A MATHEMATICAL PROGRAMMING BASED PROCEDURE FOR THE SCHEDULING OF LOTS IN A WAFER FAB

VINOD D. SHENAI

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

Master of Science
in
Industrial and Systems Engineering

September 21, 2001
Blacksburg, Virginia

Keywords: Semiconductor Manufacturing, Wafer Fab, Mathematical Programming, Scheduling, Tardiness, Lagrangian Relaxation, Cycle Time

A MATHEMATICAL PROGRAMMING BASED PROCEDURE FOR THE SCHEDULING OF LOTS IN A WAFER FAB

VINOD D. SHENAI

ABSTRACT

The semiconductor industry provides a host of very challenging problems in production planning and scheduling because of the unique features of the wafer fab. This research addresses the need to develop an approach, which can be used to generate optimal or near-optimal solutions to the scheduling problem of a wafer fab, by using Mathematical Programming for a general case of a wafer fab.

The problem is approached in two steps. First, the number of lots of different products to be released into the system during each planning period is determined, such that the total tardiness of the product orders is minimized over the planning horizon. Second, the schedule of these lots is determined so that the cycle time of each lot released into the system is minimized. Thus, the performance measures based both on due dates and cycle time are considered.

The lot release, tardiness problem is formulated as an integer linear program, and a 3-phase procedure, which utilizes a variation of the Wilkerson-Irwin algorithm, is developed. The performance of this 3-phase procedure is further improved by using insights from classical scheduling theory. The scheduling problem is formulated as a 0-1 integer linear program. An algorithm is developed for tightening the LP relaxation of this 0-1 integer linear programming model (of the scheduling problem) leading to a better performance of the branch and bound procedure used for its solution. Lagrangian relaxation is applied on a carefully chosen set of constraints in the scheduling problem, and a Lagrangian heuristic is developed for scheduling the jobs in each period of the planning horizon. Several useful insights are developed throughout to further improve the performance of the proposed algorithm.

Experiments are conducted for both the tardiness and the scheduling problems. Five experiments are conducted for the tardiness problem. Each experiment has a different combination of number of products, machines, and work orders in a small sized wafer fab (2 to 6 products, 8 to 10 station families, 15 to 30 workstations, 9 to19 work orders, and 100 to 250 lots per work order). The solutions obtained by the 3-phase procedure are compared to the optimal solutions of the corresponding tardiness problems, and the tardiness per work order for the 3-phase procedure is 0% to 25% greater than the optimal solution. But the time required to obtain the optimal solution is 22 to 1074 times greater than the time required to obtain the solution through the 3-phase procedure. Thus, the 3-phase procedure can generate almost optimal solutions and requires much smaller computation time than that required by the optimal solution.

Four experiments are conducted to test the performance of the scheduling problem. Each experiment has a different combination of number of products, machines, routes, bottleneck stations, processing times, and product mix entering the system each day in a small sized wafer fab (2 products, 8 station families, 18 workstations, and 8 to 10 lots released per day into the system). The solution quality of the schedule generated by the Lagrangian heuristic is compared to the solution provided by the standard dispatching rules available in practice. In each experiment, the cycle time of a product for each dispatching rule is divided by the best cycle time for that product over all the dispatching rules in that experiment. This ratio for the Lagrangian heuristic in each experiment and over all the experiments varies from 100% to 104%. For the standard dispatching rules, this ratio ranges from 100% to 120% in each experiment and also over all the experiments. The average of the ratio over all the experiments is the least for the Lagrangian heuristic. This indicates that for the experiments conducted, the Lagrangian heuristic consistently provides a solution that is, or is close to, the best solution and, hence, quite competitive when compared to the standard dispatching rules.

ACKNOWLEDGMENTS

This is a major milestone in my life, for which I wish to thank my parents and brother, without whom this would not have been possible. Their unconditional love and constant support over the years is something that I cannot thank them enough for. I also thank them for having faith in me, and my capabilities, and the advice they offered to me over all these years.

My deepest appreciation goes to my advisor, the Chairman of my committee, Professor Subhash C. Sarin, for his valuable assistance, guidance, and encouragement in bringing this research work to a successful completion. I highly commend him for his character and academic achievements. He showed extraordinary patience and faith in me, and was available, pretty much 24 hours a day.

I am also grateful to my thesis committee members, Dr. Yossi Bukchin, and Mr. Andy Peake for their advise and help.

I would also like to thank all my colleagues and friends that I have acquired throughout my years at Virginia Tech.

Last, but not least, I thank the Faculty and the Staff in the Grado Department of Industrial and Systems Engineering. I am honored to have been a part of such a talented and dedicated staff during my years at Virginia Tech.

I will cherish this period greatly for the rest of my life.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 BACKGROUND

The importance of semiconductor manufacturing is widely acknowledged in the U.S. manufacturing industry. Wafer fabrication is the most technologically complex and capital intensive phase in semiconductor manufacturing. The high cost of wafer fabs (over $3 billion for new 300-mm factories) and the need to pay them off quickly means that efficient designs and operating strategies are absolutely essential. But it is only recently that the production planning and scheduling problems in this environment have begun to be addressed using industrial engineering and operations research techniques.

1.1.1 Manufacturing Process of Very Large-Scale Integrated Circuits

It consists of the following steps: -

1. Wafer Fabrication

   This involves the processing of wafers of silicon or gallium arsenide in order to build up the layers and patterns of metal and wafer material to produce the required circuitry. Many of these operations have to be performed in a clean room environment to prevent particulate contamination of the wafers. The facility in which wafer fabrication takes place is referred to as a wafer fab. The same sequence of operations is repeated for each layer of circuitry on the wafer.

2. Wafer Probe

   In this stage, the individual circuits, of which there may be hundreds on each wafer, are tested electrically by means of thin probes. Wafers are then cut up into individual circuits and the defective circuits are discarded.

3. Assembly or Packaging

   The circuits are placed in plastic packages that protect them from the environment.

4. Final Testing

   Automated testing equipment is used to interrogate each integrated circuit and determine whether it is operating at the required specifications.

Assembly and final test of chips have involved fairly low investment and are labor-intensive operations with short cycle times compared to wafer fabrication. **As a result, the vast majority of research efforts in production planning and control in the semiconductor industry have been directed towards the wafer fab.**

1.1.2 Basic Processing Steps Involved in Wafer Fabrication

1. Cleaning

   This involves the removal of particulate matter before a layer of circuitry is produced.

2. Oxidation, deposition, metallization

   A layer of material is grown or deposited on the surface of the cleaned wafer. Extensive setup time is involved at this step, resulting in machines being dedicated to a limited number of operations.

3. Lithography

   This is the most complex operation, as well as the one requiring the greatest precision. A photo-resistant liquid is deposited onto the wafer and the circuitry is defined using photography. The photo-resist is first deposited and baked. It is then exposed to ultraviolet light through a mask that contains the pattern of the circuit. Finally, the exposed wafer is developed and baked.

4. Etching

   In order to define a circuit, the exposed material is etched away.

5. Ion Implantation

   Selected impurities are introduced in a controlled fashion to change the electrical properties of the exposed portion of a layer. Setup time for this operation may range from few minutes to hours.

6. Photo-resist Strip

   This process removes any photo resist remaining on the wafer.

7. Inspection and Measurement

   Each layer is inspected and measured to identify defects and guide future operations.

1.1.3 Complexities in planning and scheduling for semiconductor industry

1. Complex Product Flows (Reentrant product flows)

The number of steps involved in the processing of a wafer is high, and a number of these steps take place on the same production equipment. For example, a wafer may have to visit the photolithography processing area numerous times to have all the layers of circuitry fabricated. The fact that a lot visits a processing area more than once, is what is called the reentrant product flow.

2. Random Yields

Process yields are uncertain and vary due to environmental conditions, and problems with production equipment or material.

3. Diverse Equipment Characteristics

The characteristics of the equipment used in the semiconductor manufacturing vary widely. Some machines have significant sequence-dependent setup times, while others do not. Some work centers such as etching consists of batch processing machines, where a number of lots are processed simultaneously as a batch.

4. Equipment Downtime

The production equipment used in semiconductor manufacturing is, technologically, extremely sophisticated. It requires extensive preventive maintenance and calibration, and is still subject to unpredictable failures.

5. Production and Development in Shared Facilities

Due to the constant development of new products and processes, very often the same equipment is used for both production lots and engineering test and qualification lots.

6. Data Availability and Maintenance

The sheer volume of data in a semiconductor manufacturing facility makes its acquisition and maintenance an extremely time-consuming and difficult task.

The distinct features of a wafer fab have led to the development of numerous models and techniques for the solution of scheduling problems occurring in this environment. The semiconductor industry provides a host of very difficult and challenging problems in production planning and scheduling. The problems have been addressed from a number of different paradigms, and at present it is difficult to say which of these is more advantageous.

Also, the performance measure to be used in the evaluation of scheduling policies is a matter of contention. The two main classes of performance measures considered are those based on flow time and due dates. The arguments in favor of flow time related performance measures are increased responsiveness to market changes, better yield, and better machine utilization. Also, if flow times are short and have low variance, it results in more realistic due date assignment procedures, which further improve due date performance. In addition, it is often a major goal to maximize throughput, which can be achieved (without increasing WIP) by minimizing mean flow time. The main argument in favor of due date related performance measures is the need for customer satisfaction, which is a critical factor for survival in today’s highly competitive markets.

1.2 MOTIVATION

A majority of research on the wafer fab scheduling problem has been devoted to the development of heuristics and dispatching rules. These heuristics and dispatching rules provide good but non-optimal solutions. There exists a need to develop an approach, which can be used to generate optimal or near-optimal solutions to the scheduling problem for the different environments present in various wafer fabs. This will help in improving the performance of a fab and in making it more competitive.

Mathematical Programming can be used as a framework for developing such an approach. Much of the research in mathematical programming is directed towards the formulation and generation of optimal solution of problems, by using insights into the structure of these problems by rigorously analyzing special cases and simplified models. The solutions to these models may need to be modified manually or through an intelligent decision aid to accommodate features not captured in the model.

The assumptions of discrete deterministic data are often valid for the semiconductor manufacturing environment, where processing times are strictly specified by process recipes and a great deal of equipment is automated.

Thus, deterministic scheduling can serve as a valuable tool in solving the scheduling problem for semiconductor manufacturing by generating an optimal or near-optimal solution for static data. This solution can be modified to incorporate the stochastic nature of real-world manufacturing environment and generate optimal schedules.

The various advantages of using mathematical programming in deterministic scheduling problems are as follows:

- The objective function can be formulated to consider only one performance measure or a combination of different performance measures. Goal or parametric programming can be used to optimize the desired performance measure(s).
- The various static features of the fab can be modeled through constraints.
- Linear, non-linear and integer programming techniques can be used to obtain optimal solutions of the mathematical model.
- These optimal solutions provide much better schedules than the heuristics or the standard dispatching rules.

These advantages provide the primary motivation for this research.

1.3 PROBLEM STATEMENT

The problem that is addressed in this research can be concisely stated as follows:

Given a set of jobs waiting to be released into a fab or for loading in a processing area, determine the order in which to release these jobs so as to minimize the average cycle time.

In wafer fabs, a lot consists of a certain number of wafers held in a cassette. For example, a lot in the 200-mm wafer fab of Infineon Technologies at Richmond, Virginia is made up of 25 wafers. These lots are defined as the jobs in a wafer fab. The processing times in the route of a product are given for the entire lot of that product. The job in a wafer fab is actually a lot of a product.

A work order in a wafer fab consists of the number of wafers ordered by the customer. The number of wafers is divided by the yield rate to obtain the number of starting wafers. The number of starting wafers is divided by the number of wafers per lot in that fab, and the answer in rounded up to the nearest integer to get the number of lots in the work order. In the problem, the number of lots in each work order is given.

The due date for a work order is the date by which the customer should receive the work order. The shipping and handling lead-time of the completed work order is not covered by the scope of the problem and is assumed to be constant. Hence, the due date of a work order is the date by which the work order should be completed in the wafer fab.

The planning horizon is the time duration that is sufficient to complete all the work orders in the fab. The calculation of the planning horizon is based upon the capacity of the fab. It is the upper bound on the time required to complete all the work orders in the system.

The entire planning horizon is divided into planning periods. The duration of each planning period is assumed to be twenty-four hours in this research as explained in 4.3.1.6

A station family consists of a number of workstations. The processing step of a lot to be performed at a station family can be performed at any of the workstations of that family. Some processing areas in a wafer fab, such as etching, consists of batch processing machines (workstations), where a number of lots are processed simultaneously as a batch. Such workstations are called batching stations.

A wafer has to visit the photolithography processing area numerous times to have all the layers of circuitry fabricated. The fact that a lot visits a processing area more than once, is what is called the reentrant product flow in a wafer fab.

Lot dedication scheme refers to a policy adopted in the photolithography processing area of certain wafer fabs, where the photolithography processing operations on all the layers of a wafer are processed on the same workstation where the first layer of that wafer was processed.

Preventive maintenance schedules are drawn up months in advance. The workstation on which the preventive maintenance is performed is not available to perform any processing operation during the duration of preventive maintenance.

Unscheduled downtime occurs in a shop when a workstation breaks down. The workstation is not available to perform any processing until it is repaired.

All the above features of a wafer fab have to be incorporated in the problem formulation and solution methodology.

The problem is approached in two steps. First, the number of lots of different products to be released into the system during each planning period is determined, such that the total tardiness of the product orders is minimized over the planning horizon. Second, the schedule of these lots in the processing areas is determined so that the cycle time of each lot released into the system is minimized.

Thus, the performance measures based both on due dates and cycle time are considered.

1.4 PROBLEM SCOPE

An integer linear programming model is formulated for the tardiness problem and an algorithm is presented for obtaining good solutions.

Sarin, et al [27] have formulated a mathematical model for the scheduling of jobs in a wafer fab. The model aims to minimize the cycle time of the lots in the system and allows the system to run dry over the planning horizon. But, in reality, a manufacturing system never dries out. We continue this research by taking this feature into account and present a modification of the model by solving the problem successively over each period (e.g. a day) of the planning horizon. We present two versions of the model, (1) – when the different steps to be processed over a planning period are known, and (2) – when the different steps to be processed over each planning period are not known apriori. The models used are 0-1 integer-programming models.

We also present an algorithm for tightening the LP relaxation of the 0-1 integer linear programming model of the scheduling problem on hand. This tightening of the LP relaxation leads to better performance of the branch and bound procedure used in solving the 0-1 integer linear programming model.

Next, we look at solution strategies for solving the 0-1 integer linear programming model of the scheduling problem. We successfully apply Lagrangian relaxation on a carefully chosen set of constraints and present a Lagrangian heuristic for scheduling the jobs in each period of the planning horizon. The schedule generated by this heuristic can be implemented on the shop floor.

Experiments are conducted on different sets of problems. The solution quality and computation time for the algorithm to minimize the total tardiness are compared against the optimal solution and the computation time required to obtain the optimal solution of the total tardiness problem. The optimal value of the LP relaxation of the 0-1 integer linear programming model with and without the algorithm for tightening the LP relaxation and their proximity to the optimal integer solution of the model is examined. The solution quality of the schedule generated by the Lagrangian heuristic is compared against the solution provided by the standard dispatching rules available in practice.

1.5 PROBLEM ASSUMPTIONS

This research is based on the following assumptions:

1. The wafer fab produces multiple products.
2. A few station families have parallel machines.
3. The parallel machines in a station family are identical in all aspects.
4. Processing times for all processing steps are deterministic
5. The setup time of each processing step is incorporated in the processing time.
6. The workstations are 100% reliable and available all the time.
7. The buffer size at any workstation is infinite.
8. Due dates are provided for every order of a product.
9. All parameters are measured after system has reached steady state.
10. Products are compatible at a batching station and, hence, can be processed together at the batching station.

1.6 OUTLINE OF THESIS

The outline of the thesis is as follows: Chapter 1 gives a brief overview of the semiconductor manufacturing industry and an introduction to the problems faced in the production planning and scheduling of lots in a wafer fab. It gives a brief outline of the problem under study and also states succinctly the assumptions made. Chapter 2 gives a detailed and exhaustive literature review that is divided into four sections. Chapter 3 introduces the tardiness problem and the algorithm for obtaining a “good” solution to this problem. Chapter 4 presents the scheduling problem, explains the methodology for solving the problem over the entire planning horizon, and introduces the modifications in the formulation to incorporate specific features present in a wafer fab and to accommodate the solution methodology of the problem. An algorithm for tightening the LP relaxation of the 0-1 integer-programming model of the scheduling problem is also presented in Chapter 4. The Lagrangian relaxation methodology for solving the scheduling problem is introduced in Chapter 5. A Lagrangian heuristic is presented to obtain near-optimal solution of the scheduling problem on hand. The validity of the Lagrangian heuristic over different performance criteria is also examined. Finally, Chapter 6 provides the results of the experimentation, the conclusions and recommendations for further research.

CHAPTER 2: LITERATURE REVIEW

2.1 INTRODUCTION

The interest in the production planning and scheduling problem of the semiconductor manufacturing industry began in the late 1980's. Various techniques and models have been developed for these scheduling problems leading to a rich literature in this area.

Uzsoy, Lee and Martin-Vega [33], [34] conducted an exhaustive literature review in 1992 and 1994. They defined problems in three types of areas related to the planning and scheduling of semiconductor manufacturing facilities:

1. Performance Evaluation: Descriptive models to understand the system behavior
2. Production Planning: Long term, aggregate production planning
3. Shop Floor Control: Short term control for the processing of orders.

Apart from the above three areas, we will also take a look at the various mathematical programming models that have been developed for production planning and scheduling of wafer fabs.

2.2 PERFORMANCE EVALUATION MODELS

These models are used for evaluating the performance of a given system configuration rather than optimizing some measure of system performance. The area of performance evaluation modeling seems to be the most technically mature of the above three types of planning and scheduling problems. Traditionally, queuing and simulation models have been used to evaluate the desired performance measure.

2.2.1 Queuing Models

Queuing networks have been used extensively to model semiconductor manufacturing facilities. These models need to have a number of features for their applicability in semiconductor environments; namely multi-server nodes, general service times, general inter-arrival times, customer routing and batching, and splitting of lots. The assumptions that need to be made in order to render these models analytically tractable sometimes limit the accuracy of these models.

Wafer fabs have been modeled as multi-class queuing network [5]. Queuing theory fails to provide any analytical tools on queuing networks for reentrant flows (such as found in wafer fabs). A Markov decision problem can be solved in theory. But no models of a size corresponding to real wafer fabs can be solved in practice. Heuristic approaches include diffusion approximations that assume reflected Brownian motions in the space of queue length [35].

Hence, queuing models have been used only for evaluating the desired performance measures and not for analytically solving the scheduling problems.

2.2.2 Simulation Models

Simulation is one of the most extensively used operations research tools in the semiconductor industry. The reasons for this are the intractability of detailed analytical models of the semiconductor manufacturing process, and the steady improvement in computer technology, which makes building simulation models easier and reduces the computational expense of the resulting models.

Simulation models can also be developed at different levels of detail: a highly detailed model of a particular process step or work-center, or a more aggregate model of an entire facility or subsystem. Typically, the behavior of a wafer fab is obtained by plotting the cycle time, inventory level and throughput obtained from simulation runs against the start rate of wafers into the fab. Schoemig and Kroehn [29] have compared the effectiveness of various dispatching rules using simulation at the 200-mm wafer fab of Infineon Technologies in Germany. This study was used to implement the most effective dispatching rule (FIFO) for that fab, which lead to an improvement in the throughput of the fab.

AutoSched AP is one of the state-of-the-art simulation packages available for the semiconductor manufacturing industry. It allows the user to perform capacity planning, capacity analysis, and real-time scheduling. Major chip manufacturers like Motorola, IBM, Intel, Lucent and Infineon Technologies are the users of this simulation package.

2.3 PRODUCTION PLANNING MODELS

Production planning models are used for high-level, comparatively long-term production planning, with a planning horizon of months or weeks. A hierarchical production planning approach is commonly followed in Production Planning models.

Leachman [20] gives a corporate-level production-planning model for the semiconductor industry. Computerized routines create the input files of an aggregate-planning model and generate the linear programming formulation. The solution to this linear program yields a production plan at the process level of detail. Once an aggregate plan has been obtained, it is dis-aggregated by solving a number of linear programs to divide the volume of production planned for each product family over the individual products. The output from this model is a capacity feasible; weekly start schedule for the various facilities in the company. The one disadvantage of this method is that the planning and scheduling decisions at the shop-floor level are constrained by the production planning decisions made at a higher level.

Lee, Yea and Kim [21] have implemented a hierarchical model for production planning in a wafer fab. They introduce a planning methodology that explicitly considers the cycle time and production capacity. The objective is to satisfy the given demand while maintaining proper level of work-in-process inventory. Their approach involves two steps: requirement planning and capacitated loading calculation considering equipment capacity. In the requirement-planning step, an LP model is used to calculate the necessary new starts into the fab to satisfy the given demand. Another LP model is used in the capacitated loading procedure to select optimum “objects” from the Work-in-Process (WIP) or new starts. The model applied to a fictitious fab for a planning period of 20 weeks showed a decrease in the WIP levels and cycle time of the entire system.

Golovin [15] has discussed the issues in production planning and scheduling in the semiconductor industry. The difficulty on selecting an appropriate objective function is highlighted. A hierarchical approach is recommended based on the insight that production planning and scheduling involves different sets of decisions made at different points in time by different groups of decision makers. Aggregate information is used to make long term decisions. The detailed operational scheduling decisions are made within the constraints of the longer-term decisions already taken.

2.4 SHOP FLOOR CONTROL MODELS

Research on shop floor control is classified by the type of approach used.

2.4.1 Dispatching rules and Input Regulation Methods

The dispatching rules are used to decide what job to schedule next when a work-center becomes free. The dispatching rules are widely used in practice and a considerable body of research exists. The input regulation policies attempt to achieve shorter, more reliable flow times by releasing work to the shop in a controlled manner.

The most common dispatching rules include first-in-first-out (FIFO), shortest processing time (SPT), longest processing time (LPT), and the due date based rules (earliest due date (EDD), least slack (LS)) among others.

Kim et al [18], have developed due-date based scheduling and control policies in a multi-product semiconductor wafer fab. Their results have shown that lot release control and lot scheduling at the photolithography workstations are more important than scheduling at other workstations. Also, it is shown that new dispatching rules work better in terms of tardiness of orders than existing rules such as the EDD rule and other well-known dispatching rules for multi-machine scheduling.

Lu, Ramaswamy and Kumar [24] introduced deterministic rules based on the Least Slack (LS) policy, in order to smooth out the mean and variance of cycle times: The proposed rules are:

1. Least Slack (LS) Rule:
   $\text{Slack}_k = (\text{Set Due Date})_k - (\text{Estimated Remaining Processing Time})_k$

The highest priority is given to the lot k with the least slack. (This rule attempts to make every lot equally early or equally late).

2. Fluctuation Smoothing of Variance of the Cycle Time (FSVCT) Rule:
   S_k = (Release time of lot)$_k$ – (Estimated Remaining Processing Time)$_k$
   The highest priority is given to the lot k with the least value of S. (This rule attempts to reduce the variance of cycle time).

3. Fluctuation Smoothing policy for Mean Cycle Time (FSMCT) Rule:
   S_k = ((n/λ) - (Estimated Remaining Processing Time)$_k$
   where n is the number of the lot and λ is the throughput rate.
   The highest priority is given to the lot k with the least value of S. (This rule attempts to reduce the variance of the inter-arrival time at each buffer).

Using simulation, the authors compared the results of the above rules to those obtained using numerous other rules (such as FIFO, SRPT, EDD, etc). It appears that the "fluctuation smoothing of variance of the cycle time" (FSVCT) policy reduces the variance of the cycle time. The "fluctuation smoothing policy for mean cycle time" (FSMCT) policy decreased the mean cycle time and tends to reduce its variance by avoiding bursting in the arrivals of buffers. The authors claim to reduce the cycle time by 22.4% and its variance by 52% over the baseline FIFO policy.

The most complete study of the scheduling problem seems to be the work of Wein [36]. He reviews 12 different dispatching rules, simulates their applications on two existing fab's data under six common distributions. The extensive amount of result is clearly summarized to enable comparisons. **It appears that no rules dominate in performance** but the author observes that if the input distribution is known, some dispatching rules can be disregarded since they cannot perform better than others. The main interest of this paper lays in the great diversity of rules tested and distribution applied to realistic set of data.

The work of Kumar and Kumar [19] addresses the performance bounds of Markovian queuing network and dispatching rule. The dispatching rules considered in the article are the First Buffer First Serve (FBFS), and Last Buffer First Serve (LBFS). Those dispatching rules are specific to the reentrant flow. The authors analyzed the behavior of these policies under traffic conditions varying from light to very heavy. They observed that the efficiency of the FBFS and LBFS policies could be improved by the addition of constraints on buffer capacity. The authors consider both open and closed queuing network and establish performance bounds for different simple models. The computation of bound for such rules provides important information on the dispatching rules and enables one to select easily the most efficient policy for a particular system under a pre-defined traffic condition.

Sarin et al [27], have proposed a Bottleneck Minimal Idleness dispatching rule for scheduling lots at any workstation. This rule schedules lots such that the idleness at the bottleneck station is avoided altogether or minimized. The simulation performed on real wafer fab shows that this rule performs better as compared to the existing system at a wafer fab facility, and other rules discussed in the literature.

2.4.2 Deterministic Scheduling Algorithms

In these models, all data are assumed to be discrete, deterministic and known a priori. These assumptions place most of these problems in the area of combinatorial optimization. Even though the reentrant product flows in semiconductor manufacturing facilities resemble a job shop, there is more structure in these problems that can be exploited to develop exact or approximate solution procedures.

Graves et al. [16] model a fab as a reentrant flow-shop, where a job reenters the flow-shop a number of times before being completed. They restrict their attention to the class of cyclic schedules. They observe that since a specified production rate is to be achieved, jobs must be started into the line at that average rate. They develop a schedule that performs each operation on a lot exactly once each cycle, obtaining a cyclic schedule that repeats indefinitely and meets the desired production equipment. The length of each cycle is given by the reciprocal of the production rate. Implementation issues such as the effects of machine breakdown and expediting and the extensions of the methodology to multi-product environments are discussed.

Ahmadi et al [1] examine problems of minimizing mean flow time and makespan in flow-shops containing batch and unit-capacity machines, assuming that all jobs have identical processing times on the batching machine. They provide polynomial time algorithms for a number of cases and NP-completeness proofs and heuristics for others.

2.4.3 Control-Theoretic Approaches

Concepts from optimal control theory have been used to develop dispatching and input regulation rules for semiconductor fabrication systems. This body of work is interesting from several aspects. The derivation of policies to minimize both mean and variance of cycle time is a substantial contribution of this approach. A major question is how these types of approaches can be extended to handle some of the detailed phenomena occurring on the shop floor. The most important advantage of the control theoretic approach is that the combinatorial nature of the problem is avoided, rendering the problems more tractable.

Kumar et al. [19], [23], [24] have used control-theoretic ideas to analyze the performance of dispatching policies in wafer fabs. Lu and Kumar [21] study the performance of two classes of dispatching rules: buffer based and due date based. The buffer based rule is First Buffer First Serve, which gives priority to lots early in the process. Due date-based policies studied are Earliest Due Date and Least Slack. The authors study the system under a deterministic model of bursty arrivals, where work arrives at a given rate. They prove that all the policies above except for First Come First Serve are stable, in the sense of cycle time or deviation from due dates being bounded. A series of simulation experiments shows that Last Buffer First Serve performs well for reducing mean cycle time, while Least Slack gives good results for minimizing its variance.

2.4.4 Knowledge Based Approaches

A number of researchers have addressed shop floor control problems using artificial intelligence approaches. Savell et al. [28] Describe an expert system for scheduling a wafer fab. They point out that although individual work centers differ considerably in nature, an expert system approach can take advantage of this modularity. Another advantage of expert systems cited is that they can use more sophisticated data analysis techniques than humans on their own. The expert system consists of a central priority assignment module and equipment scheduling modules for each of the work cells. Information on routings, average cycle times and yields, current WIP location, delivery schedules and products shipped are downloaded from the company's CAM system. The expert system uses an external routine to compute how far behind or ahead of schedule each lot is. Based on this information and knowledge of special lots such as expedited orders, it assigns a priority to every lot. The equipment scheduling modules then use these priorities and processing knowledge specific to the cell to schedule the work cell.

2.5 MATHEMATICAL PROGRAMMING MODELS

These models can be further classified as linear and non-linear programming models.

Hung and Leachman [17] present a linear programming based planner and scheduler. The results given by the system are used as an input to a simulation model that validates the result. When the LP and simulation systems do not agree, the LP is reformulated until satisfactory agreement between the two models in obtained. The LP minimizes a weighted production cost and includes capacity, demand and processor availability. The use of two models ensures reliable results under varying conditions.

Glassey, Shantikumar and Seshadri [12] address the job –release problem for a single product, high volume semiconductor fab. The objective is to minimize the cycle time. The linear control rules are based on intersecting hyperplanes and determine the time of release. These rules can be applied to the fab model developed by Wein [36]. Lou and Kager [22] worked along the same direction. They minimize the WIP while maintaining the output. The validation of their model is achieved by comparing its results to two other simulation-based schedulers.

Connors and Yao [7] paid attention to the total demand for a multi-product fab. The authors define numerous technological constraints and assume a random yield. The optimization of the production is achieved with the help of six linear programs. The main contribution of this work is the integration of constraints that aim at reaching the expected demand set. The problem to meet production targets in an IBM facility motivated the design of this system.

Bai, Srivastsan and Gershwin [3] decompose the scheduling problem into a hierarchical structure to allow for the integration of non-linear and linear programming. The program applies non-linear programming data to establish long run values such as the set-up rate. At a lower level, the system executes linear programming computations to find the production rate etc. The lower level is controlled by different dispatching rules that depend on the results previously obtained.

Mehta and Uzsoy [25] deal with the scheduling of several incompatible product families. The minimization of total tardiness appears to be NP-hard. Dynamic Programming (DP) optimally solves small size systems but larger problems cannot be treated within reasonable amount of time. By applying a decomposition algorithm to the large size problem, they divide it into smaller ones that can be solved using DP. The results of the heuristic are obtained quickly and appear to be robust and near optimal.

Sarin et al. [27] present a mathematical model of a wafer fab. The integer program is formulated by assigning a binary variable to the start times of each operation of each job. The time horizon T is assumed to be discrete and specified by the user.

2.6 CONCLUSION

Various techniques and models have been developed for the scheduling problem of wafer fabs. The production planning and scheduling problems for the semiconductor manufacturing facilities are performance evaluation models, production planning models, shop floor control models, and the mathematical programming models.

The area of performance evaluation modeling is the most technologically mature of the problem areas considered in this review. There has been substantial progress both in the areas of queuing networks and simulation, and both techniques are used extensively in practice. Queuing models are used for fast, approximate analyses while simulation models are developed for detailed studies that take a longer time. The limitations of the queuing models are the assumptions on which they are based. The limitations of the simulation models are the long time necessary to develop and run the large models for complex facilities.

The production planning models are used for long term aggregate planning. In the area of production planning, a hierarchical approach is widely accepted. The relationships between the different levels of the hierarchy and between different functional groups on the same level of the hierarchy should be well defined so that a coherent solution approach can be developed.

The shop floor control models are based on dispatching rules, deterministic scheduling algorithms, control-theoretic approaches, and knowledge based approaches. Dispatching rules are the most common tools for shop floor control due to their low computational requirements. However, the dispatching rules provide myopic decisions based on local optimality. Deterministic scheduling models are computationally intractable and algorithms that are fast enough to be used in a reactive mode need to be developed. The control theoretic approach avoids the combinatorial nature of the scheduling problem. When scheduling for due dates is required, this approach becomes more difficult to apply. The strength of the knowledge-based approach is the ability of the model to capture in detail many of the constraints and interactions that occur on the factory floor. The main disadvantage is the acquisition of the appropriate domain knowledge and its maintenance.

Mathematical programming models are useful because of their ability to provide optimal or near-optimal solutions to the scheduling problems. The main disadvantage of the mathematical programming models is their computational intractability.

CHAPTER 3: LOT RELEASE POLICY TO MINIMIZE TARDINESS OF JOB ORDERS

3.1 INTRODUCTION

3.1.1 Single machine Tardiness problem

Due date based performance measures have recently gained prominence because of the increased importance of customer satisfaction in today's competitive world. Tardiness is one of the most commonly used due date based performance measure. Most of the research work for the tardiness problem has been limited to minimizing the tardiness for the n-jobs, single machine case ($1 || \Sigma T_j$). Even for such a simple case, the tardiness problem turns out to be computationally difficult. The NP-hardness of the problem was established in 1989 [26].

Most of the algorithms for solving the ($1 || \Sigma T_j$) tardiness problem are pseudo-polynomial time algorithms based on dynamic programming. These algorithms provide good solutions to the ($1 || \Sigma T_j$) tardiness problem.

3.1.2 Multiple machine Tardiness problem

The multiple machine tardiness problem ($m || \Sigma T_j$) is more difficult than the single machine tardiness problem and has not received as much attention as the latter. Most of the work for the $m || \Sigma T_j$ problem has been concentrated on the jobs with a single processing step to be scheduled on parallel machines [26].

The manufacturing environment of a wafer fab involves multiple station families. Furthermore, each of the station families contains multiple workstations or parallel machines. Also, each job is actually a work order for a product. Each work order is made up of lots, which are to be released into the system. In addition, each lot requires many processing steps in its route thus complicating the problem. A random release of these lots from different work orders into the system may lead to a delay in the completion of all the work orders, thus causing customer dissatisfaction.

Hence, a lot release policy should be devised which will ensure that the lots are released into the system such that the total tardiness of all the work orders is minimized. To this end, we formulate a mathematical model to minimize the total tardiness of the different work orders present in the wafer fab over a suitable time horizon, and develop a solution methodology.

3.2 AN INTEGER PROGRAMMING MODEL FOR THE RELEASE OF THE WORK ORDERS PRESENT IN A WAFER FAB IN ORDER TO MINIMIZE TOTAL TARDINESS.

3.2.1 Notation

i	A subscript representing a product
j	A subscript representing the work order number of a product
t	A subscript representing a single time period
m	A subscript representing a station family
n_m	Number of parallel machines in the station family m
T	An upper bound on time required to complete all the work orders
d_{ij}	Due date of work order j of product i
d'_{ij}	Modified due date of work order j of product i
C_{ij}	Completion time of work order j of product i
T_{ij}	Tardiness of work order j of product i $\{T_{ij} = \max(C_{ij} – d'_{ij}, 0)\}$
X_{ijt}	= 1, if work order j of product i is released into the system during time period t
	= 0, otherwise
p_{ijt}	Number of lots of work order j of product i released into the system during time period t
B_{it}	Maximum number of lots from any work order j belonging to product i that can be released into the system during time period t due to the capacity constraint of the system (Concept explained in section 3.2.2.4)

k_{ij}	Total number of lots in the work order j of product i
r_{im}	Number of re-entries of product i into station family m
s_{im}	Processing time of each step of product i on station family m in hours
F_{ij}	Time period in which the last lot from work order j of product i is released into the system
BLT_i	Base Lead Time of a lot of product i (Sum of the processing times over all the processing steps of product i)
R_i	Cycle Time of a lot of product i
h_t	Number of hours in time period t
U_{ij}	An upper bound on the time required to complete work order j of product i
L_{ij}	A lower bound on the time required to complete work order j of product i

3.2.2 Key concepts for the model

3.2.2.1 Completion Time of a work order

In mid 1980's, the standard cycle times of a lot in a wafer fab were ranging from ten to fourteen times the theoretical processing time of the lot. Hence, the WIP had to wait for a long time in a wafer fab. Longer wait times led to increased contamination of the wafers and this led to an increase in the yield loss. Hence, an effort was made to shorten the cycle times of wafer fabs by the semiconductor manufacturing industry. The current target and standards of the cycle time set by the industry are three to four times the minimum processing time of a wafer [2].

Lets assume that all the lots of a work order are identical and there are unlimited number of resources present in the system. In this case, each lot can begin processing as soon as it is released into the system and all the processing requirements of the lot can be completed in the theoretical time required to process its operations.

Now, under the ideal conditions in a wafer fab, if every lot is assumed to take for its processing, say n times its theoretical processing time, then the production environment can be assumed to be equivalent to an environment with unlimited number of resources. Consequently, the completion time of a work order is equal to the time its last lot is released into the system plus n times its theoretical processing time. Now, since the processing time for each lot is constant (being n times its theoretical processing time), minimization of the sum of the completion times of work orders is equivalent to the minimization of the sum of the release times of their last lots. Hence, we have shown the following result.

Proposition 3.1

If the time that a lot of a work order spends in the system is the same for all the lots of the work order, then the minimization of the sum of the release times of the last lots of the work orders is equivalent to the minimization of their completion times. As a consequence of the above result, we consider the release time of the last lot of a work order in the problem instead of its completion time.

3.2.2.2 Due Date of a work order

The due date of a work order is the date at which the work order should be completed and shipped to the customer. The Tardiness measure of a work order j of product i is defined as $T_{ij} = \max(C_{ij} - d_{ij}, 0)$.

From Proposition 3.1, the completion time C_{ij} of a work order j of product i is replaced by the time period in which the lot containing the last set of wafers of that work order is released into the system. But, the last lot spends an additional time in the system, which is equal to the cycle time of the lot. The due date of the work order should be modified to accommodate the change in the completion time of the work order considered. Hence, the due date of a work order should be decreased by the cycle time of a lot of that work order.

