wants to keep this data, it must be backed up before clearing it.

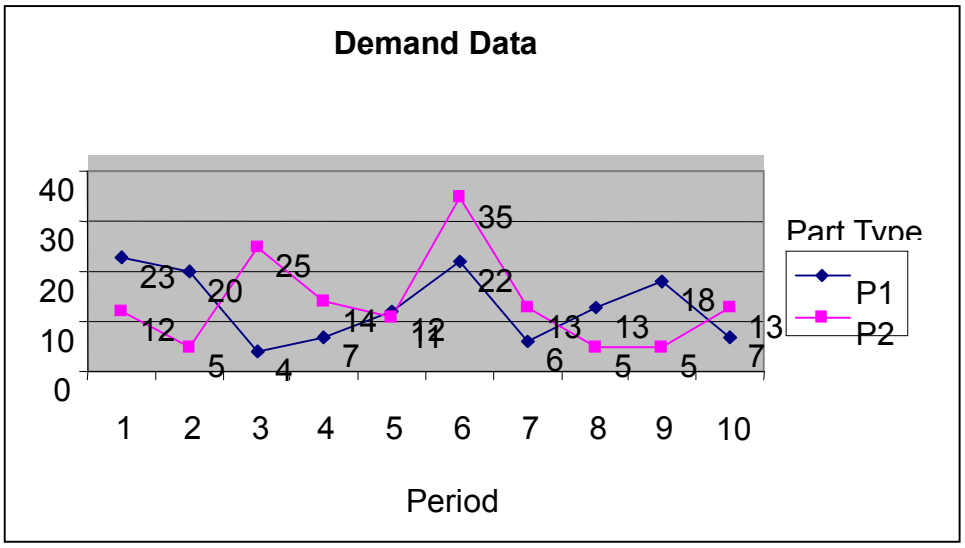


Figure B.2 A plot of the demand data

Step 3. Formulate and solve the planning problem

After loading the input, the user can click the "Solution" tab on the upper-left corner of the dialog. A new screen will appear as shown in Figure B.3.

In this screen, the user can select different cost components or constraints to be included in the problem formulation. The default cost components include inventory and shortage costs of finished

goods, while the default constraints include the flow conservation and station capacity constraints. These constraints are necessary to capture the underlying process and, hence, are not optional. The optional WIP cost reflects the cost of holding unfinished goods. If included, the solution may reflect a tendency of emptying out the WIP. The raw material cost and revenue components help in differentiating among different part types. The optional buffer capacity constraint requires that the WIP at a certain station family be no more than the physical buffer capacity. The move limit constraints establish an upper bound on the movement rate of lots. The fixed new release plan constraints, as alluded to by its name, fix the release plan of new lots in order to compare different release policies. Integer constraints may be desired when fractional yields are not considered.

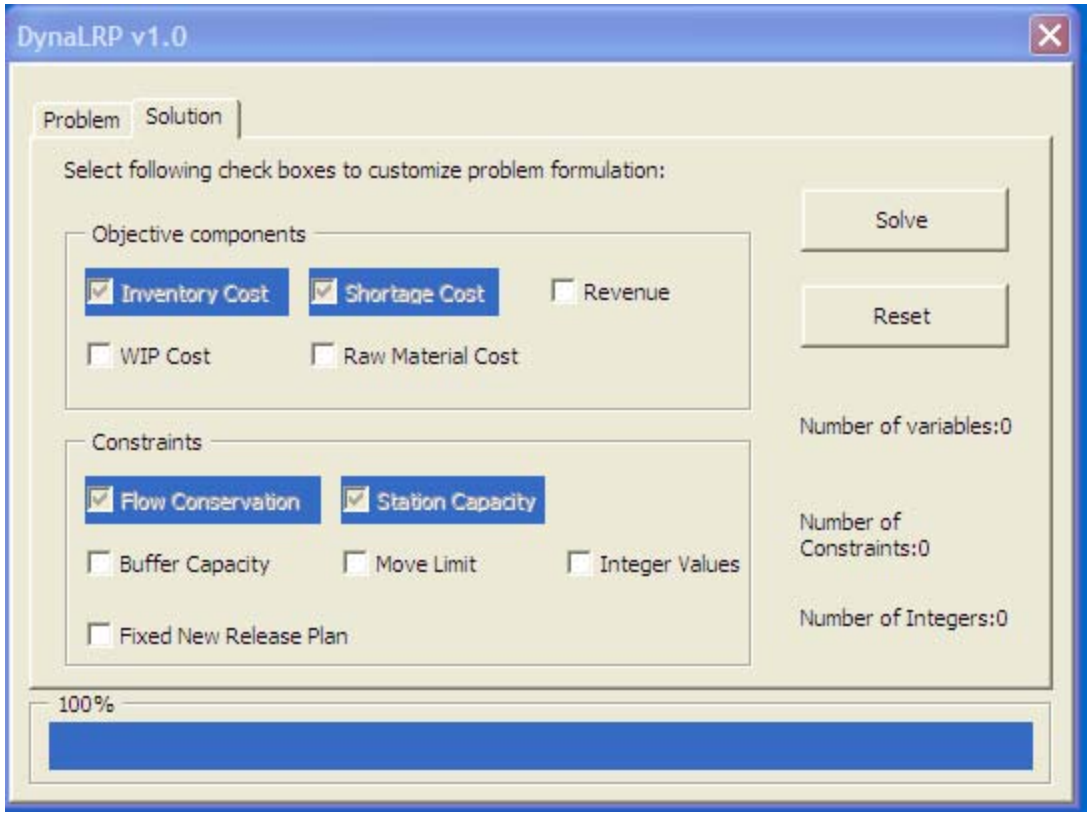


Figure B.3 A screen depicting various options for problem formulation

Click the "Solve" button to formulate and solve the customized LP problem, and the "Reset" button to remove an existing solution and its associated reports.

Step 4. View and interpret solution reports

After the optimal solution is obtained, an executive summary screen will be displayed which summarizes the major results as described in the *DynaLRP* features.

By clicking the buttons located on the executive summary screen, detailed output information can be reviewed in various customizable graphs.

Figure B.4 shows the executive summary screen while Figure B.5, B.6 and B.6 depict detailed reports regarding the release plan of the new lots, intermediate release plan and new finished goods inventory, respectively.

