Multi-objective Resource Constrained Parallel Machine Scheduling Model with Setups, Machine Eligibility Restrictions, Release and Due Dates with User Interaction

by

Luis Munoz-Estrada

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved August 2020 by the
Graduate Supervisory Committee:

Jesus Rene Villalobos, Co-Chair
John Fowler, Co-Chair
Bradley Rogers

ARIZONA STATE UNIVERSITY

December 2020

ABSTRACT

This dissertation explores the use of deterministic scheduling theory for the design and development of practical manufacturing scheduling strategies as alternatives to current scheduling methods, particularly those used to minimize completion times and increase system capacity utilization. The efficient scheduling of production systems can make the difference between a thriving and a failing enterprise, especially when expanding capacity is limited by the lead time or the high cost of acquiring additional manufacturing resources.

A multi-objective optimization (MOO) resource constrained parallel machine scheduling model with setups, machine eligibility restrictions, release and due dates with user interaction is developed for the scheduling of complex manufacturing systems encountered in the semiconductor and plastic injection molding industries, among others. Two mathematical formulations using the time-indexed Integer Programming (IP) model and the Diversity Maximization Approach (DMA) were developed to solve resource constrained problems found in the semiconductor industry. A heuristic was developed to find fast feasible solutions to prime the IP models. The resulting models are applied in two different ways: constructing schedules for tactical decision making and constructing Pareto efficient schedules with user interaction for strategic decision making aiming to provide insight to decision makers on multiple competing objectives.

Optimal solutions were found by the time-indexed IP model for 45 out of 45 scenarios in less than one hour for all the problem instance combinations where setups were not considered. Optimal solutions were found for 18 out of 45 scenarios in less than

one hour for several combinations of problem instances with 10 and 25 jobs for the hybrid (IP and heuristic) model considering setups. Regarding the DMA MOO scheduling model, the complete efficient frontier (9 points) was found for a small size problem instance in 8 minutes, and a partial efficient frontier (29 points) was found for a medium sized problem instance in 183 hrs.

DEDICATION

This dissertation is dedicated to my wife Anabella, my children Anabella, Jonas, Clarissa and Lucas for their patience, support and encouragement. I also want to dedicate this manuscript to my parents Luis and Olga and my sister Susana for their support and words of wisdom. To all my family members and friends that cheered me up along the way- Thank you!

Above all, I want to thank God for giving me the strength and patience to persevere in this long journey.

ACKNOWLEDGEMENTS

The completion of this Ph. D. manuscript in Systems Engineering could not have been possible without the technical direction, assistance and guidance of my ASU faculty committee members Dr. Villalobos, Dr. Fowler and Dr. Rogers. Their technical support, and guidance were invaluable. I want to acknowledge Intel Corporation for supporting my studies and also thank my Intel fellows who supported me during this time: CC Liong, Dr. Shai Rubin, Jim Evers, Matt Ward, Ricardo Guemes, Andrea Balmores, Rital Klin, Brad Urbanek, Steve Shong, Jarrod Bowser, Lorinda Braun, Michael Vento, Dr. Chelsea Brown, Dr. Josh Prickett, Katherine Adams and Dr. Ramona Perez among others. Special thanks to Dr. Gerardo Trevino for his technical guidance in the field of operations research.

TABLE OF CONTENTS

Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1 INTRODUCTION

1.1. Introduction

When manufacturing capacity becomes the lead constraint in revenue and profit, efficient operations of the production systems can make the difference between a thriving and a failing enterprise. This is particularly true when expanding capacity is limited by the new equipment lead time or the high cost of acquiring additional manufacturing resources. Thus, increasing utilization of existing capacity is extremely important, not only for the manufacturing industry, but also for the economies of the communities in which they operate. One of the industries that can benefit significantly from efficient and effective manufacturing operations is the electronics industry, specifically the semiconductor industry.

According to Hunter et al. (2002) semiconductor manufacturing fabrication lines are considered some of the most complex manufacturing processes in the world. This is due to multiple factors such as highly re-entrant and long process flows, rework, and variable yields. The fabrication of 200 mm and 300 mm wafers require over 1,000 processing, inspection and test steps (Brown et al., 2010). Figure 1-1 depicts a generic, highly re-entrant process flow consisting of pad deposition, photoresist, lithography, etch, and strip. As shown, lots can be sent to inspection and/or measurement from any of the process steps previously mentioned. After the strip process step, lots are sent to implant and/or diffusion, insulator deposition, polish, and metal deposition. This process repeats multiple times until lots are sent to bond, assembly, and packaging.

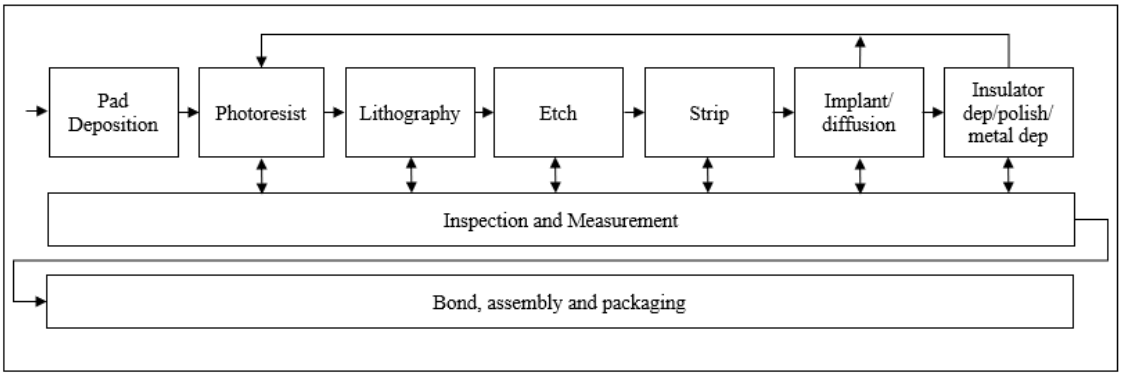


Figure 1-1: Wafer Fabrication Flow (Brown et al., 2010)

Brown et al. (2010) claim that the time to complete a lot is typically several months; however, as manufacturers are able to shrink the size of transistors and pack more chips on a single wafer, new technologies face greater manufacturing challenges, resulting in even longer throughput times.

In this dissertation, we advocate for the use of deterministic scheduling theory for the design and development of more efficient scheduling strategies as alternatives to current methods to increase the utilization and the capacity of the manufacturing system. "Sequencing and scheduling are forms of decision-making that play crucial roles in manufacturing and service industries. In the current competitive environment effective sequencing and scheduling has become a necessity for survival in the market-place" (Pinedo, 2016). Scheduling optimization is an approach widely used in manufacturing to improve the performance of a system by optimizing multiple competing objectives such as minimizing machine idle time, setups, job completion times, job tardiness and maximizing throughput. According to Hopp and Spearman (2011) increasing throughput of the factory constraint will increase the overall factory throughput to that given by the rate of the new constraint. It is worth highlighting that these complex manufacturing

systems not only arise in the electronics and semiconductor industry, but also in telecommunications and plastic injection molding industries, among others. Photolithography is an example of these complex systems in the semiconductor industry.

This dissertation/research also aims to reduce the chasm between theoretical and practical research for manufacturing scheduling systems by proposing a heuristic that interacts with the Integer Programming (IP) model to ultimately find better solutions. Before we establish the statement of the problem, it is necessary to introduce some concepts and the notations to be used throughout this manuscript.

1.2. Definition of Terms

There are multiple scheduling scenarios in the open literature including, but not limited, single machine, parallel machines, flow shop, flexible flow shop, open shop, and job shop. This research draws concepts from job shop and parallel machines scheduling since these environments are often found in the real world. Job shop scheduling can be framed as an operations research optimization problem consisting of a finite set of jobs $j \in J$ that must be processed on a finite set of parallel machines $m \in M$ requiring auxiliary resources $r \in R$ in such a way that the overall resulting production plan is optimal, or at least efficient, with respect to one or more measures of performance. An auxiliary resource is an additional scarce resource needed to process a job. A formal definition is provided below. According to Çaliş and Bulkan (2015) "The job shop scheduling problem is one of the most important and complicated problems which has been known to be NP-hard. That is, the high complexity of a given problem makes it extremely difficult to find the optimal solution within reasonable time in most cases".

Unlu and Mason (2010) stated that parallel machine scheduling is also NP hard. It is noteworthy that a considerable amount of literature has been published in the field of scheduling during the past 50 years. These studies cover a wide spectrum of problems ranging from deterministic to stochastic optimization, single vs. multiple machines, single vs. multiple resources, single vs. multiple objectives, job vs. family, and setup vs. no setup times. According to Allahverdi (2015) "tens of thousands of papers, addressing different scheduling problems, have appeared in the literature since the first systematic approach to scheduling problems was undertaken in mid-1950s". However, a limited number of publications in the scheduling field have focused on practical models using time-indexed, mix integer linear programing (MILP) and Integer Programing (IP) methods for parallel machines with auxiliary resources and multiple competing objectives including sequence-dependent setup times.

Despite the fact that the majority of scheduling researchers perceive time-indexed IP/MILP formulations unattractive for real-time applications, Avella et al. (2017) were able to challenge this claim by developing a time-indexed IP approach capable of solving to optimality real-life instances of airplane runway scheduling. On a similar note, the author of this dissertation conducted a comparison of six single machine formulations and the time-indexed IP outperformed five other single machine formulations for problem instances between 10 and 65 jobs. It is also important to highlight that sequence-dependent setups have an adverse and significant impact on the efficiency of the manufacturing systems since machines must pause processing while waiting for an auxiliary resource to be made available. Thus, as previously stated, scheduling theory is

proposed in this study to aid manufacturers become more efficient through the application of scheduling algorithms and best practices.

According to Pinedo (2016) the following framework and notations are able to succinctly capture the majority of the deterministic scheduling problems that arise in general manufacturing environments:

- **Resource scheduling** can be defined as the assignment of jobs to limited processing resources (i.e. machines, tools, auxiliary resources, etc.) in order to find the schedule or sequence that optimizes an objective function.
- **Auxiliary resource** is defined in this dissertation as an additional scarce resource that can be shared across multiple machines and it is required to be loaded at one machine in order to process a job. If the auxiliary resource is available, but the machine is not, then the job cannot be processed. Vice-versa, if the machine is available, but the auxiliary resource is not available, then the job cannot be processed either.
- **Setup** is defined as the idle time a machine incurs when two jobs are not compatible with each other; hence, an auxiliary resource swap is required.
- **Process time (p_j)** is the time to process job *j* on any identical parallel machine.
- **Release date (r_j)** is the time when job *j* arrives at the system also known as ready time.
- **Due date (d_j)** is the committed shipping date for job *j* or completion date.
- **Weight (w_j)** is a priority factor denoting the importance of the job *j* relative to other jobs in the system.