Hence, the modified due date of a work order is given as

$d'_{ij} = d_{ij} – R_l$, and hence

$d'_{ij} = d_{ij} - n * BLT_i$

where n is a pre-specified factor, usually ≥ 3.

In the subsequent portion of this chapter and all subsequent chapters, the due date of a work order will refer to the modified due date of the work order.

3.2.2.3 Stability of the system

The stability of the system should not be disturbed by the release policy. When excessive number of lots are released into the system, then the WIP at the workstations will increase thereby leading to greater queue times and, hence, an increase in the cycle time. Subsequently, the system will become unstable. If the number of lots released into the system is less than the capacity available, then the system will be underutilized and throughput will decrease. Hence, the release policy should be such that the maximum capacity of the system is utilized without overloading the system.

The WIP flowing through the system can be considered to be a fluid flowing through the system [37]. For the stability of the flow through the system, it is required that the amount of fluid entering the system during a time period should be less than or equal to the rate of flow of fluid flowing through the system during the given time period.

Using this condition for stability, the release policy should be such that the total processing time required for lots released into the system during a time period t should be less than or equal to the maximum machine capacity available during that time period t [37]. This property for system stability will be incorporated as the capacity constraint.

3.2.2.4 Maximum number of lots of a product family i to be released in a time period

The bottleneck of a system determines its throughput rate. Goldratt [14] terms this phenomenon as the Drum-Buffer-Rope concept. The maximum number of lots of a product that can be processed by the bottleneck machine in a single time period is used to determine the throughput and, hence, its input rate into the system in that time period. We can use this methodology to determine the maximum number of lots of any work order of a product that can be processed in a single time period.

Each lot of a product i visits the various station families a certain number of times in its route. Each station family can process a certain number of lots in each time period due to the limited capacity available in that time period. If the number of re-entries of a lot of product i into a station family m is s_{im}, the processing time of a lot of product i at station family m at each entry is r_{im}, (and thus the total processing that a lot of product i undergoes at station family m is s_{im}* r_{im}), the amount of time in a single time period t is h_t, the number of workstations at station family is n_m, (and consequently the total capacity available at station family m in time period t is h_t* n_m), then the number of lots of product i that can be processed at station family m in time period t, when the entire capacity of that station family is devoted to that product, is

$$M_{mit} = \left\lfloor \frac{h_t * n_m}{s_{im} * r_{im}} \right\rfloor \qquad i = 1, 2,3,\ldots \qquad m = 1,2,3,\ldots \qquad t = 1, 2,3,\ldots,T$$

The bottleneck station processes the minimum number of lots in a single time period amongst all station families. The number of lots of product i that can be processed by the bottleneck station family in time period t, when the entire capacity of the bottleneck is devoted to product i, is given by

$$B_{it} = \min_m M_{mit} \qquad i = 1, 2,3,\ldots \qquad t = 1,2,3,\ldots\ldots T$$

Hence, for any work order j of a product i, the maximum number of lots that can be processed by the system in a time period t is B_{it}. The maximum throughput and input rate of any work order j belonging to product i in time period t is B_{it}.

3.2.2.5 Magnitude of time available in each time period

We assume that each time period t is of equal time duration. We consider this duration to be one day. The wafer fab runs continuously for 3 shifts in a day and, hence, the time available in a single time period is 24 hours. Therefore, **h_t = h = 24 hours** and **B_{it} = B_i.**

3.2.3 Integer Programming Model

The integer-programming model determines the number of lots, of each work order, to be released into the system, in each time period, over the entire planning horizon, so that the total tardiness of the work orders is minimized.

Objective Function

Minimize the Tardiness of the work orders of all the products

Min $\sum_{i} \sum_{j} T_{ij}$

Constraints

1. (Tardiness Definition Constraint)

The Tardiness of a work order should be greater than the difference between the Completion time and the modified due date of the work order.

$T_{ij} \geq C_{ij} - d'_{ij}$ i = 1, 2,3,…. j = 1,2,3,….

2. (Tardiness Positive Value Constraint)

The Tardiness of a work order should be a positive value

$T_{ij} \geq 0$ i = 1, 2,3,…. j = 1,2,3,….

3. (Completion Time Constraint)

The completion time of a work order j of product i is equal to the time period t in which the last lot of the work order j is released into the system.

$C_{ij} \geq t\, X_{ijt}$ i = 1, 2,3,…. j = 1,2,3,…. t = 1, 2,3,….,T

4. (Maximum Input Rate Constraint for each Product)

The number of lots of a work order j of product i released into the system in time period t is less than or equal to the maximum number of lots of product i which can be processed by the system in time period t.

$$p_{ijt} \le B_{it} X_{ijt} \qquad i = 1, 2,3,\ldots \qquad j = 1,2,3,\ldots \qquad t = 1, 2,3,\ldots,T$$

5. (Completion of each Work Order Constraint)

The sum of all the lots of a work order released into the system over the planning horizon is equal to the total number of lots in the work order.

$$\sum_{t=1}^{T} p_{ijt} = k_{ij} \qquad i = 1, 2,3,\ldots \qquad j = 1,2,3,\ldots$$

6. (Capacity constraint)

The total number of lots, over all the work orders, released into the system should satisfy the capacity of the system.

$$\sum_{i} \sum_{j} p_{ijt} * r_{im} * s_{im} \le h_t * n_m \qquad m = 1, 2,3,\ldots \qquad t = 1,2,3,\ldots,T$$

7. (Bottleneck Constraint)

The maximum number of lots of a product i that can be processed in a single time period, t, is equal to the maximum number of lots processed by the bottleneck station of the product.

$$M_{mit} = \left\lfloor \frac{h_t * n_m}{s_{im} * r_{im}} \right\rfloor \qquad i = 1, 2,3,\ldots \qquad m = 1,2,3,\ldots \qquad t = 1, 2,3,\ldots,T$$

$$B_{it} = \min_{m} M_{mit} \qquad i = 1, 2,3,\ldots \qquad t = 1,2,3,\ldots\ldots T$$

$B_{it} = B_i$ and $h_t = h$ (from section 3.2.2.5)

8. (Upper Bound on Completion Time of a Work Order Constraint)

An upper bound on time required for completing a work order j of product i,

$$U_{ij} = \left\lceil \frac{k_{ij}}{B_i} \right\rceil \qquad i = 1, 2,3,\ldots. \qquad j = 1,2,3,\ldots.$$

9. (Lower Bound on Completion Time of a Work Order Constraint)

A lower bound on time required for completing a work order j of product i,

$$L_{ij} = \left\lfloor \frac{k_{ij}}{B_i} \right\rfloor \qquad i = 1, 2,3,\ldots. \qquad j = 1,2,3,\ldots.$$

10. (Planning Horizon Constraint)

The maximum number of time periods to release all the lots in the system.

$$T = \sum_i \sum_j U_{ij} \qquad i = 1, 2,3,\ldots. \qquad j = 1,2,3,\ldots.$$

11. $T_{ij}, C_{ij}, p_{ijt}, M_i, U_{ij} \geq 0$; Integer

12. $X_{ijt} = 0, 1$

This completes the formulation.

The tardiness problem, 1 || ΣT_j is NP-Hard. Similarly, the above IP problem for the m-machine tardiness problem is also NP-Hard and cannot be solved in polynomial time. Hence, we need to develop a heuristic that can give us a good solution in a reasonable amount of time.

3.3 METHODOLOGY TO SOLVE THE PROBLEM OF MINIMIZING THE TARDINESS OF WORK ORDERS PRESENT IN A WAFER FAB

We solve this problem in three distinct phases. In the first phase, we apply the Wilkerson-Irwin algorithm [4] for minimizing the tardiness of the $1 \,||\, \Sigma T_j$ problem to the $m \,||\, \Sigma T_j$ problem of the wafer fab by appropriately modifying the system. In the second phase, we improve upon the solution obtained in Phase 1. In the third phase, we feed the solution from Phase 2 back to the Integer problem and add further constraints derived from the classical scheduling theory, to the Integer problem, and solve the Integer problem using the CPLEX optimization solver. The iterations are terminated once a good solution is obtained in a reasonable amount of computation time. The definition of a good solution and the maximum allowable computation time is based on the experimental study. Each of these phases is discussed next.

3.3.1 Phase 1

The Wilkerson-Irwin (W-I) algorithm is a heuristic procedure to minimize the tardiness of a single machine problem. We extend this heuristic procedure to the parallel machine environment of a wafer fab in order to obtain a good initial solution. A good initial solution helps the integer problem to converge to near optimal solutions faster in Phases 2 & 3.

To extend the W-I algorithm, we consider an aggregated version of the parallel machine environment of a wafer fab as follows:

- First, the wafer fab is assumed to devote its manufacturing capacity to only one work order j of product i at a time.
- Let t’ be the time period in which the last lot from work order j of product i is released into the system.
- Then, lots from any other work order j’ of any product i’ are released into the system only after time period t’ i.e. (from time period t’+1,… T).

- Starting from time period t'+1, all the lots from work order j' of product i' are released into the system in each successive time period until and including the time period $(t'+1) + U_{i'j'} - 1$.
- In each time period $t = t'+1, t'+2, \ldots, (t'+1) + U_{i'j'} - 2$, we release $p_{ijt} = B_{it}$ number of lots into the system.
- In the time period $t = (t'+1) + U_{i'j'} - 1$, the number of lots released into the system, $p_{ijt} \leq B_{it}$.
- Thus, the system processes one work order after the other.
- Each work order j of product i can be considered as a single job with processing time U_{ij}.
- The entire wafer fab can be considered as a single machine that processes jobs with processing times U_{ij}.

Now, the tardiness problem for the wafer fab can be considered as a single machine, n job tardiness problem on which the W-I algorithm can be applied.

The wafer fab will process only one work order at a time. There may exist sufficient capacity in the system to simultaneously process lots from work orders of other products. This capacity of the system is not utilized in Phase 1. Hence, the solution that is provided by the W-I algorithm for the aggregated version of the wafer fab will have idle capacity in the system. This issue is addressed in Phase 2 of the problem.

The solution methodology of Phase 1 is summarized as follows:

Step 1. Consider the entire manufacturing system as a single machine.
Step 2. Consider each work order as a single job.
Step 3. Obtain the modified due date d'_{ij} and upper bound on the completion time of an order U_{ij} as per section 3.2.2.2 and constraint 8 respectively.
Step 4. Consider U_{ij} as the processing time of a job on the single machine (system).
Step 5. The W-I algorithm can be applied to this environment to minimize the tardiness of the reduced problem.
Step 6. After applying the W-I algorithm, we get the sequence of the jobs (work orders) to be scheduled on the single machine (entire system).

Step 7. The work orders are ranked according to the sequence generated by the W-I algorithm.

3.3.2 Phase 2

- The W-I algorithm will rank the jobs ij (work order) on the single machine (wafer fab).
- Each job i'j' will begin processing after the completion of the previous job ij.
- The completion time t' of the job ij is the time period in which the last lots of the work order ij are released into the system (from Proposition 3.1).
- The next job i'j' will start processing only in the time period t'+1.
- There may be capacity available in time period $t \leq t'$ for processing lots from other work orders ij, but this capacity is not utilized.
- Hence, the solution that is provided by the W-I algorithm for the aggregated version of the wafer fab would be a "loose" solution with many capacity "gaps".

The capacity gaps will occur in the time periods in which lots from work order ij are being released into the system according to the W-I algorithm, but there may exist sufficient capacity to release lots from some other work order i'j'. We remove these capacity gaps in Phase 2 of the solution methodology. Note that,

- Lots from work orders of different products can be released into the system in the same time period while guaranteeing stability of the system.
- Simultaneous release of lots will lead to a decrease in the total numbers of time periods required to release all the lots of a work order as compared to an individual release of lots.
- This can be considered as further "compression" of the schedule generated by the W-I algorithm.
- This will lead to decrease in completion times of the work orders and a corresponding decrease in the tardiness of the work orders.
- It is intuitive that this "compression" of the schedule will lead to a much better objective function value than the one provided by the W-I algorithm in Phase 1.

“Compression” Heuristic to eliminate idle capacity

The following heuristic is employed to improve the solution obtained from the W-I algorithm used in Phase 1.

<u>Step 1</u>. (Initialization) Rank the work orders in ascending order according to the schedule generated by the W-I algorithm in Phase 1.
<u>Step 2</u>. In the 1st time period, take the 1st ranked job (work order j) and release the maximum number of lots (B_i) for the product i of the work order j into the system.
<u>Step 3</u>. Go through the schedule, and select the least ranked work order i’j’ over all the remaining work orders.
<u>Step 4</u>. Determine the feasibility of releasing lots ($p_{i'j'1} \leq B_{i'}$) of the work order i’j’ into the system such that the system stability is not violated (system capacity is not exceeded). If system stability is not violated, then release as many lots as possible from that work order into the system.
<u>Step 5</u>. Repeat steps 3 and 4 until all the work orders of every product are evaluated for releasing lots into the system such that the system capacity is not exceeded.
<u>Step 6</u>. Go to Step 7. Set t = 2.
<u>Step 7</u>. For the tth time period (t = 2, 3, 4, …, T), release as many lots as possible ($p_{i''j''t} \leq B_{i''}$) of the least ranked work order j’’ of product i’’ in the system.
<u>Step 8</u>. Repeat steps 3, 4, and 5.
<u>Step 9</u>. Go to Step 10. Set t = t+1.
<u>Step 10</u>. Repeat step 7,8 and 9 until time period T or when no lot of any work order remains to be released in the system.

3.3.3 Phase 3

The solution from Phase 2 is used as a starting solution for the integer problem. This solution can be improved further by using an integer-programming solver. We use the CPLEX 6.6 optimization software to solve the integer problem. The tardiness problem is NP-Hard and the time required for the problem to provide the optimal solution is not bounded by a polynomial in the size of the problem.

Additional constraints are imposed on the integer problem, so that the formulation can be tightened and a good solution can be obtained in a relatively small computation time.

1. Dominance Property

We use the following Lemma derived from classical scheduling theory [26].

"If $p_j < p_k$ and $d_j < d_k$, then there exists an optimal sequence in which job j is scheduled before job k".

This Lemma as applied to the present integer problem will be

"If $U_{ij} < U_{i'k}$ and $d_{ij} < d_{i'k}$, then there exists an optimal sequence in which work order j is completed before work order k, $\forall$ i, i'".

This lemma defines a dominance property (relationship) between different work orders. This property is incorporated in the integer problem as

$C_{ij} \leq C_{i'j'}$ $\forall$ ij and i'j' satisfying the above Lemma and ij $\neq$ i'j'

2. Lower Bound on Completion Time

The second set of constraints are formulated by Constraint Logical Programming

The W-I algorithm provides us with rankings for all work orders. It also gives sequential rankings between the work orders of the same product. We know that no work order can be completed before the lower bound on the time required for completing the work order L_{ij}. We use the rankings from the W-I algorithm and these lower bounds to formulate lower bounds for the completion times of the work orders.

$C_{ij} \geq L_{ij} + C_{i j}''$ i = 1, 2,3,…. j = 1,2,3,….

Where $C_{i j}'' = 0$ when ij is the 1st ranked work order in a product,

$= \Sigma L_{ij}$ of all previously ranked work orders of the same product.

3. Upper Bound on Completion Time

The third set of constraints provides an upper bound on completion times of work orders.

$C_{ij} \leq C_{ij}'$ where C_{ij}' is the tentative completion time provided by the W-I algorithm.

These sets of constraints tighten the problem further and help the problem to converge to a good solution faster.

The integer problem is now solved using an integer programming software. The problem converges to a very good solution within a reasonable amount of computation time even when not reaching to the optimal solution. The solution obtained is used to determine the lot release policy into the system. We now show the efficacy of the solution methodology by solving a numerical example.

3.4 NUMERICAL EXAMPLE

The wafer fab system for the numerical system is summarized below. The workstation data and product routes are provided in Appendix A.

6 Products	10 Station Families	25 Work stations	One time period = 24 hrs

The data for the work orders of the 6 products is provided in Table 3.1. For each work order, the product number is provided along with the due date and the number of lots in the work order. The upper bound for completing the work order is also computed using the formula from constraint 8 of the integer problem in section 3.2.3.

Table 3.1 Data for the work orders of the 6 products

Product Number	**Order Number**	**Number of lots in the order**	**Upper bound on time for completing the work order**	**Due date of the order**
i	**j**	$\mathbf{k_{ij}}$	$\mathbf{U_{ij}}$	$\mathbf{d'_{ij}}$
1	1	100	9	30
1	2	250	21	100
1	3	150	13	180
1	4	150	13	260
2	1	150	19	50
2	2	100	13	120
2	3	100	13	200
3	1	150	19	60
3	2	250	32	130
3	3	100	13	210
3	4	100	13	280
4	1	250	21	70
4	2	200	17	170
5	1	100	12	60
5	2	150	17	130
5	3	100	12	230
5	4	100	12	280
6	1	150	13	40
6	2	100	9	150

In Table 3.2, the maximum number of lots of a product that can be released into the system in a single time period is computed using the formula from constraint 7 of the integer problem in section 3.2.3.

Table 3.2 Maximum number of lots of a product that can be released into the system in a single time period

Product	Maximum number of lots released in a single time period (B_i)
1	12
2	8
3	8
4	12
5	9
6	12

3.4.1 Phase 1

The W-I algorithm is applied to the work orders by considering the completion time and the due date of each work order. The tentative lot release policy is provided in Table 3.3.

Table 3.3. Tentative lot release policy produced by the W-I algorithm

Rank	Product Number	Order #	# of lots in the order	Processing Time	Due date of order	Completion Time	Tardiness
r	i	j	k_{ij}	U_{ij}	d'_{ij}	C_{ij}	T_{ij}
1	1	1	100	9	30	9	0
2	6	1	150	13	40	22	0
3	2	1	150	19	50	41	0
4	5	1	100	12	60	53	0
5	3	1	150	19	60	72	12
6	4	1	250	21	70	93	23
7	1	2	250	21	100	114	14
8	2	2	100	13	120	127	7
9	5	2	150	17	130	144	14
10	6	2	100	9	150	153	3
11	4	2	200	17	170	170	0
12	1	3	150	13	180	183	3
13	2	3	100	13	200	196	0
14	3	3	100	13	210	209	0
15	5	3	100	12	230	221	0
16	3	2	250	32	130	253	123
17	1	4	150	13	260	266	6
18	5	4	100	12	280	278	0
19	3	4	100	13	280	291	11

The rank r for any work order j of a product i, is the rank computed by the W-I algorithm. The product number, the work order number, the number of lots in the order, the processing time, and the due date for the work order are obtained from Table 3.1. The completion time for the work order is computed by using the processing time for each work order and its rank determined by the W-I algorithm in Phase 1. The time required to release all the lots into the system is 291 days.

The tardiness value for each work order of each product is obtained by comparing the due date and the completion time of the work order. The Total Tardiness value at the end of Phase 1 is $\Sigma T_{ij} = 216$ days. This solution obtained in Phase 1 is improved by the "compression" heuristic in Phase 2.

3.4.2 Phase 2

In phase 2, the "compression" heuristic is applied on the solution obtained in phase 1. The detailed implementation of the lot release policy developed in phase 2 is provided in Appendix A and is summarized in Table 3.4

The rank r for any work order j of a product i, is the rank computed by the compression heuristic. The product number, the work order number, the number of lots in the order, the processing time, and the due date for the work order are obtained from Table 3.1. The completion time for the work order is computed by using the processing time for each work order and its rank determined by the compression heuristic in Phase 2. The time required to release all the lots into the system is 271 days compared to the 291 days in Phase 2. This suggests that Phase 2 can improve the solution from Phase 1.

The tardiness value for each work order of each product is obtained by comparing the due date and the completion time of the work order. The Total Tardiness value at the end of Phase 2 is $\Sigma T_{ij} = 151$ days compared to the 216 days from Phase 1. Thus the solution from Phase 2 is definitely better than the solution from Phase 1.

Finally, Phase 3 improves the solution obtained from Phase 2.

Table 3.4 Summary of lot release policy produced by the Compression Heuristic

Rank	**Product Number**	**Order #**	**# of lots in the order**	**Due date of order**	**Completion Time**	**Tardiness**
r	**i**	**j**	**k_{ij}**	**d'_{ij}**	**C_{ij}**	**T_{ij}**
1	1	1	100	30	9	0
2	6	1	150	40	21	0
3	2	1	150	50	40	0
4	5	1	100	60	51	0
5	3	1	150	60	70	10
6	4	1	250	70	87	17
7	1	2	250	100	108	8
8	2	2	100	120	121	1
9	5	2	150	130	137	7
10	6	2	100	150	142	0
11	4	2	200	170	158	0
12	1	3	150	180	171	0
13	2	3	100	200	183	0
14	3	3	100	210	196	0
15	5	3	100	230	207	0
16	3	2	250	130	238	108
17	1	4	150	260	247	0
18	5	4	100	280	259	0
19	3	4	100	280	271	0

3.4.3 Phase 3

The dominance property between different work orders is determined as per the lemma defined in section 3.3.3. The lower bound and upper bounds on the completion times are also established from the W-I algorithm.

The integer problem is input into the integer programming software CPLEX 6.6. The tardiness problem is a NP-Hard and may not converge to optimality in a tractable amount of time. Hence, we terminate the solution procedure when we do not see any improvement in the objective function for a time period that is 200% of the time required for obtaining the incumbent integer solution. We also set the minimum run time of the optimization solver to 30 minutes.

In Table 3.5, we provide the results of the tests done on the above problem, which support the efficacy of the 3-phased solution approach. The problem is solved under 6 different cases and the results for each different case can be compared.

In the first case, the integer problem is solved using CPLEX, and without using the 3 phase methodology. We observe that the value of the integer solution obtained is quite large. In the 2nd case, the solution obtained by using phase 1 of the 3-phase methodology is inserted as an upper bound on the solution. This does not lead to any improvement in the solution quality.

In the 3rd case, the solution obtained by using phase 2 of the 3-phase methodology is inserted as an upper bound on the solution. This leads to a drastic improvement in the solution quality from the 1st two cases. In the 4th case, the dominance property constraints from phase 3 are incorporated in addition to the 3rd case. This leads to a further improvement in the solution value.

In the 5th case, the lower bound constraint from phase 3 is incorporated in addition to the 4th case. This improves the solution quality. Finally, the best solution value is obtained under the 6th case when all the constraints of the phase 3 are incorporated in addition to the 3rd case.

This shows that each additional constraint in the 3-phase methodology improves the solution. Hence, we can say that each constraint of the 3-phase methodology is required to improve the solution quality and obtain a good solution.

Table 3.5 Results for efficacy of the 3-phased solution approach

Case #	Additional Constraints in the Integer Problem	Time for LP Solution	Present Integer Solution	Time for Present Integer Solution	Node at which Incumbent Integer Solution obtained	Time Branch and Bound terminated	Node at which Branch and Bound terminated
		(sec)		(sec)		(sec)	
1	None	121.16	1045	-	-	45222.64	26267
2	Phase 1 Solution as Upper Bound	123.72	-	-	-	35236.95	84132
3	Phase 2 Solution as Upper Bound	123.02	120	10704.9	9808	10740.99	10047
4	Phase 3, 1st constraint	97.02	114	11005.3	4461	11222.67	4477
5	Phase 3, 1st and 2nd constraint	88.12	102	9575.04	4100	10596.05	6192
6	Phase 3, 1st, 2nd and 3rd constraint	9.31	99	386.97	1892	2021.71	8000

The detailed implementation of the lot release policy developed in phase 3 is provided in Appendix A and is summarized in Table 3.6

The completion time for any work order j of a product i, is the completion time provided by the integer solution. The product number, the work order number, the number of lots in the order, the processing time, and the due date for the work order are obtained from Table 3.1. The rank r for the work order is computed by using the completion time for the work order. Earlier the completion time of a work order, lower is the rank of that work order. The time required to release all the lots into the system is 253 days compared to the 271 days in Phase 2. This suggests that Phase 3 can improve the solution from Phase 2.

The tardiness value for each work order of each product is obtained by comparing the due date and the completion time of the work order. The Total Tardiness value at the end of Phase 3 is Σ T_{ij} = 99 days compared to the 151 days from Phase 2. Thus the solution from Phase 2 is definitely better than the solution from Phase 1.

Table 3.6 Summary of lot release policy at the end of phase 3

Rank	**Product Number**	**Order #**	**# of lots in the order**	**Due date of order**	**Completion Time**	**Tardiness**
r	**i**	**j**	**k_{ij}**	**d'_{ij}**	**C_{ij}**	**T_{ij}**
1	1	1	100	30	9	0
2	6	1	150	40	22	0
3	2	1	150	50	41	0
4	5	1	100	60	53	0
5	3	1	150	60	72	12
6	4	1	250	70	83	13
7	1	2	250	100	110	10
8	2	2	100	120	122	2
9	5	2	150	130	135	5
10	6	2	100	150	149	0
11	4	2	200	170	170	0
12	1	3	150	180	183	3
13	3	2	250	130	184	54
14	2	3	100	200	196	0
15	3	3	100	210	209	0
16	5	3	100	230	221	0
17	1	4	150	260	241	0
18	5	4	100	280	249	0
19	3	4	100	280	253	0

Table 3.7 summarizes the solutions obtained at the end of each phase. The total tardiness and the completion time for all the work orders at the end of each phase are compared. It is clearly seen that the solution from Phase 3 is better than the solution from Phase 2, which in turn is better than the solution from Phase 1.

Table 3.7 Total Tardiness and Completion time for each phase for 6-product problem

Phase	**Total Tardiness**	**Completion Time**
1	216	291
2	151	271
3	99	253

The optimal Total Tardiness value for this problem = 95 days. The following settings were fixed in CPLEX 6.6 for the branch and bound procedure by which the optimal solution was found.

- SET MIP STRATEGY BACKTRACK = 1 (MORE DEPTH EXPLORED IN BRANCH)
- SET MIP STRATEGY NODESELECT = 2 (BEST ESTIMATE SEARCH)
- SET MIP STRATEGY VARIABLESELECT = 2 (PSEUDO COSTS)
- SET MIP TOLERANCES OBJECTIVE VALUE DIFFERENCE = 1
- SET MIP TOLERANCES UPPER CUT OFF = 151 (from Phase 2)

Two, three, four, and five product problems were also solved using the 3-phase solution methodology. The data and detailed lot release policy for the 2-product problem is provided in Appendix B. Data for the three, four, and five product problems is provided in Appendix E. Tables 3.8, 3.9, 3.10, and 3.11 below summarize the solutions obtained for each problem at every phase. The total tardiness and the completion time for all the work orders at the end of each phase are compared. It is clearly seen that, for each problem, the solution from Phase 3 is better than the solution from Phase 2, which in turn is better than the solution from Phase 1.

Table 3.8 Total Tardiness and Completion time for each phase for 2-product problem

Phase	**Tardiness**	**Completion Time**
1	140	162
2	105	155
3	105	155

Table 3.9 Total Tardiness and Completion time for each phase for 3-product problem

Phase	**Tardiness**	**Completion Time**
1	72	165
2	44	144
3	28	136

Table 3.10 Total Tardiness and Completion time for each phase for 4-product problem

Phase	Tardiness	Completion Time
1	73	214
2	19	188
3	13	187

Table 3.11 Total Tardiness and Completion time for each phase for 5- product problem

Phase	Tardiness	Completion Time
1	367	203
2	327	197
3	327	197

Table 3.12 shows the efficacy of the 3-phase solution approach for each problem. The number of products, machines, and work orders establish the problem size. The tardiness per work order for the three-phase solution and the optimal solution is compared. The time required to obtain the three-phase solution and the optimal solution are also compared.

For each example we observe that the tardiness per work order from the 3-phase solution is only 0-2 days more than the tardiness per work order from the optimal solution. Only for the example with 3 products, this difference is 2 days. The difference for all other examples is 0-0.2 days. Thus, the 3-phase approach provides a solution that is comparable to the optimal solution for solution quality.

The biggest advantage of the 3-phase approach is the computation time. From Table 3.12, we observe that the computation time required for the optimal solution is at least 22 times the computation time required for the 3-phase approach. In the example with 5 products, the optimal solution is not obtained even after 10 hours of computational time. The Tardiness problem is strongly NP-Hard. Hence, the computation time required to obtain the optimal solution is very large compared to the computation time required for 3-Phase solution methodology.

A trade-off exists between the solution quality and the computation time required for obtaining the solution for the tardiness problem.

From the above three paragraphs we can say that the 3-phase solution methodology can provide good solutions within a reasonable computation time.

Table 3.12 Comparison between the Solution Quality and Computation Time for the 3-Phase solution methodology, and the Optimal Solution and the Computation time for the Optimal Solution to the Total Tardiness Problem

Number of Products	**Number of Machines**	**Number of Work Orders**	**3-phase approach Tardiness per Work Order**	**Optimal Solution Tardiness per Work Order**	**Time for 3-phase Solution (a)**	**Time for Optimal Solution (b)**	**Ratio of Solution Time**
			(days)	**(days)**	**(sec)**	**(sec)**	**a/b**
6 product	25	19	5.2	5.0	386.97	106309.34	274
5 product	27	17	19.2	*	1648.85	-	-
4 product	24	13	1.0	0.8	103.33	110646.26	1074
3 product	21	11	2.5	0.5	88.13	5494.55	62
2 product	18	9	11.6	11.6	5.47	123.46	22

* - Optimal solution not obtained after 36,000 seconds (10 hours) of computation time.

3.5 MODIFICATIONS IN THE TARDINESS PROBLEM AND THE SOLUTION METHODOLOGY TO INCORPORATE REAL-LIFE SITUATIONS

3.5.1 Weighted Tardiness

The problem formulation presented in this chapter minimizes the total tardiness of the work orders. The penalty for the tardiness of each work order is assumed to be equal. Often, the tardiness penalty is different for each work order. These penalties depend upon the importance of the customer, upper management commitments and priorities, and the monetary penalties associated with the tardiness. Thus, a problem minimizing only the total tardiness may not be applicable when the tardiness penalties are different. Hence, a weighted tardiness problem should be formulated.

The integer-programming model has to determine the number of lots, of each work order, to be released into the system, in each planning period, over the entire planning horizon, so that the total weighted tardiness of the work orders is minimized. The only change in the total weighted tardiness problem from the total tardiness problem is in the objective function.

Objective Function

Minimize the Weighted Tardiness of the work orders of all the products

$$\text{Min} \sum_i \sum_j w_{ij} T_{ij}$$

Subject to the constraints formulated in the total tardiness problem from section 3.2.3.

The solution methodology to the total weighted tardiness problem can be similar to the solution methodology of the total tardiness problem. In Phase 1, an algorithm to minimize the total weighted tardiness for an n-job, single machine tardiness problem can be applied to the aggregated version of the fab. In Phase 2, the “compression” heuristic can be used to fill the capacity gaps. Dominance properties from scheduling theory, and lower and upper bounds derived from the Phase 1 solution can be added to the model in Phase 3 and the problem can be solved by an integer programming solver.

3.5.2 Dynamic Arrival of Work Orders

The tardiness problem is solved for the fixed work orders on hand. The planning horizon over which the tardiness problem is solved is sufficient to process all the work orders. The management may consider accepting a work order in the middle of the planning horizon. We term this phenomenon as the dynamic arrival of work orders. These work orders may have a priority higher than or equal to the priority of the existing work orders. We briefly discuss a framework for accommodating these two situations.

3.5.2.1 Dynamic Arrival of Work Orders with Regular Priority

The tardiness problem is to be re-solved using the current status of the existing work orders and the information about the new work orders. The solution to this revised problem will provide a new lot release policy that accommodates the new work orders. Thus, the completion times of each work order and, hence the tardiness of each work order is affected. The management has to consider these revised tardiness values and hence customer satisfaction before making a decision on whether to accept the additional work orders.

3.5.2.2 Dynamic Arrival of Work Orders with High Priority

At times, management may decide to accept new work orders because of government contracts, satisfaction of prized customers, or get ahead of the competition. In these instances, the work orders have priorities higher than the existing work orders. The lots of these work orders are called ‘hot lots’.

From the moment the high priority work orders are accepted, the lot release schedule for the existing work orders is stopped. The hot lots are released into the system in each time period at a rate determined by the bottleneck station for the hot lots, until there are no more hot lots to be released into the system. Thus, the lot release schedule for the existing work orders is delayed by the number of time periods required to release all the lots of the work order with a higher priority. Hence, the completion times and tardiness of the existing work orders are increased.

3.6 CONCLUSIONS

1. An integer-programming model is formulated for determining the lot release policy into the system such that the total tardiness over all the work orders is minimized.
2. This tardiness problem is NP-Hard, like all the tardiness problems in literature.
3. A 3-phase solution methodology is proposed for this problem.
4. This methodology provides us with an acceptable solution to the problem with a relatively small amount of computational effort.
5. The quality of the solution obtained varies with the problem data as is shown by the examples presented.
6. The computation time required for obtaining the optimal solution to the tardiness problem is very large as compared to that required for obtaining a good solution from the 3-phase solution methodology.
7. This tardiness problem is amongst the very few of its kind attempted. (M-parallel machines, n-jobs, each job with its own route) and provides a fascinating challenge in combinatorial optimization.
8. Further research can be conducted on this problem to obtain better solutions with the least amount of computational effort. Special structures, if any, in the problem can be exploited to improve the computational performance and solution quality of the solution methodology.
9. The problem formulation and the solution methodology can be modified to accommodate real-life features such as different tardiness penalties for each work order, and dynamic arrival of regular and high priority work orders.
10. This solution of this problem provides us with the policy for lot release into the system. This leads us to the next step in our research i.e. the scheduling of the lots on the machines so that the cycle time of the jobs is minimized and the throughput is increased.
11. The scheduling problem is explored further in the next 2 chapters.

CHAPTER 4: SCHEDULING OF JOBS IN A WAFER FAB

4.1 INTRODUCTION

The previous chapter addressed the problem concerned with the release of lots into the system so that the average tardiness of the work orders in the fab is minimized. The lots are released into the system such that the system is not overloaded, and remains stable. At the same time, the system must not be under-utilized and as many lots as possible are released into the system during each period.

Once the lots have been released into the system, the main task is the sequencing and scheduling of these various lots at the workstations. In a wafer fab, due to the reentrant nature of the flow, lots come back to the same station family more than once in their route. This is particularly true for the photolithography processing area. Wafers have as many as eighteen layers of circuitry and, hence, they wind up back to the photolithography machines as many as seventeen times.

Thus, in a wafer fab, lots of the same products, at different stages of completion, are part of the WIP in front of a station. Moreover, frequently, multiple products exist in a wafer fab environment at the same time. The task of the scheduler is to choose a lot to be processed next from the WIP of a station, such that

- No stations downstream or upstream of this station are unnecessarily idle.
- WIP does not build up beyond control at any station.
- The queue time of a lot (hence the cycle time) is decreased.

This summarizes the complex nature of wafer fab scheduling.

The wafer fab scheduling problem has received a lot of interest in the past decade. Most of the research has been focussed on developing heuristics, which work well in the particular environment for which they have been developed. No globally dominant rules for selecting lots from the WIP have been developed, as yet, from these heuristics.

Mathematical programming techniques have not been pursued for the scheduling of jobs in a wafer fab. The main reason for the lack of progress in this approach has been the NP-Hardness of most of the scheduling problems. However, a major advantage of this approach is that it generates optimal solutions to the problem, which will definitely be better than the solutions provided by heuristics. Near-optimal solutions obtained by exploiting special structures in these mathematical models may also provide much better solutions than the heuristics.

In the subsequent part of this chapter, we develop an integer-programming model for the scheduling of jobs in a wafer fab. We explain the "flow" approach behind the formulation and the solution methodology of the problem. We present two variations of the model: (1) – when the steps to be processed in a single time period have been determined apriori and (2)– when the steps to be processed in a single time period are not determined apriori. Subsequently, a tighter LP relaxation of the integer program leading to a better performance of the branch and bound method is developed.

In Chapter 5, we present a heuristic based on the Lagrangian relaxation approach for obtaining near-optimal solutions to the scheduling problem.

4.2 FLOW APPROACH

The WIP moving through a wafer fab can be considered as a fluid moving through the system [37]. The rate of fluid entering the system should be less than or equal to the rate of fluid moving through the system. This condition ensures stability of the system; i.e. it ensures that the rate of input of the lots into the system is sufficient enough not to cause excessive build-up of WIP in the system. This condition was incorporated as the capacity constraint in the model, formulated in chapter 3, for determining the lot release policy.

Restricting the input of fluid into the system is not sufficient to guarantee the stability of the system. The rate of flow of fluid through the system should also be maintained so that there is no pile up of fluid at any point in the system. Thus, the entire capacity of the system should be utilized to the maximum extent possible, so that, there is no excessive pile up of WIP at any station. Hence, the sequencing, scheduling and processing of jobs at various stations should be such that the WIP moves along smoothly in the system and there is no excessive build up of lots at any station.

The concept of cyclic scheduling of jobs advocated by Graves et al [16] states that, during each cycle, a shop should perform all of the tasks required to complete a job, even though they are performed on different jobs. Hence, in a cycle, each facility should do a task assigned to it only once. Thus, during each cycle one job will be completed. Each cycle is repeated throughout the planning period. Thus, a number of cycles are completed in each planning period. The number of jobs completed in a planning period is equal to the number of cycles repeated in the planning period.

Extending this concept to a single planning period in our model, we can state that:
“In a single planning period, the fab should perform all the processing steps belonging to the lots that are fed into the system, although possibly on lots other than the ones that were released into the system in planning period”.