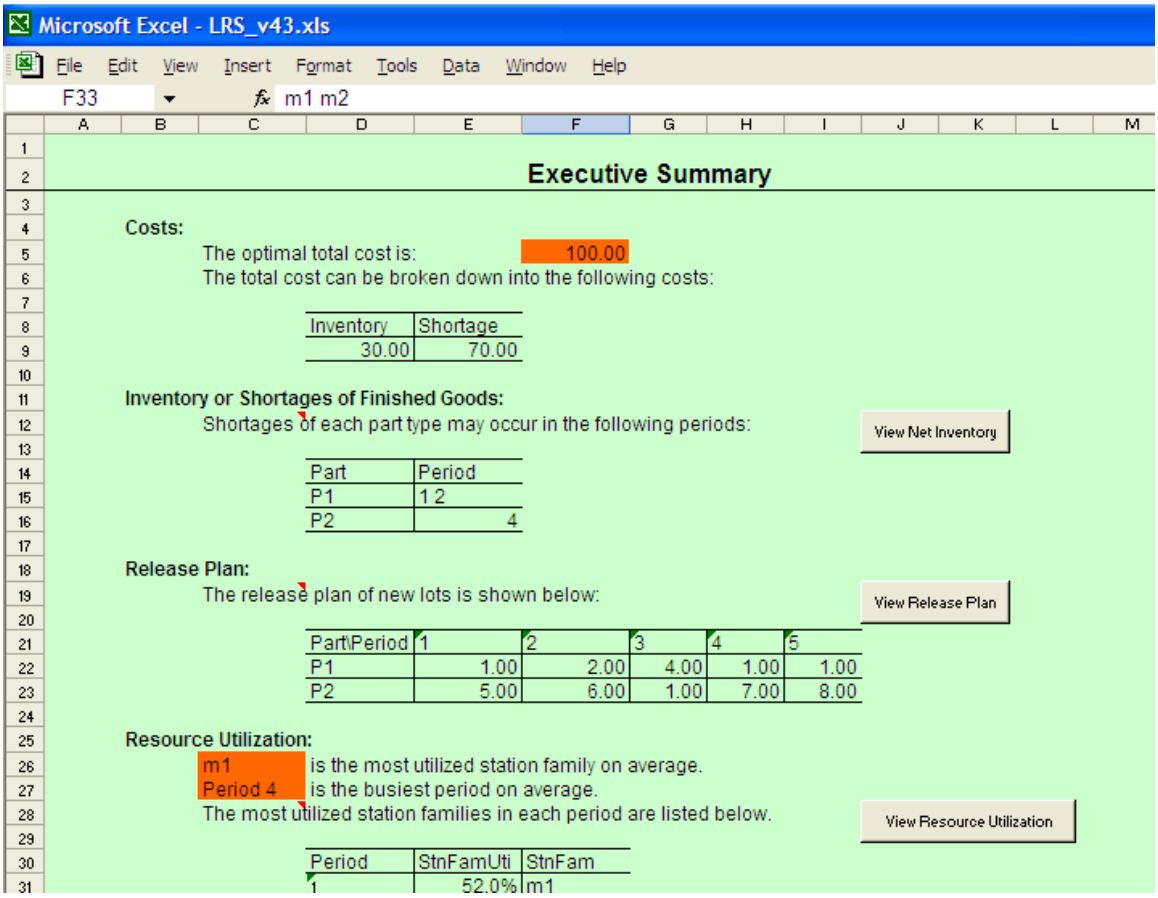


Figure B.4 Executive summary screen

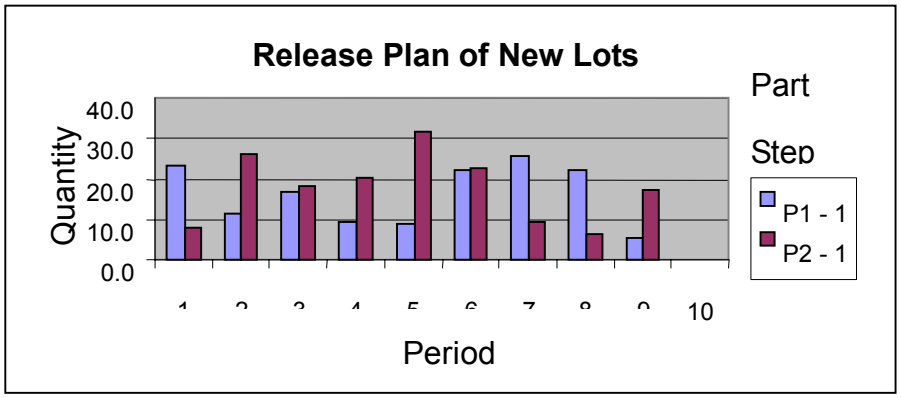


Figure B.5 Detailed report regarding the release of new lots

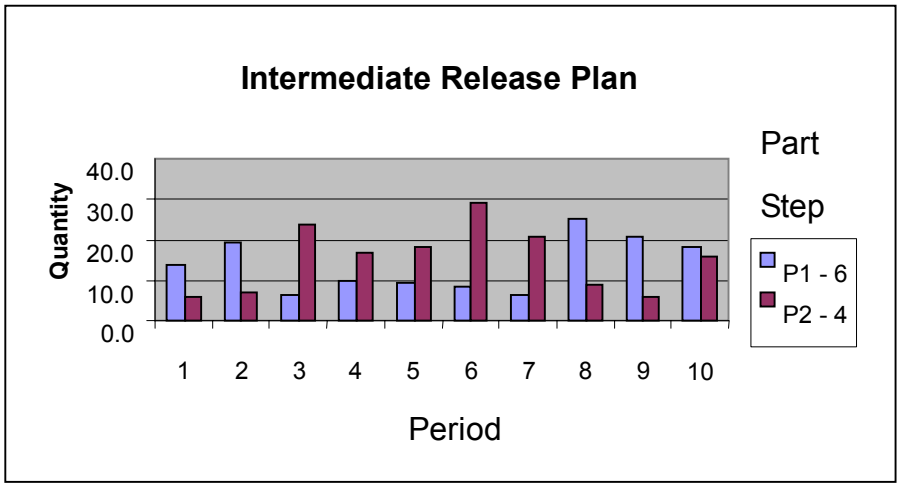


Figure B.6 Intermediate release plan

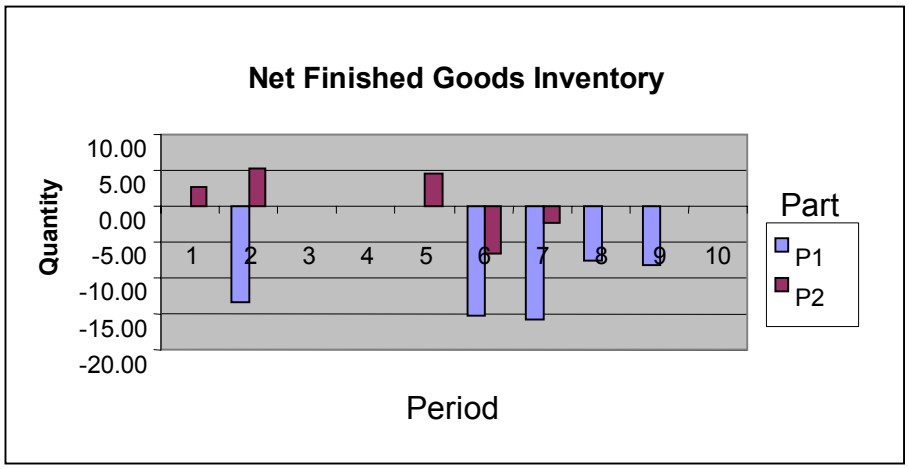


Figure B.7 Net finished goods inventory

What-if scenario analysis can be performed by changing the input data and re-solving the planning problem in DynaLRP. For example, customer due date can be determined by placing the demand in different periods and computing the total cost. This is shown in Figure 4.5 where total cost and shortage cost are plotted against due date.