- **Operation (o_j)** is the operation attribute for job *j* that denotes job's location.

Most scheduling problems in the open literature can be described as a triple α | β | γ where α field describes the machine environment, β provides details of processing characteristic and the γ field describes the objective function(s) to be minimized.

- **Identical machines in parallel (p_m)** denotes a workstation with *m* identical machines in parallel that can run process job *j* at operation *o*. if a job is restricted to run on certain machines, the subscript *Mj* is used in field β to denote the machines that can process job *j*.
- **Sequence-dependent setup times (s_{jk})** represent the set up time between job *j* and *k*. s_{0k} denotes set up time for job *k* if it is the first in the sequence and s_{j0} is the clean-up time after job *j* if it is the last job in the sequence.
- **Preemptions *(prmp)*** means job *j* can be interrupted at any time.
- **Completion time (C_j)** is the time when job *j* exits the system.
- **Makespan (C_{max})** is the equivalent to the completion time of the last job *j* processed.
- **Lateness (L_j)** is the amount of time by which the completion time of job *j* exceeds its due date (d_j). If Lateness is negative the job *j* completed early-> $L_j = C_j - d_j$
- **Tardiness (T_j)** is the positive lateness of a job where $T_i = (C_j - d_j, 0) = \max(L_j, 0)$.
- **Unit penalty (U_j)** is one if $C_j > d_j$; 0 otherwise.
- **Flowtime (F_j)** is the amount of time job *j* spends in the system.

Some of the common objective functions that appear in the scheduling literature include:

- **Total weighted completion time ($\sum w_j C_j$)** is the sum of the weighted completion times of *n* jobs also known as flow time.
- **Total weighted tardiness** ($\sum w_j T_j$) is the sum of weighted tardiness of *n* jobs.
- **Maximum lateness (*max* L)** is the worst violation of the due dates
- **Tardy jobs** ($\sum U_j$) **and weighted number of tardy jobs** ($\sum w_j U_j$) are mainly academic objective functions (Pinedo, 2016).

Multi-resource scheduling can be defined as the assignment of jobs to a machine and auxiliary resources simultaneously in order to find the schedule that optimizes the objective function or functions. For instance, Bitar et al. (2016) investigated a memetic algorithm for unrelated parallel machine scheduling that considered auxiliary resources (i.e. lenses) with two optimization criteria tested separately, minimize weighted flow time and maximize number of products that are processed denoted as $R_m|aux|min \sum W_j F_j , \sum Product\ Count$.

Ham (2018) investigated a parallel machine scheduling problem with auxiliary resources in semiconductor manufacturing with a single objective to minimize the maximum completion time $P_m|s_{jk}, r_j, M_j, aux \mid \sum C_j$.

The practice of considering two or more optimization objectives simultaneously is known as Multi-objective optimization (MOO). Addressing two or more objectives is not commonly used in optimization models that have been developed for job scheduling. However, the use of more than one objective is a common practice in the semiconductor industry (Ham, 2018). Multi-objective optimization (MOO) is defined as the assignment of jobs to resources with the goal to simultaneously optimize *k* objective functions

defined as: $[f_1(x), f_2(x), \dots, f_k(x)]$ and forming a vector function F(x): defined as: $[f_1(x), f_2(x), \dots, f_k(x)]^{\mathrm{T}}$. In most cases, the objectives are usually in conflict with each other and a solution must be found in which the values of all the objective functions are acceptable to the user (Chiandussi et al., 2012). Next, we present the statement of the problem.

1.3. Statement of the Problem

The focus of this dissertation is to develop efficient model-based procedures for the scheduling of identical parallel machines with shared, constrained, auxiliary resources with sequence-dependent setups, machine eligibility restrictions, job release and due dates with single and multiple objectives. The need to effectively schedule finite resources exist across multiple industries including semiconductor fabrication and assembly, mold injection, and printed circuit board (PBC) assembly, amongst others. Thus, improvements in modeling to get efficient scheduling solutions will have a large impact on those manufacturing systems subject to capacity constraints, especially if the system being optimized is the overall factory constraint. As previously stated, increasing throughput at the factory lead-constraint will increase the overall factory throughput dictated by the new constraint.

We aim to develop a single objective time-indexed IP model and a MOO time-indexed IP model that resemble practical manufacturing systems. We also aim to develop a heuristic capable of solving large problems to prime the time-indexed IP models by providing feasible initial solution. The proposed models can be defined as $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$ and $P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|\sum w_j C_j, \sum T_j, T_{max}$

Figure 1-2 depicts the environment we plan to study where jobs must be processed by one machine and one auxiliary resource simultaneously. Also, the auxiliary resource cannot be assigned to more than one machine at the same time.

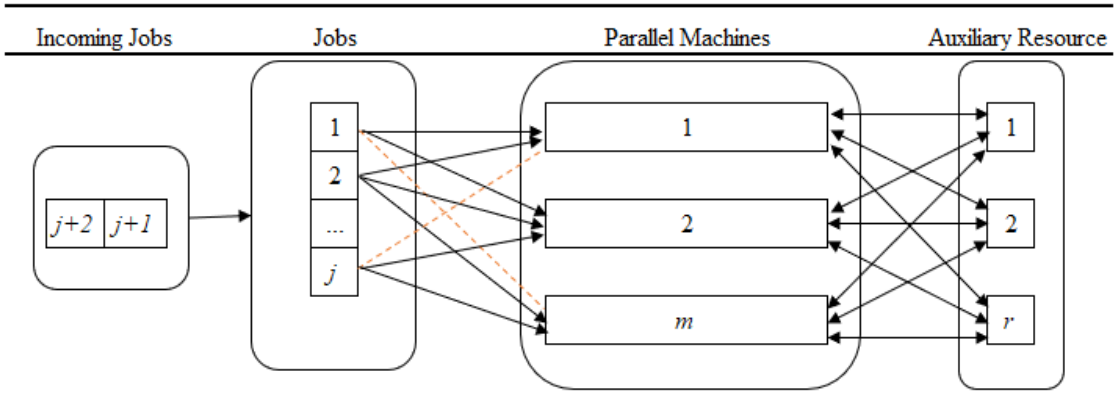


Figure 1-2: Generic Manufacturing System with Parallel Machines and Auxiliary Resources

This environment has a set of jobs *J* either ready to be processed ("ready jobs") or incoming to the system. These jobs can be processed in any of the unrestricted *M* parallel machines, provided that they are fitted with an instance *r* of the set of scarce auxiliary resources *R*, which are shared with other machines. Note the dotted line between jobs and machine in Figure 1-2 denotes a job-machine restriction. Incoming jobs require a release date to prevent assignment before they are available. Due dates are also incorporated to meet customer on-time delivery commitments. Once a job is assigned to machine-auxiliary resource pair, that job must complete processing. That is, preemption is not allowed. Once a job has been assigned to a combination of resources, that job may need a setup time. A setup time is incurred when job *j* is not compatible with the previous job and a different auxiliary resource is required to be added to the machine to run the job. Thus, the machine must temporarily idle for the auxiliary resource setup. If jobs are

compatible with each other they belong to the same family since no setup is required. In practice, a family consists of a combination of specific product and an operation; however, for this study a family is created by randomly assigning jobs to family sets forcing each set to have a similar number of jobs. If a resource is transferred from one machine to another one, an additional setup time (travel time between machines) is required. Inefficient resource assignments may cause a machine to be temporarily idle until the proper auxiliary resource is installed. Typically, auxiliary resources are finite and expensive that need to be shared among different machines, thus if a resource is being used in one machine, another one may be idle waiting for this resource. Thus, generating efficient schedules not only optimizes resource assignment and utilization, but also overall factory throughput resulting in lower cycle times. Since multiple job-machine scheduling studies have been published across multiple industries we plan to keep our terminology and assumptions aligned to the open literature (Pinedo, 2016; Ham, 2018; Edis, 2009).

1.4. Dissertation Goals

One of the goals of this research is to find efficient solutions for the manufacturing system previously described by applying time-indexed IP model in conjunction with heuristics. Our goal is to develop a framework that allows collaboration among exact methods and heuristics to provide more tools and options to practitioners looking to implement more sophisticated models to solve real-world practical problems. Another goal is to develop multi-objective (MOO) models to enable the decision maker to interact with the model in order to test different schedules and hypothesis via efficient frontier.

Not only should the model allow user input restrictions, but also help the decision maker trust the optimization model and its parameters. It needs to be noted that our goal is to use the MOO model for strategic purposes and not real-time scheduling.

It is important to highlight that there is a paucity of publications in the scheduling field focusing to bridge the gap between theory and practice. Pinedo (2016) states that it is not clear how stylized and elegant mathematical models studied by academic researchers can be applied to real-world scheduling problems since they tend to differ significantly. Table 1-1 presents key differences between academic research and industry needs as listed by Pinedo (2016). These differences are particularly related to objective functions, job-machine availability, job priorities, penalty functions and dynamic environments.

Table 1-1: Key Differences Between Academic and Industry Needs

Key differences	**Theoretical scheduling assumptions**	**Real-world scheduling needs**
Objective function	Deal with single objective function	Require two or more objectives
Jobs in the system	All jobs are ready to be scheduled	Jobs continue to arrive, dynamic environment
Machine assumptions	Simplified machine restrictions	Real processing restrictions may require more involved constraints
Reactive scheduling	Do not emphasize in resequencing after random changes occur	Require reactive schedules to accommodate minor/major random event changes
Job priorities (weights)	Same weight is applied across all time periods	Weights typically fluctuate over time
Machine availability	Assume machine is available 100% of the time	Machine availability fluctuates due to unscheduled maintenance

The topic related to incorporating human expertise to scheduling has received mixed reviews and will be covered in detail in the literature review since there is controversy on benefits of incorporating expert knowledge into the solution. Some researchers state that complex scheduling systems are beyond the cognitive capability of human beings (Steffen, 1986) and (Fox, 1990). However, other authors state there are potential benefits (Reinschmidt et al., 1990) and (Framinan and Ruiz, 2010). We suggest and propose that human experts and decision makers should be able to provide feedback to scheduling systems.