Thus, in a planning period, each station family should complete all the processing steps assigned to it for all the lots entering the system in that planning period, although the steps may be performed possibly on different lots.

This essentially means that, in a single planning period and at each station family, lots should be selected from the WIP such that all the processing steps, assigned to the station family for the lots entering the system in that planning period, are scheduled. Thus, this policy determines the lots from the WIP to be scheduled at each station family in every planning period.

By limiting the release of lots into the system in chapter 3, the system is not overloaded in any planning period. At the same time, the implementation of the above lot selection policy for the WIP, ensures that the system is not underutilized and the WIP keeps on flowing through the system at a constant rate. This flow of WIP continues from stage to stage, i.e. from one time period to the succeeding time period.

This is the basic concept behind the “Flow” Approach.

The application of this flow concept to the static or dynamic lot release policies that may prevail in a wafer fab is explained below.

4.2.1 Flow Approach for a static lot release policy.

- Under the static lot release policy, in each time period, the same product mix is released into the system, that is, for a product the same number of lots are released into the system in each time period.
- According to the flow approach, at every station family, lots from the WIP are scheduled so that all the steps assigned to the station family for the product mix are completed in a single time period.
- Thus, in every planning period, at every station family, the same product mix of lots from the WIP is scheduled.
- The steps that need to be processed in a single planning period are known.
- There exists no need to schedule any more number of steps in a single planning period than what is determined above because the system capacity is being sufficiently utilized.
- Thus, the rate of flow of WIP is constant is every planning period.
- Also the inflow of lots is constant in every planning period.

- Thus, the scheduling problem is cyclic in nature for the static lot release policy.
- In every planning period, the same scheduling problem is solved.
- It is quite intuitive to see that, as a result, the type of WIP in front of every station family is the same at the beginning of every planning period.
- If the WIP and workstation availability data is the same at the beginning of every planning period, the scheduling problem is solved only once and the solution is repeated over every planning period.

The first model that we formulate is the scheduling problem for a wafer fab under the static lot release policy.

Proposition 4.1

Application of flow approach to the scheduling problem for a wafer fab under the static lot release policy completely determines the processing steps of the lots to be scheduled in a planning period.

Proof:

Consider a product f with a lot release rate of n_f lots per planning period. Let the number of processing steps in the route of a lot i of product f be J_{fi}. To maintain the stability of the system, the output rate of each product must be equal to the input rate of that product. Hence, n_f lots of the product f must be completed in the given planning period.

Thus, n_f lots of the product must complete the $J_{fi}{}^{th}$ processing step. To maintain the flow of the WIP, n_f lots of the product must complete the J_{fi}-1^{th} processing step. By induction and by proceeding one step backward through the route, n_f lots of the product f must complete each of the steps in the route of the product f. The n_f lots scheduled at each of the processing steps are the earliest lots from product f available for processing at those steps.

Hence, for a given product f, the earliest n_f lots which arrive at each of the processing steps in the route of the product f should be scheduled. This process is repeated for each of the products released into the system.

This leads to the establishment of the lots that need to be processed at each processing step of each product. QED

In the scheduling problem under the static lot release policy, only those processing steps that should be processed in the planning period are included. Thus, at each workstation, only those lots that should affect the scheduling decision are considered. The problem size is also reduced because only the relevant processing steps are incorporated in the problem.

Proposition 4.2

For a given product f, the earliest n_f lots which arrive at each of the processing steps in the route of the product f should be scheduled at that processing step.

4.2.2 Flow approach for a dynamic lot release policy

According to the lot release policy problem in chapter 3, the lot release policy is never static. The product mix of the lots released into the system is not constant in every planning period. This is the most likely scenario in a real life wafer fab. Hence, we need to study the change, if any, in the flow approach for a dynamic lot release policy.

- For a dynamic lot release policy, the WIP in the system is different in each planning period. The product mix of the WIP in a time period may be different from the product mix of the lots released into the system in that planning period.
- Hence, the lots from the WIP that should be definitely scheduled in a planning period cannot be pre-selected.
- But, the WIP should flow smoothly through the system and the system should not become overloaded due to build up of inventory at any station family.
- Hence, the formulation of the scheduling problem under the static lot release policy is modified for the scheduling problem under the dynamic lot release policy.

In the scheduling problem, the lots and processing steps of those lots that may be scheduled in the planning period should be identified.

The total amount of processing performed on a lot in a planning period cannot exceed the total time available in a planning period. Hence, for all the lots in the system, the steps that can possibly be processed in the planning period can be identified.

Proposition 4.3

Application of flow approach to the scheduling problem for a wafer fab under the dynamic lot release policy completely determines the processing steps of the lots that may be scheduled in a planning period.

Proof:

Let a lot i of the product f be a part of the WIP at the start of the planning period, the next step of the lot that can be scheduled be j_{fi}, the total time available in the current planning period be T, and the last processing step in the route of the lot be J_{fi}.

Set $k = j_{fi}$.

Let the processing time for the lot at step k be p_k. Let the total amount of processing possible on the lot in the planning period be P_{fi}.

Set $P_{fi} = p_k$.

$k = k + 1$ and $P_{fi} = P_{fi} + p_k$.

Repeat while $k \leq J_{fi}$ or $P_{fi} \leq T$.

If $k = J_{fi}$ then all the processing steps from j_{fi} to J_{fi} of the lot i of product f may be scheduled in the current planning period.

If $P_{fi} > T$, then only the processing steps from j_{fi} to k of the lot i of product f may be scheduled in the current planning period.

This process is repeated for all the lots of all the product in the system. This leads to the establishment of all the processing steps over all the lots that may be processed during the current planning period. QED

In the scheduling problem under the dynamic lot release policy, only those processing steps that may be processed in the planning period are included. Thus at each workstation, only those lots that can affect the scheduling decision are considered. The problem size is also reduced because only the relevant processing steps are incorporated in the problem.

4.2.3 Flow approach to determine release point of lots in a single planning period

4.2.3.1 Single Product

With reference to the cyclic scheduling policy for a single product [16], a planning period consists of many cycles. One lot of the product is completed by the system at the end of each cycle. Thus, the number of lots of the product completed by the system in a planning period is equal to the number of cycles in the planning period. The stability of the fab is ensured when the number of lots of a product entering the system equals the number of lots of that product leaving the system in a planning period. At the start of each cycle, one lot of the product enters the system and at the end of the cycle, one lot leaves the system.

Proposition 4.4
Lots of a product are released into the system at equal time intervals given by the ratio of the planning period duration and the number of lots released into the system of that product.

That is, if n is the number of lots of the product released into the system in a planning period with total time duration T, then, the lots will be released into the system at time intervals given by

$$t = (k-1) * \left\lfloor \frac{T}{n} \right\rfloor \qquad k = 1,2,3,\ldots,n$$

4.2.3.2 Multiple Products

In a wafer fab with multiple products, the length of a cycle for each product is different. The number of lots of a product completed by the system is equal to number of cycles of that product in the planning period. The stability of the fab is ensured when the number of lots of a product entering the system equals the number of lots of that product leaving the system in a planning period. Hence, the number of lots of a product entering the system is equal to number of cycles of that product in the planning period. At the start of each cycle one lot of the product enters the system and at the end of the cycle, one lot leaves the system. Thus, lots from the product are released into the system at time intervals that are given by the ratio of the planning period duration and the number of lots released into the system of that product.

That is, if n_f is the number of lots of the product f released into the system in a planning period with total time duration T, then, the lots of a product f will be released into the system at time intervals given by

$$t_f = (k-1) * \left\lfloor \frac{T}{n_f} \right\rfloor \qquad k = 1,2,3,\ldots,n_f$$

4.3 INTEGER PROGRAMMING MODEL FOR SCHEDULING OF JOBS IN A WAFER FAB

4.3.1 Key concepts for the model

4.3.1.1 Motivation

Sarin et al [27] have formulated an integer program for minimizing the makespan in a wafer fab. In other words, the problem is formulated to complete a given set of lots. This implies that the wafer fab is allowed to run dry. But, in actual working conditions, a wafer fab never runs dry. Lots are released into the system and lots exit the system all the time. Hence, a more realistic model of a wafer fab should allow for this dynamic entry-exit of the lots in the system.

4.3.1.2 Dynamic Programming Approach

The mathematical model that we have formulated is solved stage by stage similar to that in the dynamic programming approach. Each stage is a planning period for which we completely solve the scheduling problem. The solution from the previous stage is used as one of the inputs for the succeeding stage. The schedule generated for a stage is followed during the corresponding planning period. The status of the shop at the end of the corresponding planning period is used as input for the next stage. The other input for the next stage will be the information about the lots being released into the system during that stage, and the availability of machines or personnel.

4.3.1.3 Validity of the objective function

The objective function of the model should take into account the stage by stage solution procedure of the problem.

Proposition 4.5

Minimizing the sum of the completion time of those processing steps of any lot that will be processed in the current planning period will minimize the cycle time of the lots.

Proof:

The objective of the scheduling problem is to minimize the cycle time of the lots released into the system. The cycle time of a lot corresponds to the completion time of the last step in the route of the lot. Hence, usually, only the completion time of the last step of the lot is included in the objective function.

But, in the stage by stage solution procedure it may happen that a lot is processed partially in one stage and completed in the next stage. The completion time of the last step will not only depend on the scheduling of its processing steps in the final stage, but also on the scheduling of the steps in the previous stages.

Because the processing steps are in series, the minimization of the completion times of all the processing steps of a lot will minimize the completion time of the last step and hence, the completion time of the lot. QED

The equipment in a wafer fab is highly automated and reliable. Hence, the processing times pertaining to a station family can be assumed as constant.

Completion time of a step of a lot is equal to the sum of the processing time and the beginning time of that step of the lot. Since, the processing time is assumed to be constant, the completion time of a step of a lot can be replaced by the beginning time of the step of that lot. This leads us to the following result,

Proposition 4.6

The objective function can be formulated as “Minimize the summation of the beginning times of the steps of all the lots in the system”.

4.3.1.4 Variables in the objective function

The model uses the start times of each operation of a lot at a particular machine as a binary variable.

4.3.1.5 Precedence constraints between lots from the same work order

Lots from the same work order may be released into the system in different planning periods as per the lot release policy obtained from the solution of the tardiness problem in chapter 3. Also, lots from the same work order released into the system in the same planning period are released at different points of time in that planning period as per section 4.2.3. It is possible that during the flow of WIP through the system, lots from the same work order, at the same stage of completion, may be competing at the same time to be scheduled on the same station family. However, the lot that was released earlier in the system has spent more time in the system than the lot that has been released later.

Proposition 4.7

For lots of the same product at the same stage of completion, the lot released into the system earlier should be scheduled before the lot that was released later.

Proof:

Consider two lots A and B, where Lot A is released into the system before lot B. Lot A has spent more time in the system than lot B.

U_A = Time spent by lot A in the system until now.

U_B = Time spent by lot B in the system until now.

P_t = Processing time at the step where lots A and B are competing

R_t = Processing time for remainder of the routes of lots A and B.

Assume that once a lot has completed the current step, there is no further queuing time for the lot. That is, the lot exits the system after completing the remaining processing steps without any queuing time.

Case 1: Lot B is scheduled before lot A.

The cycle time of lot B

$C_B = U_B + P_t + R_t$

The cycle time of lot A

$C_A = U_A + 2P_t + R_t$

As Lot A was released into the system before lot B

$U_A > U_B$

Consequently, $C_A > C_B$

Average cycle time,

$C_1 = (U_A + U_B + 3P_t + 2R_t)/2$

The difference in the cycle times

$S_1 = C_A - C_B = U_A - U_B + P_t$

Case 2: Lot A is scheduled before lot B.

The cycle time of lot A

$C_A' = U_A + P_t + R_t$

The cycle time of lot B

$C_B' = U_B + 2P_t + R_t$

Average cycle time

$C_2 = (U_A + U_B + 3P_t + 2R_t)/2$

The difference in the cycle times

$S_2 = C_A' - C_B' = U_A - U_B - P_t$

From above, note that the average cycle time is the same, i.e. $C_1 = C_2$, while $S_1 > S_2$

Therefore, the difference in the cycle times is more in case 1 than in case2. In other words, in case 1, the variability in the cycle time is more than the variability in the cycle time in case 2. From above, the scheduling of jobs released earlier in the system over jobs released later in the system, at the same step, helps to reduce the variability in the cycle time. QED.

Apart from the main objective of minimizing the cycle time of the lots, the other objective of scheduling is the minimization of the variability in the cycle times of the lots. Hence, this condition is captured in the model by incorporating precedence constraints for lots of the same product.

Apart from reducing the variability in the cycle time, this constraint also helps the branch and bound procedure, employed in solving the integer program, by preventing it from exploring symmetrical solutions, thus cutting off a branch of the tree and improving the computational efficiency.

4.3.1.6 Time Duration of a stage

A wafer fab is a dynamic manufacturing environment. Machines may breakdown, lots may get rejected due to quality considerations or personnel may not turn up for a shift. All these factors affect the manufacturing capacity of the fab. Hence, when the scheduling problem is solved, the problem should extend into the future time horizon only as much as it is feasible to do so. At the same time, the time horizon should not be so small that the problem has to be solved frequently, thereby incurring unnecessary computational burden. Thus, the time horizon should be chosen such that it strikes a “correct” balance between the above two factors.

For the scheduling problem formulated, the time period for solving the scheduling problem is chosen as one day and hence, the model will schedule the jobs over this period. The implicit assumption behind this is that the shop supervisor and scheduler know the status about the availability of personnel and the machines at the beginning of the day (except during unscheduled downtime). Thus, the shop status is considered as “frozen” over a time period of one day. Therefore, the scheduling generates a solution for the supervisor to use that day.

4.3.1.7 Duration of a time unit in the model

A single planning period in the scheduling problem is equal to twenty-four hours. We can choose a single time unit in the scheduling problem to be one second, one minute or one-hour. Selecting a small time unit such as one second will provide more accuracy to the model. The processing times are never expressed in a time unit smaller than the second. But choosing such a small time unit will mean that we have to solve the scheduling problem over 86400 seconds. This will make the problem computationally intractable. Choosing a larger time unit such as one hour or half an hour will mean that we will be rounding off the processing times thus reducing the accuracy of the model. To forge a compromise between the competing aspects of model accuracy and model tractability, we have chosen the individual time unit in each time period to be 6 minutes. Thus, we have 10 time units in one hour and 240 time units in a single day (planning period). All processing times will be expressed in terms of the time unit of 6 minutes.

4.3.1.8 Lot Dedication Scheme

Lot dedication scheme refers to a policy adopted in the photolithography processing area of certain wafer fabs, where all the layers of a wafer are processed on the same workstation where the first layer of that wafer was processed. This scheme is captured in the model by incorporating lot dedication constraints.

Once the workstation, on which the photolithography processing operation for the first layer of a wafer is performed, is known, only that workstation is considered for the photolithography operation of all the remaining layers of the wafer.

In the scheduling problem of a wafer fab for a single planning period, the photolithography operation on the first layer of the wafer may or may not have been performed. Also, the first layer and some of the successive layers may or may not be performed in the same planning period. Separate constraint formulations for each of the above cases in the scheduling problem under the static and dynamic lot release policies are presented.

4.3.2 Notation

f	A subscript representing the product of the lot to be scheduled
F	Total number of products in the system
i	A subscript representing the lot number
I_f	Total number of lots of product f in the system
j	A subscript representing the processing step of the lot
J_{fi}	Set of processing steps in the route of the lot i of product f that will be processed (under static lot release policy) or may be processed (under dynamic lot release policy) in a single planning period
k	A subscript representing the station family on which the lot is to be scheduled
b_k	Number of workstations in station family k
b	A subscript representing the number of the workstation where the lot is to be scheduled
t	A subscript representing an individual time unit at which the lot is to be scheduled
X_{fijkbt}	= 1, if lot processing step j of lot i of product f is scheduled on station b of station family k at time unit t = 0, otherwise
$I(k)$	Set of operations that can be processed on station family k
p_{fijk}	Processing time of step j of lot i of product f on station family k
T	Upper bound on the time horizon. $T = 240$ time units
B	Set of batching machines
U_k	Maximum batch size at station family k
L_k	Minimum batch size at station family k
$K(f,i,j)$	Set of station families on which operation j of lot i of product f can be processed
Y_{fij}	= 1, if processing step j of lot i of product f is scheduled in the planning period = 0, otherwise
Note:	The value of every Y_{fij} is known apriori for a planning period in the scheduling problem under the static lot release policy case (Refer section 4.2.1) and hence Y_{fij} is not included in the scheduling problem, while it is not known in the scheduling problem under the dynamic lot release policy case (Refer section 4.2.2) and the solution of the scheduling problem provides the value of Y_{fij}.

4.3.3 Integer programming model

4.3.3.1 Model for the static lot release policy

Under this policy, the processing steps of the various lots in the system that should be scheduled are determined apriori (Proposition 4.1). The scheduling problem has to determine the time at which each lot should be scheduled at a workstation of the appropriate station family.

Objective function:

Minimize, over all lots, the completion time of those processing steps that are to be scheduled in the corresponding time period.

$$\text{Min} \sum_{f=1}^{F} \sum_{i=1}^{I_f} \sum_{j \in J_{fi}} \sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t * X_{fijkbt}$$

Constraints

1. (Unique processing of an operation of a job)

Each step of a job must be processed only once

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} X_{fijkbt} = 1 \qquad f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$$

2. (Sequential undertaking of the jobs)

Step j+1 of any lot must start only after the completion of step j of that lot

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} (t + p_{fijk}) * X_{fijkbt} \leq \sum_{k' \in K(f,i,j+1)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} t * X_{fij+1k'b't}$$

$f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$

3. (Precedence constraints between lots of the same product)

Lot i of a product should be scheduled before lot i+1 of that product at any step j. The lots are numbered according to the chronological order of their release into the wafer fab. (Refer section 4.3.1.5)

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t * X_{fijkbt} \quad \leq \quad \sum_{k' \in K(f,i+1,j)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} t * X_{fi+1jk'b't}$$

f = 1,2,….,F i = 1,2,…..,I_f $\forall$ j $\in$ J_{fi}

4. (Capacity constraint for non-batching stations)

All stations, except batching stations, have a capacity of 1. At any given time, the number of lots at a non-batching machine must be less than or equal to 1.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \quad \leq \quad \mathbf{1} \qquad \forall k \notin B, \forall b_k$$

where,

- y_{fijkbt} is a column vector with T+1 elements and consists of zeros except from the t^{th} element to the (t + p_{fijk} –1) element, where it consists of one's.
- **1** is a column vector with T+1 elements consisting of one's.

1. (Minimum capacity constraint for batching stations)

At batching stations, a number of jobs can be processed simultaneously. A batching machine can start processing the lots once the minimum number of lots required to start processing are reached. This capacity constraint can be captured by a constraint that is identical to that in constraint 4.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \quad \geq \quad L_k \bullet D_t \qquad \forall k \in B, \forall b_k$$

where

- $\mathbf{L_k}$ is a column vector with T+1 elements with each element equal to L_k, where L_k is the minimum batch size at the batching machine.
- D_t = 1, if a batch of size of atleast L_k is being processed at the batching station
 = 0, otherwise

2. (Maximum capacity constraint for batching stations)

A batching station can only process as many lots simultaneously as the maximum batch size.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \leq U_k \qquad \forall k \in B, \forall b_k$$

where

- $\mathbf{U_k}$ is a column vector with T+1 elements with each element equal to U_k, where U_k is the maximum batch size at the batching machine.

1. (No partial overlapping of jobs at a batching station)

At a batching station, the subsequent operation must precede the previous operation by at-least the processing time of the previous operation.

$$\sum_{t'=0}^{T} t' * X_{f'i'j'k'b't'} + \alpha\ p_{f'i'j'k'} + K_2 = \sum_{t=0}^{T} t * X_{fijkbt} + \beta\ p_{fijk} + K_1$$
$$\forall k \in B, \forall b_k$$

where,

α and β are binary variables.

K_1 and K_2 are non-negative real variables

$\alpha + \beta \leq 1$

$K_1 \leq \beta * T$

$K_2 \leq \alpha * T$

2. (Lot dedication constraints)

Case 1

In this case, the photolithography operation on the first layer of the wafers in a lot will be scheduled in the current planning period and the photolithography operation for any of the subsequent layers of the wafers in that lot will not be scheduled in the same planning period, then no additional constraint is required.

Case 2

The photolithography operation on the first layer of the wafers in a lot will be scheduled in the current planning period and the photolithography operation for some of the subsequent layers of the wafers in that lot will be scheduled in the same planning period.

$$\sum_{t=0}^{T} X_{f'i'1k'b't} = \sum_{t=0}^{T} X_{f'i'j'k'b't}$$

$\forall\ f' \in F'$ $\forall\ i' \in I_f'$ $\forall\ j' = 2,3,\ldots, J_{fi}'$ $\forall\ k' \in K'(f,i,j)$ $\forall\ b' = 1,2,..,b_{k'}$

where

F' is the set of all products for which one of the lots will undergo processing at the photolithography processing step.

I_f' is the set of all lots of product f that will undergo processing at the photolithography processing step.

J_{fi}' is the highest ranked layer of a lot i of product f that will have its photolithography operation scheduled in the planning period.

K'(f,i,j) is the set of station families at which the photolithography processing operation j of lot i of product f can be performed.

$b_{k'}$ is the number of workstations in the station family k'

Case 3

In case the photolithography operation on some of the layers of the wafers in a lot was scheduled in the previous planning periods and the photolithography operation for any of the subsequent layers of the wafers in that lot will not be scheduled in the current planning period, then no additional constraint is required.

Case 4

The photolithography operation on some of the layers of the wafers in a lot was scheduled in the previous planning periods and the photolithography operation for some of the subsequent layers of the wafers in that lot will be scheduled in the current planning period.

$$\sum_{t=0}^{T} X_{f'i'j'k'b't} = 1$$

$\forall f' \in F' \qquad \forall i' \in I_f' \qquad \forall j' \in J_{fi}' \qquad k' = K'(f,i,1) \qquad b' = b_{k'}$

where

F' is the set of all products for which one of the lots will undergo processing at the photolithography processing step.

I_f' is the set of all lots of product f that will undergo processing at the photolithography processing step.

J_{fi}' is the set of photolithography operations of a lot i of product f that will be scheduled in the planning period.

K'(f,i,1) is the station family at which the photolithography processing operation for the first layer of the wafers of lot i of product f is scheduled.

$b_{k'}$ is the workstation in station family k' at which the photolithography processing operation for the first layer of the wafers of lot i of product f is scheduled.

This completes the formulation of the model for scheduling the jobs in a wafer fab under a static lot release policy.

4.3.3.2 Model for the dynamic lot release policy

Under this policy, the processing steps of the various lots in the system that should be definitely scheduled are not predetermined (Proposition 4.3). The scheduling problem has to determine the processing steps of the lots and the time at which each lot should be scheduled at a workstation of the appropriate station family.

This model requires some modifications from the model of the static lot release policy.

Objective function:

Minimize over all lots the completion time of those processing steps that are scheduled in the corresponding time period.

Min $\sum_{f=1}^{F} \sum_{i=1}^{I_f} \sum_{j \in J_{fi}} \sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t * X_{fijkbt} \quad - M * \sum_{f=1}^{F} \sum_{i=1}^{I_f} \sum_{j \in J_{fi}} Y_{fij}$

Where M is a penalty for not scheduling the job as early as possible. M ≥ T

Constraints

1. (Unique processing of each step of a job if selected for processing)

Each step of a job must be processed only once or not at all.

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} X_{fijkbt} = Y_{fij}$$

$f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$

2. (Sequential undertaking of the jobs)

Step j+1 of any lot must start only after the completion of step j of that lot

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} (T-t-p_{fijk}) * X_{fijkbt} \geq \sum_{k' \in K(f,i,j+1)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} (T-t) * X_{fij+1k'b't}$$

$f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$

3. (Sequential undertaking of jobs for the Y_{fij} variable)

Step j+1 of any lot can be processed only if step j of that lot is processed.

$Y_{fij} \leq Y_{fij+1} \qquad f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$

4. (Precedence constraints between lots of the same product)

Lot i of a product should be scheduled before lot i+1 of that product at any step j. The lots are numbered according to the chronological order of their release into the wafer fab. (Refer section 4.3.1.5)

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} (T-t) * X_{fijkbt} \geq \sum_{k' \in K(f,i+1,j)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} (T-t) * X_{fi+1jk'b't}$$

$f = 1,2,\ldots,F$ $\quad i = 1,2,\ldots,I_f$ $\quad \forall\, j \in J_{fi}$

5. (Precedence constraints between lots of the same product for the Y_{fij} variable)

Lot i+1 of a product can be scheduled at any step j only if lot i is scheduled at that step.

$$Y_{fij} \leq Y_{fi+1j} \qquad f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$$

6. (Capacity constraint for non-batching stations)

All stations, except batching stations, have a capacity of 1. At any given time, the number of lots at a non-batching machine must be less than or equal to 1.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \leq \mathbf{1} \qquad \forall k \notin B, \forall b_k$$

where,

- y_{fijkbt} is a column vector with T+1 elements and consists of zeros except from the t^{th} element to the $(t + p_{fijk} - 1)$ element, where it consists of one's.
- **1** is a column vector with T+1 elements consisting of one's.

1. (Minimum capacity constraint for batching stations)

At batching stations, a number of jobs can be processed simultaneously. A batching machine can start processing the lots once the minimum number of lots required to start processing are reached. This capacity constraint can be captured by a constraint that is identical to that in constraint 4.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \geq \mathbf{L}_k \bullet D_t \qquad \forall k \in B, \forall b_k$$

where

- $\mathbf{L}_k$ is a column vector with T+1 elements with each element equal to L_k, where L_k is the minimum batch size at the batching machine.
- $D_t = 1$, if a batch of size atleast L_k is being processed at the batching station
  $= 0$, otherwise

2. (Maximum capacity constraint for batching stations)

A batching station can only process as many lots simultaneously as the maximum batch size.

$$\sum_{t=0}^{T} \sum_{I(k)} y_{fijkbt} \bullet X_{fijkbt} \leq U_k \qquad \forall k \in B, \forall b_k$$

where

- $\mathbf{U_k}$ is a column vector with T+1 elements with each element equal to U_k, where U_k is the maximum batch size at the batching machine.

1. (No partial overlapping of jobs at a batching station)

At a batching machine, the subsequent operation must precede the previous operation by at-least the processing time of the previous operation.

$$\sum_{t'=0}^{T} t' * X_{f'i'j'k'b't'} + \alpha\, p_{f'i'j'k'} + K_2 + T * (1 - \sum_{t'=0}^{T} t' * X_{f'i'j'k'b't'}) =$$

$$\sum_{t=0}^{T} t * X_{fijkbt} + \beta\, p_{fijk} + K_1 + T * (1 - \sum_{t=0}^{T} t * X_{fijkbt})$$

$$\forall k \in B, \forall b_k$$

where,

α and β are binary variables.

K_1 and K_2 are non-negative real variables

$\alpha + \beta \leq 1$

$K_1 \leq \beta * T$

$K_2 \leq \alpha * T$

2. (Lot Dedication Constraints)

Case 1

In this case, the photolithography operation on the first layer of the wafers in a lot may be scheduled in the current planning period and the photolithography operation for any of the subsequent layers of the wafers in that lot will not be scheduled in the same planning period, then, no additional constraint is required.

Case 2

The photolithography operation on the first layer of the wafers in a lot may be scheduled in the current planning period and the photolithography operation for some of the subsequent layers of the wafers in that lot may be scheduled in the same planning period.

$$\sum_{t=0}^{T} X_{f'i'1k'b't} \leq \sum_{t=0}^{T} X_{f'i'j'k'b't}$$

$\forall f' \in F'$ $\forall i' \in I_f'$ $\forall j' = 2,3,\ldots, J_{fi}'$ $\forall k' \in K'(f,i,j)$ $\forall b' = 1,2,..,b_{k'}$

where

F' is the set of all products for which one of the lots may undergo processing at the photolithography processing step.

I_f' is the set of all lots of product f that may undergo processing at the photolithography processing step.

J_{fi}' is the highest ranked layer of a lot i of product f that may have its photolithography operation scheduled in the planning period.

K'(f,i,j) is the set of station families at which the photolithography processing operation j of lot i of product f can be performed.

$b_{k'}$ is the number of workstations in the station family k'

Case 3

In case the photolithography operation on some of the layers of the wafers in a lot was scheduled in the previous planning periods and the photolithography operation for any of the subsequent layers of the wafers in that lot will not be scheduled in the current planning period, then, no additional constraint is required.

Case 4

The photolithography operation on some of the layers of the wafers in a lot was scheduled in the previous planning periods and the photolithography operation for some of the subsequent layers of the wafers in that lot may be scheduled in the current planning period.

$$\sum_{t=0}^{T} X_{f'i'j'k'b't} = Y_{f'i'j'}$$

$\forall f' \in F' \qquad \forall i' \in I_f' \qquad \forall j' \in J_{fi}' \qquad k' = K'(f,i,1) \quad b' = b_{k'}$

where

F' is the set of all product for which one of the lots may undergo processing at the photolithography processing step.

I_f' is the set of all lots of product f that may undergo processing at the photolithography processing step.

J_{fi}' is the set of photolithography operations of a lot i of product f that may be scheduled in the planning period.

K'(f,i,1) is the station family at which the photolithography processing operation for the first layer of the wafers of lot i of product f is scheduled.

$b_{k'}$ is the workstation in station family k' at which the photolithography processing operation for the first layer of the wafers of lot i of product f is scheduled.

This completes the formulation of the model for scheduling the jobs in a wafer fab under a dynamic lot release policy.

4.3.3.3 Additional features in the models to incorporate real-life situations

1. Preventive Maintenance

Preventive maintenance is performed on various machines from time to time. The schedule for preventive maintenance is drawn up months in advance. Hence, the unavailability of machine for the duration of preventive maintenance is known. This unavailability can be accommodated in both the models in the capacity constraints for all the stations. When a workstation is down from a period t to t+p, the capacity of the workstation for that time duration is set to zero. Thus, no job can be processed at that workstation during that time duration.

2. Unscheduled Workstation Breakdown

Workstations may breakdown in the middle of their operations. The workstation needs to be repaired so that it can start processing jobs again. The time spent in repairing the workstation is called unscheduled workstation downtime. Based upon the type of breakdown on a particular machine, the amount of time required to repair the machine can be estimated. When the machine breaks down, the scheduling model should be solved once again. The shop status information at the time of breakdown should be considered in the model. The capacity of the workstation that has broken down should be set to zero for the time duration of expected repair time.

3. Hot Lots

Hot lots are lots that have a priority higher than all other lots in the fab. At any workstation the hot lots have a higher priority than any other lot for scheduling. This feature can be incorporated in the model by adding a penalty term for not scheduling the hot lots as soon as possible at any workstation. Hence, in the objective function for the model with a dynamic lot release policy, the penalty term "M" for the hot lots can be higher than for other lots in the wafer fab. Or, the Y_{fij} variables for the processing steps of the hot lots can be fixed to a value of one (i.e. those processing steps have to be scheduled in the planning period). Thus, the scheduling of the processing steps of a hot lot can be ensured.

4. Due Date Constraint for the Last Lot of a Work Order

The due date of a work order is the date the work order should be completed. In the tardiness problem presented in chapter 3, the modified due date of the work order is the due date of the work order less the expected cycle time of the last lot of the product of the work order (refer section 3.2.2.2). Thus the modified due date of a work order is the date by which the last lot of the work order should be released into the system. The completion time of work order in the tardiness problem is the date on which the last lot of the work order is released into the system (refer section 3.2.2.1). The solution of the tardiness problem provides the completion time and hence the tardiness of the work orders.

The expected cycle time of any lot of a product is based upon the historical data. The wafer fab has the capacity to process the lot within the expected cycle time. If the last lot of the work order is not completed within the expected cycle time, then the work order is delayed and hence, the tardiness value of the work order is increased and hence the total tardiness value obtained from the solution of the tardiness value is increased.

To prevent the violation of the tardiness values obtained from the solution of the tardiness problem, the following constraint can be incorporated in both the models

$$\sum_{k \in K(f,i',J'fi')} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t * X_{fi'J'fi'kbt} \leq T - p_{fi'J'fi'}$$

where

i' is the last lot of a work order

$J'_{fi'}$ is the last processing step of the lot i' of product f.

$p_{fi'J'fi'}$ is the processing time of step J'_{fi} of lot i' of product f.

The time period during which the above constraint is incorporated is the time period of completion of the last lot of the work order as per the solution to the tardiness problem plus the expected cycle time of the last lot of the product of the work order.

This constraint should be incorporated in the models only if the last processing step of the last lot of the work order has not been scheduled in the previous time periods.

4.3.4 Methodology to solve the integer problem

The 0-1-integer problem that we have formulated for the scheduling of the jobs in a wafer fab is a NP-Hard problem. The time required to solve the problem is not a polynomial in the size of the problem. Problems for scheduling of jobs in a large size shop have been known to be computationally intractable. Hence, algorithms that exploit the special structure of the problem on hand have been developed for most of the scheduling problems.

An attempt to solve the formulated scheduling problem by the branch and bound procedure of the CPLEX 6.6 integer programming software fails to provide a solution in reasonable computation time. Hence, we explore other methodologies for solving this scheduling problem.

A branch and bound procedure typically begins by solving the LP relaxation of the problem on hand and starts branching on those variables that have a fractional value. One method for improving the performance of the branch and bound procedure is to improve its LP relaxation as much as possible. We explore the tightening of the LP relaxation of the scheduling problem in the next section of this chapter.

4.4 A TIGHTER LP RELAXATION OF THE INTEGER PROGRAMMING MODEL FOR SCHEDULING IN A WAFER FAB

4.4.1 Introduction

The integer-programming model for the scheduling of lots in a wafer fab has been formulated in the previous section. The start times of the lots on the workstations are represented as binary variables. The scheduling problem is solved over a single time period of usually, one-day. Even a moderate sized problem has a large number of binary variables in this formulation. A problem with 22 lots consisting of 2 products, 18 workstations, and an average of 14 processing steps in each route has nearly 80,000 binary variables that represent the start times of the lots on the various machines along their routes. This scheduling problem is NP-Hard and can take an excessive amount of time for solution.

Hence, there is a need to develop a method that can help in obtaining a good solution to the integer program in a reasonable amount of time. The current section presents such a method. In the subsequent parts of this chapter, we briefly explain the branch and bound procedure that is typically used to solve an integer program. We explain the motivation for the method that we have explored to solve the integer program faster. Finally, we present some results and conclusions from the experiments performed on this method.

4.4.2 Branch and Bound Procedure

4.4.2.1 Description

The Branch and Bound (B&B) Procedure is the most widely used method for solving the pure integer and mixed-integer programming models. Sierksma [32] provides a succinct description of the B&B method. “Basically, the B&B method solves a model by breaking up its feasible region into successively smaller subsets (branching), calculating bounds on the objective value over each corresponding sub-model (bounding), and using them to discard certain sub-models from further consideration (fathoming). The bounds are obtained by replacing the current sub-model by an easier model (relaxation), such that the solution of the latter yields a bound for the former. The procedure ends when each sub-model has either produced an infeasible solution, or has been shown to contain no better solution than the one already at hand. The best solution found during the procedure is an optimal solution”.

4.4.2.2 Shortcomings of the Branch and Bound Procedure

To compute a lower (or upper) bound for each sub-problem, the procedure has to solve a linear programming (LP) relaxation (of the integer problem) in which the integer restrictions on all the integer variables are relaxed. The integrality gap that exists between the value of the LP relaxation, ν(LP), and that for the integer problem, ν(IP), given by | ν(LP) - ν(IP) |, is often too large in relation to ν(IP), requiring extensive branching to resolve the values of the integer variables.

4.4.3 Motivation for tighter LP relaxation

It is well known in discrete optimization literature that in order to be able to solve reasonably sized instances of challenging classes of problems, two particular features must come into play. First, a good model of the problem must be constructed in the sense that it affords a tight underlying linear programming representation, and second, any inherent special structures must be exploited, both in the process of model formulation and in algorithmic development [31].

Most of the commercially available software for solving integer programming models use the LP relaxation within their solution process to compute lower (upper) bounds at the nodes of a branch and bound procedure. A tighter LP relaxation will mean that the B&B procedure will not require extensive branching to obtain an integer solution.

One possible option to overcome a large value of the integrality gap is to reduce it before solving the problem by the B&B procedure. This will require a reformulation of the problem into a tighter model. The characteristic of this “tight” model is that it more

closely approximates the convex hull of integer feasible solutions. Thus, the model should “chop off” a large region from the convex hull of LP feasible solutions and should be closer to the convex hull of integer feasible solutions.
Additional constraints can be incorporated into the problem that will chop off regions from the convex hull of LP feasible solutions, and make the formulation tighter. Thus, by the addition of the “right” type of constraints to integer programs, the solvability of the problem will be enhanced, even though the problem size is increased.

4.4.4 Earliest Start Times (EST) and Latest Start Times (LST)

PERT-CPM is a widely used technique to establish the earliest start times (EST) and the latest start times (LST) for activities in a project. The earliest start time of an activity establishes the time before which the activity cannot begin because the previous activities in the precedence network of the project cannot be completed before the earliest start time. The latest time of an activity is the time beyond which the start of the activity cannot be delayed because, doing so will delay the subsequent activities in the precedence network of the project. This will lead to an overall delay in the execution of the project.