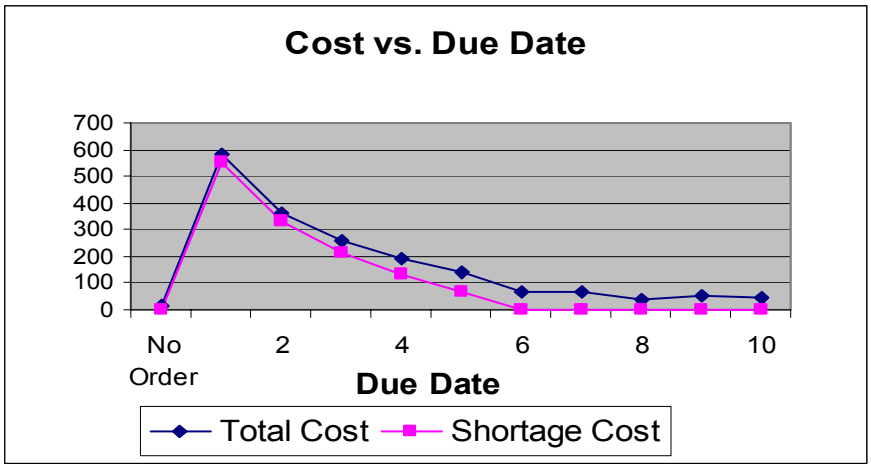


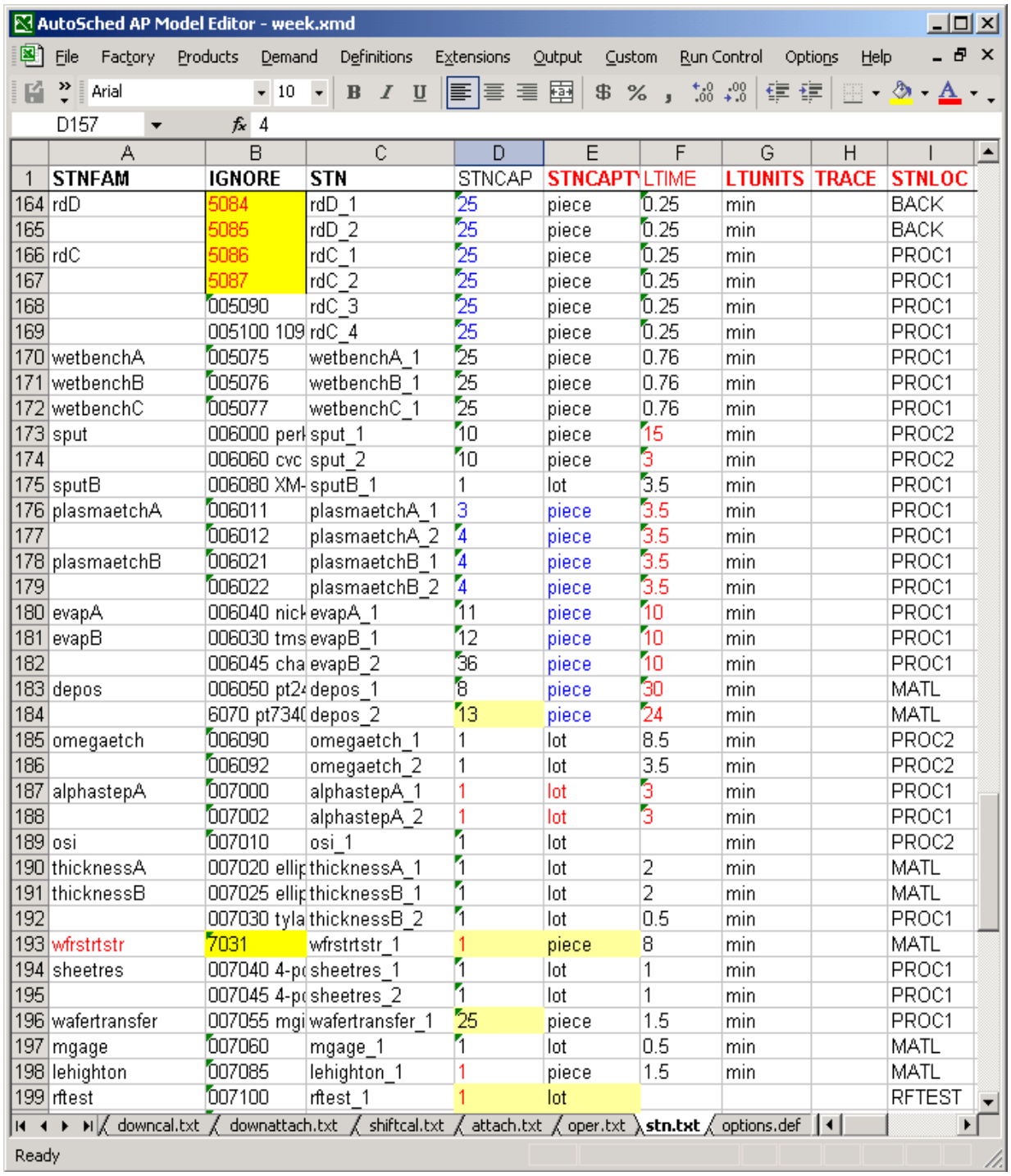
Figure B.8 What-if analysis: total cost and shortage cost vs due date

Appendix C. Simulation Modeling of the Wafer Fab

We used AutoSched AP 7.1 as the simulation software because it contains features pertinent to the semiconductor manufacturing environment. AutoSched AP provides flexible customization capability to meet our specific requirements in modeling the wafer fabrication facility. Its Excel based modeling tool also provides convenience in data manipulation, which is an important factor when building a complex simulation model (see the Model Editor interface in Figure C.1).

The simulation model contains the following primary data files:

- ***Station file***: It defines the main resource in the factory. These stations represent constraints on the capacity of the manufacturing system. Each station is a machine or work area where work is performed. There are about 110 pieces of equipment in the fab. Each piece of equipment is modeled as a station. Usually stations that perform same processes fall in the same station family. Appropriate properties of the station were identified and added into the model, such as capacity, dispatching rules, batching and processing delays. Figure C.1 shows an example of station file in Model Editor's interface.
- ***Part file***: It defines the types of products that are manufactured in the facility. The part file specifies the name of each part type and the route that it follows in the fab. The fab produces a broad line of products that falls in 7 process types. In each process type, there are a few to many variations in designs and recipes. Each of these variations can be a part type. In the model developed, we incorporated only those part types that are currently marketed. There are 24 part types in all.