We believe there is merit in studying semiconductor manufacturing systems given the recommendations previously stated by Allahverdi (2015) and Ham (2018). While multiple publications have successfully applied IP and MILP models with heuristics to solve parallel machine scheduling problems, to our knowledge no single study has addressed this problem by applying the MOO MDA resource constrained parallel machining scheduling model with and without setups using the time-indexed IP model. Our proposed research should contribute to multiple manufacturing industries to address tactical and strategic practical applications. Our problem should be of great relevance, not only in the semiconductor industry, but also in the molding injection industry amongst other.

1.5. Benefits of Proposed Research

According to Brown et al. (2010) advanced queueing models and optimization techniques can become an integral part of predicting factory bottlenecks, prioritizing continuous-improvement efforts, planning capital equipment investments, and managing

factory lead times. The return on the investment reported by optimizing IBM's fab was over $30 million in capital avoidance and over $700,000 per year in lower operating expenses in addition to lead times and variability reduction. We estimate today's complex manufacturing facilities could realize similar benefits or greater as each technology becomes more complex driving inefficiencies to a higher degree. Commercial software vendor Applied Materials published "With Scheduling, customers can defer or eliminate investment in additional lithography equipment by increasing tool utilization. Customer results reported litho equipment utilization > 1% and throughput improvement of 2.1%" (Applied Materials, 2012).

Another publication claims the following "To demonstrate the effectiveness of the scheduler's objective function capabilities, we used samples of production data to generate a production schedule; the same data was compared with pure heuristics-only real time dispatching. In this test, the software significantly improved cycle time for the highest priority lots by ~20%, then improved the second highest priority lots by <5%. The cycle time for lower priority lots remained unchanged. This example verified that the objective function capability in the scheduling component works and helps customers meet their KPIs, in this case, improving cycle time" (Marteney, 2011). We believe a practical scheduling framework combined with effective IP models and heuristics has potential to outperform existing rule-based heuristics in many manufacturing industries. Avella et al. (2017) stated that most of the scheduling research using time-indexed IP/MILP report that these models are unattractive for real-time applications due to computation times likely to grow too large. In their paper, the authors reverse this claim

by developing a time-indexed IP/MILP approach capable of solving to optimality real-life instances of airplane runway scheduling. The following section presents the organization of this manuscript.

1.6. Organization of the Dissertation

This dissertation is divided in three phases. Phase I and II aim to generate efficient schedules for small, medium and large problem instances under one hour. Phase III aims to generate a MOO framework and strategic models—Pareto efficient frontier— to enable decision maker to interact with the schedule/model. Feedback is given to the decision maker on the goodness of the edited schedule in the form of distance metric to the closest optimal point. The remaining part of this dissertation proceeds as follows.

Chapter 2 presents an overview of the existing literature related to semiconductor scheduling, optimization models and practical production scheduling. This chapter also discusses the limitations and gaps of the existing literature.

Chapter 3 presents the methodology, research objectives, a heuristic-optimization framework and assumptions for the problem instances data input.

In chapter 4, we present the phase I model with the objective to minimize total completion time for the resource constrained parallel machine scheduling model with release dates, job-machine eligibility restrictions $P_m|r_j, M_j, aux\ 1|\sum C_j$ for five problem instance sizes, three complexity levels and three machine eligibility restriction levels

In chapter 5, we present the phase II mathematical models consisting on phase I formulation with addition of sequence-dependent setups and resource travel time between machines $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$ for five problem instance sizes, three complexity

levels and three machine eligibility restriction levels. A heuristic that primes the optimization IP model is also presented in this chapter.

In chapter 6, we present phase III mathematical formulation consisting of phase I and II models combined with the DMA lexicographic MOO model for parallel machine scheduling denoted as $P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha, \sum w_j C_j, \sum T_j, T_{max})$.

In chapter 7 and 8 we present the results for the three phases, conclusions and future research respectively.

CHAPTER 2 LITERATURE REVIEW

Many comprehensive surveys have been published since 1979 up to a more recent comprehensive survey by Allahverdi (2015) which listed hundreds of papers followed by the suggestion that there is a need for more research in scheduling models that mimic real-world practical problems. This chapter aims to summarize publications related model optimization, semiconductor scheduling and practical production scheduling. This chapter is subdivided in three subsections as depicted in Figure 2-1. Our problem ("OP") lies in the intersection among semiconductor scheduling presented in section 2.1; optimization modeling presented in section 2.2; and practical production scheduling presented in section 2.3. Finally, section 2.4 presents literature review conclusions and our contributions.

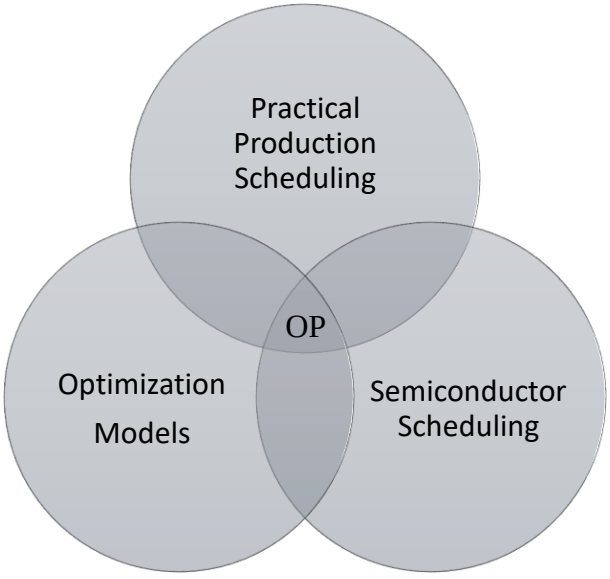


Figure 2-1: Literature Review Sections

2.1. Semiconductor Scheduling

Publications related to semiconductor scheduling and real-time dispatching rules are presented in this section. Resource scheduling in semiconductor industry is considered a complex task. In fact, semiconductor manufacturing fabrication lines are considered some of the most complex manufacturing processes in the world, mainly due to long and highly re-entrant process flows, rework, and variable yields (Hunter et al., 2002). This section provides an overview on the scheduling publications in semiconductor manufacturing with emphasis in photolithography machine scheduling.

2.1.1. Real-time Dispatching Rules

For this study, we define dispatching rules as follows: "A dispatching rule is used to select the next job to be processed from a set of jobs awaiting service. The dispatching rule selected can be very simple or extremely complex" (Blackstone et al., 1982). Simple rules range from selecting jobs randomly or oldest job to more complex rules selecting jobs based on customer's due date if the inventory has dropped below a pre-specified threshold. One of the comprehensive studies on dispatching rules across multiple job shop scenarios conducted by Blackstone et al. (1982) concluded that there is no one single rule that is superior to the rest under all circumstances. The study analyzed a total of 34 rules; however, the authors only provided a summary for the top eight rules which are shown in Table 2-1.

Table 2-1: Dispatching Rule Summary

Dispatching rule	**Description**
Shortest Imminent Operation (SI)	Selects job with shortest processing time
Truncated SI	F_j={time until due date-remaining processing time-Parameter} for job *j* if F_j>0 select job by SI. if F_j<=0 select job smallest F_j
Earliest due date	job selected based on earliest due date
Least slack (dynamic)	Time since job entered operation (smallest value)
Least slack-per-operation (dynamic)	Time since it entered operation (smallest value)
Critical ratio	Days remaining until due date/ Lead time remaining
FIFO (as a 'control' rule)	Job selected as first-in-first-out basis
COVERT	Delay cost over time remaining

Blackstone et al. (1982) used flowtime, lateness, and tardiness as the key measurement criteria to study and compare dispatching rules performance. The reader is referred to section 1.1 for definitions presented in table above. Blackstone et al. (1982) concluded that shortest imminent operation (SI) rule was generally the best rule when the shop floor had either tight or loose due dates, or the shop did not set due dates. If due dates were present the truncated SI metric performed well since it allows the decision maker to enter the parameters to balance SI and $F_{j.}$

Similarly, Min and Yih (2003) conducted a study on wafer fabrication scheduling approaches and proposed a wafer fabrication scheduler aiming to select real-time dispatching decisions based on user-defined objectives. The proposed strategy was to select rules initiated by both machines and vehicles transporting lots in the factory. A simulation model was used to generate the expected relationship between the selected decision variables, dispatching rules, current system status, and performance metrics of a

semiconductor manufacturing fab. The decision variables, dispatching rules and objectives applied in their study are listed below:

- Decision variables: lot selected by critical machine, non-critical machine, stocker (buffer) rule, and vehicle in monorail rule.
- Dispatch rules linked to each of the decision variables: First-In-First-out (FIFO), Shortest Remaining Processing Time (SRPT), Earliest Due Date (EDD), Critical Ratio (CR), Lowest Remaining Space in Stocker (LRSS) and In Bay First (IBF) among others.
- Objectives to optimize were mean flow time, slack time and total remaining processing time.

A brief summary of previous approaches to wafer fabrication scheduling was also presented by Min and Yih (2003) and it is depicted in Table 2-2.

Table 2-2: Approaches to Wafer Fabrication Scheduling (Min and Yih, 2003)

Author	Approaches to Wafer Fabrication Scheduling	Summary
Li et al. (1996)	Minimum inventory variability schedule (MIVS)	Reduced variability via simulation model using WIP level as indicator for variability between output and processing rate of station downstream
Lu and Kumar (1991)	Dispatching rules: buffer-based and due-date based rules	BFR: focuses on processing step to perform or buffer to serve. DDBR: Focus on earliest due date and least Slack. Simulation results show that BBR policy performs well reducing mean cycle time. Least slack policy gives good results for minimizing variance
Kim et al. (1998a)	minimize mean tardiness using dispatching rules adapted for photo and non-photo workstations	Simulation results showed that dispatching rules in photo workstations have a greater effect on performance than dispatching rules in non-photo workstations
Baek et al. (1998)	Spatial Adaptation Procedure (SAP)	Simulation and Taguchi experimental design applied to select the most appropriate dispatching rule. Results showed that the SAP method reduced mean cycle time when compared to a single decision rule policy
Hung and Chen (1998)	simulation-based dispatching rule to reduce a cycle time in a fab	Aim to predict waiting times and flow times using 2-phase simulation (parent and children simulations). The results outperformed other static dispatching rules that use only queue information at the time of dispatch
Nakata et al. (1999)	Justice/Moral method dynamically detects a bottleneck machine and feeds work to the machine at an appropriate time	Bottleneck synchronized with all the machines in the line controlling progress speed of all lots. Simulation results indicated that cycle time can be reduced by ~ 13%, throughput increase ~10% vs FIFO rule
Lin et al. (2001)	Vehicle dispatching policies via simulation	Simulation results show dispatching policy has a significant impact on average transportation time, waiting time, throughput and vehicle utilization. Combining shortest distance and nearest vehicle and first encounter first-served rule outperformed the other rules

Min and Yih (2003) suggest that most studies have focused on algorithm development for single objective (i.e. cycle time reduction and throughput increase) and single dispatching decision variables. However, semiconductor scheduling is complex; hence, rule selections must consider multiple objectives and multiple decision variables in order to utilize resources efficiently. The authors also concluded that their proposed system (scheduler) was able to select effective dispatching strategies given the

complexity of semiconductor wafer fabrication systems. The authors also concluded that no single dispatching strategy consistently dominated the others in all situations.