A scheduling problem is usually solved over a definite time horizon. The starting times of the various steps of a job are assumed to lie somewhere over the entire time horizon of the scheduling problem. A closer look at the scheduling problem will reveal that there exist certain points in time before which steps of the lots cannot start processing because the previous steps in the route cannot be completed before that time. This implies that there exist earliest start times for the steps of any lot. By eliminating the time periods from the beginning of the scheduling problem to the EST’s for each processing step, the solution space of the problem can be restricted. By eliminating the binary variables corresponding to the above time period, the problem size will be reduced and a tighter formulation of the problem will be obtained. The convex hull of LP feasible solutions can be expected to be closer to the convex hull of integer solutions. Hence, the performance of the branch and bound algorithm should improve, and result in a reduction in the computational time to obtain the optimal integer solution.

The scheduling problem can be made tighter by computing the latest start times of the processing steps of the jobs. The LST can be computed for the processing steps of all the jobs for a scheduling problem following a static lot release policy. The LST cannot be computed for the processing steps of all the jobs in the fab for a scheduling problem under the dynamic lot release policy as will be explained in the subsequent sections.

The incorporation of the EST and the LST constraints in the scheduling problem should

- Eliminate a large number of variables from the problem.
- Provide an LP relaxation that closely resembles the convex hull of integer feasible solutions.
- Considerably reduce the amount of computations necessary to obtain an integer feasible solution.

4.4.5 Incorporation of EST and LST as constraints

The EST_{fij} and the LST_{fij} determined for processing step j of lot i of product f are incorporated into the scheduling problem by the following constraints.

1. Constraint for EST

Each operation can start only on or after its EST

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t \; X_{fijkbt} \quad \geq \quad EST_{fij}$$

f = 1,2,….,F i = 1,2,…..,I_f $\forall$ j $\in$ J_{fi}

2. Constraint for LST

Each operation should start only on or before its LST

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t \; X_{fijkbt} \quad \leq \quad LST_{fij}$$

f = 1,2,….,F i = 1,2,…..,I_f $\forall$ j $\in$ J_{fi}

Separate strategies are employed to determine the EST's and the LST's for the scheduling problem under the static lot release policy and the dynamic lot release policy.

4.4.6 EST for scheduling problem under the static lot release policy.

Application of flow approach to the scheduling problem for a wafer fab under the static lot release policy completely determines the processing steps of the lots to be scheduled in a planning period. (From Proposition 4.1)

For a given product f we have to schedule the earliest n_f lots which arrive at each of the processing steps in the route of the product f. (From Proposition 4.2)

The lots of the same product follow the precedence constraints between them at each processing step. Lots of a product released into the system earlier are processed before the lots of the product released later, at the same processing step. (From Proposition 4.7)

Thus, the identity and chronological order of all the lots that need to be processed at each processing step of each product is established. Next, the EST's for each of these processing steps of the lots should be established. The data about the station families at each of the processing steps, the availability of these workstations and the lots should be considered while deciding the EST's.

Consider a station family k with m workstations at which the j^{th}-processing step of product f will be performed. Let n_f be the number of lots of product f in the system. At the beginning of the current planning period, the workstations may still be working on jobs from the previous time period. Hence the workstation b will be available at some time t_b.

Arrange the m workstations in non-decreasing order of availability. Thus the m workstations will be available at time $t_1 \le t_2 \le t_3 \le, \ldots, \le t_m$. Number the workstations in non-decreasing order of availability.

Due to the stage by stage solution approach, the workstations may be completing the processing operation on a lot scheduled in the previous stage, when the current planning period (stage) starts. The earliest available workstation may be available at the start of the planning period or it may be available after the start of the planning period.

Hence, the earliest available time of the workstations may be > 0. This leads us to the following result:

Proposition 4.8

The earliest time in the current planning period at which the workstation is available for processing is such that $0 \le t_1 \le t_2 \le t_3 \le \ldots\ldots\ldots \le t_m$.

Proposition 4.9

The earliest available time of a workstation will always be less than the processing time at that station. ($t_1 \le t_2 \le t_3 \le, \ldots, \le t_m < p_{fij}$)

Proof: (by contradiction)

Consider a workstation k, which is available the earliest at time t_k in the planning period, because a job from the previous planning period is still being processed. Assume $t_k \ge p_{fij}$. Then, the start time of the job that is still being completed on the workstation = $t_k - p_{fij} \ge 0$

This means that the job was scheduled on the machine at a time $t_k - p_{fij} \ge 0$, i.e. the job was scheduled on the machine in the current planning period or the workstation was available at a time $0 \le t_k' < t_k$, a contradiction. QED

Allocation of EST_{stn}

Consider a hypothetical case, in which all the n_f lots of product f are available for processing at the start of the planning period, and the station family is going to devote its entire capacity to only these lots.

From the Proposition 4.7, the earliest lot amongst the n_f lots should be scheduled at the earliest available time. The second earliest lot can be scheduled at the second earliest available time, and so on.

Case 1: The number of lots n_f is less than or equal to number of workstations m. ($n_f \le m$)

- Assign the times $t_1 \le t_2 \le t_3 \le, \ldots, \le t_{nf}$ as the EST_{stn} to the n_f lots.
- A few workstations of the station family may remain idle if $n_f < m$.

- Then, each of the lots has been allotted an EST_{stn}.

Case 2: The number of lots n_f is greater than the number of workstations m. ($n_f > m$)

- Assign the times $t_1 \leq t_2 \leq t_3 \leq, \ldots, \leq t_m$ as the EST_{stn} to the first m lots out of the n_f lots.
- The next m lots are allotted their EST's by the same procedure, but the earliest start times for these m lots will be t_m+p_j, where p_j is the processing time for each lot on the j^{th} processing step and t_m is the available time of the m^{th} machine. The schematic representation is provided in figure 4.1
- Continue this forward recursion until all the n lots are assigned earliest start times.

Note that the lots are not assigned to specific workstations. The workstations are used only to determine the EST's of the lots.

This procedure is repeated for all the processing steps of all the lots of all the products in the system. We call these EST values as EST_{stn}, because these values are obtained based on the station data.

Proposition 4.10

The methodology adopted in assigning the EST_{stn} to the lots does not cause a violation of the precedence constraint amongst lots of the same product at the same processing step. (i.e, with reference to figure 4.1, lot m+1 is not scheduled before lot m, i.e. $t_m < t_{m+1}$)

Proof:

EST_{stn} for lot m = t_m

EST_{stn} for lot m+1 = $t_{m+1} = t_1 + p_{fij}$

$t_1 \leq t_2 \leq t_3 \leq, \ldots, \leq t_m < p_{fij}$ (from Proposition 4.9)

$0 \leq t_1 \leq t_2 \leq t_3 \leq \ldots\ldots\ldots \leq t_m$ (from Proposition 4.6)

$t_m < p_{fij}$

$0 \leq t_1$

$\therefore t_m + 0 < t_1 + p_{fij}$

$\therefore t_m < t_1 + p_{fij}$

$\therefore t_m < t_{m+1}$

QED

Allocation of EST_{lot}

- A lot may not be available for processing at the start of the planning period because it may be finishing its processing on another machine from the previous period, or due to the time of release of the lot into the system.
- Let the earliest time a lot will be available for processing be denoted by EST_{lot}.

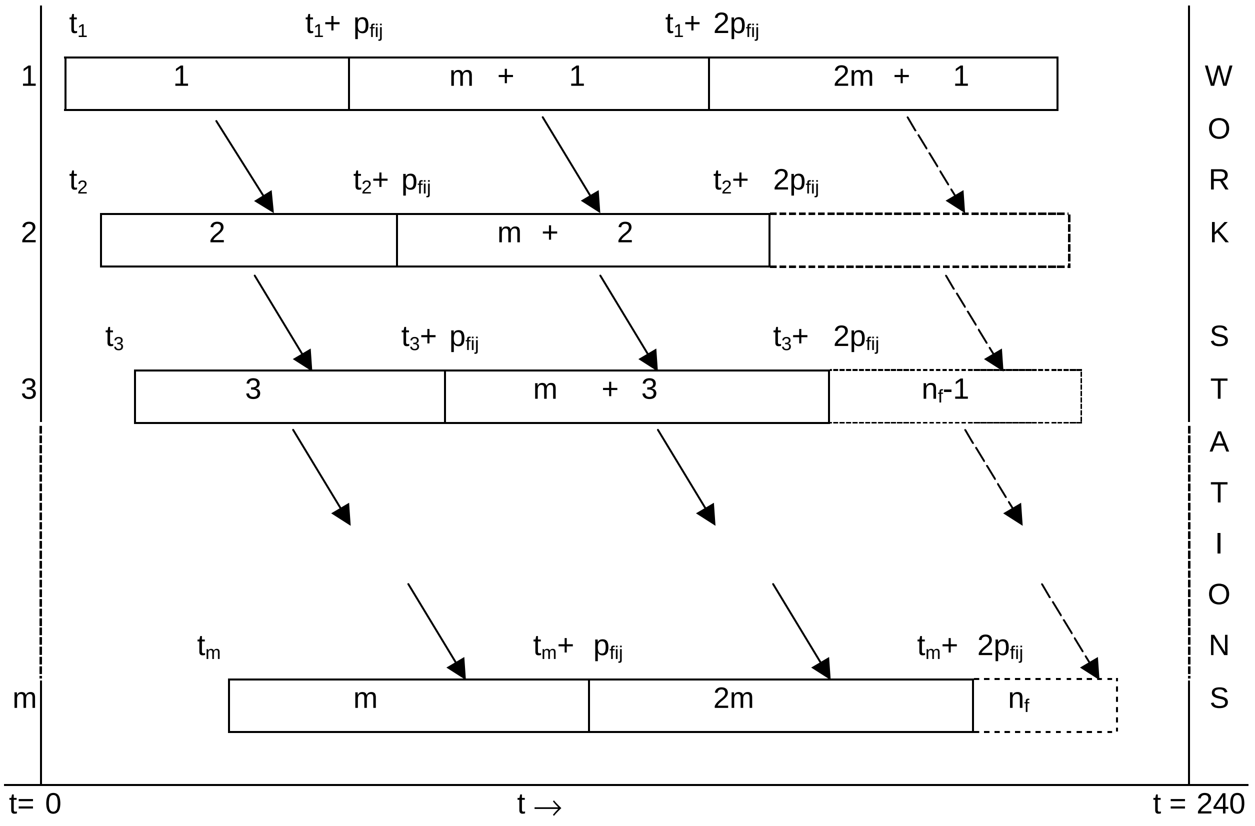

Figure 4.1 Allocation of EST_{stn} to lots of a product at a processing step

Allocation of EST_{new}

- The value of EST_{stn} and EST_{lot} may be different for the processing step of any lot.
- A modified value of the EST for each processing step of each lot is calculated as $EST_{new} = \max(EST_{stn}, EST_{lot})$.

Allocation of EST_{fnal}

- Let the number of processing steps of a lot to be processed in the current planning period be x (identified in Proposition 4.1).
- The EST_{new} of all these processing steps have been determined.
- A step k cannot start before the step k-1 is completed.
- The earliest a step k can start is $EST_{new}(k-1) + p_{k-1}$, where p_{k-1} is the processing time for step k-1. Let this value of EST be denoted by EST_{final}.
- This value of the EST obtained is the final EST value and is incorporated in the scheduling problem.

We now present the algorithm to determine the EST for the processing steps of each lot in the system.

4.4.7 Algorithm to establish EST of each processing step of a lot in a single planning period at a wafer fab with static lot release policy

Initialization

Step 1: Let f = 1,2,3,…,F be the products in the system in each planning period.
Step 2: Let n_f be the number of lots of product f released into the system in each planning period.
Step 3: Let $j_f \in \{1,2,3,\ldots,J_f\}$ be the processing steps in the route of product f. (Thus n_f lots will be completed at each step j_f in each time period).
Step 4: Let b_k be the number of workstations in station family k = 1,2,3,…,K
Step 5: For every station family, arrange the workstations in non-decreasing order of their earliest available times, $t_1 \leq t_2 \leq t_3 \leq, \ldots, \leq t_{bk}$

Allocation of EST_{stn}

Step 6: For the processing step j_f of a product f, identify the earliest n_f lots in the system that have to be processed on step j_f. Set y= 0. Let p_{fj} denote the processing time of step j_f of product f.

Step 7: If step j_f is to be performed on station family k, then assign the times $t_1 \leq t_2 \leq t_3 \leq, ..., \leq t_{bk}$ as the EST_{stn} to the first b_k (out of n_f) lots of the product f at step j. Set y = 1

Step 8: Assign the times $t_1 + yp_{fj} \leq t_2 + yp_{fj} \leq t_3 + yp_{fj} \leq, \ldots, \leq t_{bk} + yp_{fj}$ as the EST_{stn} to the next b_k {from (y*m+1) to ((y+1)*m)} lots of the product f at step j_f. Refer figure 4.1. Set y = y+1

Step 9: Repeat the step 8 until all the n_f lots of step j of product f have been assigned their EST_{stn}'s.

Step 10: Repeat steps 6,7, 8, and 9 for all steps $j_f \in \{1,2,3,\ldots,J_f\}$ for the product f

Step 11: Repeat steps 6,7,8,9, and 10 for all the products in the system

Computation of EST_{new}

Step 12: Let the earliest time a lot becomes available for processing be denoted by EST_{lot}.

Step 13: For each processing step of a lot, compute the modified value of the EST by the formula $EST_{new} = \max(EST_{stn}, EST_{lot})$.

Computation of EST_{final}

Step 14: Let x_{fij} represent the number of processing steps to be performed on a lot i of product f in the current planning period identified by Proposition 4.1. Let x = 1, 2, 3, …., x_{fij}. Let p_x be the processing time for step x. Set x = 1

Step 15: For x = 1, $(EST_{final})_x = (EST_{new})_x$.

Step 16: $(EST_{final})_{x+1} = (EST_{final})_x + p_x$

Step 17: Set x = x+1

Step 18: Repeat steps 16 and 17 while $x \leq x_{fij}$

Step 19: Repeat steps 14, 15, 16, 17, and 18 for all the lots in the system

Step 20. Terminate the algorithm.

4.4.8 LST for scheduling problem under the static lot release policy.

The methodology to determine the LST for the scheduling problem of a wafer fab under the static lot release policy is similar to the methodology for determining the EST. In the scheduling problem, the worst-case scenario for scheduling a lot will be that the lot will start processing during the last time unit of a planning period. This will be the LST for those lots that are at the bottom of the list of lots to be processed at a processing step. To determine the LSTs of the lots higher in the list we just have to go one step backward recursively, just as we went forward recursively when determining the ESTs. The last time unit in the scheduling problem is 240 as explained in chapter 3.

Allocation of LST_{stn}

- Consider a station family k with m workstations at which the j^{th}-processing step of product f will be performed.
- Let n_f be the number of lots of product f in the system.
- In the scheduling problem, the worst case scenario for scheduling a lot will be for the lot to start processing during the last time unit of a planning period. Hence, in the worst case, the workstation b may start processing at T.
- Thus, the m workstations may start processing the last m lots at a time $t_1 = t_2 = t_3 =,\ldots, = t_m = T = 240$

Consider the extreme case in which all the n_f lots of product f are available for processing at the end of the planning period, and the station family is going to devote its entire capacity to only these lots of the product.

From the Proposition 4.5, the latest lot amongst the n_f lots should be scheduled at the latest starting time. The second latest lot can be scheduled at the second latest available time, and so on.

Case 1: The number of lots n_f is less than or equal to number of workstations m. ($n_f \leq m$)

- Assign the times $t_1 = t_2 = t_3 =,\ldots, = t_{nf} = T = 240$ as the LST_{stn} to the n_f lots.
- A few workstations of the station family may remain idle if $n_f < m$.
- Thus, each of the lots has been allotted an LST_{stn}.

Case 2: The number of lots n_f is greater than the number of workstations m. ($n_f > m$)

- Assign the times $t_1 = t_2 = t_3 =, \ldots, = t_{nf} = T = 240$ as the LST_{stn} to the last m lots out of the n_f lots.
- The next m lots are allotted their LST's by the same procedure, but the latest start times for these m lots will be $t_m - p_j$, where p_j is the processing time for each lot on the j^{th} processing step and t_m is the latest starting time of the m^{th} machine. The schematic representation is provided in Figure 4.2
- Continue this backward recursion until all the n lots are assigned earliest start times.

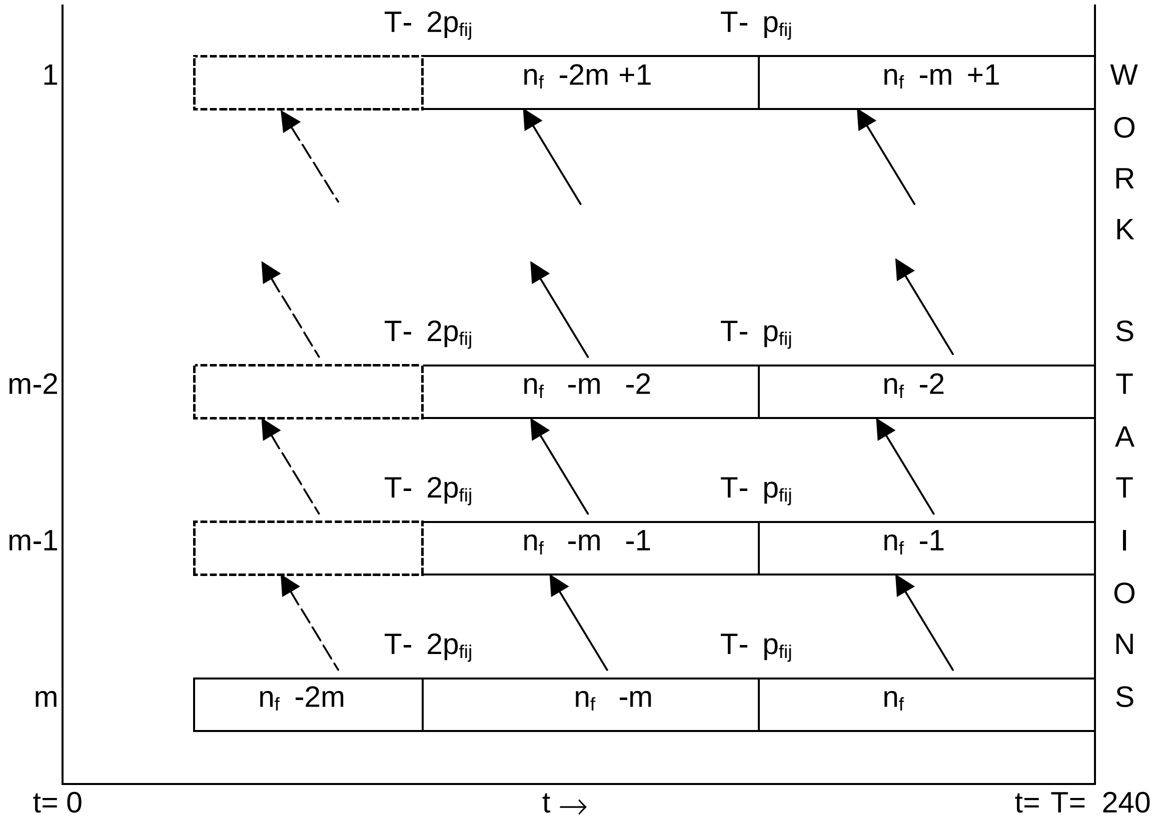

Figure 4.2 Allocation of LSTstn to lots of a product at a processing step

Note that the lots are not assigned to specific workstations. The workstations are used only to determine the LST's of the lots.

This procedure is repeated for all the processing steps of all the lots of all the products in the system. We call these LST values as LST_{stn}, because these values are obtained based on the station data.

Proposition 4.11

The methodology adopted in assigning the LST_{stn} to the lots does not cause a violation of the precedence constraint amongst lots of the same product at the same processing step. (that is, with reference to Figure 4.2, lot n_f - m+1 is not scheduled before lot n_f - m, i.e. $t_{nf -- m} < t_{nf -- m +1}$)

Proof:

LST_{stn} for lot $n_f – m + 1 = t_{nf -- m +1} = T = 240$

LST_{stn} for lot $n_f – m = t_{nf -- m} = T - p_{fij}$

$\Rightarrow t_{nf -- m} < t_{nf -- m +1}$

QED

Allocation of LST_{lot}

- A processing step of a lot may be available for processing until the last time unit in the planning period.
- Let the latest time a lot will be available for processing be denoted by LST_{lot}.

Allocation of LST_{new}

- The value of LST_{stn} and LST_{lot} may be different for the processing step of any lot.
- A modified value of the LST for each processing step of each lot is calculated as $LST_{new} = \min(LST_{stn}, LST_{lot})$.

Allocation of LST_{fnal}

- Let the number of processing steps of a lot to be processed in the current planning period be x (identified in Proposition 4.1).
- The LST_{new} of all these processing steps have been determined.
- A step k cannot start before the step k-1 is completed.
- The latest a step k can start is $LST_{new}(k+1) - p_k$, where p_k is the processing time for step k. Let this value of LST be denoted by LST_{final}.

- This value of the LST obtained is the final LST value and is incorporated in the scheduling problem.

This procedure is repeated for all the processing steps of all the lots of all the products in the system.

4.4.9 Algorithm to establish LST of each processing step of a lot in a single planning period at a wafer fab with static lot release policy

Initialization

Step 1: Let f = 1,2,3,…,F be the products in the system in each planning period.
Step 2: Let n_f be the number of lots of product f released into the system in each planning period
Step 3: Let $j_f \in \{1,2,3,\ldots,J_f\}$ be the processing steps in the route of product f. (Thus n_f lots will be completed at each step j_f in each planning period).
Step 4: Let b_k be the number of workstations in station family k = 1,2,3,…,K
Step 5: For every station family, the last available time unit in the planning period will be T= 240. For each workstation the latest time a lot can start processing is the 240th time unit. Hence, $t_1 = t_2 = t_3 =, \ldots,= t_{bk} = 240$

Allocation of LST_{stn}

Step 6: For the processing step $j_f \in \{1,2,3,\ldots,J_f\}$ of a product f, identify the earliest n_f lots in the system that have to be processed on step j. Set y= 0. Let p_{fj} denote the processing time of step j of product f.
Step 7: If step j is to be performed on station family k, then assign the times $t_1 = t_2 = t_3 =, \ldots,= t_{bk} = 240$ as the LST_{stn} to the last b_k (out of n_f) lots of the product f at step j from the n_f lots selected in step 6. Set y = 1
Step 8: Assign the times $t_1 - yp_{fj} = t_2 - yp_{fj} = t_3 - yp_{fj} =, \ldots,= t_{bk} - yp_{fj}$ as the LST_{stn} to the next to last b_k {from $(n_f - (y+1)*b_k + 1)$ to $(n_f - y*b_k)$} lots of the product f at step j. refer figure 4.2. Set y = y+1
Step 9: Repeat the step 8 until all the n_f lots of step j of product f have been assigned their LST_{stn}'s.

Step 10: Repeat steps 6,7, 8, and 9 for all steps $j_f \in \{1,2,3,\ldots,J_f\}$ for the product f

Step 11: Repeat steps 6,7,8,9, and 10 for all the products f in the system

Computation of LST_{new}

Step 12: Let the latest time a lot can start processing be denoted by LST_{lot}. For the static lot release policy problem the LST_{lot} for all lots is T = 240 time units.

Step 13: For each processing step j of a lot i, compute the modified value of the LST by the formula $LST_{new} = \min(LST_{stn}, LST_{lot})$.

Computation of LST_{final}

Step 14: Let x_{fij} represents the number of processing steps to be performed on a lot i of product family f in the current time period identified by Proposition 4.1. Let x = 1, 2, 3, …., x_{fij}. Let p_x be the processing time for step x. Set $x = x_{fij}$

Step 15: For $x = x_{fij}$, $(EST_{final})_x = (EST_{new})_x$.

Step 16: $(EST_{final})_{x-1} = (EST_{final})_x - p_{x-1}$

Step 17: Set x = x-1

Step 18: Repeat steps 16 and 17 while $x \geq 1$

Step 19: Repeat steps 14, 15, 16, 17, and 18 for all the lots in the system

Step 20: Terminate the algorithm.

4.4.10 EST for scheduling problem under the dynamic lot release policy

The methodology to determine the EST for the scheduling problem of a wafer fab with a dynamic lot release policy is similar to the methodology for determining the EST for the scheduling problem of a wafer fab under the static lot release policy.

Application of flow approach to the scheduling problem for a wafer fab under the dynamic lot release policy completely determines the processing steps of the lots that may be scheduled in a planning period (Proposition 4.3).

The lots of the same product follow the precedence constraints between them at each processing step. The lots of a product released into the system earlier are processed before the lots of the product released later, at the same processing step. (From Proposition 4.7)

Thus, the identity and chronological order of all the lots that may be processed at each processing step of each product is established for the scheduling problem under the dynamic wafer release policy.

The rest of the procedure for establishing the EST for the processing steps for each of the above lots is the same as the procedure adopted for establishing the EST for the lots under the static lot release policy.

We now present the algorithm to determine the EST for the processing steps of each lot in the system for a wafer fab with a dynamic lot release policy.

4.4.11 Algorithm to establish EST of each processing step of a lot in a single time period at a wafer fab with dynamic lot release policy

Initialization

Step 1: Let $f = 1,2,3,\ldots,F$ be the product s in the system in the time period.
Step 2: Let n_f be the number of lots of product f in the system in the time period.
Step 3: Let $j_{fi} = \{x_{fi},\ldots,J_{fi}\}$ be the remaining processing steps in the route of lot i of product f.
Step 4: Let b_k be the number of workstations in station family $k = 1,2,3,\ldots,K$
Step 5: For every station family, arrange the workstations in non-decreasing order of their earliest available times, $t_1 \leq t_2 \leq t_3 \leq, \ldots, \leq t_{bk}$

Selection of steps for each lot (as per Proposition 4.3)

Step 6: Let x_{fi} denote the next processing step for lot i of product f. Set $j = x_{fi}$. For lot i of product f let the earliest time the lot becomes available for processing be denoted by $(EST_{lot})_{fij}$. Thus the EST for step j of lot i will be $(EST_{lot})_{fij}$.
Step 7: $(EST_{lot})_{fij+1}$ for step j+1 of lot i will be $(EST_{lot})_{fij} + p_{fij}$. Set $j = j+1$
Step 8: Repeat step 7 while $j \leq J_{fi}$ or $(EST_{lot})_{fij} + p_{fij} \leq T = 240$

Step 9: Delete from the scheduling problem all those steps j of lot i of product f for which the $(EST_{lot})_{fij}$ is greater than T = 240, because these processing steps will be processed in the next planning period and hence, in a different scheduling problem.

Allocation of EST_{stn}

Step 10: For the processing step j of a product f, identify all the n_f lots determined by Proposition 4.3 that can be processed on step j. Set y= 0. Let p_{fj} denote the processing time of step j of product f.

Step 11: If step j of product f is to be performed on station family k, then assign the times $t_1 \le t_2 \le t_3 \le, \ldots, \le t_{bk}$ as the EST_{stn} to the first b_k (out of n_f) lots of the product f at step j. Set y = 1

Step 12: Assign the times $t_1 + yp_{fj} \le t_2 + yp_{fj} \le t_3 + yp_{fj} \le, \ldots, \le t_{bk} + yp_{fj}$ as the EST_{stn} to the next b_k {from (y*m+1) to ((y+1)*m)} lots of the product f at step j_f. Refer figure 4.1. Set y = y+1

Step 13: Repeat the step 12 until all the n_f lots of step j of product f have been assigned their EST_{stn}'s.

Step 14: Repeat steps 10,11, 12, and 13 for all steps j for the product f

Step 15: Repeat steps 10,11,12,13 and 14 for all the products in the system

Computation of EST_{new}

Step 16: For each processing step j of a lot i, compute the modified value of the EST by the formulae $EST_{new} = \max(EST_{stn}, EST_{lot})$.

Computation of EST_{final}

Step 17: Let x_{fij} represents the number of processing steps to be performed on a lot j of product f in the current time period identified by Proposition 4.3. Let x = 1, 2, 3, …., x_{fij}. Let p_x be the processing time for step x. Set x = 1

Step 18: For x = 1, $(EST_{final})_x = (EST_{new})_x$.

Step 19. $(EST_{final})_{x+1} = (EST_{final})_x + p_x$

Step 20: Set x = x+1

Step 21: Repeat steps 19 and 20 while $x \le x_{fij}$

Step 22: Repeat steps 17, 18, 19, 20, and 21 for all the lots in the system

Step 23: Delete all those processing steps j of lots of product f for which the $(EST_{lot})_{fij}$ is greater than T = 240, because those processing steps will be scheduled in the next planning stage and hence, in another scheduling problem.
Step 24: Terminate the algorithm.

4.4.12 LST for scheduling problem under the dynamic lot release policy

The methodology to determine the LST for the scheduling problem of a wafer fab under the dynamic lot release policy is similar to the methodology for determining the LST under the static release policy. In the scheduling problem, the worst case scenario for scheduling a lot will occur when the lot will start processing at the last time unit of a planning period. Under the dynamic lot release policy, we do not pre-determine the processing steps of a lot that will be processed in a planning period. Even if the lots get processed, they may start processing at the last time unit of a planning period. As the last time unit in the scheduling problem is 240 (explained in 4.3.17), we assign LST = T = 240 for all the processing steps identified in the algorithm for EST.

4.4.13 Algorithm to establish LST of each processing step of a lot in a single planning period at a wafer fab under the dynamic lot release policy

Step 1: Select all the processing steps of all lots over all products that were identified in the algorithm for computing the EST under the dynamic lot release policy.
Step 2: Assign the LST of all the steps identified above as LST_{final} = T = 240.
Step 3: Terminate the algorithm

4.4.14 Numerical Example

To illustrate the significance of the integrality gap, consider the scheduling problem for a small sized wafer fab. Detailed problem data given in Appendix C is summarized below.

Product Number	Processing steps in route	Input rate per day (lots/day)	Initial WIP
1	14	7	9
2	11	3	3

8 Station Families	18 Workstations	One time period = 24 hours

In Table 4.1, the values of the optimal integer solution and the linear programming relaxation are presented. The absolute value of the integrality gap and the difference of the integrality gap from the optimal integer solution value are compared. The time and number of iterations required for the linear programming relaxation are also provided.

Table 4.1 Solution of the LP relaxation and comparison of ν(LP) with ν(IP)

ν(IP), Optimal Integer Solution	15317.00
ν(LP), LP relaxation value	13082.00
Integrality gap, $\mid \nu(LP) - \nu(IP) \mid$ (absolute)	2235.00
Integrality gap, % difference from ν(IP)	14.59%
Solution Time for ν(LP)	522.89 seconds of computer time
Number of Iterations for ν(LP)	54326

The integrality gap for this problem is 14.59%. Such a large value of the integrality gap implies that the branch and bound algorithm will require extensive branching to reach the optimal integer optimal solution.

In Table 4.2, the solution values obtained for the above numerical example with and without the EST/LST constraints are compared. The percentage difference of the integrality gap from the optimal integer solution is 14.59% when the EST/LST constraints are not introduced in the problem. This difference is only 0.75% when the EST/LST constraints are incorporated into the problem. This shows that the EST/LST constraints are effective in reducing the integrality gap. The solution time and the number of iterations required to obtain the linear programming relaxation value are also reduced. Almost 80% of the variables are reduced in the problem due to the EST/LST constraints. This helps in reducing the problem size and the computational burden of the problem. This is the reason for the reduction in the solution time and the number of iterations required for obtaining the linear programming relaxation.

Table 4.2 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 1st numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Optimal Integer Solution	15317.00	15317.00
ν(LP), LP relaxation value	13082.00	15201.09
Integrality gap, | ν(LP) - ν(IP) | (absolute)	2235.00	115.91
Integrality gap, % difference from ν(IP)	14.59%	0.75%
Solution Time for ν(LP) {computer time in seconds}	522.89	66.59
Number of Iterations for ν(LP)	54326	10048
Number of Variables in the Problem	74202	14054
Number of Variables Eliminated by EST	0	34942
Number of Variables Eliminated by LST	0	25206
Number of Variables Eliminated by EST (%)	0.00%	47.09%
Number of Variables Eliminated by LST (%)	0.00%	33.97%

The EST/LST algorithm is applied on those scheduling problems for which the optimal solution was found, so that the integrality gap with and without the EST/LST constraints can be compared.

The scheduling problem in Table 4.2 has a static lot release policy. The scheduling problem in Table 4.3 has a dynamic lot release policy. Due to the modified formulation for the scheduling problem with a dynamic lot release policy, the solution values in Table 4.3 are negative. The scheduling problems in Tables 4.4, 4.5, 4.6, and 4.7 have static lot release policy and hence the solution values are positive.

Similar to Table 4.2, Tables 4.3 to 4.7 compare the integrality gap for each scheduling problem with and without the EST/LST constraints. The solution time and number of iterations required for the linear programming relaxation, with and without the incorporation of the EST/LST constraints are also compared. The number of variables eliminated by the EST/LST constraints, and hence the reductions of the problem size are noted. From each of the Tables 4.3-4.7, we can observe that the EST/LST constraints are effective in reducing the integrality gap and the number of variables in the problem.

Table 4.3 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 2nd numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Best Known Integer Solution	-14970	-14970
ν(LP), LP relaxation value	-18265.22	-15173.01
Integrality gap, | ν(LP) - ν(IP) | (absolute)	3295.22	203.01
Integrality gap, % difference from ν(IP)	22.01%	1.35%
Solution Time for ν(LP) {computer time in seconds}	424.65	172.56
Number of Iterations for ν(LP)	49781	20675
Number of Variables in the Problem	71336	31292
Number of Variables Eliminated by EST	0	34884
Number of Variables Eliminated by LST	0	5160
Number of Variables Eliminated by EST (%)	0.00%	48.90%
Number of Variables Eliminated by LST (%)	0.00%	7.23%

Table 4.4 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 3rd numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Best Known Integer Solution	15293	15293
ν(LP), LP relaxation value	12978	15143.36
Integrality gap, | ν(LP) - ν(IP) | (absolute)	2315	149.64
Integrality gap, % difference from ν(IP)	15.13%	0.97%
Solution Time for ν(LP) {computer time in seconds}	508.36	48.54
Number of Iterations for ν(LP)	52513	7715
Number of Variables in the Problem	73746	12685
Number of Variables Eliminated by EST	0	34356
Number of Variables Eliminated by LST	0	26705
Number of Variables Eliminated by EST (%)	0.00%	46.58%
Number of Variables Eliminated by LST (%)	0.00%	36.31%

Table 4.5 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 4th numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Best Known Integer Solution	16624	16624
ν(LP), LP relaxation value	14475	16229.13
Integrality gap, | ν(LP) - ν(IP) | (absolute)	2149	394.87
Integrality gap, % difference from ν(IP)	12.92%	2.37%
Solution Time for ν(LP) {computer time in seconds}	704.14	316.05
Number of Iterations for ν(LP)	69753	39072
Number of Variables in the Problem	81940	26150
Number of Variables Eliminated by EST	0	31195
Number of Variables Eliminated by LST	0	24595
Number of Variables Eliminated by EST (%)	0.00%	38.07%
Number of Variables Eliminated by LST (%)	0.00%	30.01%

Table 4.6 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 5th numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Best Known Integer Solution	15530	15530
ν(LP), LP relaxation value	12989	15251.86
Integrality gap, | ν(LP) - ν(IP) | (absolute)	2541	278.14
Integrality gap, % difference from ν(IP)	16.36%	1.79%
Solution Time for ν(LP) {computer time in seconds}	466.15	51.22
Number of Iterations for ν(LP)	51266	8001
Number of Variables in the Problem	73746	14340
Number of Variables Eliminated by EST	0	34381
Number of Variables Eliminated by LST	0	25025
Number of Variables Eliminated by EST (%)	0.00%	46.62%
Number of Variables Eliminated by LST (%)	0.00%	33.93%

Table 4.7 Comparison of performance parameters without and with the EST/LST constraints incorporated for the 6th numerical example

Performance Parameter	**Without EST/LST**	**With EST/LST**
ν(IP), Best Known Integer Solution	15283	15283
ν(LP), LP relaxation value	13013	15180.30
Integrality gap, | ν(LP) - ν(IP) | (absolute)	2270	102.7
Integrality gap, % difference from ν(IP)	14.85%	0.67%
Solution Time for ν(LP) {computer time in seconds}	422.10	40.29
Number of Iterations for ν(LP)	45788	7089
Number of Variables in the Problem	73746	14288
Number of Variables Eliminated by EST	0	34338
Number of Variables Eliminated by LST	0	25120
Number of Variables Eliminated by EST (%)	0.00%	46.56%
Number of Variables Eliminated by LST (%)	0.00%	34.06%

The integrality gap in the above problems with and without the EST/LST constraints is summarized below in Table 4.8. We can observe that the EST/LST constraints are very effective in reducing the integrality gap and the solution time required for the linear programming relaxation.

Table 4.8 Integrality gap with and without EST/LST constraints from above problems

PROBLEM	Integrality Gap %		Solution Time For LP Relaxation	
NUMBER	With EST/LST	Without EST/LST	With EST/LST	Without EST/LST
1	0.75%	14.59%	66.59	522.89
2	1.35%	22.01%	172.56	424.65
3	0.97%	15.13%	48.54	508.36
4	2.37%	12.92%	316.05	704.14
5	1.79%	16.36%	51.22	466.15
6	0.67%	14.85%	40.29	422.10

Thus, the incorporation of constraints for EST and LST of processing steps leads to a

- Significant improvement in the LP relaxation value of the scheduling problem
- Tighter formulation of the scheduling problem
- Reduction in the total number of variables
- Reduction in the computation time to obtain the LP relaxation value

4.4.15 Provision of Upper Bounds for the scheduling problem.

An integer feasible solution can be obtained by applying a standard dispatching rule to the scheduling problem. The solution that is provided by such heuristics is always integer feasible. This integer feasible solution can serve as an upper bound to the scheduling problem.