	A	B	C	D	E	F	G	H	I
1	STNFAM	IGNORE	STN	STNCAP	STNCAPTY	LTIME	LTUNITS	TRACE	STNLOC
164	rdD	5084	rdD_1	25	piece	0.25	min		BACK
165		5085	rdD_2	25	piece	0.25	min		BACK
166	rdC	5086	rdC_1	25	piece	0.25	min		PROC1
167		5087	rdC_2	25	piece	0.25	min		PROC1
168		005090	rdC_3	25	piece	0.25	min		PROC1
169		005100 109	rdC_4	25	piece	0.25	min		PROC1
170	wetbenchA	005075	wetbenchA_1	25	piece	0.76	min		PROC1
171	wetbenchB	005076	wetbenchB_1	25	piece	0.76	min		PROC1
172	wetbenchC	005077	wetbenchC_1	25	piece	0.76	min		PROC1
173	sput	006000 perl	sput_1	10	piece	15	min		PROC2
174		006060 cvc	sput_2	10	piece	3	min		PROC2
175	sputB	006080 XM-	sputB_1	1	lot	3.5	min		PROC1
176	plasmaetchA	006011	plasmaetchA_1	3	piece	3.5	min		PROC1
177		006012	plasmaetchA_2	4	piece	3.5	min		PROC1
178	plasmaetchB	006021	plasmaetchB_1	4	piece	3.5	min		PROC1
179		006022	plasmaetchB_2	4	piece	3.5	min		PROC1
180	evapA	006040 nick	evapA_1	11	piece	10	min		PROC1
181	evapB	006030 tms	evapB_1	12	piece	10	min		PROC1
182		006045 cha	evapB_2	36	piece	10	min		PROC1
183	depos	006050 pt2	depos_1	8	piece	30	min		MATL
184		6070 pt734(	depos_2	13	piece	24	min		MATL
185	omegaetch	006090	omegaetch_1	1	lot	8.5	min		PROC2
186		006092	omegaetch_2	1	lot	3.5	min		PROC2
187	alphastepA	007000	alphastepA_1	1	lot	3	min		PROC1
188		007002	alphastepA_2	1	lot	3	min		PROC1
189	osi	007010	osi_1	1	lot		min		PROC2
190	thicknessA	007020 ellip	thicknessA_1	1	lot	2	min		MATL
191	thicknessB	007025 ellip	thicknessB_1	1	lot	2	min		MATL
192		007030 tyla	thicknessB_2	1	lot	0.5	min		PROC1
193	wfrstrtstr	7031	wfrstrtstr_1	1	piece	8	min		MATL
194	sheetres	007040 4-p	sheetres_1	1	lot	1	min		PROC1
195		007045 4-p	sheetres_2	1	lot	1	min		PROC1
196	wafertransfer	007055 mgi	wafertransfer_1	25	piece	1.5	min		PROC1
197	mgage	007060	mgage_1	1	lot	0.5	min		MATL
198	lehighton	007085	lehighton_1	1	piece	1.5	min		MATL
199	rftest	007100	rftest_1	1	lot				RFTEST

Figure C.1 Station file shown in AutoSched AP's Model Editor

- ***Route file***: It defines the processing steps that parts must go through in the fab. Each step in the route uses a station, which is the main resource that processes lots.

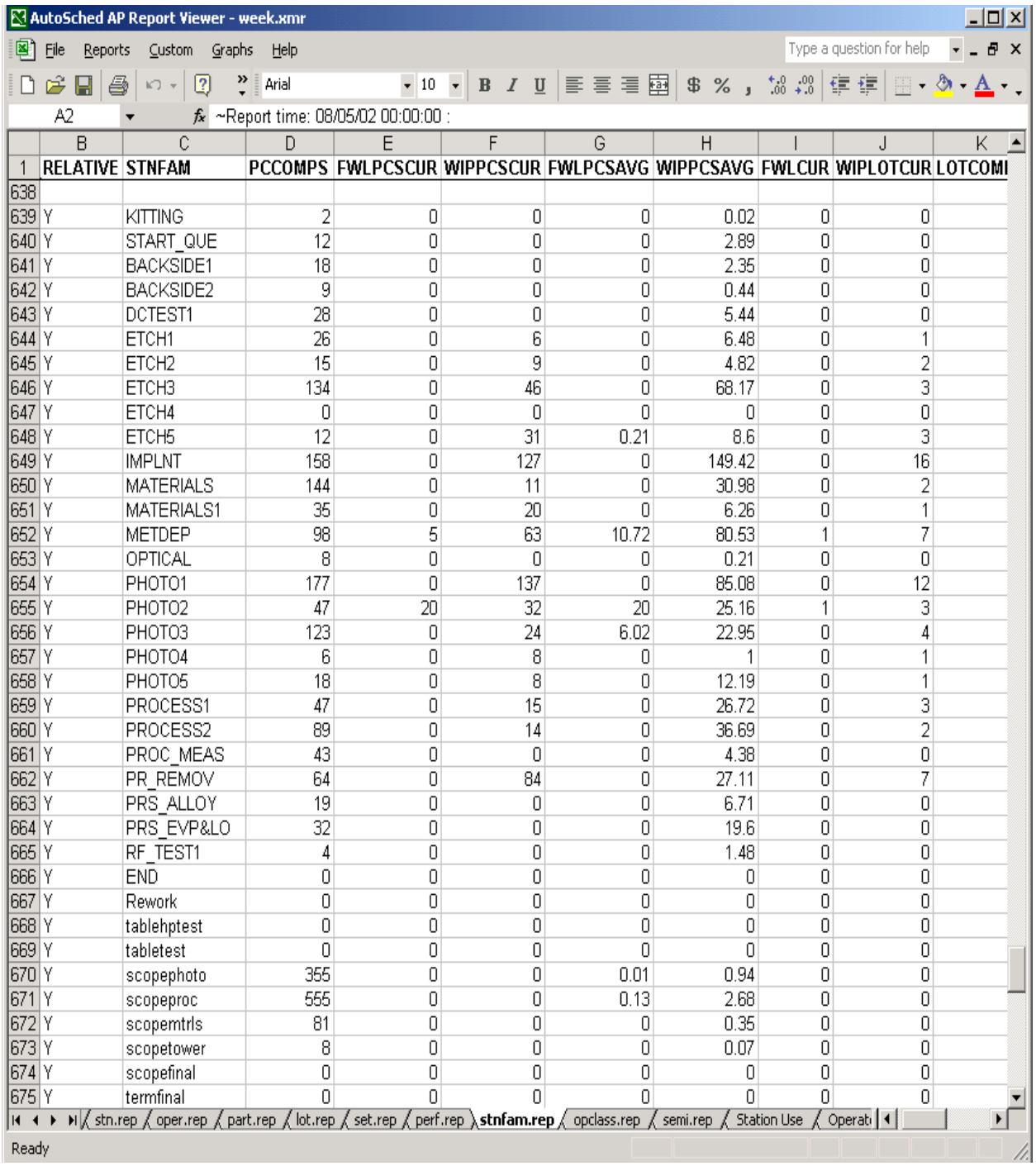
Steps can also use other resources as well, such as operators. In the semiconductor-manufacturing environment, there are usually a large number of processing steps in a route. These are grouped into operations such that one operation finishes one major transformation in the whole procedure. In the fab, one operation is completed in one pull area. Therefore, it is appropriate to model an operation as a step and the associated pull area as the station used at this step. We call such a step as "a primary step". A sequence of primary steps forms a primary route. The real processing steps that are contained in an operation form a sub route. This sub route is specified in a primary step. In this way, the model will run exactly like the real pull system operation. That is, the pull area determines which lot to work on, and then the selected lot will follow the sub route that is defined by its current primary step. For each part type, there is a primary route. Hence, there are 24 primary routes in total. Similarly, there are about 100 operations that are associated with 100 sub routes.