Mönch et al. (2011) published a comprehensive survey of semiconductor manufacturing problems, solution techniques, and challenges in semiconductor manufacturing operations. The authors identified typical scheduling problems encountered in semiconductor manufacturing systems. Not only were batch scheduling problems presented but also parallel machine scheduling problems and scheduling problems with auxiliary resources which is the focus of this study. The key metrics used in this survey are: (a) Cycle time, (b) Throughput, and (c) On-time delivery performance measures. Table 2-3 presents the characteristics of some of the scheduling problems found in wafer fabrication lines.

Table 2-3: Characteristics of Scheduling Problems (Mönch et al., 2011)

Work area	Parallel machines	Dedication	Cluster tools	Bottleneck	Batching	Sequence-dependent setups	Auxiliary resources
Oxidation Deposition (CVD/PVD) Diffusion	Y	Y	N	N	p-batching with incompatible families	N	N
Lithography	Y	Y	Y	Y	s-batching with job availability	Y	reticles
Etch	Y	Y	Y	Y	N	N	N
Ion Implantation	Y	Y	N	Y	N	Y	N
Planarization	Y	Y	N	N	N	N	N

Mönch et al. (2011) cited the work of Park et al., (1999) who conducted a simulation analysis to study reticle management issues as a result of different storage and inspection policies, and the relationship between cycle time and product mix. They concluded that jobs wait due to reticle unavailability and/or machine availability.

Mönch et al. (2011) also cited the work of Akcali and Uzsoy (2000), the photolithography work center scheduling problem assuming that an operation requiring a reticle type can be assigned to at most *r* machines, where *r* is the existing number of reticles of the corresponding type that are available in their simulation-based analysis.

Cakici and Mason (2007) studied the scheduling heuristics based on dispatching rules for the problem $P_m|r_j, aux|\sum w_j C_j$ motivated by stepper scheduling problems (as cited Mönch et al. (2011)).

A slightly different approach to job scheduling research is proposed by Zhang et al. (2017) based on Industry 4.0. This concept started from a German government project in the high-tech industry and it is gaining momentum with researchers. The key idea behind Industry 4.0 is to transition from centralized scheduling to more intelligent distributed manufacturing systems supported by mass customization, Cyber-Physics Systems, Digital Twin, and SMAC (Social, Mobile, Analytics, Cloud). The authors presented over 120 papers written as early as 1954 (Johnson's Optimal two and three stage production schedules with setup times included) to the most recent algorithms related to metaheuristic methods such as ant colony and particle swarm optimization depicted in Figure 2-1.

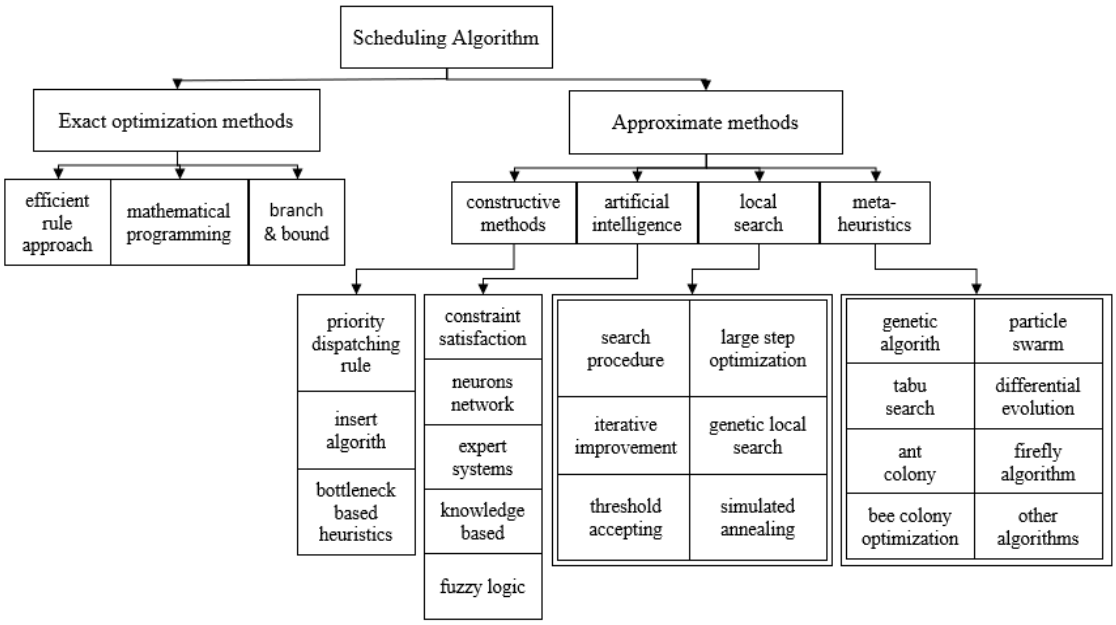


Figure 2-1: Scheduling methods (Zhang et al., 2017)

One of the conclusions reached by the authors is that manufacturers must rely on technology to enable smart objects or entities interact with each other to optimally allocate resources based on static and dynamic scheduling. Zhang et al. (2017) propose that researchers should focus more on Multi-resource Flexible Job Shop (MrFJSP) and Multi-plants Flexible Job Shop MpFJSP algorithms to prepare for industry 4.0. The meta-heuristic methods shown in Figure 2-1 are gaining more momentum and Genetic Algorithms is one of the most frequently cited meta-heuristic methods in the open literature (Zhang et al., 2017). Overall, the concept proposed include scheduling decentralization and autonomous decisions; flexibility and adaptability; integration and networking in environments with rising complexity. It is worth to highlight the authors are proposing more research on approximate methods instead of focusing on exact mathematical methods. This brief overview of publications in semiconductor dispatching and optimization models leads to a review of publications related to machine scheduling

with auxiliary resources in the photolithography functional area which is considered one of the most complex areas in wafer fabrication lines.

2.1.2. Photolithography Machine Scheduling

Photolithography—the process step in semiconductor manufacturing in which circuit patterns are transferred onto the wafer thru a lens known as a reticle—is a complex manufacturing process due to its highly reentrant nature. It is necessary to subject the wafer under fabrication to several applications of the lithography machines to get the final version of the edged circuitry, leading to the reentrant nature of the semiconductor manufacturing process. Akcalt et al. (2001) claim that "Photolithography is usually the bottleneck process with the most expensive equipment in a wafer fab. Being one of the processes that is repeated the most during fabrication, any reduction in photolithography cycle time will reduce overall fab cycle-time". According to Yan et al. (2013) lithography is the major constraint in semiconductor manufacturing due to limited number of expensive resources and its process complexity. Consequently, it is imperative that semiconductor manufacturers seek more effective ways to manage photolithography and other factory constraints in order to improve key performance indicators.

Ham and Cho (2015) proposed a two-phase approach to machine scheduling in lithography by integrating dispatching logic with MILP technique. For phase 1, the following is considered:

Sets

- *J* set of jobs {j}
- R set of auxiliary resources {r}

- M set of machines $\{m\}$

Parameters:

- C_{rm} shipping (=setup) cost of auxiliary resource *r* to machine *m*,
- P_j processing time of job *j*
- T_{prev} previous iteration's completion time for machine *m.*

Decision variables:

- X_{jrm} 1 if job j is assigned to machine m with auxiliary resource *r.*
- *Yrm* 1 if auxiliary resource *r* is assigned to machine *m*
- *Tm* completion time of machine *m*
- *Gm+* amount of production overachievement by machine *m*,
- Gm^- amount of production underachievement by machine *m*
- C_{MAX} completion time of the last completing machine
- C_{MIN} completion time of the first completing machine.

The mathematical model for the two-phase approach has four terms in the objective function aiming to minimize overall cycle time while balancing the load among machines. The constraints can be summarized at a high level as follows:

- minimize auxiliary resource setup cost
- calculation of completion times
- prevent model from over and under achieving the output targets.

The phase 1 model utilizes a MILP method known as the transportation model where jobs, auxiliary resources and machines are assigned in order to minimize cycle time as previously described. Phase 2 is described as the lower-stage level that optimizes job

dispatch sequence for each machine by iterating between phase 1 model and a rule-based heuristic in phase 2. It can be observed that this two-phase practical approach is quite simple, but practical and outperforms existing rule-based heuristics (Ham and Cho, 2015).

Three years later, Ham (2018) published the "Scheduling of Dual Resource Constrained Lithography Production" model. The dual resource constrained (DRC) problem arises when a finite number of machines and auxiliary resources are needed simultaneously to process a job. This problem becomes more challenging when auxiliary resource must be tracked, and a setup time is incurred as a function of auxiliary resource location. This means, an auxiliary resource swap within the machine may take two minutes when an auxiliary resource swap between machines may take 20-40 minutes. The concept is to apply a hybrid approach using MILP model and a constraint programing (CP) model to solve the complex DRC problem. The scheduling problem was decomposed into two sub-decisions: a) assign jobs to auxiliary resources and machines without sequence (MILP) and b) let the CP solver start a search in a set of decision variables referred as warm start, aiming to reduce search time.

The overall results showed that CP alone outperformed the hybrid model initially proposed. The author stated that CP is a promising method to reach optimality for large-scale problems that once were intractable. CP proved optimality for the small (20 jobs), medium (50 jobs), large (100 jobs) and industry-size model (200 jobs) within two minutes and a gap of <1% GAP against the best solution found. It is important to mention that Ham, (2018) concluded in his paper that his MILP model used was based on

positional and assignment variables which turned out to be inefficient for the given problem due to the large number of binary variables. His proposed MILP model took several hours to find an optimal solution even for a small problem instance (10 jobs). As future research Ham (2018) proposed to explore time-indexed IP/MILP models as an alternative to his exact model.