4.5 CONCLUDING REMARKS

1. Separate scheduling problems for the wafer fab under the static lot release policy and under the dynamic lot release policy are formulated.
2. The formulations take into account the unique features of a wafer fab (e.g. batching, lot dedication, and hot lots).
3. Real life features such as preventive maintenance and unscheduled workstation breakdown can be accommodated in the problem formulation.
4. The flow approach for formulating the objective function is explained.
5. The key concepts necessary for understanding the formulation are provided.
6. The scheduling problem is NP-Hard like most scheduling problems in the literature.
7. The incorporation of constraints for the EST and the LST of the processing steps of a lot contributes to a significant tightening of the LP relaxation.
8. The reduction of the integrality gap should lead to a significant improvement in the branch and bound procedure employed to solve the scheduling problem.
9. The amount of reduction in the integrality gap depends upon the problem data.
10. Integer feasible solutions to the scheduling problem obtained by applying dispatching rules to the scheduling problem can serve as upper bounds to the problem.
11. Various ways to exploit the structural properties of the scheduling problem should be explored to solve the scheduling problem within a reasonable amount of time. The research effort in this direction is presented in the next chapter.

CHAPTER 5: LAGRANGIAN HEURISTIC FOR SCHEDULING IN A WAFER FAB

5.1 INTRODUCTION

The scheduling problem for a wafer fab is formulated as a 0-1-integer program in chapter 4. The start times of lots on the machines are defined as binary variables. Separate formulations are presented for the wafer fabs under the static lot release policy and the dynamic release policy. This integer problem is NP-Hard and the time required to solve it increases exponentially with the size of the problem. The concept of the earliest start times and the latest start times of the processing steps of lots is utilized to tighten the LP relaxation of the scheduling problem. This provides a tighter lower bound on the optimal integer feasible solution. The upper bound on the optimal integer feasible solution can be obtained by applying any standard dispatching rule (FIFO, EDD, and LPR) to the scheduling problem.

The branch and bound procedure is not able to solve the scheduling problem in a time bounded by a polynomial in the size of the problem despite the tightening of the lower and upper bounds on the optimal integer feasible solution. Thus, the scheduling problem for a wafer fab remains computationally intractable. Hence, other methodologies should be explored which can be used either in conjunction with the branch and bound procedure or independently to solve the wafer fab scheduling problem.

5.2 MOTIVATION

It is well known in integer programming literature that combinatorial optimization problems come in two varieties. The “easy” problems can be solved in a time bounded by a polynomial in the size of the problem. The algorithms for the large class of “hard” problems require exponential time in the worst-case.

Many hard problems can be viewed as easy problems complicated by a relatively small set of side constraints. “Dualizing the side constraints produces a Lagrangian problem that is easy to solve and whose optimal value is a lower bound (for minimization problems) on the optimal value of the original problem. The Lagrangian problem can thus be used in place of a linear programming relaxation to provide bounds in a branch and bound algorithm" [9]. The optimal solution to the Lagrangian problem is often nearly feasible to the original problem and can be made feasible with some judicious tinkering.

The integer problem for the scheduling of a wafer fab can be viewed as an easy problem complicated by a set of constraints. This property of the scheduling problem should be examined by relaxing one set of constraints each time and by solving the relaxed problem. The ease of solving the relaxed problem should be studied and the Lagrangian relaxation approach to solve the scheduling problem should be pursued accordingly.

The theory of the Lagrangian relaxation approach is explained in the next section of the chapter. The adaptation and modification of the technique to the scheduling problem, the results and the conclusions are presented in the subsequent sections of this chapter.

5.3 LAGRANGIAN RELAXATION

5.3.1 Introduction

"Lagrangian relaxation is based upon the observation that many difficult integer programming problems can be modeled as a relatively easy problem complicated by a small set of constraints. To exploit this observation, a Lagrangian problem is created in which the complicating constraints are replaced with a penalty term in the objective function involving the amount of violation of the constraints and their dual variables." [10] The Lagrangian problem is relatively easy to solve and provides a lower bound (for a minimization problem) on the optimal value of the original problem.

The purpose in constructing and solving one or more dual problems is to make the search for a solution to the original problem more efficient by exploiting the following properties:

1. The duals are easier to solve than the primal.

2. Feasible dual solutions provide lower bounds to the primal problem objective function and those of any primal sub-problems derived during the search.
3. Lagrangian minimization may yield optimal or good, feasible solutions to the primal.

5.3.2 Basic Construction

Let the scheduling problem be represented as a general integer program P.
Let Z be the objective function value.

Problem P

$Z =$ $\min cx$

s.t. $Ax \leq b$

$Dx \leq e$

$x \geq 0$ and integral

where x is n X 1, b is m X 1, e is k X 1 and all other matrices have conformable dimensions. Let LP denote problem P with the integrality constraint on x relaxed and let Z_{LP} denotes the optimal value of LP.

The constraints of P are assumed to be partitioned into the two sets $Ax \leq b$ and $Dx \leq e$ so as to make it easy to solve the Lagrangian problem LR_u given below. The problem LR_u is "easy to solve" relative to P.

Problem LR_u

$Z_D(u) =$ $\min \{cx + u(Ax - b)\}$

s.t. $Dx \leq e$

$x \geq 0$ and integral

$u \geq 0$

where $u = (u_1, \ldots\ldots , u_m)$ is a vector of Lagrange multipliers.

$Z_D(u)$ is finite for all u because of the following assumptions in the problem LR_u

- P is feasible.
- The set $X = \{x \mid Dx \leq e, x \geq 0$ and integral$\}$ of feasible solutions to LR_u is finite.

It can be shown that $Z_D(u) \leq Z$ by assuming an optimal solution x* to P and observing that

$$Z_D(u) \quad \leq \quad cx^* + u(Ax^* - b) \quad \leq \quad Z$$

The inequality in this relation follows from

- The definition of $Z_D(u)$
- The equality from $Z = cx^*$ and
- The inequality from $Ax^* \leq b$.

5.3.3 Optimality Conditions

The underlying goal of the Lagrangian technique is to try to establish the following sufficient optimality conditions:

The pair (x*, u*), where x* is 0-1 and $u^* \geq 0$, satisfies the optimality conditions for the 0-1 Integer Problem P if

1. $Z(u^*) = cx^* + u^*(Ax^* - b)$
2. $u^*(Ax^*-b) = 0$
3. $Ax^* \leq b$

If the 0-1 solution x* satisfies the optimality conditions for some u*, then x* is optimal in Problem P.

5.3.4 Strategy for application of the Lagrangian technique

The strategy for the application of Lagrangian techniques is to first find an optimal u* in the dual problem. Once this is done, a complementary $x' \in X$ is explored for which the optimality conditions hold by calculating one or more solutions x satisfying $Z_D(u^*) = cx^* + u(Ax^* - b)$.

There is no guarantee that this strategy will succeed because

(a) There may be no u* optimal in the dual for which the optimality conditions can be made to hold for some $x' \in X$;
(b) The specific optimal u* calculated does not admit the optimality conditions for any $x^* \in X$;
(c) The specific x* (or x*'s) in X, selected by minimizing the Lagrangian, do not satisfy the optimality conditions of the original problem although some other $x'' \in X$ which is sub optimal to the Lagrangian may satisfy them [28].

5.3.5 Relation between the Lagrangian solution and the original problem

Proposition 5.1

The value of the optimal solution to the scheduling problem P is always greater than or equal to the value of the corresponding feasible solution in the Lagrangian problem.

i.e. $Z_D(u^*) \leq Z$

Proof:

For an optimal solution x* to problem P

$Z = cx^*$

$Ax^* \leq b$

$Ax^* - b \leq 0$

For the corresponding feasible solution in the Lagrangian problem LR_u

$u^* \geq 0$

$u^*(Ax^* - b) \leq 0$

$\therefore cx^* + u(Ax^* - b) \quad \leq \quad cx^* \quad = \quad Z$

$Z_D(u^*) = cx^* + u^*(Ax^* - b)$

$\therefore Z_D(u^*) \quad \leq \quad Z$

QED

The relation $Z_D(u) \leq Z^*$ will always hold good, but with the non-convex structure of the primal problem there can be no guarantee that $Z_D(u) = Z^*$. In general, it is not possible to guarantee finding u for which $Z_D(u) = Z$, but it may happen for particular problem instances. If $Z_D(u)$ cannot produce an optimal solution to Z, then a duality gap is said to exist between the primal and the dual problem.

Proposition 5.2

If the value of the optimal solution to the scheduling problem P is always greater than the value of the optimal dual solution in the Lagrangian (dual) problem

i.e. $Z_D(u^*) < Z$, then a duality gap exists between the primal and the dual problem.

5.3.6 Parameters affecting the Lagrangian solution

A number of parameters need to be decided before applying the Lagrangian relaxation.

- Selection of an appropriate value for u
- Obtaining a value for u for which $Z_D(u)$ is equal or nearly equal to Z
- Choice between competing relaxations, i.e. different Lagrangian relaxations and the linear-programming relaxation
- Using LR_u to obtain feasible solutions for P.
- Quality of the solutions obtained.
- Integrating the lower and upper bounding capabilities of the Lagrangian problem with branch and bound methodology.

5.3.7 Determining optimal value of the Lagrange multipliers

The duality gap between the primal problem and the dual problem should be reduced as much as possible. Hence, the scheme for determining u should have as it's objective, of obtaining optimal or near optimal solutions to the dual problem D.

Problem D

$Z_D = \max_u Z_D(u)$

s.t. $Dx \le e$

$x \ge 0$ and integral

$u \ge 0$

where $u = (u_1, \ldots\ldots, u_m)$ is a vector of Lagrange multipliers.

Problem D has a number of important structural properties that make it feasible to solve. The set $X = \{x \mid Dx \le e, x \ge 0 \text{ and integral}\}$ of feasible solutions for LR_u is assumed to be finite. Hence, the set X can be represented as $X = \{x^t, t = 1, \ldots, T\}$.

Problem D can now be expressed as the following linear program with many constraints.

Problem $\overline{D}$

$Z_D = \max w$,

$w \le cx^t + u(Ax^t - b), t = 1, \ldots, T$

Proposition 5.3

The value of the solution to the dual Lagrangian problem LR_u is always less than or equal to the optimal value of the problem P.

Proof:

The optimal solution to the problem P is Z*. The optimal solution to the dual problem is $Z_D(u^*)$. A duality gap may exist between the primal and the dual problem (from Proposition 5.2). Hence, $Z_D(u^*) \le Z^*$

For a value of u^t, the value of $Z_D(u) = cx^t + u^t(Ax^t - b)$.The optimal value of the dual problem $\overline{D}$, $Z_D = \max_u Z_D(u)$.

$\therefore Z_D = Z_D(u^*)$

$\therefore Z_D(u^*) = \max_u Z_D(u)$

$\therefore Z_D(u) \le Z_D(u^*)$

$\therefore Z_D(u) \le Z^*$

QED

Proposition 5.4

The Lagrangian solution always provides a lower bound to the original problem P, because $Z_D(u) \leq Z^*$.

Proposition 5.5

The scheme for determining u (i.e. solution of the dual problem LR_u) obtains optimal or near optimal solutions to the problem D (maximizes the lower bound).

The LP dual of $\overline{D}$ is a linear program with many columns.

Problem $\overline{P}$

$$Z_D = \min \sum_{t=1}^{T} \lambda_t c x^t$$

$$\sum_{t=1}^{T} \lambda_t A x^t = b$$

$$\sum_{t=1}^{T} \lambda_t = 1$$

$\lambda_t \geq 0$, t = 1, …. , T

Problem $\overline{P}$ with λ_t required to be integral is equivalent to P, although Problems $\overline{P}$ and LP generally are not equivalent problems.

Both Problem $\overline{D}$ and $\overline{P}$ are important constructs in the formulation of algorithms for solving Problem D due to the following observation:

Problem $\overline{D}$ shows that Z_D(u) is the lower envelope of a finite family of linear functions.

The function $Z_D(u)$ has all the nice properties, like continuity and concavity that make life easy for a hill-climbing algorithm, except for differentiability. The function is non-differentiable at any $\overline{u}$ where $LR_{\overline{u}}$ has multiple optima. Although $Z_D(u)$ is differentiable almost everywhere, it generally is non-differentiable at an optimal point.

The form of $Z_D(u)$ is shown in Figure 5.1 for m = 1 and T = 4.

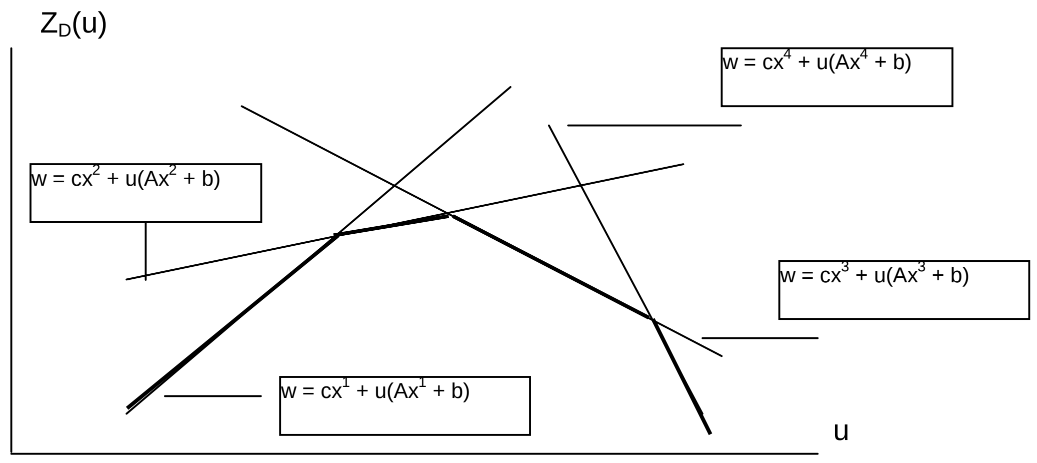

Figure 5.1 The form of $Z_d(u)$ [9]

Even though the function $Z_D(u)$ is not differentiable everywhere, directional derivatives exist in all directions at any point $u \in R^m$. Directions with positive directional derivative are the ones to use in ascent algorithms.

An m-vector y is called a subgradient of $Z_D(u)$ at $\bar{u}$ if it satisfies

$$Z_D(u) \quad \leq \qquad Z_D(\bar{u}) + y\,(u - \bar{u}), \qquad \forall u$$

Any subgradient of $Z_D(u)$ at $u = \bar{u}$ points into the half-space containing all optimal solutions to the dual problem; namely, $(u: (u - \bar{u})y \geq 0\}$ contains all optimal dual solutions. Thus, it is possible to construct ascent algorithms for the dual problem of the form $u^{k+1} = u^k + t_k y^k$, where y^k is a subgradient of $Z_D(u)$ at $u = u^k$, and t_k satisfies

$$Z(u^k + t_k y^k) \quad = \qquad \max_{t \geq 0} Z(u^k + t y^k)$$

The vector $(Ax^t – b)$ is a subgradient at any u for which x^t solves LR_u. Any other subgradient is a convex combination of these primitive subgradients. With this perspective, the well-known result that u^* and λ^* are optimal for $\overline{D}$ and $\overline{P}$ if and only if they are feasible and satisfy a complementary slackness condition can be seen to be equivalent to the obvious fact that u* is optimal for D if and only if 0 is a subgradient of $Z_D(u)$ at u*[9].

Subgradient relaxation methods can be viewed as an ascent algorithm of this general type, where t_k is chosen by a different rule. The theoretical difficulty is, however, that Z may not increase in the direction y^k although u^k is not optimal and Z increases in the direction of another subgradient.

The advantage of subgradient relaxation over the primal-dual ascent algorithm is the elimination of computational overhead. One disadvantage is the absence of guaranteed monotonically increasing lower bounds.

The subgradient method is a brazen adaptation of the gradient method in which gradients are replaced by subgradients. Given an initial value u^0 a sequence $\{u^k\}$ is generated by the rule

$$u^{k+1} = u^k + t_k(Ax^k - b)$$

where x^k is an optimal solution to LR_{u^k} and t_k is a positive scalar step size.

The justification for the subgradient relaxation method rests with its computation effectiveness. This method has proven effective for many problems.

Because the subgradient method is easy to program and has worked well on many practical problems, it has become the most popular method for D. There have also been many papers, such as Camerini et al. [6], that suggest improvements to the basic subgradient method.

The fundamental theoretical result is that $Z_D(u^k) \rightarrow Z_D$ if $t_k \rightarrow 0$ and $\sum_{i=0}^{k} t_i \rightarrow \infty$. The step size t_k used most commonly in practice is

$$t_k = \frac{\lambda_k (Z' - Z_D(u^k))}{\left\| A x^k - b \right\|^2}$$

where λ_k is a scalar satisfying $0 < \lambda_k \leq 2$ and Z' is an upper bound on Z_D, frequently obtained by applying a heuristic to P. Often the sequence λ_k is determined by setting $\lambda_0 = 2$ and halving λ_k whenever $Z_D(u)$ has failed to increase in some fixed number of iterations. This rule has performed well empirically, even though it is not guaranteed to satisfy the sufficient condition given above for optimal convergence [9].

Unless we obtain a u^k for which $Z_D(u^k) = Z^*$, there is no way of proving optimality in the subgradient method. To resolve this difficulty, the method is usually terminated upon reaching a specified iteration limit.

There is no guarantee when using subgradient optimization that $Z(u^{k+1}) > Z(u^k)$ although practice has shown that increasing lower bounds can be expected on most steps under the correct combination of artistic expertise and luck. Thus, the subgradient optimization is essentially a heuristic method with theoretical as well as empirical justification [28].

5.3.8 Selecting between competing relaxations

Two properties are important in evaluating a relaxation:

1. The sharpness of the bounds produced.
2. The amount of computation required for obtaining these bounds.

Usually selecting a relaxation involves a tradeoff between these two properties; sharper bounds require more time to compute. It is generally difficult to know whether a relaxation with sharper bounds but greater computation time will result in a branch and bound algorithm of better overall performance.

5.3.9 Obtaining Feasible Solutions

In the course of solving Problem D, it is possible that a solution to Problem LR_u will be discovered that is feasible in Problem P. Because the dualized constraints contain some inequalities, a Lagrangian problem solution can be feasible but non-optimal for P. However, it is rare that a feasible solution of either type is discovered. On the other hand, it often happens that a solution to LR_u obtained while optimizing D will be nearly feasible for P and can be made feasible with some judicious tinkering. Such a method might be called a Lagrangian heuristic.

$Z_D(u)$ is expected to be much easier to compute than Z because of the special form of X. The number of elementary operations required to compute $Z_D(u)$ is bounded by a polynomial of parameters of the problem. The algorithm may be quite efficient empirically and derived from some simple dynamic programming recursion or list processing scheme.

5.3.10 Quality of the solution obtained

The answer to this question that is available in the literature is completely problem specific and largely empirical. The reader is referred to [9] for detailed information.

5.3.11 Integration with the Branch and Bound Algorithm

Lagrangian techniques can be applied in a fail-safe manner, if they are embedded in branch and bound searches. For some discrete optimization problems, it is possible to strengthen the dual problem if it fails to yield an optimal solution to the primal solution. Under certain conditions, the dual can be successively strengthened until the optimality conditions are found to hold.

5.3.12 Tightening the dual problem

The common practice of relaxation by simply throwing away some of the constraints is equivalent to Lagrangian relaxation with $u = 0$. Permitting $u \neq 0$, but always $u \geq 0$, allows the relaxation to be tighter. Restricting the solutions permitted in the Lagrangian minimization to be a strict subset of the zero-one solutions can strengthen the dual problem. Introduction of surrogate constraints can strengthen the dual problem.

Definition of Surrogate Constraints:

A surrogate constraint is an inequality implied by the constraints of an integer program, and designed to capture useful information that cannot be extracted from the parent constraints individually but is nevertheless a consequence of their conjunction [13].

5.3.13 Conclusion

“Lagrangian relaxation is a systematic exploitation of the formal Lagrangian dual problem in integer programming. This dual problem need not be solved optimally and need not be devoid of a duality gap in order to be useful. It provides a means for fathoming, range reduction, generating improved feasible solutions, and guiding separation” [9]. In the next section of this chapter we will explore the adaptation of the Lagrangian technique for obtaining a solution to the scheduling problem for a wafer fab.

5.4 LAGRANGIAN RELAXATION TECHNIQUE FOR THE SCHEDULING PROBLEM OF A WAFER FAB

The scheduling problem for a wafer fab is a prime candidate for the application of the Lagrangian relaxation technique. We need to study and adapt the parameters affecting the Lagrangian relaxation technique to the scheduling problem.

5.4.1 Selection of the Relaxed Constraint

The selection of the constraint to be relaxed is the initial step in the adaptation of the Lagrangian technique to the scheduling problem.

5.4.1.1 Constraint for Unique processing of a step of a job

The relaxation of this constraint essentially frees the scheduling problem from the necessity to schedule any job.

Proposition 5.6

The solution value of the Lagrangian problem for scheduling of lots in a wafer fab, with the constraint for “unique processing of a step of a job” relaxed, and with $u^0 = 0$ is zero.

Proof:

The removal of the unique processing constraint removes, from the scheduling problem, the restriction of scheduling any step of the lots in the system. The objective function of the problem is to minimize the sum of the starting time of the steps of all the lots. With the removal of the necessity to schedule a step of a lot, the problem does not schedule any step of any lot, thus, minimizing the objective function.

Hence, none of the steps of any lots are scheduled. The value of X_{fijkbt} variable (in the scheduling problem formulated in chapter 4) for all the steps j of all the lots i of all the product f at any time unit t is zero. Hence, the sum of the X_{fijkbt} variable for all the steps j of all the lots i of all the product f at any time unit t is zero. The objective function value is zero. QED

Proposition 5.7

Relaxation of the "Constraints for unique processing of a step of a job" leads to a solution in which none of the operations of any job are scheduled.

Such a solution is clearly infeasible in the original problem. The selection of the appropriate dual variables will lead to a much better solution in the Lagrangian problem.

Proposition 5.8

The optimal value of the dual variable u^0 in the first iteration of the subgradient method is

u^0 = EST for each step of each lot of every product to be scheduled in the given planning period.

Proof:

The objective function for the scheduling problem formulated in chapter 4 is:

$$\text{Min} \sum_{f=1}^{F} \sum_{i=1}^{I_f} \sum_{j \in J_{fi}} \sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} t * X_{fijkbt}$$

Where the subscripts are as per their definition in chapter 4

The constraint for the unique processing of a job in the scheduling problem is:

$$\sum_{k\in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} X_{fijkbt} = 1 \qquad f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\, j \in J_{fi}$$

Let c' denote the objective function coefficient t, which is the time unit in the problem at which the steps of a lot may be scheduled. Clearly c' = t ≥ 0 for all the steps of all the lots.

Let x denote the objective function variable X_{fijkbt}, the objective function be denoted as Σc'x, and the constraint be denoted as Σx = 1, i.e. Σx – 1 = 0. Hence, the problem P can be defined as min Σc'x.

The Problem LR_u is defined as min {Σc'x – u(Σx – 1)} i.e. the Problem LR_u is defined as min {Σc'x –Σux + u}. Hence, the solution x = 0, i.e. no step is scheduled, is optimal in the Lagrangian problem LR_u for any u satisfying u ≤ c'.

Setting u^0 = min(c') for each step of each lot of every product is better than u^0 = 0 and maximizes the lower bound over all u for which x = 0 is optimal in the Lagrangian problem. The minimum c' for each step is the EST for that step.

Thus, u^0 = EST is optimal for each step of each lot to be scheduled in the planning period in the first iteration of the sub-gradient optimization method. QED

Proposition 5.9

The solution value of the Lagrangian problem for the scheduling of lots in a wafer fab, with the constraint for "unique processing of a step of a job" relaxed and with optimal value for u^0 in the first iteration of the subgradient method, is equal to the sum of the EST for each of the steps of each lot of each product.

Proof:

The optimal value of the dual variable u^0 in the first iteration of the subgradient method is u^0 = EST of each step of each lot of every product in the wafer fab in the planning period (from Proposition 5.8). For this optimal value of u^0, the value of x is zero.

The objective function can be defined as min {$\Sigma c'x - \Sigma ux + u$}. Hence, the value of the objective function is u.
u represents the sum of u^0 = EST of each step of each lot to be scheduled in the given planning period.

Hence, the solution value with optimal value of u^0 in the first iteration of the subgradient method is the sum of the EST for each of the steps of each lot of every product. QED

5.4.1.2 Capacity constraint

The capacity constraint is the largest set of constraints in the scheduling problem. For a moderate sized wafer fab with 18 workstations and 240 time units, the number of capacity constraint equations is 4338. Hence, the relaxation of this constraint results in elimination of a large number of constraints.

The optimal solution of the Lagrangian problem may not yield a solution that is feasible in the original problem. Hence, the Lagrangian solution may have to be modified by a heuristic. To obtain a feasible solution to the original problem, the heuristic will have to eliminate the infeasibility of those capacity constraints that have been violated. A large number of capacity constraints may have to be adjusted to obtain a feasible solution to the original problem and this may lead to non-optimal decisions by the heuristic.

Hence, we do not pursue the relaxation of the capacity constraint in our approach.

Proposition 5.10
The solution value of the Lagrangian problem for scheduling of lots in a wafer fab, with the capacity constraints relaxed, and with $u^0 = 0$ is the sum of the EST of all the steps of all the lots of every product in the wafer fab during the planning period.
Proof:
The objective function of the scheduling problem is the minimization of the sum of the start time of each step of each lot of every product in the system in the planning period. Hence, the scheduling problem tries to schedule each step as early as possible.

With the removal of the capacity constraint, each workstation in every station family can process multiple lots at the same time. Hence, each step can start processing from its EST.

Therefore, the solution value is the sum of the EST for all the steps of every lot of every product during the planning period. QED

5.4.1.3 Sequential undertaking of the jobs

The relaxation of this constraint leads to a schedule in which the sequential processing of the steps of a lot may not be followed.

Proposition 5.11

The solution value of the Lagrangian problem for scheduling of lots in a wafer fab with the sequential constraints relaxed, and with $u^0 = 0$ is greater than or equal to the sum of the EST of all the steps of all the lots in the planning period.

Proof:

The EST for the steps of a lot that were established in chapter 4 follows the sequential order of the steps of a lot. The objective function of the scheduling problem is the sum of the start time of each step of each lot of every product in the system in the planning period. Hence, the scheduling problem tries to schedule each step as early as possible.

The capacity constraint ensures that the machine capacity is not violated. Hence, to satisfy the capacity constraint, some of the steps of a few lots have to be scheduled later than their EST (they cannot be scheduled before the EST). Therefore, the start time of some operations of some lots may occur after their EST.

Even though the steps of a lot may not follow the sequential order of their routes when the sequential constraints are relaxed, all the steps will be scheduled at or after their EST.

Hence, the sum of the start time of the all the operations of all the lots in the system will be atleast the sum of the EST of all the steps of all the lots in the planning period. QED

Proposition 5.12

The solution value of the Lagrangian problem for scheduling of lots in a wafer fab with the sequential constraints relaxed and with $u^0 = 0$ is greater than or equal to the solution value of the corresponding Lagrangian problems with

1. The capacity constraints relaxed and $u^0 = 0$
2. The "unique processing of a step of a job constraint relaxed and $u^0 = EST$.

Proof:

The solution value of the Lagrangian problem, with the constraint for "unique processing of a step of a job" relaxed and optimal value for u^0 is the sum of the EST for each of the steps of each lot of each product in the planning period. (Proposition 5.9)

The solution value of the Lagrangian problem, with the capacity constraints relaxed, and with $u^0 = 0$ is the sum of the EST for each of the steps of each lot of each product in the planning period. (Proposition 5.10)

The solution value of the Lagrangian problem, with the sequential constraints relaxed, and with $u^0 = 0$ is greater than or equal to the sum of the EST for each of the steps of each lot of each product in the planning period. (Proposition 5.11)

Hence, the solution value of the Lagrangian problem for scheduling of lots in a wafer fab with the sequential constraints relaxed and with $u^0 = 0$ is greater than or equal to the solution value of the corresponding Lagrangian problems with

1. The capacity constraints relaxed
2. The "unique processing of a step of a job" constraint relaxed. QED

The optimal solution obtained for this relaxed problem may not be feasible in the original problem. But the only adjustment that needs to be made is that the operations of a job should be arranged in a sequential manner. The sequence of steps for a lot is known from the route of the lot and hence this adjustment can be accomplished easily. The capacity constraints ensure that the capacity of the workstations is not violated. Also the number of constraints for the sequential undertaking of the jobs is less (eg. 109 equations for the moderate sized wafer fab in Appendix C). Hence the adjustments in the solution for such a small number of constraints can be accomplished easily.

Also, the Lagrangian problem with the sequential constraints relaxed provides a higher lower bound than the corresponding Lagrangian problem with the capacity constraints or the “unique processing of a step of a job constraint” relaxed (from Proposition 5.12).

Hence, the constraints defining the sequential undertaking of the jobs are relaxed in the Lagrangian relaxation technique for the scheduling problem.

In the sequential undertaking of the jobs for a scheduling problem under the static lot release policy, Operation j+1 of any lot must start only after the completion of operation j of that lot. That is,

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} (t + p_{fijk}) * X_{fijkbt} \leq \sum_{k' \in K(f,i,j+1)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} t * X_{fij+1k'b't}$$

f = 1,2,….,F i = 1,2,…..,I_f $\forall$ j $\in$ J_{fi}

In the sequential undertaking of the jobs for a scheduling problem under the dynamic lot release policy, Operation j+1 of any lot must start only after the completion of operation j of that lot. That is,

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} (T - t - p_{fijk}) * X_{fijkbt} \geq \sum_{k' \in K(f,i,j+1)} \sum_{b'=1}^{b_{k'}} \sum_{t=0}^{T} (T - t) * X_{fij+1k'b't}$$

f = 1,2,….,F i = 1,2,…..,I_f $\forall$ j $\in$ J_{fi}

5.4.1.4 Numerical Example

The numerical example presented in chapter 4 is considered once again. The following sets of constraints from the scheduling problem are independently relaxed in separate Lagrangian problems. The value of the Lagrangian multiplier is $u^0 = 0$.

Constraint 1: Unique processing of an operation of a job
Constraint 2: Sequential undertaking of the jobs
Constraint 3: Capacity constraint

Table 5.1 Summary of results obtained by relaxing different constraints

Relaxed Constraint	Integer Solution	Solution Time	Iterations	Nodes
Constraint 1	0	12.04 sec	0	0
Constraint 2	15012	40.00 sec	479	2
Constraint 3	14838	3.14 sec	131	0

An integer feasible solution for the scheduling problem is obtained in a reasonable amount of time when each constraint is relaxed. The relaxed scheduling problems are easy to solve.

The Lagrangian problem with the “Sequential undertaking of the jobs” constraint relaxed provides the highest lower bound amongst the 3 problems (as proved in Proposition 5.12). Note that the solutions of the Lagrangian problems are not the optimal solutions to the Lagrangian problem.

5.4.2 Determination of the optimal value of the Lagrangian multipliers

The subgradient optimization method is chosen for determining the optimal solution of the Lagrangian multiplier u as discussed in section 5.3.7.

Given an initial value u^0 a sequence $\{u^k\}$ is generated by the rule
$u^{k+1} = u^k + t_k(Ax^k – b)$
where x^k is an optimal solution to $LR_u{}^k$ and t_k is a positive scalar step size given by

$$t_k = \frac{\lambda_k(Z' - Z_D(u^k))}{\left\| A x^k - b \right\|^2}$$

- The value of u^0 is set equal to zero.
- The value of λ_0 is set equal to 2. Whenever the value of $Z_D(u)$ fails to increase, the value of λ_k is halved and the optimization method continued.
- The value of Z’ is determined by applying any dispatching rule to the scheduling problem.

Unless a value for u^k is obtained for which $Z_D(u^k) = Z^*$, there is no way of proving optimality in the subgradient method. To resolve this difficulty, the method is usually terminated upon reaching a specified iteration limit.

For the scheduling problem on hand, a limit of 10 on the number of iterations was found to be sufficient to obtain a near optimal solution for u. The value of step size t approaches 0 as the number of iterations approaches 10. Hence, we terminate the subgradient optimization procedure after 10 iterations. No significant improvement in the Lagrangian solution is observed for number of iterations greater than 10.

The optimal value of the Lagrangian solution $Z_D(u^*)$ is greater than $Z_D(u^k)$ for all other u. due to the concave structure of the dual problem $\overline{D}$. From the ten iterations performed for optimizing the Lagrangian problem via the subgradient optimization method, the maximum value of $Z_D(u^k)$ is selected as the optimal solution of the Lagrangian problem.

5.4.3 Tightening of the dual problem

Introducing surrogate constraints can tighten the dual problem further.

Consider the objective function of Problem P: Z = Minimize cx. The optimal solution to this problem will be less than or equal to a solution Z' obtained by applying a dispatching rule to the problem. The Lagrangian problem can be tightened by enforcing a constraint that the solution obtained by solving the Lagrangian problem is less than or equal to Z'. That is,

1. $$c\,x \leq Z'$$

This will lead to better Lagrangian solutions and the Lagrangian solution obtained will be atleast as good as Z'.

The relaxed constraint in the Lagrangian problem is: $Ax \leq b$. This constraint can be expressed as

$$\sum_j a_{ij} * x_{ij} \quad \leq \quad b_i \qquad \forall i$$

If the optimal solution for the Lagrangian problem is feasible to the original problem, then all the “i” constraints, as defined above, are satisfied. Hence, the summation of all the “i” constraints for a solution feasible to both the Lagrangian problem and the original problem is given by:

$$\sum_i \sum_j a_{ij} * x_{ij} \quad \leq \quad \sum_i b_i \qquad \forall i, \quad \forall j$$

But, an optimal solution for the Lagrangian problem is seldom feasible in the original problem. Hence, some of the constraints from the set of relaxed constraints may not be satisfied. Hence, the “i” constraints from the optimal solution of the Lagrangian problem can be split into 2 sets i’ and i’’. The set i’ represents those constraints that are feasible to both the Lagrangian problem and the original problem. The set i’’ represents those constraints that are feasible to only the Lagrangian problem. Hence,

$$\sum_j a_{ij} * x_{ij} \quad \leq \quad b_i \qquad \forall i \in i'$$

$$\sum_j a_{ij} * x_{ij} \quad > \quad b_i \qquad \forall i \in i''$$

Inspite of the above, the summation of all the “i” rows for an optimal solution to only the Lagrangian solution may be

$$\sum_i \sum_j a_{ij} * x_{ij} \quad > \quad \sum_i b_i \qquad \forall i, \quad \forall j \quad i = i' \cup i''$$

This aggregate constraint is clearly not feasible for a solution to the original problem. Hence, the following constraint can be incorporated to tighten the Lagrangian problem.

2. $$\sum_i \sum_j a_{ij} * x_{ij} \quad \leq \quad \sum_i b_i \qquad \forall i, \quad \forall j$$

The above constraint is just the summation of all the rows of the relaxed constraint.

In a solution feasible to the original problem, all the “i” rows of the relaxed constraint are satisfied. Hence, the summation of any i’’’ number of rows out of the “i” rows of the relaxed constraint will provide an aggregate constraint such that

$$\sum_{i'''} \sum_{j} a_{i'''j} * x_{i'''j} \leq \sum_{i'''} b_{i'''} \qquad \forall i''' \in i, \quad \forall j$$

Intuitively, aggregating a sub set of relaxed constraints rather than aggregating all the relaxed constraints will tighten the dual problem further. Such a sub set can be obtained for the processing steps of each lot. Thus, we can aggregate the relaxed constraints for the processing steps of each lot. Hence, the sequential constraints for the processing steps of each lot can be aggregated and formulated as the following surrogate constraints:

Let I represent the total number of lots in the system, i’ represent a lot in the system. Hence, i’ $\in$ I, $J_{i'}$ represent the set of processing steps of the lot i’ and $j_{i'}$ represent a processing step from the set $J_{i'}$. Hence, $j_{i'} \in J_{i'}$, and we have,

3. $$\sum_{j_{i'}} a_{i'j_{i'}} * x_{i'j_{i'}} \leq \sum_{j_{i'}} b_{j_{i'}} \qquad \forall i' \in I, \quad \forall j_{i'} \in J_{i'}$$

Thus, the constraint is aggregated for only those processing steps of each lot that are included in the scheduling problem. This surrogate constraint is expected to be tighter than the surrogate constraint 2.

It is intuitive from the definition of surrogate constraints 2 and 3 that whenever constraint 3 is satisfied, constraint 2 will be satisfied, but the reverse may not be true. Hence, constraint 3 is stronger than constraint 2. Consequently, we incorporate only surrogate constraints 1 and 3 in the Lagrangian problem.

The table below shows successive tightening of the dual problem due to addition of surrogate constraints on the example problem in Appendix C.