- ***Order file***: It contains the lots and when they start in the model. Lots consist of a number of pieces of a part type specified in the part file. Here, the priority of a lot is also modeled, together with some other properties such as due date, hold status and lot type (experimental or not).
- ***Operator file***: Operators are modeled as one of the fab resources. This file defines the operators by specifying their capacities, classes etc. We also defined a shift calendar and associated operators with shifts in "shift attachment" file. These modeling features capture labor constraints in the model.
- ***Machine downtime file***: The down calendar specifies how the random events of machine breakdowns happen. Almost all the stations were associated with a down calendar.
- ***Preventive maintenance file***: The PM calendar was used to specify how the PM operation for each station is scheduled. In this way, the regular PM operations and their impacts were taken into account in the simulation model.
- ***Setup file***: Some long setups and delays were modeled in a setup file. The required setup was specified at a processing step, and the setup time was given in the setup file.

- ***ActionList file***: We use this file to customize the actions conducted by a lot at each step.

The next task was to build the shop floor control logic of the wafer fab into the model. Two major functions of the shop floor control are modeled, including lot start and lot dispatching. This was accomplished by making use of AutoSched's customization capability. AutoSched AP provides its own C++ class library that can be directly used by users to create customized rules, entities, and statistics etc. The customized code is compiled into a dynamic link library (DLL) that can be used by the simulation model as task selection rules and ranking functions.

Once the modeling tasks are accomplished, we defined the output of the simulation model by creating various reports using AutoSched AP's standard templates. For example, we specified statistics about stations, parts, operators, overall performance and individual lots etc. in their respective reports. During the simulation, these statistics were automatically generated and stored in the reports which can be viewed suing AutoSched AP's Report Viewer after the simulation is completed. Figure C.2 shows a station family report in the Report Viewer.



	B	C	D	E	F	G	H	I	J	K
1	RELATIVE	STNFAM	PCCOMPS	FWLPCSCUR	WIPPCSCUR	FWLPCSAVG	WIPPCSAVG	FWLCUR	WIPLOTCUR	LOTCOMI
638										
639	Y	KITTING	2	0	0	0	0.02	0	0	
640	Y	START_QUE	12	0	0	0	2.89	0	0	
641	Y	BACKSIDE1	18	0	0	0	2.35	0	0	
642	Y	BACKSIDE2	9	0	0	0	0.44	0	0	
643	Y	DCTEST1	28	0	0	0	5.44	0	0	
644	Y	ETCH1	26	0	6	0	6.48	0	1	
645	Y	ETCH2	15	0	9	0	4.82	0	2	
646	Y	ETCH3	134	0	46	0	68.17	0	3	
647	Y	ETCH4	0	0	0	0	0	0	0	
648	Y	ETCH5	12	0	31	0.21	8.6	0	3	
649	Y	IMPLNT	158	0	127	0	149.42	0	16	
650	Y	MATERIALS	144	0	11	0	30.98	0	2	
651	Y	MATERIALS1	35	0	20	0	6.26	0	1	
652	Y	METDEP	98	5	63	10.72	80.53	1	7	
653	Y	OPTICAL	8	0	0	0	0.21	0	0	
654	Y	PHOTO1	177	0	137	0	85.08	0	12	
655	Y	PHOTO2	47	20	32	20	25.16	1	3	
656	Y	PHOTO3	123	0	24	6.02	22.95	0	4	
657	Y	PHOTO4	6	0	8	0	1	0	1	
658	Y	PHOTO5	18	0	8	0	12.19	0	1	
659	Y	PROCESS1	47	0	15	0	26.72	0	3	
660	Y	PROCESS2	89	0	14	0	36.69	0	2	
661	Y	PROC_MEAS	43	0	0	0	4.38	0	0	
662	Y	PR_REMOV	64	0	84	0	27.11	0	7	
663	Y	PRS_ALLOY	19	0	0	0	6.71	0	0	
664	Y	PRS_EVP&LO	32	0	0	0	19.6	0	0	
665	Y	RF_TEST1	4	0	0	0	1.48	0	0	
666	Y	END	0	0	0	0	0	0	0	
667	Y	Rework	0	0	0	0	0	0	0	
668	Y	tablehptest	0	0	0	0	0	0	0	
669	Y	tabletest	0	0	0	0	0	0	0	
670	Y	scopephoto	355	0	0	0.01	0.94	0	0	
671	Y	scopeproc	555	0	0	0.13	2.68	0	0	
672	Y	scopemtrls	81	0	0	0	0.35	0	0	
673	Y	scopetower	8	0	0	0	0.07	0	0	
674	Y	scopefinal	0	0	0	0	0	0	0	
675	Y	termfinal	0	0	0	0	0	0	0	

Figure C.2 Station family report

Appendix D. IPSPP Data Format for Database Use

The IPSPP problem involves a large amount of data to model the real-life scenarios. This appendix provides a sample of the data set stored in the database table-format to facilitate the implementation of this approach.

Table D.1 Problem settings

Number of Orders	Number of Routes	Number of Machines	Number of Vehicles	Vehicle Capacity	Number of Time Periods	Number of Process Routes
20	4	4	10	100	10	4

Table D.2 Orders

Order	Route	Weight	Due	Process Route	Production Lead Time	Earliness Penalty	Tardiness Penalty	Inventory Cost
0	1	53	9	2	3	10	6	7
1	0	14	4	1	2	16	18	5
2	2	38	2	1	2	3	37	3
3	1	84	3	0	3	19	32	6
4	1	68	2	3	2	7	73	4
5	1	5	3	0	3	15	16	5
6	0	87	9	1	2	19	61	0
7	0	97	2	1	2	17	56	6
8	2	67	10	1	2	12	58	10
9	3	55	6	1	2	16	73	3
10	1	67	5	1	2	3	52	9
11	0	5	4	3	2	17	100	7
12	2	77	5	2	3	19	29	6
13	1	5	8	2	3	11	61	2
14	1	4	9	0	3	8	76	1
15	0	50	10	2	3	15	65	9
16	2	13	2	1	2	14	33	4
17	2	58	8	1	2	14	83	1
18	3	47	7	0	3	10	18	4
19	3	0	4	1	2	1	89	7