Yan et al. (2011) proposed a MILP model with the objective to meet daily production targets while reducing the number of auxiliary resource setups. A branch-and-cut method was used to solve the problem. The authors in this paper do not distinguish among different lots and focus on machine and auxiliary resource setups which differs from Ham and Cho's approach. Yan et al. (2012) modified their previous model to include load balancing and auxiliary resource expiration. The authors continue to use MILP models with capacity constraints and process requirements. The objective is to meet daily targets while keeping the load balanced to minimize auxiliary resource shortage and machine setups. In this paper, they solved the model using a two-phase approach as well, the higher phase reduced the problem range through constraint relaxation. For the lower phase they applied a similar concept to Ham and Cho, (2015) consisting on scheduling jobs and removing them from the unscheduled set.

Yan et al. (2013) proposed a photolithography machine scheduling approach based on convex hull analyses. The authors discovered that the convex hull of the problem was difficult to solve; hence, a two-phase approach was also utilized in order to solve the problem within reasonable time. Phase 1 consisted of removing certain complicated constraints (i.e. auxiliary resource setups) from the original full-size problem

in order to solve it more efficiently. In other words, the first phase reduces the ranges of the decision variables. Then, phase 2 efficiently solves the problem with the full set of constraints within a reduced decision space. For instance, the problem was simplified by fixing the machine-auxiliary and resource-layers assignment and/or fixing the number of lots scheduled in each machine-auxiliary resource combination. "consider a fab with M lithography machines (*m*), *R* auxiliary resources (*r*), *P* product types (*p*), *L* layer types (*l*) and *K* discrete periods (*k*) within a day." (Yan et al., 2013). The objective function and ten constraints are described next.

The objective function has four terms that are solved simultaneously in one step which may not lead to an optimal solution as opposed if these four terms were solved using an iterative multi-objective optimization (MOO) method. The four terms are summarized as follows:

- Meet daily targets for product *p* and layer *l.*
- The second term balances future loads, W^L denotes the weight for future stacking layer load balancing. S^{SG} and $S_g{}^{MS}$ denote set of stacking groups and set of machines in stacking group g respectively. LD_{gm} denotes load difference for a stacking group per machine.
- The third term avoids simultaneous auxiliary resources expiring, G_{pl} denotes expected expiration interval for auxiliary resources *p* and *l*. R denotes the auxiliary resource lifetime before recalibration is needed and W^{RP} is the reward-penalty weight. 4) The fourth term aims to avoid excessive auxiliary resource setups, the term y_{mr} denotes the beginning and completion points as binary

variable for a machine *m* and auxiliary resource *r* pair. The constraints for this method can be summarized at a high level as follows: 1) do not exceed resource capacity, 2) calculate processing time requirement and 3) maximize the number of lots scheduled for each setup and 4) minimize the number of machine-auxiliary resource setups.

Up to this point dispatching rules and photolithography machine scheduling have been discussed. There seems to be a shortage of publications related to parallel machine scheduling with auxiliary resources on semiconductor manufacturing. Most researchers concluded that semiconductor wafer fabrication facilities are complex and IP/MILP methods alone cannot solve industry-size problems. The proposed methods rely on hybrid concepts including, IP, MILP, CP and heuristics. It is worth mentioning that none of the publications found has attempted to use the time-indexed IP or MILP method which is proposed in this dissertation. The next subsection presents an overview of single and multiple parallel machine scheduling.

2.2. Optimization Models

Deterministic sequencing and scheduling publications began to appear as early as the 1950s, which included results by W.E. Smith, S.M. Johnson, and J.R. Jackson (Pinedo, 2008). A comprehensive survey published in 1979, "Optimization and Approximation in Deterministic Sequencing and Scheduling", listed over 100 publications related to single and parallel machine, open shop, flow shop, and job shop scheduling (Graham et al., 1979). Many other studies, including surveys and reviews, have been published in the field of deterministic scheduling. A recent comprehensive survey of scheduling problems

was published by Allahverdi (2015) listing hundreds of papers. However, due to the large amount of literature published in this field, it is impossible to discuss all the material in this review. Therefore, we must restrict our literature review to deterministic machine scheduling with single and multiple objectives. For further scheduling theory and algorithms not covered in this review, the reader is referred to sequencing and scheduling books published by (Baker, 1974; Brucker, 2007; Pinedo, 2016). Section 2.2 presents an overview of key publications related to exact mathematical models and it is further subdivided as follows: section 2.2.1 presents single machine scheduling formulations. Section 2.2.2 presents parallel machine scheduling formulations, and section 2.2.3 multi-objective optimization (MOO) concepts for machine scheduling.

It is noteworthy that publications addressing machine scheduling first appeared in the 1950s due to increasing complexity in manufacturing environments. Exact scheduling methods have been studied for many years in order to find an optimal schedule for job and machine assignment over time. These scheduling models are optimized with respect to one or multiple objectives. These scheduling models have been extended to include setup times, job release dates, and due dates which complicates the problem further. It is worth mentioning that these problems turn into very challenging combinatorial problems that become computationally intractable as the number of jobs and machines increase (i.e. 65+ jobs). Scheduling finite resources with setup times and/or costs play a very important role in manufacturing in order to deliver the right products at the right time and cost (Allahverdi, 2015).

2.2.1. Single Machine Scheduling

This subsection will focus on single machine scheduling formulations with the objective function of minimizing weighted completion time $\sum_j w_j C_j$ and weighted tardiness $\sum_j w_j T_j$ for a single machine with and without setup. Most of the research conducted on MILP scheduling models emanate from the following five key formulations (Nogueira et al., 2014)

1. Manne formulation- completion time modeled as continuous variable (Manne, 1960)
2. Potts formulation - job sequence modeled by binary linear ordering variables (Potts, 1980)
3. Wagner formulation - variables are a fixed number of slots per machine (Wagner, 1959)
4. Sousa and Wolsey formulation- discrete time periods for horizon (Sousa and Wolsey, 1992)
5. Pessoa et al. formulation- discrete time periods for horizon and precedence relationships (Pessoa et al., 2010)

Table 2-4 summarizes previous publications related to the five formulations previously mentioned. Refer to (Nogueira et al., 2014) for a full list of the references summarized below.

Table 2-4: MIP Formulations Single Machine Scheduling (Nogueira et al, 2014)

MIP Formulations	**Problem Parameters**	$\sum_j w_j C_j$	$\sum_j w_j T_j$
Manne	no parameters	Keha et al. (2009),	Keha et al. (2009),
		Queyranne and Wang (1991)	Khowala et al. (2005)
	r_j and no s_{ij}	Keha et al. (2009)	Keha et al. (2009)
	with s_{ij}	Queyranne (1993)	Queyranne (1993)
Potts	no parameters	Blazewicz et al. (1991),	Blazewicz et al. (1991),
		Chudak and Hochbaum (1999),	Keha et al. (2009),
		Keha et al. (2009)	Khowala et al. (2005),
	r_j and no s_{ij}	Dyer and Wosley (1990),	Keha et al. (2009)
		Keha et al. (2009),	
		Queyranne et al. (1994),	
		Unlu and Mason (2010)	
	with s_{ij}		Tanaka and Araki (2013)
Wagner	no parameters	Keha et al. (2009),	Keha et al. (2009),
		Khowala et al. (2005),	
		Lasserre and Queyranne (1992),	
		Queyranne et al. (1994)	
	r_j and no s_{ij}	Keha et al. (2009)	Keha et al. (2009)
Sousa and Wolsey	no parameters	Keha et al. (2009)	Bigras et al. (2008a),(2008b)
		Khowala et al. (2005)	Keha et al. (2009),
			Razaq et al. (1990),
			Sadykov (2006),
			Sadykov and Vanderbeck (2011),
			Sourd (2009a),
			Sousa and Wolsey (1992),
			Tanaka et al. (2009)
	r_j and no s_{ij}	Avella et al. (2005)	Keha et al. (2009),
		Keha et al. (2009),	Queyranne et al. (1994)
		Queyranne et al. (1994)	
Pessoa et al.	no parameters		Pessoa et al. (2010)

Some of the key conclusions from the study conducted on the five formulations previously described include that the number of jobs and the length of the time horizon significantly impact the MILP model performance. The most used formulations are those created by Manne, Sousa and Wolsey. The authors stated that "All the MIP formulations developed in this article present a polynomial number of constraints and variables...it is worth noting that as the horizon (h) >>> *n* jobs, h ∝ n, "Pessoa et al.", "Sousa and Wolsey" and "Sousa and Wolsey Improvement" formulations will grow faster than other formulations (Nogueira et al., 2014). Even though time-indexed MIP formulation has a negative reputation in some of the open literature, it is the focus on this study given preliminary experimentation results obtained in a project in which six single machine formulations were compared including discrete and continuous decision variables as well as ordering techniques. The results showed that time-indexed formulation for single machine with due-dates outperformed five other single machine formulations for problem instances of up to 65 jobs.

2.2.2. Parallel Machine Scheduling

This subsection deals with *m* parallel machines and requires that all the jobs are assigned to a single operation similarly to what was described last section. This study focuses on identical parallel machine schedules (PMS) assuming all the machines may have the same speed for a given job. A sequence-dependent setup time may be incurred for individual jobs or a family of jobs as proposed by Allahverdi (2015). This section presents publication with and without setup times as well as publications with job-machine eligibility restrictions.

Cheng and Sin (1990) presented a state-of-the-art review of PMS of more than 80 papers with publications ranging from 1959-1990. The authors stated that PMS problems have been a subject of extensive study fueled by the need to schedule incoming jobs to parallel processors in a computer system. Likewise, manufacturing environments such as machine shops encountered the problem to schedule job orders on groups of identical production facilities. Previous survey papers on PMS were presented by Graham et al., (1979). Most of the papers reviewed by Cheng and Sin (1990) focused on job and machines scheduling and excluded auxiliary resources. However, auxiliary resources are present in multiple real-life manufacturing environments such as plastic injection molding, semiconductor and electronics industries.