Table 5.2 Summary of results obtained by introducing various surrogate constraints in the scheduling problem

Surrogate Constraint	**u^0**	**$Z_D(u)$**	**Solution Time**	**Iterations**
None	0	15012	36.01 sec	481
2	0	15020	261.39 sec	1895
3	0	15039	45.48 sec	624

5.4.4 Feasibility and Quality of the Optimal Lagrangian Solution

The optimal solution of the Lagrangian problem may not be feasible for the original problem. Hence, there exists a need to develop an approach to judiciously tinker with the optimal Lagrangian solution and make it feasible to the original problem. This methodology is presented in the next section of this chapter. A Lagrangian heuristic for solving the scheduling problem is presented. Important properties of the heuristic like the validity, feasibility, tractability and quality of the solution obtained are examined.

5.4.5 Methodology for the Lagrangian Heuristic

The sub-gradient optimization method is used to obtain the optimal solution to the Lagrangian problem. The optimal solution for the Lagrangian problem is obtained by repeatedly solving the problem over the dual variable u. The sub-gradient optimization method does not guarantee an optimal solution to the Lagrangian problem. Hence the solution procedure is terminated after a certain number of iterations. Based on the experimental study on the numerical example in Appendix C, the solution procedure is terminated after ten iterations and the maximum value of the objective function, over the iterations of the sub-gradient optimization method is chosen as the optimal solution to the Lagrangian problem.

The optimal solution for the Lagrangian problem may not be feasible for the original scheduling problem because some of the processing steps of a job may not follow the sequential precedence constraints. Thus, the optimal solution for the Lagrangian problem may be a partially feasible solution, where some of the processing steps follow the sequential constraints and some do not.

Consider a lot i of product f. The processing steps to be performed on the lot in the planning period are j = {1,2,…,J_{fi}}. Let p_{fij} be the processing time of step j of lot i of product f. The EST algorithms in chapter 4 have established the EST for each of the processing steps. Let t_{fij} be the EST of step j of lot i of product family f.

$\therefore 0 \le t_{fi1} < t_{fi2} < t_{fi3} < t_{fi4} < \ldots\ldots < t_{fiJfi}$

or

$0 \le t_{fi1} + p_{fi1} = t_{fi2}$

$t_{fi2} + p_{fi2} = t_{fi3}$

and so on.

Now consider the schedule generated by the optimal Lagrangian solution.

The best schedule for step j of any lot i of any product f will always have the steps scheduled at their EST so that the objective function value is minimized. This schedule for steps j = {1,2,…,J_{fi}} of lot i of product f will be feasible both in the Lagrangian problem and the original scheduling problem, because it satisfies the sequentiality constraints. Figure 5.2 illustrates such a schedule for steps j of lot i of product f.

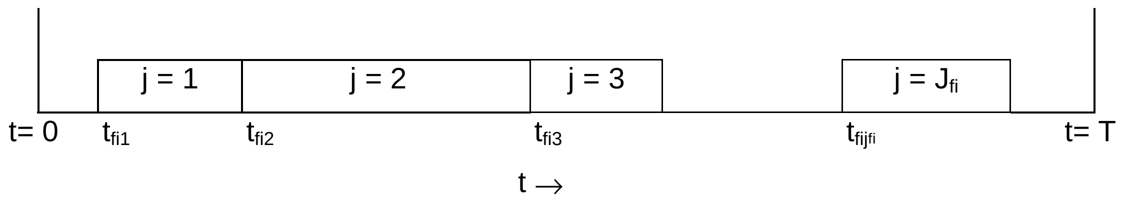

Figure 5.2 EST schedule for steps j of lot i of product f

The schedule in figure 5.2 may not be generated by the solution to the Lagrangian problem. Let $0 \le t'_{fi1}, t'_{fi2}, t'_{fi3}, t'_{fi4}, \ldots\ldots, t'_{fiJfi}$ be the point in time when step j of lot i of product f is scheduled by the Lagrangian solution.

Because of the EST constraints incorporated in the model,

$t_{fi1} \le t'_{fi1}$

$t_{fi2} \le t'_{fi2}$

and so on.

The schedule will be feasible in the original scheduling problem only if the following sets of conditions are satisfied:

$0 \le t'_{fi1} + p_{fi1} \le t'_{fi2}$

$t'_{fi2} + p_{fi2} \le t'_{fi3}$

and so on.

Figure 5.3 shows the schematic depiction of such a schedule.

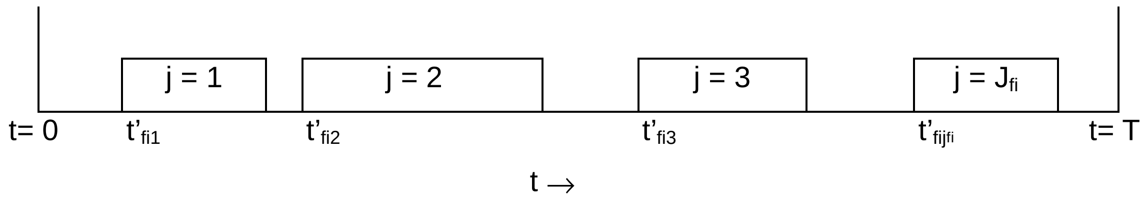

Figure 5.3 Feasible schedule for steps j of lot i of product f in Lagrangian solution

Both Figures 5.2 and 5.3 depict schedules that are feasible for step j of lot i of product f in both the Lagrangian problem and the original scheduling problem.

It is possible that such feasible schedules may not be generated by the Lagrangian solution, as the sequential constraints are relaxed in the scheduling problem. It is likely that the schedule generated by the Lagrangian solution may violate some of the sequential constraints for step j of lot i of product f and may satisfy some of the them.

Let $0 \le t''_{fi1}, t''_{fi2}, t''_{fi3}, t''_{fi4}, \ldots\ldots, t''_{fiJfi}$ be the point in time when step j of lot i of product f is scheduled by the Lagrangian solution.

Because of the EST constraints incorporated in the model,

$t_{fi1} \le t''_{fi1}$

$t_{fi2} \le t''_{fi2}$

and so on.

The schedule may or may not be feasible in the original problem. The feasibility conditions are presented in the table below

Feasibility Conditions	Infeasibility Conditions
$t'_{fi1} + p_{fi1} \leq t'_{fi2}$	$t'_{fi1} + p_{fi1} > t'_{fi2}$
$t'_{fi2} + p_{fi2} \leq t'_{fi3}$	$t'_{fi2} + p_{fi2} > t'_{fi3}$
And so on	And so on

The schedule will be feasible in the original problem for some of the steps and infeasible for the remaining steps. Figure 5.4 depicts such a schedule.

Note that for lot i of product f, the schedule developed from the solution to the Lagrangian problem will be feasible in the original scheduling problem only up to the earliest point in time when the sequential constraint between two successive processing steps j' and j' + 1 is violated. The start time of the processing steps that are downstream of step j'+1 will be affected by the start time of step j'+1. Since the start time of step j'+1 violates the sequential constraint, the start time of the processing steps downstream of step j'+1 should also be considered as violating the sequential constraints. Hence, the schedule for all the processing steps downstream of step j' + 1, including step j'+1 will be considered infeasible.

In figure 5.4, the infeasible portion of the schedule for lot i of product f is displayed with a shaded background.

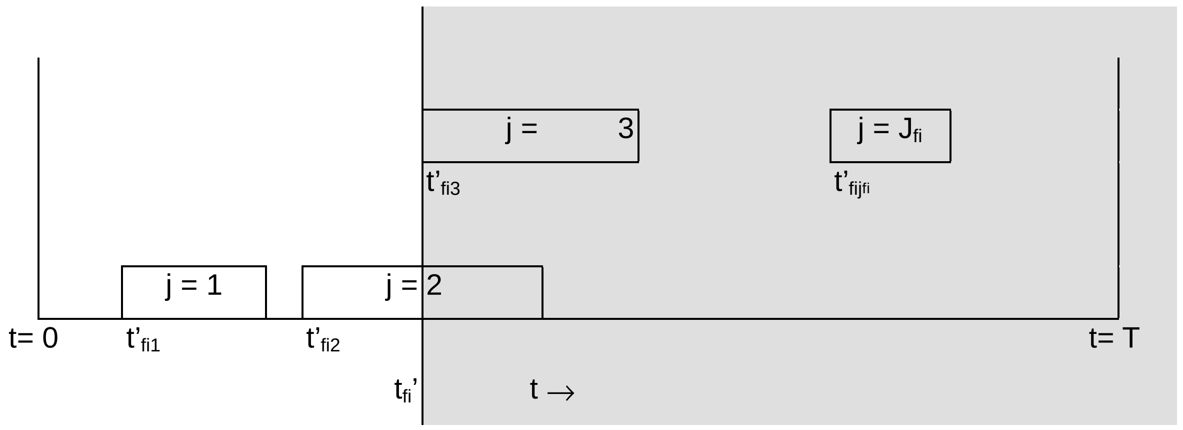

Figure 5.4 Schedule for steps j of lot i of product f in the Lagrangian solution

Thus, from the solution of the Lagrangian problem, we can obtain the feasible portion of the schedule for the lot i of a product f. The schedule will be feasible up to point t_{fi}’ and this schedule is frozen and accepted. The entire schedule after point t_{fi}’ will be infeasible.

By repeating the same procedure for all the lots in the system, we get the feasible portion of the schedule for each lot of every product. Figure 5.5 shows the feasible and infeasible portions (shaded) of the schedule for each lot in the system.

The portion of the schedule that will be feasible and infeasible will be different for each lot. The lots interact between each other indirectly via the capacity constraint. Hence, the entire schedule will be infeasible after the first point in the planning period at which the infeasibility occurs. For the Figure 5.5, this point in the planning period is t’. The entire schedule for all the lots will be infeasible after point t’;

$$t' = \min_{f,i} t_{fi}'$$

The feasible portion of the schedule is frozen and the start times of the steps of the lots are fixed for the feasible portion of the schedule. Based upon the start times and hence completion times of the steps of the lots, the EST for the steps in the remaining portion (infeasible portion) of the schedule can be updated.

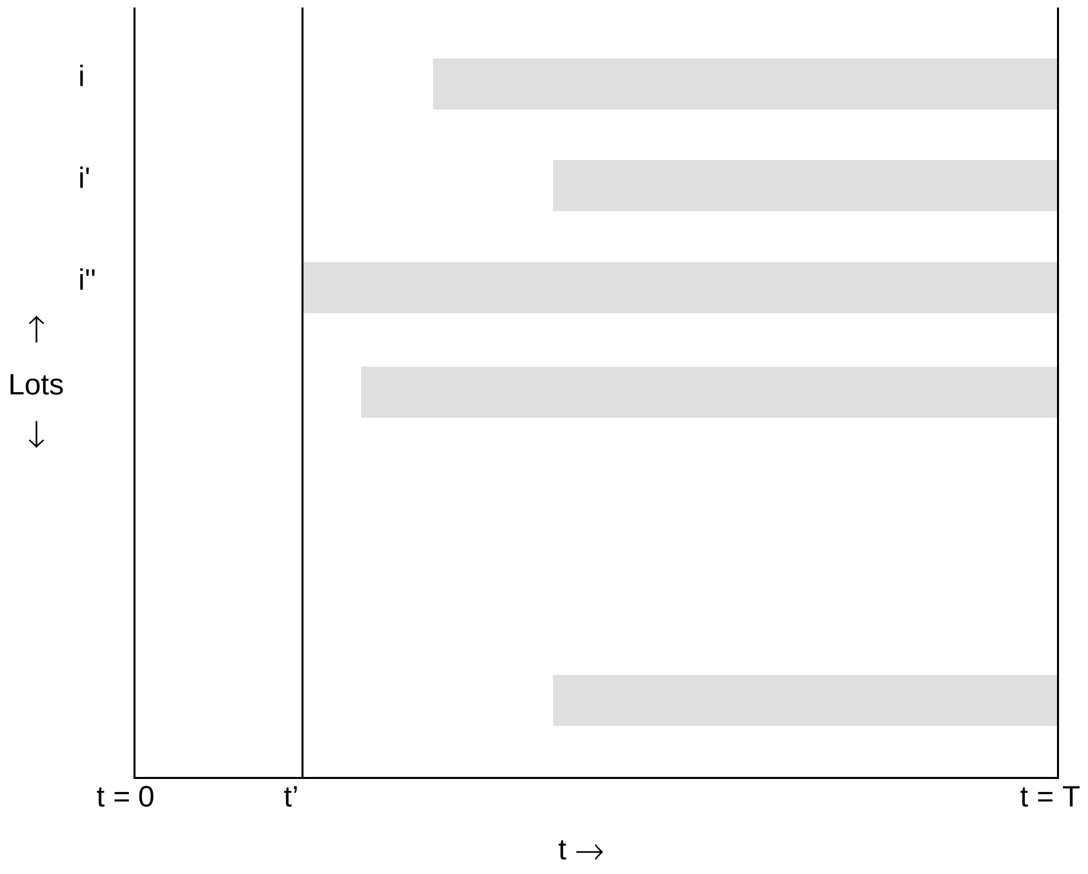

Figure 5.5 Feasible and infeasible portions of the schedules for all the lots from the Lagrangian solution

A step j of lot i of product f may be scheduled at time t'_{fij} in the feasible portion of the schedule. The EST of the step j is t_{fij}.

$\therefore t'_{fij} \geq t_{fij}$.

Consider the case where $t'_{fij} > t_{fij}$.

$\therefore t'_{fij} + p_{fij} > t_{fij} + p_{fij}$

$\therefore C_{fij} > t_{fij+1}$

where C_{fij} is the completion time of step j.

Thus, the step j is completed after the EST of step j+1. The starting time of step j has been fixed by the Lagrangian heuristic. Hence, step j is going to be completed after the

EST of step j+1. Hence, the EST of step j+1 and all other steps downstream of step j+1 should be revised.

Hence the EST of processing step j +1 and all the processing steps downstream of step j+1 that will be scheduled in the infeasible portion of the schedule are updated.

Let t''_{fij+1}, t''_{fij+2}, ….., t''_{fiJfi} be the modified EST of steps j+1, j+2,…J_{fi} of lot i of product f.
$t''_{fij+1} = t'_{fij} + p_{fij}$
$t''_{fij+2} = t''_{fij+1} + p_{fij+1}$
and so on.

A truncated Lagrangian problem is obtained which can be solved by the sub-gradient optimization method. Thus, at the end of the first stage we obtain a feasible portion of the schedule and a truncated Lagrangian problem.

This procedure is repeated at every stage until all the steps of all the lots of all the product families are scheduled. The validity and feasibility of this approach are discussed in the subsequent section.

5.4.6 Validity of the approach for the Lagrangian heuristic

The Lagrangian heuristic for scheduling of lots in a wafer fab solves the scheduling problem by a stage by stage approach. The optimal solution for the Lagrangian problem is examined for feasibility in the original problem. The earliest point in time t' where the sequential constraints are violated is observed and the schedule of the entire set of processing steps feasible to be scheduled before t' is frozen.

This approach needs to be compared vis-à-vis fixing the schedule of only a sub set of processing steps, in different Lagrangian problems, from the entire set of the feasible steps that can be fixed, and then solving the different Lagrangian problems separately.

Conjecture 1: The quality of solution, obtained as a result of fixing the entire set of earliest feasible variables from the optimal solution to the Lagrangian problem at each stage of the Lagrangian heuristic is as good as that obtained by fixing sub sets of the above set in different Lagrangian problems and solving them separately in each stage.

Proof:

The value of the binary variable X_{fijkbt} is one if step j of lot i of product f is scheduled on workstation b of station family k at time unit t, and zero otherwise as per the notation for the scheduling problem formulated in chapter 4.

We let the variable x denote the binary variable X_{fijkbt}. If the processing step j of lot i of product f is feasible on workstation b of station family k at time unit t in the Lagrangian solution, then $X_{fijkbt} = x = 1$. The entire set of x variables that are obtained in the solution to the Lagrangian problem are denoted by x^t. The set of earliest feasible variables from the solution of the Lagrangian problem is $x'^t \in x^t$.

The Lagrangian problem is defined as:

Problem LR$_u$

$Z_D(u) =$ $\min \{cx + u(Ax - b)\}$

s.t. $Dx \leq e$

$x \geq 0$ and integral

$u \geq 0$

where $u = (u_1, \ldots\ldots, u_m)$ is a vector of Lagrange multipliers.

$Z_D(u)$ is finite for all u because of the following assumptions in the problem LR$_u$

- P is feasible.
- The set $X = \{x \mid Dx \leq e, x \geq 0 \text{ and integral}\}$ of feasible solutions to LR$_u$ is finite.

The dual problem D is defined as:

Problem D

$Z_D = \max_u Z_D(u)$

The set X can be defined as X = $\{x^t, t = 1,\ldots,T\}$. Thus, the dual problem D is expressed as the following linear program with many constraints:

Problem $\overline{D}$

$Z_D = \max w$,
$w \leq cx^t + u(Ax^t - b), t = 1, \ldots, T$ - (1)

The form of $Z_D(u)$ as a function of u is shown in Figure 5.1. Thus, for each $x^t \in X$, there exists the Lagrangian solution as a function of u, which lies along the line given by (1). The optimal solution to the Lagrangian problem, considers all $x^t \in X$, t = 1,2,..,T, and produces an optimal value of u* and a set of variables $x^* \in X$ that represent the start times of the steps of the lots. Optimization of the dual problem $\overline{D}$ results in the maximum value of the function $Z_D(u)$. Thus the highest lower bound for the original scheduling problem is obtained, i.e. the value of Z_D is obtained. The duality gap that may exist between the primal and dual problem is the least when the optimal solution of the dual problem $\overline{D}$ is obtained.

From the set $x^* \in X$, only a few variables $x'^* \in x^*$, will be the earliest feasible in the original problem. When all the earliest feasible variables $x'^* \in x^*$ are fixed in the original problem, the optimal solution to the Lagrangian problem is implemented up to the first violation of feasibility of the sequential constraints in the scheduling problem.

Consider the truncated Lagrangian problem from time unit t to T. Consider the optimal solution to the Lagrangian problem $x^* \in X$, the set of earliest feasible variables $x'^* \in x^*$, and the subset of variables $x''^* \in x'^*$. The values of the variables x* interact to provide an optimal solution to the Lagrangian problem. Together with the set $x'^* \in x^*$ and $x''^* \in x'^*$, the solution x* is optimal to the Lagrangian problem. Thus, the sub set x''* and x'* also interact to provide an optimal solution to the Lagrangian problem.

When the optimization solver is solving the Lagrangian problem, it chooses the subset x''*, sets x'* and x* to provide the optimal solution to the Lagrangian problem. With reference to the subset x''*, the optimal solution to the Lagrangian problem has to contain the sets x'* and x*.

Thus, if only a subset of earliest feasible variables x''* $\in$ x'* is fixed in the Lagrangian problem and the subsequent truncated Lagrangian problem is solved, the optimal solution of the subsequent truncated Lagrangian problem has to contain the sets x'* and x*. Even if the truncated Lagrangian problems are solved with different subsets x'* fixed, the optimization solver has to choose the set x'* and x* to provide the optimal solution to the Lagrangian problem. Hence, even if the sub sets x''* are fixed in different Lagrangian problems, the optimal solution to the subsequent truncated Lagrangian problems will contain the solution x'* and x*, and finally the entire set x'* will get fixed. Thus, fixing sub sets of earliest feasible variables results in the entire set of earliest feasible variables being fixed.

Hence, in the Lagrangian heuristic, fixing only a sub set of variables x''* does not lead to a better solution and hence, the entire set of variables x''* can be fixed in each stage. QED

Conjecture 2: Fixing the entire set of earliest feasible processing steps leads to much better computational performance than fixing a subset of earliest feasible processing steps in a Lagrangian problem and solving different Lagrangian problems.

Proof:
Let the number of processing steps to be scheduled in a single planning period be N.

For the strategy of fixing all the feasible processing steps in the optimal solution of the Lagrangian problem in each stage k, let the number of stages required to schedule the entire N processing steps be n. Let n_1, n_2,…n_k.…,n_n be the number of processing steps fixed in stage k of the Lagrangian heuristic.
$\therefore n_1 + n_2 + \ldots + n_k + \ldots n_n = N$

Thus, the Lagrangian heuristic solves n Lagrangian problems to completely solve the scheduling problem. Fixing all the feasible processing steps will lead to successive reduction of the feasible region. Each truncated Lagrangian problem will be smaller than the previous truncated Lagrangian problem. Hence, the Lagrangian problem will become easier to solve in each successive stage of the solution methodology. This will lead to shorter computation times for the branch and bound algorithm in obtaining the optimal dual solution in each successive stage.

For the strategy of fixing only a subset of processing steps in different truncated Lagrangian problems from the set of feasible processing steps in the original Lagrangian problem, only those processing steps can be fixed that are scheduled at the same time in the planning period and are the earliest scheduled processing steps from the set of feasible processing steps. The feasible processing steps that are scheduled after the above processing steps cannot be fixed because the schedule for the earliest portion of the planning period should be fixed first.

The best case scenario for the strategy of fixing only a subset of processing steps in different truncated Lagrangian problems from the set of feasible processing steps in the original Lagrangian problem, is that each processing step is scheduled at different points in time in the planning period. Thus, only one feasible processing step will be scheduled earliest amongst all the feasible processing steps. Hence, only the one earliest feasible processing step is fixed in the one truncated Lagrangian problem at the end of each stage. If this procedure is continued for each stage, then the number of stages required for solving the Lagrangian problem will be equal to the total number of processing steps to be scheduled. Thus, in the best-case scenario, the number of stages required to solve the Lagrangian problem will be N.

Now, $n \leq N$. Hence, the strategy of fixing only one variable will be as good as the strategy of fixing all the variables only in the best case possible.

The worst case scenario for the strategy of fixing the subsets of processing steps in different truncated Lagrangian problems from the set of feasible processing steps in the original Lagrangian problem, is that each processing step is scheduled at the same point in time in the planning period and the subset size is one. Only one feasible processing step out of all the feasible processing steps is scheduled in a truncated Lagrangian problem. Thus the number of truncated Lagrangian problems is N. If this procedure is continued in each stage, the number of truncated Lagrangian problems will be N!.

The truncated Lagrangian problem obtained by fixing only one processing step will be definitely more harder than the problem obtained by fixing all the feasible processing steps. Also $n < N!$.

Hence, fixing all the feasible processing steps leads to much better computational performance than fixing one feasible processing step in a Lagrangian problem and solving different Lagrangian problems. QED

5.4.7 Feasibility of the Schedule Generated

The Lagrangian heuristic generates the solution to the scheduling problem by a stage by stage approach. The property of the heuristic to always generate feasible solutions should be established.

Proposition 5.13

In every stage of the Lagrangian heuristic, atleast one processing step is feasible.

Proof: (by contradiction)

Consider a truncated Lagrangian problem from time $t',\ldots, T = 240$ and $t' \geq 0$. Let us assume that the solution of the truncated Lagrangian problem will be such that no processing step is feasible.

The EST of each step for all lots to be scheduled in the truncated planning period from $t',\ldots, T = 240$ has been updated based on the start times of the steps scheduled in the planning period from $0,\ldots,t'$.

Let $j \in \{j_{fi}, j_{fi}+1,\ldots,J_{fi}\}$ be the steps of lot i of product f to be scheduled in the truncated planning period from $t',\ldots, T = 240$.

Let $t_{fij}, t_{fij+1},\ldots, t_{fiJ}$ be the EST for the processing steps j_{fi} of lot i of product f.

Let $p_{fij}, p_{fij+1},\ldots, p_{fiJ}$ be the processing time for step j_{fi} of lot i of product f.

$t_{fij} \geq t'$

$t_{fij+1} = t_{fij} + p_{fij}$

$t_{fij+2} = t_{fij+1} + p_{fij+1}$

and so on.

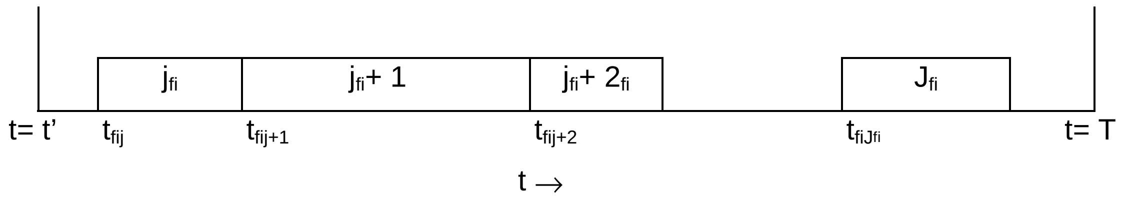

Figure 5.6 EST for steps j of lot i of product f

Let t'_{fij}, t'_{fij+1},…., t'_{fiJ} be the time at which the processing steps j_{fi} of lot i of product f is scheduled.

Case 1: $t'_{fij} + p_{fij} \leq t'_{fij+1}$

Thus, step j_{fi} +1 is scheduled after the completion of step j_{fi}, i.e. the schedule is feasible for the step j_{fi} and j_{fi} +1 for lot i.

If such a schedule is observed for step j_{fi} and j_{fi} +1 for any of the lots i in the system, then the schedule for steps j_{fi} and j_{fi} +1 will be feasible for those lots. Even if the sequential constraints between the steps step j_{fi} +1 and j_{fi} +2 are not satisfied, still we have a feasible schedule in the planning period for step j_{fi} and j_{fi} +1. Contradiction.

Hence we can freeze the schedule from t' to the first violation of the sequential constraint in the planning period, and move on to the resulting truncated Lagrangian problem.

Case 2: $t'_{fij} + p_{fij} > t'_{fij+1} > t'_{fij}$

Thus, step j_{fi} +1 is scheduled after the step j_{fi} is scheduled, but before the completion of step j_{fi}, i.e. the schedule is feasible for the step j_{fi}, but not for step j_{fi} +1 for lot i.

Even if the sequential constraints between the steps step j_{fi} +1 and j_{fi} +2 are not satisfied, still we have a feasible schedule in the planning period for step j_{fi}. Contradiction

Hence we can freeze the schedule from t' to the first violation of the sequential constraint in the planning period, and move on to the resulting truncated Lagrangian problem.

Case 3: $t'_{fij} > t'_{fij+1}$
Thus, step j_{fi} +1 is scheduled before the step j_{fi} is scheduled, i.e. the schedule is not feasible for the step j_{fi} and hence any other steps downstream of step j_{fi} for lot i.

Consider the worst case of this solution, i.e. the step j_{fi} +1 is scheduled before the step j_{fi} is scheduled for all the lots i. Also, let the step j_{fi} +1 be scheduled at its EST.
Thus, the starting time of step j_{fi} +1 is t_{fij+1}.
Step j_{fi} is scheduled at some time $t'_{fij} > t_{fij+1}$.
The EST of all other steps downstream of j_{fi} +1 is greater than or equal to $t_{fij+1} + p_{fij+1}$.
Thus, no processing step is processed is scheduled in the time period from t' to t_{fij+1}.

Thus, the system is idle from t' to t_{fij+1}.

The objective function of the scheduling problem is to minimize the start times of each step of each lot in the system. Hence, the Lagrangian problem will always try to schedule the processing step as close to its EST as possible, subject to capacity constraints.

We observe that in case 3, the system is idle from t' to t_{fij+1} and still step j_{fi} is scheduled after time t_{fij+1}.

The EST of step j_{fi} is t_{fij}.
$t' \le t_{fij} < t_{fij+1}$

Hence, to minimize the objective function value, the Lagrangian problem will schedule step j_{fi} at some time $t' \le t_{fij} < t_{fij+1}$

Thus the schedule generated, will schedule atleast one operation in the time period $t' \leq t_{fij} < t_{fij+1}$. Contradiction

Hence, atleast one operation will be feasible and the feasible duration of the scheduling problem is fixed and the next truncated Lagrangian problem is obtained. QED

In the scheduling problem under the static lot release policy, the processing steps to be scheduled in the planning period are determined apriori. Thus, the capability of the Lagrangian heuristic to definitely schedule these processing steps should be proved.

The Lagrangian problem is solved by a stage by stage approach. The first feasible portion of the schedule in the planning period is accepted at each stage and the remaining portion of the planning period is solved once again as a Lagrangian problem. Thus the feasible portion of the schedule in the planning period cumulatively increases at the end of each stage. The property of the Lagrangian heuristic to generate schedules such that no processing step determined apriori remains unscheduled at the end of the Lagrangian heuristic needs to be established.

Proposition 5.14

The Lagrangian heuristic will always generate a feasible schedule for the scheduling problem under the static lot release policy.

Proof:

Consider a truncated Lagrangian problem from time t',…., T = 240 and $t' \geq 0$.

The EST of each step for all lots to be scheduled in the truncated planning period from t',…., T = 240 has been updated based on the start times of the steps feasibly scheduled in the planning period from 0,…,t'.

The LST of each step for all lots to be scheduled in the truncated planning period from t',…., T = 240 are established from the LST algorithm in chapter 4.

Let $j \in \{j_{fi}, j_{fi}+1,\ldots,J_{fi}\}$ be the steps of lot i of product f to be scheduled in the truncated planning period from t',…., T = 240.

Let t_{fij}, t_{fij+1},…., t_{fiJ} be the EST for the processing steps j_{fi} of lot i of product f.

Let l_{fij}, l_{fij+1},…., l_{fiJ} be the LST for the processing steps j_{fi} of lot i of product f.

Let p_{fij}, p_{fij+1},…., p_{fiJ} be the processing time for step j_{fi} of lot i of product f.

$t_{fij} \geq t'$

$t_{fij+1} = t_{fij} + p_{fij}$

$t_{fij+2} = t_{fij+1} + p_{fij+1}$

and so on.

$l_{fiJ} \leq T$

$l_{fiJ-1} = l_{fiJ} - p_{fiJ-1}$

$l_{fiJ-2} = l_{fiJ-1} - p_{fiJ-2}$

and so on.

Also, $t_{fij} \leq l_{fij}$ $\forall j$

The schematic depiction for the EST and LST is shown in Figure 5.7

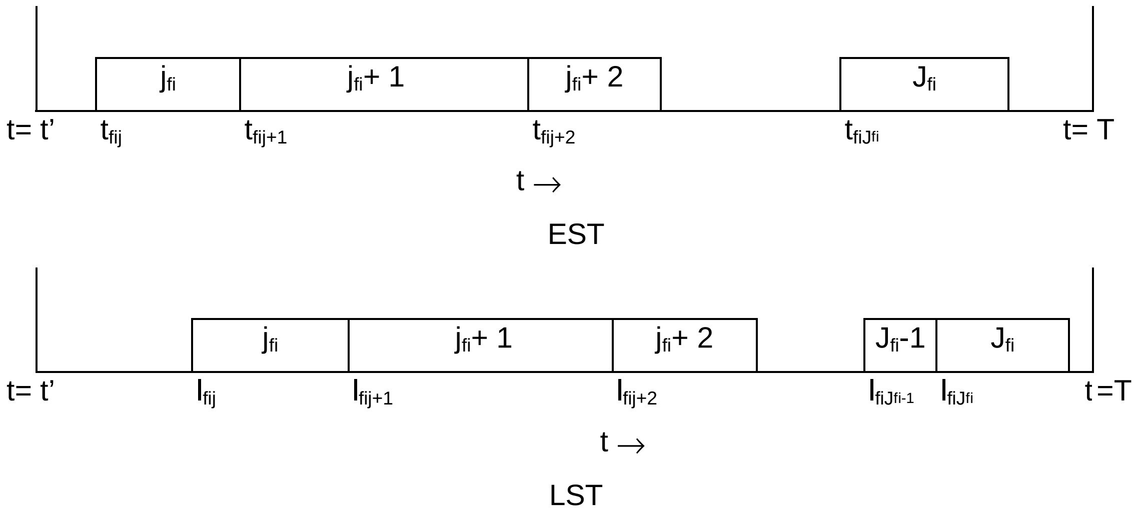

Figure 5.7 EST and LST for steps j of lot i of product f

There is sufficient capacity in the system to schedule all the processing steps determined apriori. This property is ensured by the capacity constraint in the tardiness problem formulated in chapter 3.

The binary variable X_{fijkbt} is one if step j of lot i of product f is scheduled on workstation b of station family k at time unit t, and zero otherwise, as per the notation for the scheduling problem formulated in chapter 4.

The value of the subscript t will range from the EST to the LST of the step j of lot i of product f, i.e. from t_{fij} to l_{fij}.

Let t'_{fij}, t'_{fij+1},…., t'_{fiJ} be the time at which the processing steps j_{fi} of lot i of product f is scheduled.

$\therefore\ t_{fij} \leq t'_{fij} \leq\ l_{fij}$

The “unique processing of a step” constraint in the scheduling problem is given by:

$$\sum_{k \in K(f,i,j)} \sum_{b=1}^{b_k} \sum_{t=0}^{T} X_{fijkbt} = 1 \qquad f = 1,2,\ldots,F \qquad i = 1,2,\ldots,I_f \qquad \forall\ j \in J_{fi}$$

Hence, one of the binary variables has to take a value of one and that will be the time when the processing on that step will be started. Hence, the processing step has to be scheduled in the planning period.

The LST that is established for a step j ensures that, even if the step j is scheduled at its LST, there is sufficient capacity in the system to schedule the steps downstream of step j, in the planning period.

The objective function of the scheduling problem is to minimize the start times of each step of each lot in the system. Hence, the Lagrangian problem will always try to schedule the processing step as close to its EST as possible, subject to capacity constraints.

Thus the starting times for the processing steps will be scheduled as close to the starting time unit t’ of the truncated Lagrangian problem.

Hence, the schedule will be clustered towards the start of the truncated planning period. Hence, there exists excess capacity towards the end of the planning period. Some of the processing steps will satisfy the sequential constraint while some will not.

As the problem is solved stage by stage, the infeasibilities of the processing steps are removed stage by stage.

The infeasible processing steps are carried over to the next truncated problem. But due to the LST constraints, these steps will be eventually scheduled at and before their LST, and their exists sufficient capacity to schedule all the processing steps downstream from the steps that are scheduled at their LST.
Thus, this iterative process schedules all the processing steps in the problem. QED.

Proposition 5.15
The Lagrangian heuristic will always generate a feasible schedule for the scheduling problem under the dynamic lot release policy.
Proof:
In the scheduling problem under the dynamic lot release policy, the processing steps of a lot may or may not be scheduled during the current planning period. Thus, the Lagrangian heuristic does not need to schedule all processing steps in the scheduling problem.

From Proposition 5.13, the Lagrangian heuristic will always generate a feasible solution in each stage. Thus, the processing steps that are scheduled by the Lagrangian heuristic will always satisfy the sequential constraints and hence the schedule will be feasible. The processing steps that are not scheduled during the current planning period are carried over to the next planning period. QED

The Lagrangian heuristic for scheduling of lots in a wafer fab is presented below. The solution quality and the computational tractability of the heuristic will be studied in the subsequent section.

5.4.8 Lagrangian Heuristic for Scheduling in a Wafer Fab

Step 1: Formulate the Lagrangian problem by relaxing the sequential precedence constraints and incorporating the surrogate constraints formulated in 5.4.3. The problem is formulated for time period t = 0,1,2,…,T=240.
Step 2: Set $u^0 = 0$, and $\lambda_0 = 2$. Set n = 1
Step 3: Solve the Lagrangian problem and obtain an integer feasible solution. n = n+1
Step 4: Compute the new dual variable by the subgradient optimization method
Step 5: Repeat steps 3 and 4 while $n \leq 10$.
Step 6: Choose the solution with the maximum value of $Z_D(u)$ and accept it as the optimal solution.
Step 7: Obtain the schedule for this optimal solution to the Lagrangian problem. Examine the schedule for feasibility in the original scheduling problem.
Step 8: Note the earliest point in time t' (refer figure 5.5) when the first sequential precedence constraint is violated by any step of any lot in the system. Fix the schedule of those steps that were scheduled before time t' in the planning period.
Step 9: Modify, if necessary, the EST of the steps that are to be scheduled based on the completion times of the steps fixed in step 8.
Step 10: Formulate the truncated Lagrangian problem from t = t',……,T.
Step 11: Repeat steps 2 to 10 until t' = T or a feasible schedule for the original problem is obtained for the planning period t = t',……,T.
Step 12: End of algorithm.

5.4.9 Solution Quality of the Schedule Generated

The solution generated by the Lagrangian heuristic is definitely expected to be atleast as good as the dispatching rule used to obtain an initial feasible solution Z' in the subgradient optimization method. This is so because of the surrogate constraint 1 incorporated into the Lagrangian problem.

The performance of the Lagrangian heuristic in terms of the solution quality and the nearness to the optimal solution needs to be studied further by applying the heuristic to more numerical examples.

5.4.10 Computational Tractability of the Algorithm

The scheduling problem is a NP-Hard problem and hence the computational tractability of the Lagrangian heuristic is of great importance.

In the worst case scenario while using the Lagrangian heuristic, in each iteration, only one operation over all the lots may be fixed by the optimal solution of the Lagrangian problem. Thus, the number of iterations required for completely solving the Lagrangian problem will be equal to the number of operations to be scheduled in a single planning period. In every iteration, ten sub-iterations are performed over the dual variable u which have been decided based on the experimentation on the numerical example in Appendix C.

For each of these ten sub-iterations, the termination criteria for the branch and bound algorithm used to obtain an integer feasible solution has to be decided. The termination criterion based on observation of the numerical example in Appendix C is as follows:

Terminate the branch and bound at time t minutes = max{20 min, time for 1st integer feasible solution}
Where t = the computation time for the CPLEX optimization solver

This criterion should be changed according to the hardness of the scheduling problem.

Let the number of operations to be scheduled be given as M.