Table D.3 Vehicle routes

Route	Round Trip Cost	Single Trip Time
0	2000	1
1	5000	1
2	6500	2
3	3000	2

Table D.4 Machines

Machine	Capacity
0	200
1	200
2	200
3	200

Table D.5 Processing routes

ProcessRoute	Machine	ProcTime	LeadTime
0	0	36	1
0	1	10	2
0	2	71	3
1	2	44	1
1	3	80	2
2	0	45	1
2	1	43	2
2	2	99	2
2	3	37	3
3	3	83	1
3	2	80	2

Table D.6 Experimentation results

Num Orders	Data	RLT Cut	Bound	Objective	Run time	Number of nodes	Status
10	10_10_1	No	No	25329	10801	149836	Optimal
10	10_10_1	No	Yes	24680	10801	151807	Optimal
10	10_10_1	Yes	No	24356	10801	153932	Optimal
10	10_10_1	Yes	Yes	24194	10801	169635	Optimal
10	10_10_2	No	No	28804	10801	149743	T Limit
10	10_10_2	No	Yes	28735	10801	146838	T Limit
10	10_10_2	Yes	No	28781	10801	187187	T Limit
10	10_10_2	Yes	Yes	28698	10801	161008	T Limit
10	10_10_3	No	No	23755	10801	157785	T Limit
10	10_10_3	No	Yes	23755	10801	155529	T Limit
10	10_10_3	Yes	No	23755	10801	199317	T Limit
10	10_10_3	Yes	Yes	23755	10801	158106	T Limit
10	10_10_4	No	No	33516.5	10801	150367	T Limit
10	10_10_4	No	Yes	33516.5	10801	158241	T Limit
10	10_10_4	Yes	No	33516.5	10801	144641	T Limit
10	10_10_4	Yes	Yes	33516.5	10801	147917	T Limit
10	10_10_5	No	No	32241	10801	153021	T Limit
10	10_10_5	No	Yes	32241	10801	164806	T Limit
10	10_10_5	Yes	No	32241	10801	166441	T Limit
10	10_10_5	Yes	Yes	32241	10801	166825	T Limit
6	6_10_1	No	No	9017.5	58	1877	Optimal
6	6_10_1	No	Yes	9017.5	85	1845	Optimal
6	6_10_1	Yes	Yes	9017.5	87	1767	Optimal
6	6_10_1	Yes	No	9017.5	73	1805	Optimal
6	6_10_2	No	No	12024.5	63	1621	Optimal
6	6_10_2	No	Yes	12024.5	61	1621	Optimal
6	6_10_2	Yes	No	12024.5	66	1631	Optimal
6	6_10_2	Yes	Yes	12024.5	67	1631	Optimal
6	6_10_3	No	No	9023	42	1085	Optimal
6	6_10_3	No	Yes	9023	40	1005	Optimal
6	6_10_3	Yes	No	9023	43	1011	Optimal
6	6_10_3	Yes	Yes	9023	42	959	Optimal
6	6_10_4	No	No	11002.5	83	2251	Optimal
6	6_10_4	No	Yes	11002.5	81	2239	Optimal
6	6_10_4	Yes	No	11002.5	94	2241	Optimal
6	6_10_4	Yes	Yes	11002.5	106	2231	Optimal
6	6_10_5	No	No	9012	30	729	Optimal
6	6_10_5	No	Yes	9012	30	749	Optimal
6	6_10_5	Yes	No	9012	28	705	Optimal
6	6_10_5	Yes	Yes	9012	29	707	Optimal
20	20_10_1	No	No	59288	10801	116686	T Limit
20	20_10_1	No	Yes	59329	10801	120952	T Limit
20	20_10_1	Yes	No	59288	10801	119966	T Limit
20	20_10_1	Yes	Yes	59339	10801	128707	T Limit

20	20_10_2	No	No	51820	10801	112005	T Limit
20	20_10_2	No	Yes	51809	10801	128239	T Limit
20	20_10_2	Yes	No	51820	10801	114943	T Limit
20	20_10_2	Yes	Yes	51829	10801	133619	T Limit
20	20_10_3	No	No	58036	10801	60889	T Limit
20	20_10_3	No	Yes	67973	10801	60855	T Limit
20	20_10_3	Yes	No	53509	10801	65696	T Limit
20	20_10_3	Yes	Yes	59968	10801	88088	T Limit
20	20_10_4	No	No	66025	10801	92754	T Limit
20	20_10_4	No	Yes	81844	10801	105598	T Limit
20	20_10_4	Yes	No	68728	10801	127700	T Limit
20	20_10_4	Yes	Yes	75260	10801	128006	T Limit
20	20_10_5	No	No	72462	10801	114879	T Limit
20	20_10_5	No	Yes	89049	10801	128275	T Limit
20	20_10_5	Yes	No	72471	10801	139128	T Limit
20	20_10_5	Yes	Yes	88971	10801	160438	T Limit

References

[1] J. Adams, E. Balas, and D. Zawack, "The Shifting Bottleneck Procedure for Job Shop Scheduling," *Management Science*, vol. 34, pp. 391-401, 1988.

[2] K. R. Baker, "Lot Steaming to Reduce Cycle Time in a Flow Shop," The Amos Tuck School of Business Administration, Dartmouth College, Hanover, N.H. Working Paper #203, 1988.

[3] K. R. Baker, "Lot Streaming in the 2-Machine Flow-Shop with Setup Times," *Annals of Operations Research*, vol. 57, pp. 1-11, 1995.

[4] K. R. Baker and D. F. Pyke, "Solution Procedures for the Lot-Streaming Problem," *Decision Sciences*, vol. 21, pp. 475-491, 1990.

[5] B. Bilgen and I. Ozkarahan, "Strategic tactical and operational production-distribution models: a review," *International Journal of Technology Management*, vol. 28, pp. 151-171, 2004.

[6] R. W. Bogaschewsky, U. D. Buscher, and G. Lindner, "Optimizing multi-stage production with constant lot size and varying number of unequal sized batches," *Omega-International Journal of Management Science*, vol. 29, pp. 183-191, 2001.

[7] J. Breithaupt, M. Land, and P. Nyhuis, "The workload control concept: theory and practical extensions of Load Oriented Order Release," *Production Planning & Control*, vol. 12, pp. 625–638. , 2002.

[8] J. Bukchin and M. Masin, "Multi-objective lot splitting for a single product m-machine flowshop line," *IIE Transactions*, vol. 36, pp. 191-202, 2004.

[9] J. Bukchin, M. Tzur, and M. Jaffe, "Lot splitting to minimize average flow-time in two-machine flow-shop," *IIE Transactions*, vol. 34, pp. 953-970, 2002.

[10] F. C. Cetinkaya, "Lot Streaming in a 2-Stage Flow-Shop with Set-up, Processing and Removal Times Separated," *Journal of the Operational Research Society*, vol. 45, pp. 1445-1455, 1994.

[11] F. C. Cetinkaya and M. S. Kayaligil, "Unit Sized Transfer Batch Scheduling with Setup Times," *Computers & Industrial Engineering*, vol. 22, pp. 177-183, 1992.

[12] J. Chen and G. Steiner, "On discrete lot streaming in no-wait flow shops," *IIE Transactions*, vol. 35, pp. 91-101, 2003.