For the case of identical parallel machines Edis (2009) cited the papers by Blazewicz et al. (1983) and Ventura and Kim (2000) proposing polynomial time exact algorithms for identical machines with resources. Edis (2009) also cited papers related to uniform machines (Kovalyov and Shafransky, 1998; Ruiz-Torres A.J. et al., 2007) and unrelated machines (Grigoriev et al., 2005). Multiple scholars have also conducted research on solving the sequence-dependent setup time or setup cost for non-family problems with the objective function of minimizing maximum completion time (C*max*) or tardiness (T) as depicted in Table 2-5.

Table 2-5: Non-family Scheduling with Setup

Authors	**Criterion**	**Method Applied**
Behnamian et al. (2009)	$C_{max,} \sum E_j + T_j$	Hybrid algorithm consisting of simulated annealing, Ant colony opt and variable neighborhood search.
Fan and Tang (2006)	$\sum w_j C_j$	Integer programming model, a column generation algorithm for parallel machines
Hou and Guo (2013)	C_{max}	MIP with Genetic Algorithms. Multiple resource constraints
Hu and Yao (2011)	C_{max}	MIP model, lower bound, GA
Rocha et al. (2008)	$C_{max}, \sum w_j T_j$	MIP, B&B algorithm.
Toksarı and Güner (2010)	$\sum w_{j1} E_j + w_{j2} T_j$	MIP

According to Behnamian et al. (2009) most of the publications used a MILP model for small size problems and a heuristic for larger size problems. It can be observed that MIP and genetic algorithms were applied to the formulations with completion time being at least one of the objective functions. Table 2-6 presents a summary of publications related to exact methods mixed with metaheuristics in order to schedule a family jobs (non-family) with sequence-independent setup, sequence-dependent setup and without setup times.

Table 2-6: Family Scheduling Summary with and without Setup

Authors	**Criterion**	**Method Applied**
Bettayeb et al. (2008)	$\sum w_j C_j$	Lower bounds, Branch and bound and constructive heuristic| Sequence Independent.
Schaller (2014)	$\sum T_j$	TS and GA | Sequence Independent.
Tavakkoli-Moghaddam and Mehdizadeh (2007)	$\sum w_j F_j$	Integer Linear programming, GA| Sequence Independent.
Bozorgirad and Logendran (2012)	$\sum w_j C_j$, $\sum w_j T_j$	MILP, meta-heuristic with TS| Sequence dependent.
Chung et al. (2009)	$Total\ Profit$	Two new algorithms| Sequence dependent.
Loveland et al. (2007)	TSC (Total setup cost)	Algorithm combining optimization and heuristic component
Monkman et al. (2008)	TSC (Total setup cost)	GRASP
Park et al. (2012)	$\sum T_j$	Heuristic Algorithm

Table 2-6 presents a summary of publications related to exact methods mixed with metaheuristics in order to schedule a family of jobs with and without setups. It is worth noting that summary above excludes other publications since they were not related to the topic of this dissertation. A brief summary provided by Allahverdi (2015) for family sequence-independent setup times states that Schaller (2014) found that genetic algorithms (GA) outperformed Tabu Search (TS) algorithms for the problem of parallel machines with objective function to minimize tardiness. Tavakkoli-Moghaddam and Mehdizadeh (2007) address the weighted flowtime problem for parallel machines using an integer linear programming model and they also proposed a Genetic Algorithm approach for solving large size problems. Bettayeb et al. (2008) addressed the weighted

completion time for parallel machines problem using a constructive heuristic and provided three lower bounds. A branch-and-bound algorithm was proposed, it incorporates the lower bounds showing that the algorithm is effective. As per Bettayeb et al. (2008) resource scheduling without setups has been intensively studied and Garey and Johnson (1978) proved that weighted completion time problem is strongly NP-hard. Allowing preemption reduces the complexity to polynomial time as proved by McNaughton (1959). For the family sequence-dependent setup time problem, Bozorgirad and Logendran (2012) addressed the minimization of weighted completion time and tardiness utilizing a MILP and a meta-heuristic with Tabu Search showing that a meta-heuristic works well by comparing the performance with the optimal solution for small size problem instance. Chung et al. (2009) used the objective function of maximizing total profit and applied two new algorithms that showed they are efficient.

Most of the publications reviewed in the previous sections report that more than one objective function was utilized to solve the model. However, not many researchers used multi-objective optimization (MOO) where *k* objective functions were optimized simultaneously using an iterative approach similar to method described in the next section.

2.2.3. Multi-objective Optimization (MOO)

Multi-objective optimization (MOO), also known as multi-criteria optimization, is concerned with decision making based on multiple criteria. Typically, mathematical optimization models involving more than one objective function are used in order to optimize *k* objective functions simultaneously. According to Metta (2008) these functions

form a mathematical description of performance criteria which are usually in conflict with each other. A key difference between MOO and single-objective problem is that MOO does not have a single optimal (best) solution. The Pareto method which is one way to implement MOO finds a feasible schedule that minimizes several objectives in such a way that no improvement can be made on one objective without degrading the other objective metric in the objective vector (Suresh and Mohanasundaram, 2006).

According to Suresh and Mohanasundaram (2006) the user may generate a schedule with a weighted combination of several scheduling objectives as the performance measure. MOO models allow experts to choose a Pareto optimal solution according to the existing priorities when a decision needs to be made. In this case, a Pareto optimal set is to be found as a family of best trade-off schedules. The set of Pareto solutions is called the Pareto frontier (Suresh and Mohanasundaram, 2006). Chiandussi et al. (2012) provide a wide variety of algorithms that can be successfully applied for multiple engineering projects and the following definition about MOO.

The general multi-objective optimization problem has an objective function set $F(x) = f_1(x) \dots f_k(x)$ composed of multiple single optimization objective functions that are either minimized or maximized subject to the following inequality constraints

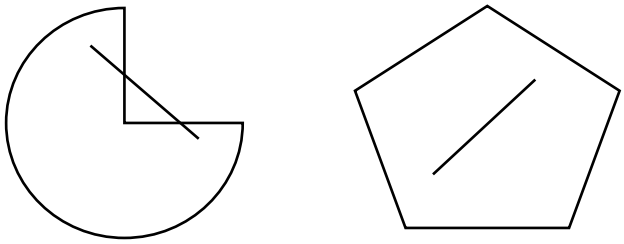
$$g_i(x) \leq 0 \; for \; i = 1 \dots m$$

$$h_j(x) = 0 \; j = 1 \dots p$$

Therefore, we have *k* objectives reflected in the *k* objective functions, m + p constraints on the objective functions and *n* decision variables (Chiandussi et al., 2012). Convexity is described as a function $f(x)$ over the domain of R if and only if

$f(\alpha x_1 + (1-\alpha)x_2) \leq \alpha f(x_1) + (1-\alpha) f(x_2)$ for the vectors $x_1 \; and \; x2 \; \in R$

Where α is a scalar in the range $0 \leq \alpha \leq 1$. In other words, the values obtained by the linear interpolation is greater than or equal to any value obtained the function above. See Figure 2-2 for an illustration of a convex and non-convex space.



a) Non-convex b) Convex

Figure 2-2: Convex and Non-Convex Space

Four MOO techniques were analyzed by (Chiandussi et al., 2012) including formulation, advantages and disadvantages.

The first technique is "Linear Combination of Weights" which is a scalar optimization method in which the objective function is the product sum of the objective function components and the weight per objective function. Two advantages of this method are the simplicity to implement and the computational efficiency. One main disadvantage is that this method is unable to generate certain portions of the Pareto front if the shape is concave (Chiandussi et al., 2012).

The second technique analyzed was the MOGA method which can be single and multi-objective optimization based on genetic algorithms know to be part of evolutionary algorithms. Some advantages of this method are the fact that supports general constraints with a mixture of real and discrete variables. It also allows users to find several members

of the Pareto optimal set in a single 'run' vs separate runs as it occurs with exact methods such as traditional mathematical programming that has one major disadvantage (computationally expensive) (Chiandussi et al., 2012).

The third technique discussed was the Global criterion method which requires the user to pre-select a goal before searching for optimality. The algorithm aims to minimize the distance between decision maker vector and ideal vector. Some advantages: Since these methods do not require a Pareto ranking, they are simple and efficient. One disadvantages is that the user must select a goal which may require extra effort to compute and results may be limited by the selected goal (Chiandussi et al., 2012).

The fourth technique is the ε-constraint method which is consider one of the best options for MOO since the model solves for one objective function at a time while transforming the other objectives into constraints. One key advantage is these methods are relatively simple to understand as only one of the original objectives is minimized while the others are transformed to constraints. Two disadvantages of this method are the potential high computationally cost and objective function encoding for certain problems can be extremely difficult (Chiandussi et al., 2012).

These methods can be grouped under one of the following categories depicted in Figure 2-3: a) A priori preference which assumes the expert has prior information allowing a decision first, then perform a search; b) A posteriori Preference performs a search before making any decisions and do not require prior preference information from the decision maker; c) Progressive Preference which is a back-and-forth combination of search and decision making; and at a high level, it finds a non-dominated solution, then

gets input from the decision maker with respect to non-dominated solution found, and modify the preferences of the objectives accordingly. Finally, repeat previous steps until user signals to end search. There are many other algorithms available to solve MOO, a taxonomy is presented in Figure 2-3.