Hence the total time required to obtain a feasible schedule in the worst case is

$$M*10*t \text{ minutes.}$$

Proposition 5.16
In the worst case, the computation time required to obtain a feasible schedule for the scheduling problem by the Lagrangian heuristic is bounded by a polynomial in the size of the problem.

Hence the Lagrangian heuristic methodology used to schedule lots in a wafer fab is definitely computationally tractable.

5.4.11 Numerical Example

The numerical example from Appendix C is solved to illustrate the performance of the Lagrangian heuristic. The numerical example solved is for a wafer fab with a static lot release policy. Hence, the operations to be scheduled in the planning period are known. In all, these are 131.

The table below shows the optimal value of the Larangian solution at each stage. Each stage corresponds to one iteration of the heuristic. The ten sub-iterations required for computing the optimal value of the Lagrangian problem at each stage (iteration) are given in Appendix D.

Table 5.3 Summary of the solution of each stage of the Lagrangian problem

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	15107.57	15093	3411.85	15	15
2	15209.68	15187	144.28	14	29
3	15246.00	15226	173.19	19	48
4	15293.00	15264	47.59	20	68
5	15331.15	15329	45.44	28	96
6	15346.21	15332	18.43	7	103
7	15343.86	15340	2.9	1	104
8		15351	0	27	131

Thus, the value of the solution generated by the Lagrangian heuristic is 15351. The total computation time for the heuristic is 3843.68 seconds.

The optimal value of the problem is 15317, which has been obtained by solving the problem by the CPLEX optimization solver. The computation time required for obtaining the optimal solution is quite large compared to the computation time required for the Lagrangian heuristic.

Thus, the Lagrangian heuristic has generated a solution which is 0.22 % from the optimal solution.

The Lagrangian heuristic has the potential to generate optimal or near-optimal solutions to the scheduling problem. Further tests on additional sets of problems should be conducted to determine the average performance of this heuristic.

5.5 EXPERIMENTS AND RESULTS

Four experiments are conducted to study the performance of the Lagrangian heuristic. The detailed data for each experiment is presented in Appendix F.

All the experiments are performed on systems that have reached a steady state. The systems have a starting WIP and hence are said to be “full”. The scheduling problems are solved for dynamic lot release policy because it represents the real life situations.

The performance measures used to evaluate the solution are the average and standard deviation of the cycle time. Each experiment has multiple products. Hence, the cycle time of each product type is computed separately. The solution obtained by the Lagrangian heuristic is compared to the solution obtained by using the standard dispatching rules available in the literature.

The Lagrangian heuristic is denoted as LAG in the tables comparing the solution of the Lagrangian heuristic with the standard dispatching rules.

A scheduling problem is solved for each day of the experiment. The Lagrangian heuristic provides a feasible solution to the scheduling problem. The detailed solution for each scheduling problem is presented in Appendix F.

Experiments 1 and 2 have the same wafer fab system, and are conducted to study the performance of the Lagrangian heuristic in a wafer fab in which:

- Different products in the system have different routes and processing times.
- Each product passes through the same bottleneck station.
- The product mix that is inputed into the system is different for each planning period.

The difference between experiment 1 and 2 is the different product mix that is fed into the system during each planning period in each experiment.

Experiment 3 is conducted to study the performance of the Lagrangian heuristic in a wafer fab in which:

- Different products pass through different bottlenecks.
- Different products in the system have different routes and processing times.
- The product mix that is inputed into the system is different for each planning period.
- The wafer fab system is completely different from the one in experiments 1 and 2.

Experiment 4 is conducted to study the performance of hot lots in the wafer fab system. Hot lots of the different products are input into the system during different planning periods.

5.5.1 EXPERIMENT 1

The system is utilized to the maximum extent possible (i.e. as many lots as possible are released into the system without violating the capacity constraints). Tables 5.4 and 5.5 summarize the problem data and the lot release policy respectively for this experiment.

Table 5.4 Summary of wafer fab system in experiment 1

Number of products	2
Number of planning periods (days)	3
Number of station families	8
Number of workstations	18

Table 5.5 Lot Release Policy in experiment 1

DAY	**Product Number**	**Number of lots released in a single time period (M_i)**
1	1	0
	2	10
2	1	9
	2	1
3	1	5
	2	5

In Table 5.6, the cycle time and the standard deviation of the cycle time obtained from the Lagrangian heuristic is compared to the corresponding values obtained using the standard dispatching rules. The standard dispatching rules and their abbreviation are listed below.

FIFO – First In First Out
LBA – Least Balance Ahead
LLA – Least Lots Ahead
LPR – Least percentage of processing time remaining
LS – Least Slack
LTR – Least Time Remaining

The best results for each performance measure for each product are shaded.

Table 5.6 Summary of results of experiment 1

PRODUCT #	CYCLE TIME	FIFO	LBA	LLA	LPR	LS	LTR	LAG
1	AVG	39:58:30	38:09:36	38:09:36	37:55:12	37:53:34	37:54:00	37:45:51
	STD DEV	0.12	0.13	0.13	0.15	0.15	0.16	0.15
2	AVG	40:14:34	38:09:26	38:09:26	37:22:43	39:20:18	37:22:43	37:29:34
	STD DEV	0.34	0.32	0.32	0.33	0.31	0.33	0.32
LOTS	COMPLETED	26	27	27	27	27	28	28

On the average, the solution obtained by the lagrangian heuristic performs better than most of the standard dispatching rules.

5.5.2 EXPERIMENT 2

The system is under utilized by releasing less number of lots than the maximum capacity of the system. Tables 5.7 and 5.8 summarize the problem data and the lot release policy respectively for this experiment.

Table 5.7 Summary of wafer fab system in experiment 2

Number of products	2
Number of planning periods (days)	3
Number of station families	8
Number of workstations	18

Table 5.8 Lot Release Policy in experiment 2

DAY	**Product Number**	**Number of lots released in a single time period (M_i)**
1	1	5
	2	2
2	1	5
	2	2
3	1	4
	2	4

In Table 5.9, the cycle time and the standard deviation of the cycle time obtained by the lagrangian heuristic is compared to the corresponding values obtained using the standard dispatching rules. The best results for each performance measure for each product are shaded.

Table 5.9 Summary of results of experiment 2

PRODUCT #	CYCLE TIME	FIFO	LBA	LLA	LPR	LS	LTR	LAG
1	AVG	41:19:42	41:28:14	40:35:18	40:41:00	39:04:06	43:31:49	40:24:00
	STD DEV	4:27:15	6:35:29	5:38:34	6:17:28	5:15:52	6:13:39	5:33:04
2	AVG	35:19:43	33:45:26	33:58:17	36:12:00	37:59:09	29:51:26	30:55:43
	STD DEV	4:26:03	4:01:09	4:00:43	6:53:25	6:16:08	1:23:59	2:20:32
LOTS	COMPLETED	24	24	24	25	26	26	26

On the average, the solution obtained by the lagrangian heuristic performs better than most of the standard dispatching rules.

5.5.3 EXPERIMENT 3

The system is utilized to the maximum extent possible (i.e. as many lots as possible are released into the system without violating the capacity constraints). Tables 5.10 and 5.11 summarize the problem data and the lot release policy respectively for this experiment.

Table 5.10 Summary of wafer fab system in experiment 3

Number of products	2
Number of scheduling problems solved	4
Number of station families	8
Number of workstations	17

Table 5.11 Lot Release Policy in experiment 3

DAY	**Product Number**	**Number of lots released in a single time period (M_i)**
1	1	7
	2	2
2	1	4
	2	5
3	1	3
	2	6
4	1	5
	2	4

In Table 5.12, the cycle time and the standard deviation of the cycle time obtained by the Lagrangian heuristic is compared to the corresponding values obtained using the standard dispatching rules. The best results for each performance measure for each product are shaded.

Table 5.12 Summary of results of experiment 3

PRODUCT #	CYCLE TIME	FIFO	LBA	LLA	LPR	LS	LTR	LAG
1	AVG	40:42:51	41:10:43	42:40:43	38:27:00	43:05:00	38:30:00	39:36:51
	STD DEV	6:01:06	5:56:27	7:02:34	5:22:31	3:34:45	6:07:37	6:24:58
2	AVG	47:03:20	42:42:00	44:33:00	46:37:48	47:30:00	46:06:40	43:23:20
	STD DEV	6:20:30	4:47:28	3:44:15	5:22:52	4:57:44	3:59:54	4:06:50
LOTS	COMPLETED	25	26	26	26	21	25	25

On the average, the solution obtained by the Lagrangian heuristic performs better than most of the standard dispatching rules.

5.5.4 EXPERIMENT 4

The system processes lots with higher priority. Tables 5.13 and 5.14 summarize the problem data and the lot release policy respectively for this experiment.

Table 5.13 Summary of wafer fab system in experiment 4

Number of products	2
Number of scheduling problems solved	3
Number of station families	8
Number of workstations	18

Table 5.14 Lot Release Policy in experiment 4

DAY	**Product Number**	**Number of lots released in a single time period (M_i)**
1	1	5 regular priority, 2 high priority
	2	2 regular priority
2	1	5 regular priority
	2	2 regular priority, 2 high priority
3	1	5 regular priority
	2	2 regular priority
4	1	5 regular priority
	2	2 regular priority

In Table 5.15, the cycle time and the standard deviation of the cycle time obtained by the Lagrangian heuristic are compared to only those obtained by one standard dispatching rule namely, Highest Priority First. The best results for each performance measure for each product are shaded.

Table 5.15 Summary of results of experiment 4

PRODUCT #	CYCLE TIME	PRIORITY	LAG	High Priority
1	AVG	Regular	44:39:51	46:43:51
	STD DEV		6:53:31	7:33:06
1	AVG	Hot Lot	36:36:00	38:30:00
	STD DEV		2:07:17	2:07:17
2	AVG	Regular	39:43:00	39:55:00
	STD DEV		6:00:45	2:59:25
2	AVG	Hot Lot	40:30:00	30:00:00
	STD DEV		4:56:59	0:42:26

The Lagrangian heuristic is not able to efficiently process the hot lots of both the products.

5.5.5 DISCUSSION OF RESULTS

In each experiment, a standard dispatching rule provides the least cycle time for a product. But, there is no standard dispatching rule that provides the least cycle time for all the products in the same experiment. Typically, the cycle times obtained through a dispatching rule favor one product over the other products. For example, the rule LTR (Least Time Remaining) favors products with smaller cycle times. Hence, the dispatching rules provide a very good solution for only one product but not for the others.

On the other hand, in each experiment, the cycle times obtained by the Lagrangian heuristic are quite close to the least cycle time for each product. Thus, the Lagrangian heuristic provides a good solution (cycle time) for each product and is not biased towards a product. Hence, it generates ‘balanced’ cycle times for each product, when all the lots have regular priorities.

It is difficult to justify that having ‘balanced’ cycle times for each product is better than having a very good solution for only one product, and vice versa. The choice will be dictated by the needs of the management.

Table 5.16 compares the consistency of each dispatching rule in providing good solutions. In each experiment, product, and dispatching rule, the cycle time is divided by the best cycle time for that product in that experiment. This ratio is expressed as a percentage. Thus, this is a ratio of the cycle time of each product, for each dispatching rule, in each experiment, to the best cycle time of each product, in each experiment.

The average of this ratio over all the experiments for each dispatching rule is provided in the bottom row of Table 5.16. The average of the ratio is the least for the Lagrangian heuristic. This shows that for the experiments conducted, the Lagrangian heuristic provides a solution that is, or close to, the best solution.

Table 5.16 Comparison of ratio of the cycle time of each product, for each dispatching rule, in each experiment, to the best cycle time of each product, in each experiment

Experiment	PRODUCT	FIFO	LBA	LLA	LPR	LS	LTR	LAG
1	1	106%	101%	101%	100%	100%	100%	100%
1	2	108%	102%	102%	100%	105%	100%	100%
2	1	106%	106%	104%	104%	100%	111%	103%
2	2	118%	113%	114%	121%	127%	100%	104%
3	1	106%	107%	111%	100%	112%	100%	103%
3	2	110%	100%	104%	109%	111%	108%	102%
Average		109%	105%	106%	106%	109%	103%	102%

Hence, it is safe to say that the Lagrangian heuristic is consistent and competitive when compared to the standard dispatching rules.

A graphical depiction of this consistency is provided in Figure 5.7. The data for Figure 5.7 is from Table 5.16. The figure clearly shows that the ratio for the Lagrangian heuristic varies from 100% to 104%. For most of the dispatching rules, this ratio ranges from 100% to 120%. For a few dispatching rules this ratio is greater than 120% and is up to 127%.

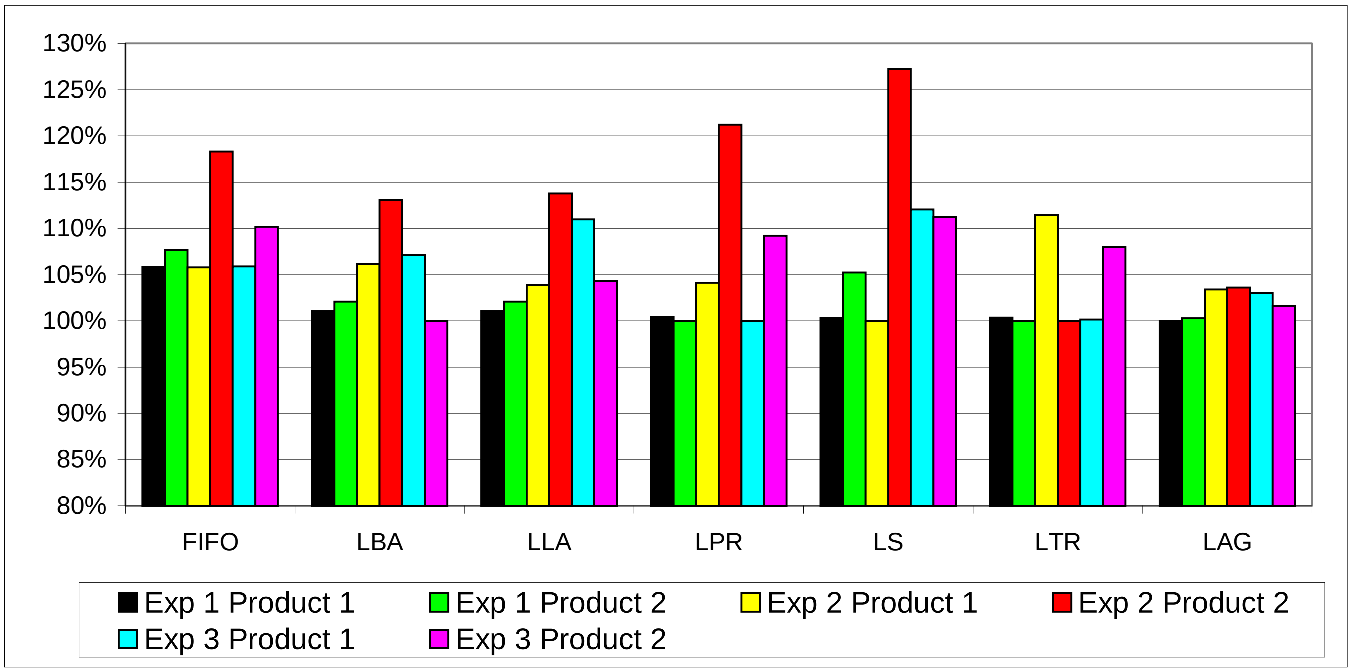

Figure 5.8 Comparison of ratio of the cycle time of each product and dispatching rule, in each experiment, to the best cycle time of a product in that experiment

In experiment 4, we study the ability of the Lagrangian heuristic to accommodate hot lots in its solution methodology. The Lagrangian heuristic is not able provide the least cycle time for the hot lots of all the products in the system. Hence, the Lagrangian heuristic is not able to process hot lots efficiently under the existing problem formulation.

One possible reason for the inability of the Lagrangian heuristic to process the hot lots efficiently may be the penalties assigned to the processing steps of the hot lots. Currently, the penalty for the processing steps of the hot lots is 480 compared to the penalty of 240 for the processing steps of regular lots. In this experiment, the ratio of the number of regular lots to hot lots in the wafer fab is 5:1. We can assume that the number of processing steps for the hot lots and regular lots is the same. Hence, the total penalty for regular lots is 5*240 = 1200. The penalty for the hot lots is 1*480 = 480. Thus, we see that the total penalty for the hot lots is less than the total penalty of the regular lots. Hence, the Lagrangian heuristic might not be able to prioritize the scheduling of hot lots over the regular lots.

One possible solution might be to increase the penalty assigned to the hot lots. Another possible solution might be to change the current problem formulation.

5.6 CONCLUSIONS

1. Many hard problems can be viewed as easy problems complicated by a relatively small set of side constraints.
2. The Lagrangian relaxation technique exploits this property of the hard problems in combinatorial optimization to provide tighter lower (upper) bounds for minimization (maximization) problems.
3. The solution to the Lagrangian problem may be judiciously tinkered with to obtain an integer feasible solution to the original integer problem.
4. The scheduling problem for a wafer fab is a prime candidate for the application of the Lagrangian relaxation technique.
5. Surrogate constraints are introduced to tighten the dual problem obtained by applying the Lagrangian relaxation.
6. A Lagrangian heuristic is developed to obtain near-optimal solutions to the scheduling problem for a wafer fab.
7. The solution quality and the capability of the heuristic to generate feasible schedules are examined and discussed.
8. The validity of the approach for the heuristic is examined and proved.
9. On the average, the quality of the solution obtained by the Lagrangian heuristic is competitive when compared to the solution obtained by the standard dispatching rules.
10. The solution obtained through the Lagrangian heuristic can be improved by introducing additional surrogate constraints that will make the problem tighter and close the integrality gap between the LP solution and the Integer solution.
11. The LST for each step of each product should be predicted more accurately. This will help to tighten the relaxed problem and provide better solutions to the scheduling problem.
12. The computation time required for the lagrangian heuristic is large. The computation times need to be reduced in order to make the lagrangian relaxation method commercially viable. This aspect can be explored in further research.

CHAPTER 6: SUMMARY AND CONCLUSION

6.1 RESEARCH OVERVIEW

This research has been motivated by the potential of mathematical programming techniques to provide optimal or near-optimal solutions to the scheduling problems in wafer fabs. Two important issues were addressed in this research for a wafer fab. These are as follows:

- Determination of the optimal lot release policy into the system. A system with an insufficient lot release rate is underutilized, resulting in lower WIP and lower throughput rate. On the other hand, a system that is overloaded has greater WIP and higher cycle times due to the excessive waiting time of the lots in the system. Hence, the lot release policy should be such that the system is sufficiently utilized and does not have excess WIP, and at the same time should meet the due date criterion.
- Determination of the optimal scheduling policy for processing lots in the system. The scheduling of the lots affects their cycle times and hence is important in determining the effectiveness of the system.

The solution methodology takes into consideration two different types of parameters to evaluate the effectiveness of the system:

- Tardiness of the work orders of the products
- Cycle time of the lots of the work orders.

The tardiness problem determines the lot release policy of the work orders. The scheduling problem is solved for the cycle time criterion as it is more relevant given the lot release policy resulting from the tardiness problem.

6.2 SUMMARY

Chapter 1 gives a brief introduction to the manufacturing process of the semiconductor industry. The problem that is addressed in this research is presented and explained.

A detailed literature survey is presented in chapter 2. The various areas of production planning and scheduling in the semiconductor industry that have been covered by the problems in the literature are given. The various techniques and approaches used for production planning and scheduling are discussed.

Chapter 3 presents a three phased methodology to minimize the total tardiness of all the work orders in the system. The solution provides the number of lots to be released into the system in each planning period of the entire planning horizon. The three-phase methodology provides acceptable solutions to the tardiness problem within a reasonable amount of computation time.

The scheduling problem for minimizing the cycle time of the lots released into the system is presented in chapter 4. Additional constraints formulated to capture specific features of a wafer fab are incorporated in the problem. Separate formulations are presented for a wafer fab under the static lot release policy and under the dynamic lot release policy. The earliest start times and the latest start times are established for every processing step that could be scheduled in the planning period for which the scheduling problem is solved.

Chapter 5 presents the Lagrangian relaxation approach for solving the scheduling problem. The Lagrangian heuristic developed solves the scheduling problem by an iterative approach. The Lagrangian heuristic is shown to provide feasible solutions to the scheduling problem. Experiments are conducted on different sets of conditions in the wafer fab. The Lagrangian heuristic generates consistently ‘good’ solutions for the scheduling problem. The cycle time for each product is close to the minimum cycle time of that product provided by any other dispatching rule. In comparison, the dispatching rules provide an optimal cycle time for only one product and provide longer cycle times for other products. Hence, the Lagrangian heuristic is a better method than the standard dispatching rules used in the industry.

6.3 FUTURE RESEARCH

Potential ideas for areas of future research:

- Develop a heuristic to solve the weighted tardiness problem for the work orders in a wafer fab, and thus establish the lot release policy.
- Incorporate additional constraints in the tardiness problem to tighten the problem and solve the problem in a much faster time.
- Develop a methodology to predict the latest start time (LST) of the processing steps of the lots in a dynamic manufacturing environment. This will help to reduce the number of variables in the problem, and thus reduce the problem size. This will reduce the computation time and enable us to apply the Lagrangian heuristic to larger wafer fabs.
- Incorporate additional surrogate constraints in the scheduling problem. This will tighten the LP relaxation, thus reducing the integrality gap, and hence the Lagrangian solution will provide a tighter upper bound to the original scheduling problem.
- Exploit the structure, if any, of the scheduling problem, so that the scheduling problem is decomposed and the Lagrangian heuristic provides a much faster solution to the scheduling problem.
- Enhance the formulation of the scheduling problem so that the hot lots can be incorporated into the solution methodology in a much better way.

The potential benefits of mathematical programming for scheduling in a wafer fab, over other methods, are shown to be significant in this research. The positive results presented here and the great potential for improvement warrants additional study of mathematical programming as a scheduling tool for the semiconductor industry.

BIBLIOGRAPHY

[1] Ahmadi A. J., Ahmadi R., Dasu S., and Tang C., "Batching and Scheduling jobs on batch and discrete processors" Operations Research, Vol. 40, 750-763, 1992.

[2] Atherton L. F., and Atherton R. W., Wafer Fabrication: Factory Performance and Analysis, Kluwer Academic Publishers, 1995

[3] Bai X., Srivatsan N., and Gershwin S. B., " Hierarchical Real Time Scheduling of a Semiconductor Fabrication Facility, IEEE/CMPT International Electronics Manufacturing Technology Symposium, 312-317, 1990

[4] Baker K. R., Introduction to Sequencing and Scheduling, John Wiley & Sons, 1974

[5] Billings R., and Hasenbein J., "A survey of applications of fluid models to semiconductor operations." Proceedings of the International Conference on Modeling and Analysis of Semiconductor Manufacturing, 97- 103, 2000.

[6] Camerini P. M., Fratta L., and Maffioli F., "On improving relaxation methods by modified gradient techniques" Mathematical Programming Study 3, 26-35, 1975

[7] Connors D. P., and Yao D. D., "Methods for Job Configuration in Semiconductor Manufacturing" IEEE Transactions on Semiconductor Manufacturing, Vol. 9, No. 3, 401-411, August 1996

[8] Fisher M. L., Northup W. D., and Shapiro J. F., "Using Duality to solve Discrete Optimization Problems: Theory and Computational Experience" Mathematical Programming Study 3, 56-94, 1975

[9] Fisher M. L., "The Lagrangian Relaxation Method for solving Integer Programming Problems" Management Science, Vol. 27, No. 1, 1-18, January 1981

[10] Fisher M. L., "An Applications Oriented Guide to Lagrangian Relaxation" Interfaces, Vol. 15, No. 2, 10-21, March-April 1985

[11] Geoffrion A. M., "Lagrangian Relaxation for Integer Programming" Mathematical Programming Study 2, 82-114, 1974

[12] Glassey C. R., Shantikumar J. G., and Seshadri S., "Linear Control Rules for Production Control of Semiconductor Fabs" IEEE Transactions on Semiconductor Manufacturing, Vol. 1, No. 1, 36-46, February 1998

[13] Glover F., "Surrogate Constraints" Operations Research, Vol. 16, 741-749, 1968

[14] Goldratt E., and Cox J., The Goal, North River Press, 1992

[15] Golovin J. J., " A Total Framework for Semiconductor Production Planning and Scheduling, " Solid State Technology, 167-170, May 1986

[16] Graves S., Meal H., Stefek D., and Zeghmi A., "Scheduling of Re-entrant flow shops." Journal of operations management, Vol. 3, No. 4, 197-207, 1983

[17] Hung Y. F., and Leachman R. C., "A Production Planning Methodology for Semiconductor Manufacturing based on Iterative Simulation and Linear Programming Calculations" IEEE Transactions on Semiconductor Manufacturing, Vol 9, No. 2, 257-269, May 1996

[18] Kim Y., Kim J., Lim S., and Jun H., "Due date based scheduling and control policies in a multi-product semiconductor wafer fabrication facility" IEEE Transactions on Semiconductor Manufacturing, Vol. 11, No. 1, 155-164, February 1998.

[19] Kumar S., and Kumar P. R., "Performance Bounds for Queuing Networks and Scheduling Policies" IEEE Transactions on Automatic Control, Vol. 39, No. 8, August 1994.

[20] Leachman R. C., “Preliminary Design and Development of a Corporate-Level Production Planning System for the Semiconductor Industry” Operations Research Center, University of California, Berkeley, February 1986.

[21] Lee Y., Kim S., Yea S., and Kim B., “Production Planning in a semiconductor wafer fab considering variable cycle times.” Computers in Industrial Engineering, Vol. 33, Nos. 3-4, 713-716, 1997.

[22] Lou S. X. C., and Kager P. W., “A Robust Production Control Policy for VLSI Wafer Fabrication” IEEE Transactions on Semiconductor Manufacturing, Vol. 2, No. 4, 159-164, November 1989.

[23] Lu S., and Kumar P. R., “Distributed Scheduling based on due dates and buffer priorities” IEEE Transactions on Automatic Control, Vol. 36, 1406-1416, 1991

[24] Lu S., Ramaswamy D., and Kumar P. R., “Efficient Scheduling policies to reduce mean and variance of cycle time in semiconductor manufacturing plants”, IEEE Transactions on Semiconductor Manufacturing, Vol. 7, No. 3, August 1994.

[25] Mehta S. V., and Uzsoy R., “Minimizing Total Tardiness on a Batch Processing Machine with Incompatible Job Families” IIE Transactions, Vol. 30, 165-178, 1998

[26] Pinedo M., Scheduling: Theory, Algorithms, and Systems, Prentice Hall, 1995

[27] Sarin S., Shikalgar S., and Shenai V., “Modeling and Analysis for the reduction of cycle time at a wafer fabrication facility” Proceedings of the International Conference on Semiconductor Manufacturing Operational Modeling and Simulation, 2001.

[28] Savell D., Perez R., and Koh S., “Scheduling Semiconductor Wafer Production: An Expert System Implementation” IEEE Expert, Vol. 4, No. 3, 9-15, 1989

[29] Schoemig A., and Kroehn N., “Comparison of Dispatching Rules for a Large Semiconductor Manufacturing Facility.” Proceedings of the International

Conference on Semiconductor Manufacturing Operational Modeling and Simulation, 2001.

[30] Shapiro J. F., “A Survey of Lagrangian Techniques for Discrete Optimization” Annals of Discrete Mathematics, Vol. 5, 113-138, 1979

[31] Sherali H. D., and Driscoll P. J., “Evolution and state-of-the-art in Integer Programming” Journal of computational and applied mathematics. 124, no. 1, 318-340, 2000

[32] Sierksma G., Linear and Integer Programming: Theory and Practice, Marcel Dekker Inc., 1996

[33] Uzsoy R., Lee C. Y., and Martin-Vega L., “A review of production planning and scheduling models in the semiconductor industry – Part I: System characteristics, performance evaluation and production planning,” IIE Transactions, Scheduling and Logistics, vol. 24, 47-61, 1992.

[34] Uzsoy R., Lee C. Y., and Martin-Vega L., “A review of production planning and scheduling models in the semiconductor industry – Part II: Shop-Floor Control” IIE Transactions, Scheduling and Logistics, vol. 26, 44-55, 1994.

[35] Wein L., “Scheduling networks of queues: heavy traffic analysis of a multi-station network with controllable inputs.” Operations Research 40, S312-S334, 1992

[36] Wein L., “Scheduling Semiconductor Wafer Fabrication.” IEEE Transactions on Semiconductor Manufacturing, Vol. 1, No. 3, 115-129, 1988

[37] Weiss G., “On Optimal Draining of Re-entrant Fluid Lines.” The IMA Volumes in Mathematics and its Applications, Stochastic Networks, Vol. 71, 91-103, 1995

APPENDICES

APPENDIX A: DATA FOR THE 6 PRODUCT TARDINESS PROBLEM IN CHAPTER 3

Workstation data

Station Family	**Number of Workstations**
1	3
2	3
3	4
4	3
5	2
6	2
7	3
8	2
9	1
10	2

Routes for the 6 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	7	3
5	4	3
6	1	1
7	2	2
8	3	7
7	7	3
10	1	1
11	2	2
12	9	2.5
13	4	3
14	8	3

PRODUCT 3		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	9	2.5
12	10	3
13	3	7
14	5	1.5
15	7	3
16	1	1
17	4	3
18	8	3

PRODUCT 4		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	1	1
10	4	3
11	8	3

PRODUCT 5		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	6	2.5
10	1	1
11	9	2.5
12	4	3
13	8	3

PRODUCT 6		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	10	3
10	1	1
11	2	2
12	4	3
13	8	3

Detailed Lot Release policy developed in Phase 2

Time Period t	Work Order ij	# of lots	Work Order ij	# of lots	Work Order ij	# of lots	Time Period t	Work Order ij	# of lots	Work Order ij	# of lots	Work Order ij	# of lots
1	11	12					136	52	9	62	3		
2	11	12					137	52	8	62	4		
3	11	12					138	62	12				
4	11	12					139	62	12				
5	11	12					140	62	12				
6	11	12					141	62	12				
7	11	12					142	62	3	42	9		
8	11	12					143	42	12				
9	11	4	61	8			144	42	12				
10	61	12					145	42	12				
11	61	12					146	42	12				
12	61	12					147	42	12				
13	61	12					148	42	12				
14	61	12					149	42	12				
15	61	12					150	42	12				
16	61	12					151	42	12				
17	61	12					152	42	12				
18	61	12					153	42	12				
19	61	12					154	42	12				
20	61	12					155	42	12				
21	61	10	21	2			156	42	12				
22	21	8					157	42	12				
23	21	8					158	42	11	13	1		

24	21	8					159	13	12				
25	21	8					160	13	12				
26	21	8					161	13	12				
27	21	8					162	13	12				
28	21	8					163	13	12				
29	21	8					164	13	12				
30	21	8					165	13	12				
31	21	8					166	13	12				
32	21	8					167	13	12				
33	21	8					168	13	12				
34	21	8					169	13	12				
35	21	8					170	13	12				
36	21	8					171	13	5	23	5	53	1
37	21	8					172	23	8				
38	21	8					173	23	8				
39	21	8					174	23	8				
40	21	4	51	5	41	3	175	23	8				
41	51	9	41	3			176	23	8				
42	51	9	41	3			177	23	8				
43	51	9	41	3			178	23	8				
44	51	9	41	3			179	23	8				
45	51	9	41	3			180	23	8				
46	51	9	41	3			181	23	8				
47	51	9	41	3			182	23	8				
48	51	9	41	3			183	23	7	33	1		
49	51	9	41	3			184	33	8				
50	51	9	41	3			185	33	8				
51	51	5	41	3	31	4	186	33	8				
52	31	8					187	33	8				
53	31	8					188	33	8				
54	31	8					189	33	8				
55	31	8					190	33	8				
56	31	8					191	33	8				
57	31	8					192	33	8				
58	31	8					193	33	8				
59	31	8					194	33	8				
60	31	8					195	33	8				
61	31	8					196	33	3	53	6	14	3
62	31	8					197	53	9	14	3		
63	31	8					198	53	9	14	3		
64	31	8					199	53	9	14	3		
65	31	8					200	53	9	14	3		

66	31	8					201	53	9	14	3		
67	31	8					202	53	9	14	3		
68	31	8					203	53	9	14	3		
69	31	8					204	53	9	14	3		
70	31	2	41	10			205	53	9	14	3		
71	41	12					206	53	9	14	3		
72	41	12					207	53	3	14	1	32	6
73	41	12					208	32	8				
74	41	12					209	32	8				
75	41	12					210	32	8				
76	41	12					211	32	8				
77	41	12					212	32	8				
78	41	12					213	32	8				
79	41	12					214	32	8				
80	41	12					215	32	8				
81	41	12					216	32	8				
82	41	12					217	32	8				
83	41	12					218	32	8				
84	41	12					219	32	8				
85	41	12					220	32	8				
86	41	12					221	32	8				
87	41	12					222	32	8				
88	12	12					223	32	8				
89	12	12					224	32	8				
90	12	12					225	32	8				
91	12	12					226	32	8				
92	12	12					227	32	8				
93	12	12					228	32	8				
94	12	12					229	32	8				
95	12	12					230	32	8				
96	12	12					231	32	8				
97	12	12					232	32	8				
98	12	12					233	32	8				
99	12	12					234	32	8				
100	12	12					235	32	8				
101	12	12					236	32	8				
102	12	12					237	32	8				
103	12	12					238	32	4	14	8		
104	12	12					239	14	12				
105	12	12					240	14	12				
106	12	12					241	14	12				
107	12	12					242	14	12				

108	12	10	22	2			243	14	12				
109	22	8					244	14	12				
110	22	8					245	14	12				
111	22	8					246	14	12				
112	22	8					247	14	12				
113	22	8					248	54	9				
114	22	8					249	54	9				
115	22	8					250	54	9				
116	22	8					251	54	9				
117	22	8					252	54	9				
118	22	8					253	54	9				
119	22	8					254	54	9				
120	22	8					255	54	9				
121	22	2	52	7			256	54	9				
122	52	9	62	3			257	54	9				
123	52	9	62	3			258	54	9				
124	52	9	62	3			259	54	1	34	8		
125	52	9	62	3			260	34	8				
126	52	9	62	3			261	34	8				
127	52	9	62	3			262	34	8				
128	52	9	62	3			263	34	8				
129	52	9	62	3			264	34	8				
130	52	9	62	3			265	34	8				
131	52	9	62	3			266	34	8				
132	52	9	62	3			267	34	8				
133	52	9	62	3			268	34	8				
134	52	9	62	3			269	34	8				
135	52	9	62	3			270	34	8				
							271	34	4				

Detailed Lot Release policy developed in Phase 3

Time	Work	# of	Work	# of	Work	# of	Work	# of	Time	Work	# of	Work	# of	Work	# of	Work	# of
Period	Order	lots	Order	lots	Order	lots	Order	lots	Period	Order	lots	Order	lots	Order	lots	Order	lots
t	ij		ij		ij		ij		t	ij		ij		ij		ij	
1	11	12							127	52	5	32	4	42	3		
2	11	12							128	52	5	32	4	42	3		
3	11	12							129	52	5	32	4	42	3		
4	11	12							130	52	5	32	4	42	3		
5	11	12							131	52	5	32	4	42	3		
6	11	4	61	8					132	52	5	32	4	42	3		
7	11	12							133	52	5	32	4	42	3		
8	11	12							134	52	5	32	4	62	3		
9	11	12							135	52	5	32	4	62	3		
10	61	12							136	62	8	32	4				
11	61	12							137	42	8	32	4				
12	61	10	21	2					138	32	5	42	4	62	2		
13	61	12							139	42	8	32	4				
14	61	8	21	4					140	62	8	32	4				
15	61	8	21	4					141	62	8	32	4				
16	61	12							142	42	8	32	4				
17	61	12							143	62	8	32	4				
18	61	12							144	62	8	32	4				
19	61	12							145	42	8	32	4				
20	61	12							146	42	8	32	4				
21	61	12							147	42	8	32	4				
22	61	8	21	4					148	62	8	32	4				
23	21	8							149	62	8	32	4				
24	21	7	51	2					150	42	8	32	4				
25	21	7	51	2					151	13	8	32	4				
26	21	8							152	13	8	32	4				
27	21	7	51	2					153	32	8						
28	21	7	51	2					154	13	8	32	4				
29	21	7	51	2					155	42	8	32	4				
30	21	7	51	2					156	42	8	32	4				
31	21	7	51	2					157	42	8	32	4				
32	21	7	51	2					158	13	8	32	4				
33	21	7	51	2					159	13	8	32	4				
34	21	7	51	2					160	42	8	32	4				
35	21	7	51	2					161	42	8	32	4				
36	21	7	51	2					162	42	8	32	4				
37	21	8							163	42	8	32	4				
38	21	7	51	2					164	42	8	32	4				
39	21	7	51	2					165	13	8	32	4				
40	21	7	51	2					166	42	8	32	4				
41	21	7	51	2					167	42	8	32	4				

42	51	5	31	4	41	3			168	42	8	32	4				
43	51	5	31	4	41	3			169	42	8	32	4				
44	51	9	41	3					170	42	8	32	4				
45	51	5	31	4	41	3			171	32	4	13	8				
46	51	5	31	4	41	3			172	32	4	13	8				
47	51	5	31	4	41	3			173	32	4	13	8				
48	51	5	31	4	41	3			174	32	4	13	8				
49	51	9	41	3					175	32	5	13	6				
50	51	5	31	4	41	3			176	32	4	13	8				
51	51	5	31	4	41	3			177	32	4	13	8				
52	51	5	31	4	41	3			178	32	4	13	8				
53	51	5	31	4	41	3			179	32	4	13	8				
54	31	6	41	4					180	32	4	13	8				
55	41	8	31	4					181	32	4	13	8				
56	31	8							182	23	4	13	8				
57	31	8							183	32	4	13	8				
58	31	8							184	32	8						
59	41	8	31	4					185	23	8						
60	41	8	31	4					186	23	8						
61	41	8	31	4					187	23	8						
62	41	8	31	4					188	23	8						
63	41	8	31	4					189	23	8						
64	41	8	31	4					190	23	8						
65	41	8	31	4					191	23	8						
66	31	8							192	23	8						
67	31	8							193	23	8						
68	41	6	31	4	12	2			194	23	8						
69	31	8							195	23	8						
70	31	8							196	23	8						
71	31	8							197	33	8						
72	41	8	31	4					198	33	8						
73	41	12							199	54	4	33	4	14	3	53	1
74	41	12							200	33	8						
75	41	12							201	33	8						
76	41	12							202	33	8						
77	41	12							203	33	8						
78	41	12							204	33	8						
79	41	12							205	33	8						
80	41	12							206	33	8						
81	41	12							207	33	8						
82	41	12							208	33	8						
83	41	12							209	33	8						
84	12	12							210	54	9	14	3				
85	12	12							211	53	9	14	3				
86	12	12							212	53	9	14	3				
87	12	12							213	53	9	14	3				
88	12	12							214	53	9	14	3				
89	12	12							215	53	9	14	1				

90	12	8	22	4					216	53	9						
91	12	12							217	53	9						
92	12	12							218	53	9	14	3				
93	12	12							219	53	9	14	3				
94	12	8	22	4					220	53	9	14	3				
95	12	10	22	2					221	53	9						
96	12	12							222	34	2						
97	12	8	22	4					223	54	9						
98	52	5	22	4	12	3			224	34	8						
99	12	8	22	4					225	14	8	34	4				
100	12	8	22	4					226	14	12						
101	12	8	22	4					227	14	12						
102	12	8	22	4					228	14	12						
103	12	8	22	4					229	54	9	14	3				
104	12	12							230	54	9	14	3				
105	12	8	22	4					231	14	12						
106	52	5	22	4	62	3			232	54	6	34	3				
107	12	8	22	4					233	14	12						
108	52	5	22	4	12	3			234	14	12						
109	12	12							235	54	9	14	3				
110	12	8	22	4					236	54	9	14	3				
111	52	5	22	4	62	3			237	14	12						
112	52	5	22	4	62	3			238	54	9	14	3				
113	52	5	22	4	62	3			239	54	9	14	3				
114	52	5	22	4	62	3			240	34	8						
115	52	5	22	4	62	3			241	14	12						
116	52	5	22	4	62	3			242	34	8						
117	52	5	22	4	42	3			243	54	9						
118	52	5	62	3	32	2	22	2	244	34	6						
119	52	5	22	4	62	3			245	34	7						
120	52	5	22	4	62	3			246	34	8						
121	52	7	42	3	32	2			247	34	8						
122	52	5	22	4	62	3			248	34	8						
123	52	9	42	3					249	54	9						
124	52	5	32	4	42	3			250	34	7						
125	52	9	62	3					251	34	8						
126	52	5	32	4	42	3			252	34	8						
									253	34	7						

APPENDIX B: DATA FOR THE 2 PRODUCT TARDINESS PROBLEM IN CHAPTER 3

2 Product	8 Station Families	18 Work stations	One time period = 24 hrs

Workstation data

Station Family	**Number of Workstations**
1	2
2	2
3	3
4	3
5	2
6	1
7	3
8	2

Routes for the 2 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	1	1
10	4	3
11	8	3

Product Number	**Maximum number of lots released in a single time period (B_i)**
1	9
2	10

Data for the work orders of the 2 products

Product Number	**Order Number**	**Number of lots in the order**	**Upper bound on time for completing the work order**	**Due date of the order**
i	**j**	**k_{ij}**	**U_{ij}**	**d'_{ij}**
1	1	100	12	15
1	2	150	17	35
1	3	100	12	60
1	4	250	28	95
1	5	200	23	120
2	1	100	10	20
2	2	250	25	50
2	3	200	20	80
2	4	150	15	100

Summary of solution in phase 1

Rank	**Product Number**	**Order #**	**# of lots in the order**	**Processing Time**	**Due date of order**	**Completion Time**	**Tardiness**
r	**i**	**j**	**k_{ij}**	**U_{ij}**	**d'_{ij}**	**C_{ij}**	**T_{ij}**
1	1	1	100	12	15	12	0
2	2	1	100	10	20	22	2
3	1	2	150	17	35	39	4
4	1	3	100	12	60	51	0
5	2	2	250	25	50	76	26
6	2	3	200	20	80	96	16
7	2	4	150	15	100	111	11
8	1	5	200	23	120	134	14
9	1	4	250	28	95	162	67

The time required to release all the lots into the system is 162 days.