[13] J. A. Chen and G. Steiner, "Discrete lot streaming in two-machine flow shops," *INFOR*, vol. 37, pp. 160-173, 1999.

[14] H.-N. Chiu, J.-H. Chang, and C.-H. Lee, "Lot streaming models with a limited number of capacitated transporters in multistage batch production systems," *Computers & Operations Research*, vol. 31, pp. 2003-2020, 2004.

[15] K. Dogan and M. Goetschalckx, "A primal decomposition method for the integrated design of multi-period production-distribution systems," *Iie Transactions*, vol. 31, pp. 1027-1036, 1999.

[16] Z. Drezner, A. Z. Szendrovits, and G. O. Wesolowsky, "Multi-Stage Production with Variable Lot Sizes and Transportation of Partial Lots," *European Journal of Operational Research*, vol. 17, pp. 227-237, 1984.

[17] S. Eilon, in *Elements of Production Planning and Control*. New York: MacMillan, 1962, pp. 227-263.

[18] S. Elhedhli and J. L. Goffin, "Efficient production-distribution system design," *Management Science*, vol. 51, pp. 1151-1164, 2005.

[19] J. W. Fowler, G. L. Hogg, and S. J. Mason, "Workload control in the semiconductor industry," *Production Planning & Control*, vol. 13, pp. 568-578, 2002.

[20] J. M. Framinan, P. L. Gonzalez, and R. Ruiz-Usano, "The CONWIP production control system: review and research issues," *Production Planning & Control*, vol. 14, pp. 255-265, 2003.

[21] C. Gimenez and E. Ventura, "Logistics-production, logistics-marketing and external integration - Their impact on performance," *International Journal of Operations & Production Management*, vol. 25, pp. 20-38, 2005.

[22] C. A. Glass, J. N. D. Gupta, and C. N. Potts, "Lot Streaming in 3-Stage Production Processes," *European Journal of Operational Research*, vol. 75, pp. 378-394, 1994.

[23] C. A. Glass and C. N. Potts, "Structural properties of lot streaming in a flow shop," *Mathematics of Operations Research*, vol. 23, pp. 624-639, 1998.

[24] F. Glover and H. D. Sherali, "Foundation-penalty cuts for mixed-integer programs," *Operations Research Letters*, vol. 31, pp. 245-253, 2003.

[25] M. Goetschalckx, C. J. Vidal, and K. Dogan, "Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms," *European Journal of Operational Research*, vol. 143, pp. 1-18, 2002.

[26] J. J. Golovin, "A Total Framework for Semiconductor Production Planning and Scheduling," in *Solid State Technology*, vol. 29, 1986, pp. 167-170.

[27] S. K. Goyal, "Note on Manufacturing Cycle Time Determination for a Multistage Economic Production Quantity Model," *Management Science*, vol. 23, pp. 332-333, 1976.

[28] S. K. Goyal and A. Z. Szendrovits, "A constant lot size model with equal and unequal sized batch shipments between production stages," *Engineering Costs and Production Economics*, vol. 10, pp. 203-210, 1986.

[29] N. G. Hall, G. Laporte, E. Selvarajah, and C. Sriskandarajah, "Scheduling and lot streaming in flowshops with no-wait in process," *Journal of Scheduling*,

vol. 6, pp. 339-354, 2003.

[30] N. G. Hall and C. N. Potts, "Supply chain scheduling: Batching and delivery," *Operations Research*, vol. 51, pp. 566-584, 2003.

[31] F. W. Harris, *Operations and Cost*. Chicago: A. W. Shaw Co., 1915.

[32] W. J. Hopp and M. L. Spearman, *Factory physics: foundations of manufacturing* Boston: Irwin/McGraw-Hill, 2001.

[33] M. A. Hoque and S. K. Goyal, "On lot streaming in multistage production systems," *International Journal of Production Economics*, vol. In Press, Corrected Proof, 2004.

[34] M. A. Hoque and B. G. Kingsman, "An optimal solution algorithm for the constant lot-size model with equal and unequal sized batch shipments for the single product multi-stage production system," *International Journal of Production Economics*, vol. 42, pp. 161-174, 1995.

[35] T. K. Hwang and S. C. Chang, "Design of a Lagrangian relaxation-based hierarchical production scheduling environment for semiconductor wafer fabrication," *IEEE Transactions on Robotics and Automation*, vol. 19, pp. 566-578, 2003.

[36] A. A. Kalir and S. C. Sarin, "Optimal solutions for the single batch, flow shop, lot-streaming problem with equal sublots," *Decision Sciences*, vol. 32, pp. 387-397, 2001.

[37] A. A. Kalir and S. C. Sarin, "Constructing near optimal schedules for the flow-shop lot streaming problem with sublot-attached setups," *Journal of Combinatorial Optimization*, vol. 7, pp. 23-44, 2003.

[38] D. H. Kropp and T. L. Smunt, "Optimal and Heuristic Models for Lot Splitting in a Flow-Shop," *Decision Sciences*, vol. 21, pp. 691-709, 1990.

[39] S. Kumar, T. P. Bagchi, and C. Sriskandarajah, "Lot streaming and scheduling heuristics for m-machine no-wait flowshops," *Computers & Industrial Engineering*, vol. 38, pp. 149-172, 2000.

[40] J. W. Lawton, A. Drake, R. Henderson, L. M. Wein, R. Whitney, and D. Zuanich, "Workload regulating wafer release in a GaAs fab facility," presented at Semiconductor Manufacturing Science Symposium, 1990. ISMSS 1990., IEEE/SEMI International, 1990.

[41] R. C. Leachman, "Modeling Techniques in Automated Production Planning for Semiconductor Industry," in *Optimization in Industry*, T. A. Ciriani and R. C. Leachman, Eds. New York: Willy & Sons, 1993.

[42] R. C. Leachman and T. F. Carmon, "On Capacity Modeling for Production Planning with Alternative Machine Types," *IIE Transactions*, vol. 24, pp. 62-72, 1992.

[43] Y. H. Lee, J. Park, and S. Kim, "Experimental study on input and bottleneck scheduling for a semiconductor fabrication line," *IIE Transactions*, vol.

34, pp. 179-190, 2002.

[44] C.-L. Li, G. Vairaktarakis, and C.-Y. Lee, "Machine scheduling with deliveries to multiple customer locations," *European Journal of Operational Research*, vol. 164, pp. 39-51, 2005.

[45] C.-L. Li and W.-Q. Xiao, "Lot streaming with supplier-manufacturer coordination," *Naval Research Logistics*, vol. 51, pp. 522-542, 2004.

[46] C. L. Li and W. Q. Xiao, "Lot streaming with supplier-manufacturer coordination," *Naval Research Logistics*, vol. 51, pp. 522-542, 2004.

[47] C. H. Martin, D. C. Dent, and J. C. Eckhart, "Integrated Production, Distribution, and Inventory Planning at Libbey-Owens-Ford," *Interfaces*, vol. 23, pp. 68-78, 1993.