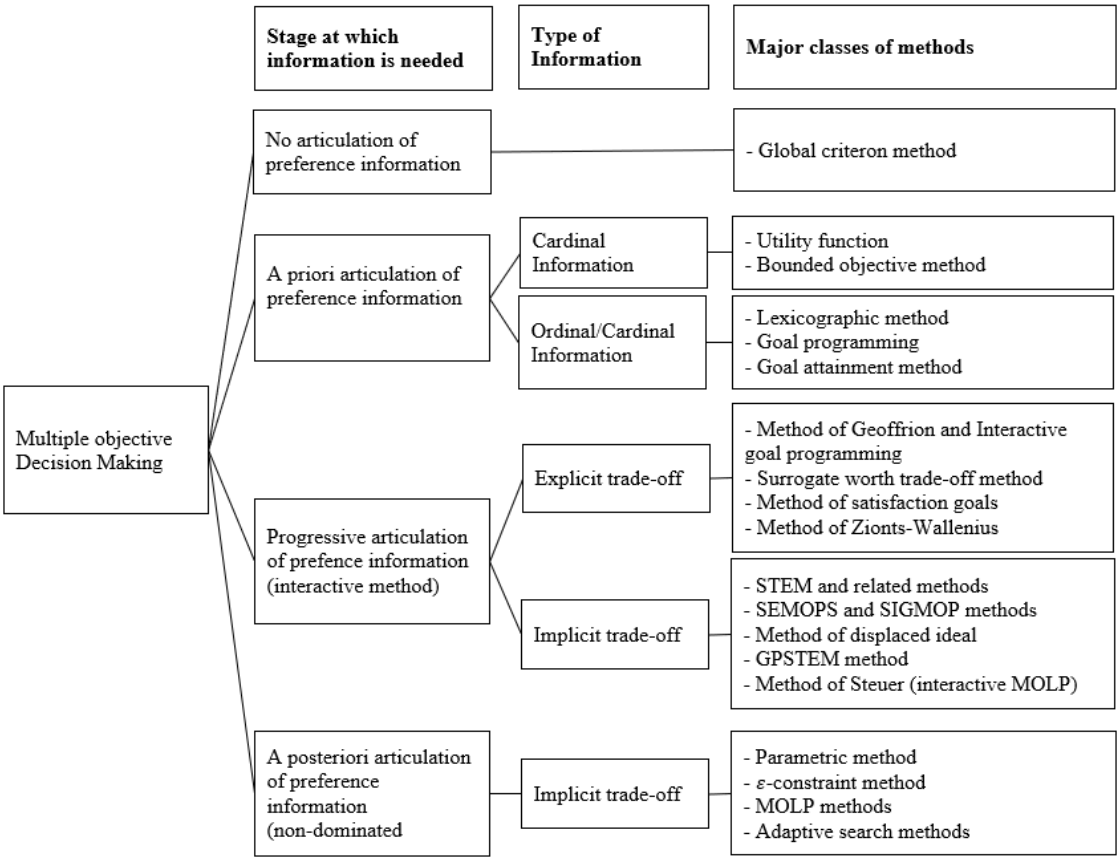


Figure 2-3: MOO Taxonomy Summary (Chiandussi et al, 2012)

As per Behnamian et al. (2010) multi-objective models always lead to a choice of a solution among all efficient solutions known as a posteriori approach. Hence, a set of best trade-off schedules called the Pareto optimal must be generated. A Diversity Maximization Approach (DMA) for MOO was proposed by Masin and Bukchin (2008) to generate the efficient frontier for MILP and combinatorial problems which is aligned

with our research interest and goals. The advantage of using the DMA for multi-objective optimization is that it has been successfully applied to MILP formulations for machine scheduling and a well-diversified partial efficient frontier is guaranteed to be obtained.

The following MOO formulations and definitions are excerpts from Masin and Bukchin, (2008) publication with minor formulation changes since Xue and Villalobos, (2012), identified a typo in the M-DMA algorithm steps.

For the MOO DMA preliminaries, Masin and Bukchin, (2008) assumed the minimization of multiple objective functions presented by the set *K* and each objective function has its own weight defined as w_k:

$$\min Z = \sum_{k \in K} w_k f^{(k)}(x)$$

s.t $x \in X \; where \; X \; is \; the \; set \; of \; feasible \; solutions$

Given $y = f(x) \in Y$ the solution obtained from the k^{th} objective function is described as $y^{(k)} = f^{(k)}(x)$. Let us define $y_1 = y_2$ if $y_1^{(k)} = y_2^{(k)}$ and $y_1 \leq y_2$ if $y_1^{(k)} \leq y_2^{(k)}$ for all *k*. Two solutions $x_1 and \; x_2$ are equivalent if $y_1 = f(x_1) = \; y_2 = f(x_2)$ and one solution $x_1 and \; x_2$ dominate another one if $y_1 = f(x_1) \leq \; y_2 = f(x_2)$ in this case we say y_2 is dominated by y_1. Y_{eff} represents the complete efficient frontier while Y_{par} represents the partial set of Pareto optimal solutions. $y = f(x)$ is called efficient if *x* is Pareto optimal. The, the diversity measure $\alpha_E(y)$ is defined as follows:

$$\alpha_E(y) = max_{y_e \in E} \left(min_{1 \leq k \leq W} \frac{y^{(k)} - y_e^{(k)}}{\Delta_{ke}} \right)$$

Where Δ_{ek} is a positive scaling coefficient and *E* is defined as the partial efficient frontier $E \in Y_{eff}$. It is important to accurately estimate the scaling factor Δ_{ke} in order to obtain an accurate representation of the efficient frontier for each iteration. (Masin and Bukchin, 2008) proposed $\Delta_{ek} = R_k = max_{y \in Y_{eff}} y^{(k)} - min_{y \in Y_{eff}} y^{(k)} \; \forall \; y_e \in E, k = 1, = 1 \ldots W$.

$$\Delta_{ek} = \begin{cases} R_k, if \; R_k > 0; \\ 1, otherwise. \end{cases}$$

Masin and Bukchin (2008) define [P1**]** which finds the initial point $Y_e \in E$**:**

$$\textbf{[P1]} \;\; \min Z = \sum_{k \in K} w_k f^{(k)}(x) \;\; \text{s.t } x \in X$$

where $W_k \; k = 1 \ldots W$are positive weights for k^{th} objective function. The subset *E* to be used to build the efficient frontier starts empty, then one point per iteration is added until a full efficient frontier is obtained or a pre-defined ε is provided.

$$\textbf{[P2]} \;\; \min Z = lex \min\left(\alpha, \sum_{k \in K} w_k f^{(k)}(x)\right)$$

$$\text{s.t} \min \alpha = max_{y_e \in E} \left(min_{1 \le k \le W} \frac{f^{(k)}(x) - y_e^{(k)}}{\Delta_{ke}} \right) \; x \in X.$$

Then the non-linear constraint α used as constraint for **P2** is linearized using binary variables.

$$[\textbf{P3}] \min Z = lex \min\left(\alpha, \sum_{k \in K} w_k f^{(k)}(x)\right)$$

$$\text{s.t. } \alpha \ge \beta_e \forall y_e \in E,$$

$$\beta_e \le \frac{f^{(k)}(x) - y_e^{(k)}}{\Delta_{ek}} \forall \; y_e \in E, k = 1 \ldots W,,$$

$$\beta_e = \sum_{k=1}^{W} \frac{\gamma_{2ke} - y_e^{(k)} \gamma_{1ke}}{\Delta_{ek}} \forall \; y_e \in E,$$

$$\sum_{k=1}^{W} \gamma_{1ke} = 1 \; \forall e \in E$$

$$\gamma_{2ke} \geq f^{(k)}(x) + (\gamma_{1ke} - 1)M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \leq f^{(k)}(x) + (1 - \gamma_{1ke})M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \leq \gamma_{1ke} M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{1ke} \in \{0,1\} \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \geq 0 \; \forall y_e \in E, k = 1, \dots, W$$

$$x \in X.$$

Where M is a large number. The A-DMA iterative algorithm is described next.

Table 2-7: A-DMA Algorithm (Masin and Bukchin (2008))

Step	Action
1	Solve [P1] Let $y^* = f(x^*)$ be optimal value Set $E=\{y^*\}$ Select $\varepsilon \geq 0 \; and/or \; sample \; size$
2	Solve [P3] Let $y^* = f(x^*)$ be optimal value

3	IF $\alpha(y^*) < -\varepsilon \; or \; Npoints \leq Threshold$ THEN
	$E = E \cup y^*$
	Go to step 2**
	ELSE
	Stop;
	END IF
	**Original paper says step 1 (Typo), should be step 2.

As the algorithm iterates a single efficient point (y^*)is added to E when the absolute value of $\alpha(y^*)$ is less than or equal to its value in the previous iteration. The user has three options to stop the algorithm early:

i. Set $\varepsilon = 0$ to find the whole efficient frontier or selected number of efficient points

ii. Set $\varepsilon = 0$ to find the whole efficient frontier or after predetermined time

iii. Set $\varepsilon \geq 0$ to desired resolution of the efficient frontier.

Selecting the right value for ε may not be an easy task in practice; hence, one can set a desired number of solutions when implementing the algorithm which is what we propose in this study.

DMA MOO was also applied to solve the multi-objective optimization problem for primary planning of the "Inspection Effort Allocation (IEA) problem for a point-of-entry Inspection (Xue and Villalobos, 2012). It is worth mentioning that a diversity maximization algorithm was also developed to enable the decision maker to select the proper solution easily. If the decision maker decided to override the proposed solutions, the model would run and immediately find a new optimal solution by adding the user-input as a new constraint that must be satisfied.

2.3. Practical Production and Manufacturing Scheduling

This subsection provides an overview of publications in manufacturing scheduling. focusing in the framework of the systems, the multiple components required for a successful practical scheduling system and user-model interaction. “While the literature on manufacturing scheduling models and solution procedures is extensive, very little has been written on how to bring these models and procedures into practice. This has given rise to the so-called ‘‘gap” between the theory and practice of scheduling“ (Framinan and Ruiz, 2010). Our goal with this subsection is to shed light on expert systems and practical production scheduling systems aiming to close the gap between theory and practice. This subsection combines ideas and frameworks presented since 1980’s until today and

summarizes the key concepts focusing on the overall structure and problem models. It is worthwhile to mention that most scheduling papers tend to focus on techniques and/or methodologies rather than the overall system and the structure of these tools (Dios and Framinan, 2016). That is, problem modeling and problem solving are at the forefront of the research.

According to Romero-Silva et al. (2015) there is not an established notion of the term "practical scheduling"; however, several authors (Hoitomt et al., 1993; Olsen, 1999) have used this term for studies related to scheduling. Even though the proposed technique could be applicable to real-world practical problems the authors did not focus on the real-world production scheduling topic. For this study, the term practical production and/or manufacturing scheduling systems will be used to refer to real-world scheduling problems and systems.

Framinan and Ruiz (2010) summarizes key components of a scheduling system architecture based on the type of functionality: scope of the system, problem modeling, problem solving, problem evaluation, reactive scheduling, capacity analysis, user interface and integration of existing business information systems. Even though this is not the only framework in the open literature, it provides a significant list of references related to this line of work for practitioners to successfully design, plan and develop a practical scheduling system.

Similarly, the proposed framework from (Romero-Silva et al., 2015) summarizes the topics into: manufacturing environment, production schedule, scheduling goals, incoming jobs, schedule construction and knowledge update, representation of

manufacturing environment, manufacturing environment status perception and schedule construction update activity. Some of the key findings for succesful designing of real-world scheduling systems is to consider all the elements that are relevant to environment in question. Another key finding is that most practical problems consider more than one goal and many papers in the academic literature review focus on single objective. It is also noteworthy that scheduling parameter inputs provided by content expert vary from person to person; hence, it is important to obtain information from multiple experts and multiple sources in order to design robust system. A practical scheduling system should be flexible, capable of reacting to changes in the environment and able to repair or reschedule resources in the shortest amount of time.