The Total Tardiness value at the end of Phase 1 is $\Sigma T_{ij} = 140$

Summary of solution in phase 2

Rank	**Product Number**	**Order #**	**# of lots in the order**	**Due date of order**	**Completion Time**	**Tardiness**
r	**i**	**j**	**k_{ij}**	**d'_{ij}**	**C_{ij}**	**T_{ij}**
1	1	1	100	15	12	0
2	2	1	150	20	20	0
3	1	2	150	35	37	2
4	1	3	100	60	48	0
5	2	2	150	50	70	20
6	2	3	250	80	90	10
7	2	4	250	100	105	5
8	1	5	100	120	128	8
9	1	4	150	95	155	60

The time required to release all the lots into the system is 155 days.

The Total Tardiness value at the end of Phase 2 is $\Sigma T_{ij} = 105$

Detailed Lot Release policy developed in Phase 2

Time Period	Work Order	# of lots	Work Order	# of lots	Work Order	# of lots	Time Period	Work Order	# of lots	Work Order	# of lots
t	ij		ij		ij		t	ij		ij	
1	11	9	21	1			78	23	10		
2	11	9	21	1			79	23	10		
3	11	9	21	1			80	23	10		
4	11	9	21	1			81	23	10		
5	11	9	21	1			82	23	10		
6	11	9	21	1			83	23	10		
7	11	9	21	1			84	23	10		
8	11	9	21	1			85	23	10		
9	11	9	21	1			86	23	10		
10	11	9	21	1			87	23	10		
11	11	9	21	1			88	23	10		
12	11	1	21	9			89	23	10		
13	21	10					90	23	10		
14	21	10					91	24	10		
15	21	10					92	24	10		
16	21	10					93	24	10		
17	21	10					94	24	10		
18	21	10					95	24	10		
19	21	10					96	24	10		
20	21	10					97	24	10		
21	12	9	22	1			98	24	10		
22	12	9	22	1			99	24	10		
23	12	9	22	1			100	24	10		
24	12	9	22	1			101	24	10		
25	12	9	22	1			102	24	10		
26	12	9	22	1			103	24	10		
27	12	9	22	1			104	24	10		
28	12	9	22	1			105	24	10		
29	12	9	22	1			106	15	9		
30	12	9	22	1			107	15	9		
31	12	9	22	1			108	15	9		
32	12	9	22	1			109	15	9		
33	12	9	22	1			110	15	9		
34	12	9	22	1			111	15	9		

35	12	9	22	1			112	15	9		
36	12	9	22	1			113	15	9		
37	12	6	22	1	13	3	114	15	9		
38	13	9	22	1			115	15	9		
39	13	9	22	1			116	15	9		
40	13	9	22	1			117	15	9		
41	13	9	22	1			118	15	9		
42	13	9	22	1			119	15	9		
43	13	9	22	1			120	15	9		
44	13	9	22	1			121	15	9		
45	13	9	22	1			122	15	9		
46	13	9	22	1			123	15	9		
47	13	9	22	1			124	15	9		
48	13	7	22	3			125	15	9		
49	22	10					126	15	9		
50	22	10					127	15	9		
51	22	10					128	15	2	14	7
52	22	10					129	14	9		
53	22	10					130	14	9		
54	22	10					131	14	9		
55	22	10					132	14	9		
56	22	10					133	14	9		
57	22	10					134	14	9		
58	22	10					135	14	9		
59	22	10					136	14	9		
60	22	10					137	14	9		
61	22	10					138	14	9		
62	22	10					139	14	9		
63	22	10					140	14	9		
64	22	10					141	14	9		
65	22	10					142	14	9		
66	22	10					143	14	9		
67	22	10					144	14	9		
68	22	10					145	14	9		
69	22	10					146	14	9		
70	22	10					147	14	9		
71	23	10					148	14	9		
72	23	10					149	14	9		
73	23	10					150	14	9		
74	23	10					151	14	9		
75	23	10					152	14	9		
76	23	10					153	14	9		

77	23	10					154	14	9		
							155	14	9		

The table below shows the efficacy of the 3-phased solution approach.

Additional Features in the Integer Problem	**Time for LP Solution**	**Present Integer Solution**	**Time for Present Integer Solution**	**Node at which Incumbent Integer Solution obtained**	**Time Branch and Bound terminated**	**Node at which Branch and Bound terminated**
None	6.68	179	738.58	9921	1800.38	22404
Phase 1 Solution as Upper Bound	6.78	-	-	-	1810.30	21466
Phase 2 Solution as Upper Bound	6.92	105	61.85	60	1800.05	14206
Phase 3, 1st constraint	8.53	105	54.49	91	1828.68	10169
Phase 3, 1st and 2nd constraint	5.33	105	36.87	72	1830.41	9685
Phase 3, 1st, 2nd and 3rd constraint	0.42	105 (optimal)	5.47	112	123.46 (optimal)	7072

Summary of solution in phase 3:

Rank	**Product Number**	**Order #**	**# of lots in the order**	**Due date of order**	**Completion Time**	**Tardiness**
r	**i**	**j**	**k_{ij}**	**d'_{ij}**	**C_{ij}**	**T_{ij}**
1	1	1	100	15	12	0
2	2	1	150	20	20	0
3	1	2	150	35	37	2
4	1	3	100	60	51	0
5	2	3	150	50	70	20
6	2	3	250	80	90	10
7	2	4	250	100	105	5
8	1	5	100	120	128	8
9	1	4	150	95	155	60

Detailed Lot Release policy developed in Phase 3

Time Period t	Work Order ij	# of lots	Work Order ij	# of lots	Work Order ij	# of lots	Time Period t	Work Order ij	# of lots	Work Order ij	# of lots
1	21	9	11	1			78	23	10		
2	11	9	21	1			79	23	10		
3	11	9	21	1			80	23	10		
4	11	9	21	1			81	23	10		
5	11	9	21	1			82	23	10		
6	11	9	21	1			83	23	10		
7	11	9	21	1			84	23	10		
8	11	9	21	1			85	23	10		
9	11	9	21	1			86	23	10		
10	11	9	21	1			87	23	10		
11	11	9	21	1			88	23	10		
12	11	9	21	1			89	23	10		
13	21	10					90	23	10		
14	21	10					91	24	10		
15	21	10					92	24	10		
16	21	10					93	24	10		
17	21	10					94	24	10		
18	21	10					95	24	10		
19	21	10					96	24	10		
20	21	10					97	24	10		
21	12	9	22	1			98	24	10		
22	12	9	22	1			99	24	10		
23	12	9	22	1			100	24	10		
24	12	9	22	1			101	24	10		
25	12	9	22	1			102	24	10		
26	12	9	22	1			103	24	10		
27	12	9	22	1			104	24	10		
28	12	9	22	1			105	24	10		
29	12	9	22	1			106	15	9		
30	12	9	22	1			107	15	9		
31	12	9	22	1			108	15	9		
32	12	9	22	1			109	15	9		
33	12	9	22	1			110	15	9		
34	12	9	22	1			111	15	9		
35	12	9	22	1			112	14	7	15	2
36	12	6	13	3	22	1	113	15	9		
37	12	9	22	1			114	15	9		
38	13	9	22	1			115	15	9		
39	13	9	22	1			116	15	9		
40	13	9	22	1			117	15	9		
41	22	10					118	15	9		
42	13	9	22	1			119	15	9		

43	13	9	22	1			120	15	9		
44	13	9	22	1			121	15	9		
45	13	9	22	1			122	15	9		
46	13	7	22	3			123	15	9		
47	22	10					124	15	9		
48	13	9	22	1			125	15	9		
49	13	9	22	1			126	15	9		
50	13	9	22	1			127	15	9		
51	22	10					128	15	9		
52	22	10					129	14	9		
53	22	10					130	14	9		
54	22	10					131	14	9		
55	22	10					132	14	9		
56	22	10					133	14	9		
57	22	10					134	14	9		
58	22	10					135	14	9		
59	22	10					136	14	9		
60	22	10					137	14	9		
61	22	10					138	14	9		
62	22	10					139	14	9		
63	22	10					140	14	9		
64	22	10					141	14	9		
65	22	10					142	14	9		
66	22	10					143	14	9		
67	22	10					144	14	9		
68	22	10					145	14	9		
69	22	10					146	14	9		
70	22	10					147	14	9		
71	23	10					148	14	9		
72	23	10					149	14	9		
73	23	10					150	14	9		
74	23	10					151	14	9		
75	23	10					152	14	9		
76	23	10					153	14	9		
77	23	10					154	14	9		
							155	14	9		

APPENDIX C: DATA FOR 2 PRODUCT SCHEDULING PROBLEM IN CHAPTER 5

Products	Station Families	Work stations	One time period	One time unit
2	8	18	24 hrs	6 minutes

Workstation data

Station Family	**Number of Workstations**
1	2
2	2
3	3
4	3
5	2
6	1
7	3
8	2

Routes for the 2 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	1	1
10	4	3
11	8	3

LOT RELEASE POLICY

Product Number	Number of lots released in a single time period (M_i)
1	7
2	3

AVAILABILITY OF WORKSTATIONS

Station Family	Workstation Number	Available at Time Unit
1	1	6
1	2	6
2	1	0
2	2	3
3	1	36
3	2	66
3	3	8
4	1	20
4	2	0
4	3	13
5	1	0
5	2	11
6	1	13
7	1	10
7	2	0
7	3	13
8	1	0
8	2	10

AVAILABILITY OF LOTS

Sr. No.	Product Number	Lot Number	Available at Time Unit
1	1	639	20
2	1	739	10
3	1	140	13
4	1	240	3
5	1	340	13
6	1	440	13
7	1	540	11
8	1	640	36
9	1	740	66
10	1	141	0
11	1	241	34
12	1	341	68
13	1	441	102
14	1	541	136
15	1	641	170
16	1	741	204
17	2	418	6
18	2	419	6
19	2	420	8

20	2	421	0
21	2	422	80
22	2	423	160

PROCESSING STEPS TO BE SCHEDULED IN THE CURRENT TIME PERIOD

Product #	**Lot #**	**Processing Steps**	**Processing Time**	**Station Family**	**EST**	**LST**
1	639	14	30	8	20	150
1	739	12	10	1	10	140
1	739	13	30	4	20	150
1	739	14	30	8	50	180
1	140	11	30	7	13	110
1	140	12	10	1	43	140
1	140	13	30	4	53	150
1	140	14	30	8	83	180
1	240	9	15	5	3	100
1	240	10	25	6	18	115
1	240	11	30	7	43	140
1	240	12	10	1	73	170
1	240	13	30	4	83	180
1	240	14	30	8	113	210
1	340	7	10	1	13	70
1	340	8	20	2	23	80
1	340	9	15	5	43	100
1	340	10	25	6	58	115
1	340	11	30	7	83	140
1	340	12	10	1	113	170
1	340	13	30	4	123	180
1	340	14	30	8	153	210
1	440	6	30	4	13	70
1	440	7	10	1	43	100
1	440	8	20	2	53	110
1	440	9	15	5	73	130
1	440	10	25	6	88	145
1	440	11	30	7	113	170
1	440	12	10	1	143	200
1	440	13	30	4	153	210
1	440	14	30	8	183	240
1	540	5	30	7	11	40
1	540	6	30	4	41	70
1	540	7	10	1	71	100
1	540	8	20	2	81	110
1	540	9	15	5	101	130
1	540	10	25	6	116	145
1	540	11	30	7	141	170
1	540	12	10	1	171	200
1	540	13	30	4	181	210
1	540	14	30	8	211	240

1	640	4	15	5	36	55
1	640	5	30	7	51	70
1	640	6	30	4	81	100
1	640	7	10	1	111	130
1	640	8	20	2	121	140
1	640	9	15	5	141	160
1	640	10	25	6	156	175
1	640	11	30	7	181	200
1	640	12	10	1	211	230
1	640	13	30	4	221	240
1	740	4	15	5	66	95
1	740	5	30	7	81	110
1	740	6	30	4	111	140
1	740	7	10	1	141	170
1	740	8	20	2	151	180
1	740	9	15	5	171	200
1	740	10	25	6	186	215
1	740	11	30	7	211	240
1	141	1	10	1	6	20
1	141	2	20	2	16	30
1	141	3	70	3	36	50
1	141	4	15	5	106	120
1	141	5	30	7	121	135
1	141	6	30	4	151	165
1	141	7	10	1	181	195
1	141	8	20	2	191	205
1	141	9	15	5	211	225
1	141	10	25	6	226	240
1	241	1	10	1	34	55
1	241	2	20	2	44	65
1	241	3	70	3	64	85
1	241	4	15	5	134	155
1	241	5	30	7	149	170
1	241	6	30	4	179	200
1	241	7	10	1	209	230
1	241	8	20	2	219	240
1	341	1	10	1	68	95
1	341	2	20	2	78	105
1	341	3	70	3	98	125
1	341	4	15	5	168	195
1	341	5	30	7	183	210
1	341	6	30	4	213	240
1	441	1	10	1	102	125
1	441	2	20	2	112	135
1	441	3	70	3	132	155
1	441	4	15	5	202	225
1	441	5	30	7	217	240
1	541	1	10	1	136	140

1	541	2	20	2	146	150
1	541	3	70	3	166	170
1	541	4	15	5	236	240
1	641	1	10	1	170	210
1	641	2	20	2	180	220
1	641	3	70	3	200	240
1	741	1	10	1	204	210
1	741	2	20	2	214	220
1	741	3	70	3	234	240
2	418	10	30	4	6	180
2	418	11	30	8	36	210
2	419	7	15	5	6	155
2	419	8	30	7	21	170
2	419	9	10	1	51	200
2	419	10	30	4	61	210
2	419	11	30	8	91	240
2	420	3	15	5	8	70
2	420	4	30	7	23	85
2	420	5	30	4	53	115
2	420	6	10	1	83	145
2	420	7	15	5	93	155
2	420	8	30	7	108	170
2	420	9	10	1	138	200
2	420	10	30	4	148	210
2	420	11	30	8	178	240
2	421	1	10	1	6	30
2	421	2	70	3	16	40
2	421	3	15	5	86	110
2	421	4	30	7	101	125
2	421	5	30	4	131	155
2	421	6	10	1	161	185
2	421	7	15	5	171	195
2	421	8	30	7	186	210
2	421	9	10	1	216	240
2	422	1	10	1	80	85
2	422	2	70	3	90	95
2	422	3	15	5	160	165
2	422	4	30	7	175	180
2	422	5	30	4	205	210
2	422	6	10	1	235	240
2	423	1	10	1	160	230
2	423	2	70	3	170	240

APPENDIX D: TABLEAU FROM THE ITERATIONS OF THE LAGRANGIAN HEURISTIC IN CHAPTER 5

1st iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15039.00	15039.00	0.00	45.48	624	0	0	0
2	1	14500.12	15350.00	-849.88	1434.6	9880	581	0	0
3*	2	14898.69	15350.00	-451.31	1344.13	131778	7010	0.08	11.7729
4	3	14989.15	15322.00	-332.85	0.26	2	0	0	0
5	4	14941.15	15276.00	-334.85	294.37	21161	762	0	0
6	5	15061.16	15123.00	-61.84	13.99	660	0	0	0
7	6	15034.14	15307.00	-272.86	51.39	1037	9	0	0
8	7	15066.81	15180.00	-113.19	42.69	819	2	0	0
9	**8**	**15107.57**	**15093.00**	**14.57**	14.34	727	0	0	0
10	9	15087.95	15175.00	-87.05	170.6	12246	814	0	0

3411.85

2nd iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15145.00	15145.00	0.00	15.06	461	1	0	0
2	1	14947.24	15350.00	-402.76	40.11	789	17	0	0
3	2	15124.59	15302.00	-177.41	11.12	571	0	0	0
4	3	15138.54	16256.00	-1117.46	11.06	541	0	0	0
5	4	15145.40	15226.00	-80.60	11.11	516	0	0	0
6	5	15152.70	15281.00	-128.30	11.24	589	0	0	0
7	6	15177.55	15176.00	1.55	11.06	493	0	0	0
8	7	15137.21	15322.00	-184.79	11.17	552	0	0	0
9	8	15178.33	15194.00	-15.67	11.21	482	0	0	0
10	9	**15209.68**	**15187.00**	**22.68**	**11.14**	**583**	**0**	**0**	**0**

144.28

3rd iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15202.00	15202.00	0.00	10.96	376	1	0	0
2	1	15044.22	15347.00	-302.78	91.73	2138	294	0	0
3	2	15192.76	15339.00	-146.24	8.16	429	0	0	0
4	3	15225.77	15298.00	-72.23	8.1	386	0	0	0
5	4	15219.25	15245.00	-25.75	8.13	401	0	0	0
6	5	15239.75	15254.00	-14.25	8.11	385	0	0	0
7	6	15223.88	15263.00	-39.12	13.57	528	1	0	0
8	7	15242.32	15253.00	-10.68	8.19	438	0	0	0
9	8	15244.95	15206.00	38.95	8.12	380	0	0	0
10	9	**15246.00**	**15226.00**	**20.00**	**8.12**	**375**	**0**	**0**	**0**

173.19

4th iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15258.00	15258.00	0.00	7.17	269	0	0	0
2	1	15221.48	15320.00	-98.52	4.5	332	0	0	0
3	2	15277.29	15285.00	-7.71	4.48	291	0	0	0
4	3	15268.80	15283.00	-14.20	4.48	285	0	0	0
5	4	15277.41	15321.00	-43.59	4.56	383	0	0	0
6	5	15283.35	15305.00	-21.65	4.47	318	0	0	0
7	6	15284.94	15285.00	-0.06	4.49	281	0	0	0
8	7	15285.34	15304.00	-18.66	4.47	323	0	0	0
9	8	15290.27	15294.00	-3.73	4.48	293	0	0	0
10	9	**15293.90**	**15264.00**	**29.90**	**4.49**	**299**	**0**	**0**	**0**
					47.59				

5th iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15310.00	15310.00	0.00	4.57	193	0	0	0
2	1	15316.88	15338.00	-21.12	3.01	205	0	0	0
3	2	15310.57	15310.00	0.57	2.96	201	0	0	0
4	3	15321.13	15310.00	11.13	2.96	212	0	0	0
5	4	15325.44	15319.00	6.44	5.05	244	0	0	0
6	5	15322.02	15310.00	12.02	5.12	227	4	0	0
7	6	15327.68	15328.49	-0.81	5.17	243	3	0	0
8	7	15328.18	15338.00	-9.82	5.17	252	3	0	0
9	**8**	**15331.15**	**15329.00**	**2.15**	**5.82**	**267**	**9**	**0**	**0**
10	9	15330.95	15319.00	11.95	5.61	283	6	0	0
					45.44				

6th iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15332.00	15332.00	0.00	1.74	76	0	0	0
2	1	15339.38	15332.00	7.38	1.75	79	0	0	0
3	2	15341.38	15340.00	1.38	1.74	81	0	0	0
4	3	15343.63	15332.00	11.63	1.75	78	0	0	0
5	4	15344.26	15347.00	-2.74	3	101	5	0	0
6	5	15345.00	15347.00	-2.00	1.75	87	0	0	0
7	6	15345.63	15347.00	-1.37	1.75	87	0	0	0
8	7	15345.35	15332.00	13.35	1.74	83	0	0	0
9	8	15345.98	15340.00	5.98	1.62	87	0	0	0
10	**9**	**15346.21**	**15332.00**	**14.21**	**1.59**	**79**	**0**	**0**	**0**
					18.43				

7th iteration

#	u	Z(u)	Z	u(Ax-b)	Time (sec)	Iterations	Nodes	gap %	gap
1	0	15340.00	15340.00	0.00	1.45	49	0	0	0
2	1	15343.86	15340.00	3.86	1.45	49	0	0	0

2.9

Note – This iteration was terminated in 2 sub-iterations only. The fixing of just one variable led to a feasible schedule for the rest of the operations

APPENDIX E: DATA FOR THE 3, 4, AND 5 PRODUCT TARDINESS PROBLEMS IN CHAPTER 3

DATA FOR THE 3 PRODUCT TARDINESS PROBLEM IN CHAPTER 3

3 Products	10 Station Families	21 Work stations	One time period = 24 hrs

Workstation data

Station Family	**Number of Workstations**
1	2
2	3
3	1 (batching station)
4	3
5	2
6	2
7	3
8	2
9	1
10	2

Routes for the 3 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	6
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	9	2.5
12	10	3
13	3	6
14	5	1.5
15	7	3
16	1	1
17	4	3
18	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	6
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	6	2.5
10	1	1
11	9	2.5
12	4	3
13	8	3

PRODUCT 3		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	6
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	10	3
10	1	1
11	2	2
12	4	3
13	8	3

Data for the work orders of the 3 products

Product Number	**Order Number**	**Number of lots in the order**	**Upper bound on time for completing the work order**	**Due date of the order**
i	**j**	**k_{ij}**	**U_{ij}**	**d'_{ij}**
1	1	150	17	20
1	2	90	10	60
1	3	110	13	110
2	1	160	18	30
2	2	240	27	70
2	3	90	10	105
2	4	120	14	120
3	1	100	9	40
3	2	250	21	80
3	3	140	12	100
3	4	170	15	135

Product Number	**Maximum number of lots released in a single time period (B_i)**
1	9
2	9
3	12

DATA FOR THE 4 PRODUCT TARDINESS PROBLEM IN CHAPTER 3

4 Products	10 Station Families	24 Work stations	One time period = 24 hrs

Workstation data

Station Family	**Number of Workstations**
1	2
2	3
3	4
4	3
5	2
6	2
7	3
8	2
9	1
10	2

Routes for the 4 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	9	2.5
12	10	3
13	3	7
14	5	1.5
15	7	3
16	1	1
17	4	3
18	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1

8	2	2
9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

PRODUCT 3		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	10	3
10	1	1
11	2	2
12	4	3
13	8	3

PRODUCT 4		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	6	2.5
10	1	1
11	9	2.5
12	4	3
13	8	3

Product Number	**Maximum number of lots released in a single time period (B_i)**
1	8
2	12
3	12
4	9

Data for the work orders of the 4 products

Product Number	**Order Number**	**Number of lots in the order**	**Upper bound on time for completing the work order**	**Due date of the order**
i	**j**	**k_{ij}**	**U_{ij}**	**d'_{ij}**
1	1	250	32	100
1	2	200	25	190
2	1	150	13	20
2	2	100	9	100
2	3	100	9	180
3	1	150	13	30
3	2	250	21	100
3	3	100	9	140
3	4	100	9	180
4	1	100	12	25
4	2	250	28	90
4	3	150	17	150
4	4	150	17	180

DATA FOR THE 5 PRODUCT TARDINESS PROBLEM IN CHAPTER 3

5 Products	10 Station Families	27 Work stations	One time period = 24 hrs

Workstation data

Station Family	**Number of Workstations**
1	2
2	3
3	4
4	4
5	2
6	2
7	4
8	2
9	2
10	2

Routes for the 4 products

PRODUCT 1

Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	6
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	9	2.5
12	10	3
13	3	6
14	5	1.5
15	7	3
16	1	1
17	4	3
18	8	3

PRODUCT 2

Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	6
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5

8	7	3
9	10	3
10	1	1
11	2	2
12	4	3
13	8	3

PRODUCT 3		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	6
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	6	2.5
10	1	1
11	9	2.5
12	4	3
13	8	3

PRODUCT 4		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	6
4	7	3
5	4	3
6	1	1
7	2	2
8	3	6
9	7	3
10	1	1
11	2	2
12	9	2.5
13	4	3
14	8	3

PRODUCT 5		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	6
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2

9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

Product Number	**Maximum number of lots released in a single time period (B_i)**
1	8
2	16
3	16
4	8
5	16

Data for the work orders of the 5 products

Product Number	**Order Number**	**Number of lots in the order**	**Upper bound on time for completing the work order**	**Due date of the order**
i	**j**	**k_{ij}**	**U_{ij}**	**d'_{ij}**
1	1	150	19	20
1	2	90	12	50
1	3	120	15	90
2	1	200	13	30
2	2	150	10	65
2	3	100	7	105
2	4	90	6	165
3	1	250	16	25
3	2	110	7	80
3	3	120	8	145
4	1	100	13	40
4	2	160	20	95
4	3	110	14	140
5	1	130	9	45
5	2	240	15	85
5	3	140	9	125
5	4	160	10	170

APPENDIX F: DATA FOR EXPERIMENTS IN CHAPTER 5

DATA FOR EXPERIMENT 1, 2 AND 4 IN CHAPTER 5

Products	**Station Families**	**Work stations**	**One time period**	**One time unit**
2	8	18	24 hrs	6 minutes

Workstation data

Station Family	**Number of Workstations**
1	2
2	2
3	3
4	3
5	2
6	1
7	3
8	2

Routes for the 2 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	2	2
3	3	7
4	5	1.5
5	7	3
6	4	3
7	1	1
8	2	2
9	5	1.5
10	6	2.5
11	7	3
12	1	1
13	4	3
14	8	3

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	1
2	3	7
3	5	1.5
4	7	3
5	4	3
6	1	1
7	5	1.5
8	7	3
9	1	1
10	4	3
11	8	3

Product Number	Processing steps in route	Input rate per day (lots/day)	Initial WIP
1	14	7	9
2	11	3	3

AVAILABILITY OF WORKSTATIONS

Station Family	Workstation Number	Available at Time Unit
1	1	6
1	2	6
2	1	0
2	2	3
3	1	36
3	2	66
3	3	8
4	1	20
4	2	0
4	3	13
5	1	0
5	2	11
6	1	13
7	1	10
7	2	0
7	3	13

8	1	0
8	2	10

AVAILABILITY OF LOTS

Sr. No.	Product Number	Lot Number	Available at Time Unit
1	1	639	20
2	1	739	10
3	1	140	13
4	1	240	3
5	1	340	13
6	1	440	13
7	1	540	11
8	1	640	36
9	1	740	66
10	2	418	6
11	2	419	6
12	2	420	8

DATA FOR EXPERIMENT 3 IN CHAPTER 5

Products	**Station Families**	**Work stations**	**One time period**	**One time unit**
2	8	17	24 hrs	6 minutes

Workstation data

Station Family	**Number of Workstations**
1	1
2	2
3	3
4	3
5	2
6	2
7	3
8	1

Routes for the 2 products

PRODUCT 1		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	0.5
2	2	1
3	3	3
4	4	5
5	2	1
6	5	2
7	6	2.5
8	4	5
9	2	1
10	5	2
11	6	2.5
12	3	3
13	8	1

PRODUCT 2		
Step Number	**Station Family**	**Processing Time (hrs)**
1	1	0.5
2	2	1.5
3	7	6
4	3	2.5
5	5	3
6	6	2.5
7	7	6
8	3	2.5
9	5	3
10	6	2.5
11	3	2.5
12	8	1

Product Number	Processing steps in route	Input rate per day (lots/day)	Initial WIP
1	13	5	6
2	12	4	5

AVAILABILITY OF LOTS

Sr. No.	**Product Number**	**Lot Number**	**Available at Time Unit**
1	1	639	20
2	1	739	10
3	1	140	13
4	1	240	3
5	1	340	13
6	1	440	13
7	1	540	11
8	1	640	36
9	1	740	66
10	2	418	6
11	2	419	6
12	2	420	8

AVAILABILITY OF WORKSTATIONS

Station Family	**Workstation Number**	**Available at Time Unit**
1	1	0
2	1	9
2	2	0
3	1	5
3	2	25
3	3	5
4	1	0
4	2	20
4	3	0
5	1	1
5	2	16
6	1	20
6	2	6
7	1	41
7	2	20
7	3	0
8	1	0

SUMMARY OF EACH STAGE FOR EXPERIMENT 1

DAY 1

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15428.04	-15180	3667.8	14	14
2	-15264.13	-15205	2593.18	19	33
3	-15129.46	-15129	125.71	11	44
4		-15026	562.16		

Total Time = 6948.85 seconds

DAY 2

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15544.4	-15217	4192.19	8	8
2	-15378.1	-15245	2537.31	21	29
3	-15246.8	-15217	680.58	17	46
4		-15111	132.22		

Total Time = 7542.30 seconds

DAY 3

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15517.57	-15534	4667.62	30	30
2	-15423.13	-15227	1011.58	16	46
3		-15276	1483.29		

Total Time = 7162.49 seconds

SUMMARY OF EACH STAGE FOR EXPERIMENT 2

DAY 1

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-16820.23	-16270	2982.16	14	14
2	-16794.16	-16406	2315.00	4	18
3	-16076.00	-16247	1852.44	8	26
4	-16068.6	-16081	1242.20	7	33
5	-16073.12	-15988	1108.83	30	63
6	-15681.53	-15701	846.84	6	69
7	-15568.18	-15470	331.47	19	88
8	-15503.07	-	28.88	19	98
9		-15479	3.42		

Total Time = 10707.82 seconds

DAY 2

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15142.49	-14472	3328.90	8	8
2	-14807.99	-14772	2424.19	14	22
3	-14317.65	-14485	1208.68	14	36
4	-14392.36	-14233	391.65	18	54
5	-14049.18	-14116	178.87	12	66
6		-13906	116.14		

Total time = 7648.43 seconds

DAY 3

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-13325.09	-13423	2185.64	12	12
2	-13038.91	-12840	1824.39	5	17
3	-12079.87	-12062	1204.38	11	28
4	-11950.92	-11735	849.28	14	42
5	-11793.06	-12007	540.84	6	48
6	-11807.96	-11963	128.74	15	63

7		-11331	149.48		

Total time = 6882.75 seconds

SUMMARY OF EACH STAGE FOR EXPERIMENT 3

DAY 1

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-17134.78	-16295	2115.64	9	9
2	-17146.39	-16331	1193.17	8	17
3	-17032.62	-15906	4379.38	7	24
4	-16621.52	-16227	303.74	19	43
5	-16442.78	-15587	169.29	15	58
6	-16039.85		127.37	18	76
7		-15657	837.44		

Total Time = 9126.03 seconds

DAY 2

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-19074.53		778.23	10	10
2	-18267.96		587.20	9	19
3	-18894.39		869.29	15	34
4	-18431.16		258.45	19	53
5		-16844			

Total time = 2493.17 seconds

DAY 3

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15878.94		773.33	9	9
2	-15795.76		649.31	9	18
3	-15801.5		462.08	15	33
4	-15705.33		225.76	13	46
5		-14980	282.27		

Total time = 2392.75 seconds

DAY 4

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-15785.24		3412.59	14	14
2	-15653.89		711.86	7	21
3	-15764.06		351.24	8	29
4	-15563.29		383.13	17	46
5		-14775	841.91		

Total time = 5700.73 seconds

SUMMARY OF EACH STAGE FOR EXPERIMENT 4

DAY 1

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-21551.40	-21507	593.29	10	10
2	-21239.22	-21388	302.45	18	28
3	-21230.43	-21096	285.30	19	47
4		-20809	493.20		

Total Time = 1674.24 seconds

DAY 2

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-24621	-24400	690.28	14	14
2	-23338.60	-22883	483.20	16	30
3	-20965	-21054	350.93	13	43
4		-20745	248.83		

Total time = 1152.24 seconds

DAY 3

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-20649.80	-20489	832.92	9	9
2	-20737.90	-20291	630.38	14	23
3	-19896.70	-20074	492.20	18	41
4	-20189.16	-19567	273.48	13	54
5		-19262	430.84		

Total time = 2659.82 seconds

DAY 4

Stage	$Z_D(u^*)$	Z*	Total Time (secs)	Number of Variables Fixed	Cumulative Number of Variables Fixed
1	-14669.70	-14326	417.24	15	15
2	-13660.40	-13641	240.86	18	33
3	-13468.24	-13270	120.97	12	45

4		-13833	340.73		

Total time = 1119.8 seconds

VITA

VINOD D. SHENAI

Vinod D. Shenai was born on January 30, 1977 in Bombay (Mumbai), India. He did his schooling in Bombay and obtained the degree of Bachelor of Science in Production Engineering from the University of Bombay, India. During his undergraduate program he interned at Larsen & Toubro at Bombay for six months. He worked in the capacity of a Manufacturing Engineer at the Gas Station manufacturing unit of Larsen & Toubro in Bombay after completion of his undergraduate program. It was at this job that he realized the importance of application of optimization techniques in the manufacturing environment, and he decided to pursue his Master's degree in Industrial Engineering at Virginia Tech. During his MS, he maintained the strong academic achievements he had shown during his undergraduate studies. He worked for six months as a Research Assistant at Infineon Technologies, Sandston, Virginia, with Dr. Sarin as the Principle Investigator. After graduation, he will be working as a Business Analyst at the Richmond office of Capital One Financial, where he will be involved in the use of optimization techniques to drive business growth and needs. His research interests are Sequencing & Scheduling, Optimization, and Operations Management.