[48] G. L. Nemhauser, and Wolsey,L.A. , *Integer and Combinatorial Optimization*: John Wiley & Sons, 1999.

[49] Y. B. Park, "An integrated approach for production and distribution planning in supply chain management," *International Journal of Production Research*, vol. 43, pp. 1205-1224, 2005.

[50] C. N. Potts and K. R. Baker, "Flow-Shop Scheduling with Lot Streaming," *Operations Research Letters*, vol. 8, pp. 297-303, 1989.

[51] M. J. D. Powell, "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," *Computer Journal*, vol. 7, pp. 155-162, 1964.

[52] G. Pundoor and Z. L. Chen, "Scheduling a production-distribution system to optimize the tradeoff between delivery tardiness and distribution cost," *Naval Research Logistics*, vol. 52, pp. 571-589, 2005.

[53] B. Ramachandran and J. F. Pekny, "Dynamic factorization methods for using formulations derived from high order lifting techniques in the solution of the quadratic assignment problem," in *State of the Art in Global Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Dordrecht: Kulwer Academic Publishers, 1996, pp. 75-92.

[54] K. G. Ramakrishnan, M. G. C. Resende, and P. M. Pardalos, "A branch and bound algorithm for the quadratic assignment prolbem using a lower bound based on linear programming," in *State of the Art in Global Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Dordrecht: Kulwer Academic Publishers, 1996, pp. 57-74.

[55] R. V. Ramasesh, H. Z. Fu, D. K. H. Fong, and J. C. Hayya, "Lot streaming in multistage production systems," *International Journal of Production Economics*, vol. 66, pp. 199-211, 2000.

[56] A. M. Sarmiento and R. Nagi, "Review of integrated analysis of production-distribution systems," *IIE Transactions*, vol. 31, pp. 1061-1074, 1999.

[57] G. Schmidt and W. E. Wilhelm, "Strategic, tactical and operational

decisions in multi-national logistics networks: a review and discussion of modelling issues," *International Journal of Production Research*, vol. 38, pp. 1501-1523, 2000.

[58] A. Sen, E. Topaloglu, and O. S. Benli, "Optimal streaming of a single job in a two-stage flow shop," *European Journal of Operational Research*, vol. 110, pp. 42-62, 1998.

[59] H. D. Sherali and W. P. Adams, "A decomposition algorithm for a discrete location-allocation problem," *Operations Research*, vol. 32, pp. 879-900, 1984.

[60] H. D. Sherali and W. P. Adams, "A Hierarchy of Relaxations between the Continuous and Convex-Hull Representations for Zero-One Programming-Problems," *Siam Journal on Discrete Mathematics*, vol. 3, pp. 411-430, 1990.

[61] H. D. Sherali, W. P. Adams, and P. J. Driscoll, "Exploiting special structures in constructing a hierarchy of relaxations for 0-1 mixed integer problems," *Operations Research*, vol. 46, pp. 396-405, 1998.

[62] H. D. Sherali and E. L. Brown, "A quadratic partial assignment and packing model and algorithm for the airline gate assignment problem," in *Quadratic Assignment and Related Problems*, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, P. M. Pardalos and H. Wolkowicz, Eds. Providence: American Mathematical Society, 1994, pp. 343-364.

[63] H. D. Sherali and P. J. Driscoll, "On Tightening the Relaxations of Miller-Tucker-Zemlin Formulations for Asymmetric Traveling Salesman Problems " *Operations Research,*, vol. 50, pp. 656-669, 2002.

[64] H. D. Sherali, S. C. Sarin, and P.-F. Tsai, "A class of lifted path and flow-based formulations for the asymmetric traveling salesman problem with and without precedence constraints," *Discrete Optimization*, vol. 3, pp. 20-32, 2006.

[65] D. Sipper and J. R. L. Bulfin, *Production: Planning, Control and Integration*. New York: The McGraw-Hill Companies, Inc., 1997.

[66] M. L. Spearman, D. L. Woodruff, and W. J. Hopp, "Conwip - a Pull Alternative to Kanban," *International Journal of Production Research*, vol. 28, pp. 879-894, 1990.

[67] M. L. Spearman and M. A. Zazanis, "Push and Pull Production Systems - Issues and Comparisons," *Operations Research*, vol. 40, pp. 521-532, 1992.

[68] C. Sriskandarajah and E. Wagneur, "Lot streaming and scheduling multiple products in two-machine no-wait flowshops," *IIE Transactions*, vol. 31, pp. 695-707, 1999.

[69] G. Steiner and W. G. Truscott, "Batch Scheduling to Minimize Cycle Time, Flow Time, and Processing Cost," *IIE Transactions*, vol. 25, pp. 90-97, 1993.

[70] A. Z. Szendrovits, "Manufacturing Cycle Time Determination for a Multistage Economic Production Quantity Model," *Management Science*, vol. 22, pp. 298-308, 1975.

[71] D. Trietsch, "Optimal Transfer Lots for Batch Manufacturing: A Basic Case and Extension," Naval Postgraduate School, Monterey,CA, Technical Report 1987.

[72] D. Trietsch, "Polynomial Transfer Lot Sizing Techniques for Batch Processing on Consecutive Machines," Naval Postgraduate School, Monterey,CA, Technical Report 1989.

[73] D. Trietsch and K. R. Baker, "Basic Techniques for Lot Streaming," *Operations Research*, vol. 41, pp. 1065-1076, 1993.

[74] W. Truscott, "Production Scheduling With Capacity Constrained Transportation Activities," *Journal of Operations Management*, vol. 6, pp. 333-348, 1986.

[75] R. Uzsoy, C. Y. Lee, and L. A. Martinvega, "A Review of Production Planning and Scheduling Models in the Semiconductor Industry .1. System Characteristics, Performance Evaluation and Production Planning," *Iie Transactions*, vol. 24, pp. 47-60, 1992.

[76] R. Uzsoy, C. Y. Lee, and L. A. Martinvega, "A Review of Production Planning and Scheduling Models in the Semiconductor Industry .2. Shop-Floor Control," *Iie Transactions*, vol. 26, pp. 44-55, 1994.

[77] R. G. Vickson, "Optimal lot streaming for multiple products in a two-machine flow shop," *European Journal of Operational Research*, vol. 85, pp. 556-575, 1995.

[78] R. G. Vickson and B. E. Alfredsson, "Two- and three-machine flow shop scheduling problems with equal sized transfer batches," *International Journal of Production Research*, vol. 30, pp. 1551-1574, 1992.

[79] L. M. Wein, "Scheduling Semiconductor Wafer Fabrication," *IEEE Transactions on Semiconductor Manufacturing*, vol. 1, pp. 115-130, 1988.

[80] G. Winz, J. Potoradi, and P. T. Lim, "Improvement of Production Logistics for Backend Manufacturing," presented at Semiconductor Manufacturing Conference Proceedings, 1999 IEEE International Symposium on, 1999.

[81] W. I. Zangwill, "Minimizing a Function Without Calculating Derivatives," *Computer Journal*, vol. 10, pp. 293-296, 1967.