2.3.1. Incorporate Human Expertise

As previously mentioned in Chapter one, Ahn (2008) stated "human computation problems" are those large-scale computational problems that often cannot be solved by either computer or humans alone. The goal is to harnesses human brainpower to solve complex problems that computers may not be able to solve in relatively short periods of time. That is, computers should enhance human intelligence instead of replacing it. Thus, this subsection focuses on the topic related to incorporating human expert knowledge in the scheduling system. (Framinan and Ruiz, 2010) reported that there is controversy in the open literature whether incorporating expert knowledge into the solution is a good or bad idea. According to Steffen (1986) and Fox (1990) some of the practical real-world scheduling problems have a complexity that goes beyond the cognitive capability of the human brain. Others claim that if human experts exist the scheduling system

incorporating their knowledge would only automate good or bad decisions (Kanet and Adelsberger, 1987). On the other hand, other authors state there are potential benefits to the overall scheduling system if human experts are able to interact with the model (Reinschmidt et al., 1990) and (Framinan and Ruiz, 2010). For instance, if the scheduling tool is programmed in a way that is open and flexible so that expert knowledge can be captured and translated into objectives and constraints it would be a potential benefit. According to (Framinan and Ruiz, 2010) it is preferable to support the decision maker with enhanced systems rather than replacing them with a scheduling tool that may not comprehend and represent practical problems.

User-model interactive systems also open up the opportunity for experts to better understand the problem they are trying to solve and the scheduling tool can also serve as a training platform for less experienced schedulers. (Stevenson et al., 2009) claim that their workload control tool and production planning concept with user-model interaction allows users to be trained. Their decision support system provides an action-learning package for end-users and improves decision-making experience related to parameter settings (i.e. due dates), acceptance or rejection of jobs and scheduling intervention among others. Similarly, Framinan and Ruiz (2010) suggest that it would be interesting to make the suggestions of the system more transparent to the scheduler so that he or she can learn from the system and gain insight to the decisions proposed.

2.3.2. User-model Interaction in Artificial Intelligence

According to Chai et al. (2009) many applications in the field of computer science are starting to employ information extraction (IE) and integration (II) programs to infer

knowledge from unstructured data. Chai et al. (2009) propose a solution for users to provide feedback and for IE/II programs to automatically process feedback. The proposed model was capable of incorporating recent feedback with historical user feedback using "hlog", a declarative IE/II language. These applications can be related to simple website feedback applications to improve customer satisfaction.

A similar approach is utilized in crowdsourcing applications which rely on human computation to extract knowledge from text, sound, video and images. Since it is relatively easy for humans to understand text, sound and video but hard for algorithms (Parameswaran et al., 2012), humans are often tasked with text feature extraction. However, it is not easy for systems to interact with users in order to obtain feedback. Thus, a declarative query model is proposed. "The goal is to design a query processor and optimizer that automatically decomposes the query into small unit tasks that are answerable by humans. These unit tasks could be as simple as comparing two items, rating an item or answering a Boolean question." (Parameswaran et al., 2012). Support Vector Machines (SVM) are popularly used in text classification; however, these models rely on large amounts of data properly labeled in order to train the model. If labeled data does not exist companies like Yahoo pay humans to label the data (Raghavan and Allan, 2007). On the other hand, there applications where the user is willing to label data which will result in increased user satisfaction. For example, personalized news or email filtering an algorithm can be used to ask the user to label as little as possible to decrease the tediousness of the process and improve customer satisfaction (Raghavan and Allan, 2007).

Xue and Villalobos (2012) incorporated user input into the flexible inspection system for a port of entry. One of their objectives was to present a set of solutions to choose from and evaluate some plans enacted by the inspector.

In summary, it is evident that researchers–mainly in the computer science field–are relying more on user feedback/interaction to extract knowledge or preferences that can be incorporated into their models or applications. However, the concept of user-model interaction does not appear to be widely studied in the scheduling field. Thus, our proposed research should contribute to bridge this gap.

2.4. Conclusion and Literature Contribution

This review focused on three basic lines of research most relevant to the problem of interest this study: 1) scheduling techniques in semiconductor industry, 2) Optimization models for single and parallel machine scheduling including multi-objective optimization (MOO), and 3) practical manufacturing scheduling systems including user-model interaction and system architecture for highly modular systems. Overall, we observed that machine scheduling has received a significant amount of attention over the last few decades with over 1,000 research papers published with three comprehensive reviews. According to Allahverdi (2015) scheduling problems with setup times/costs are growing. However, considering most scheduling environments involve setup operations, only 10% of the research in scheduling includes setup times/costs. Hence, more research on scheduling problems with explicit consideration of setup times/costs is needed. He also proposes more research that considers family setup time for the parallel and job shop environments. The multiple criteria scheduling problems constitute less than 10% of the

papers, while in real life many scheduling problems require multiple criteria. Therefore, there exists a need to address more scheduling problems with multiple criteria. Finally, he concluded that there are about 500 papers, which he summarized and concluded that most of them use heuristic methods such as Genetic algorithms while only 50 papers used MIP and 30 used branch and bound (exact methods). Our proposed models and approach are well aligned with many of the recommendations proposed by (Allahverdi, 2015) from his third comprehensive scheduling review, we highlight some salient point and state the advance made in this dissertation

- "The research on scheduling problems with setup times/costs is still less than 10 percent of the available research on scheduling problems while most scheduling environments involve setup operations. Hence, more research on scheduling problems with explicit consideration of setup times/costs is needed" (Allahverdi, 2015). Our proposed phase II model includes sequence-dependent setup times and addresses this recommendation.
- "...for the single machine environment with family setup time case, about 75 percent of the papers addressed the sequence-independent problem. This indicates the need for addressing the sequence-dependent scheduling problems in single machine" (Allahverdi, 2015). Our formulation addresses the multiple identical parallel machine with sequence-dependent scheduling problem; hence, setting $m=1$ aligns with the recommendation previously stated.
- "...in the current competitive work environment, firms strive to save in every possible way, and minimizing work-in process is a major cost saving. It has been

observed that the total completion time performance measure has been addressed by a relatively smaller number of papers for the single machine, parallel machine, and job shop environments. It has been also observed that total tardiness performance measure has been only utilized in a few papers for the job shop environments. Therefore, there is a need to consider these performance measures in those scheduling environments" (Allahverdi, 2015). Our phase III objective function addresses this recommendation since we are minimizing total weighted completion time and total tardiness.

- For the Diversity Maximization Approach (DMA) for MOO presented by Masin and Bukchin, (2008) with MILP scheduling problem, our model allows user-model interaction for the parallel machine scheduling with shared auxiliary resource which addresses Masin and Bukchin's, (2008) recommendation to explore an interactive DMA with the decision maker resulting in a more effective method since the user could focus on the relevant parts of the efficient frontier. Our proposed research is going to address this recommendation since we plan to enable model-user interaction for the DMA MOO model in phase III.
- Ham (2018) suggested to apply the time-indexed MILP model instead of "positional & assignment" variables used in Ham's study. Similarly, Avella et al. (2017) successfully applied MILP model in runway scheduling problems reversing the opinion that time-indexed MILP models are unattractive for real-time applications or large scale models. Our formulations address these two

recommendations by applying time-indexed IP and MILP models for the three phases.

- Edis (2009) claims that most of the research on parallel machine scheduling neglects machine eligibility restrictions. Our three phases apply job-machine eligibility restrictions to mimic real-world practical models.

Finally, our MOO model addresses the following recommendation: "The multiple criteria scheduling problems constitute less than 10 percent of the papers while in real life many scheduling problems require multiple criteria. Therefore, there exists a need to address more scheduling problems with multiple criteria" (Allahverdi, 2015).

CHAPTER 3 METHODOLOGY

This section presents the methodology used to carry out our research and is subdivided into seven sections. Section 3.1 presents traditional scheduling background, section 3.2 presents the envisioned framework of study, section 3.3 presents research objectives, section 3.4-3.6 present phase I, II and III of the research, respectively. Finally, Section 3.7 presents the solution approach we adopted in this dissertation.

3.1. Background into Semiconductor and Traditional Scheduling

The ultimate purpose of the research presented in this dissertation is to develop decision support tools through which semiconductor manufacturers can improve factory performance resulting in cycle time reduction, increased on-time delivery, and maximal throughput. There are multiple factors that need to be considered before devising a scheduling framework for semiconductor manufacturing.

Leachman (2013) states that the competition in the semiconductor industry has evolved due to multiple factors such as proprietary designs, pricing, quoted lead times, and on-time delivery commitments. Even though the semiconductor industry has improved on-time delivery to as high as 90% success rate, there are organizational barriers that make production planning and production scheduling a difficult undertaking. Higher pressure to meet customer requirements has increased the need to improve factory performance and the ability to generate efficient and effective production schedules that meet customer expectations. Semiconductor manufacturers are currently striving to optimize production around the globe; however, there are key challenges these

manufacturers face, such as the need to consistently improve cost and reduce cycle time to maximize revenue, and increased time spent planning and managing factory resources driven by higher product variation and volume (Applied Materials, 2012). Semiconductor manufacturers are realizing that productivity is key to improve the on-time delivery and increasing efficiency in the wafer fabrication lines is a must to remain competitive. Photolithography is one area that is highlighted by researchers and software vendors as a key opportunity for improvement due to the complexity of assigning jobs to machines, largely due to machines being partially restricted (i.e. not capable of running all jobs) and shared limited auxiliary resources.

It is important to understand how this study contributes to production planning and control hierarchy presented in Figure 3-1, which illustrates long term strategic decisions, short-term scheduling, and control issues in manufacturing. Hopp and Spearman (2011) state that long term decisions occur at the strategic level where the basic function is to establish a production environment capable of meeting the plant's goals. Forecasting and planning (workforce and capacity) are the major activities here.

The basic function of the tactical tools is to receive input from the long-range plans and turn them into a general plan of action in the form of WIP/quota settings. The scheduling module takes the master production schedule quotas and customer demands and translates them into a work schedule for the near term. The focus of this dissertation is to develop efficient mathematical formulations and heuristics that are able to sequence and schedule jobs in complex manufacturing systems.

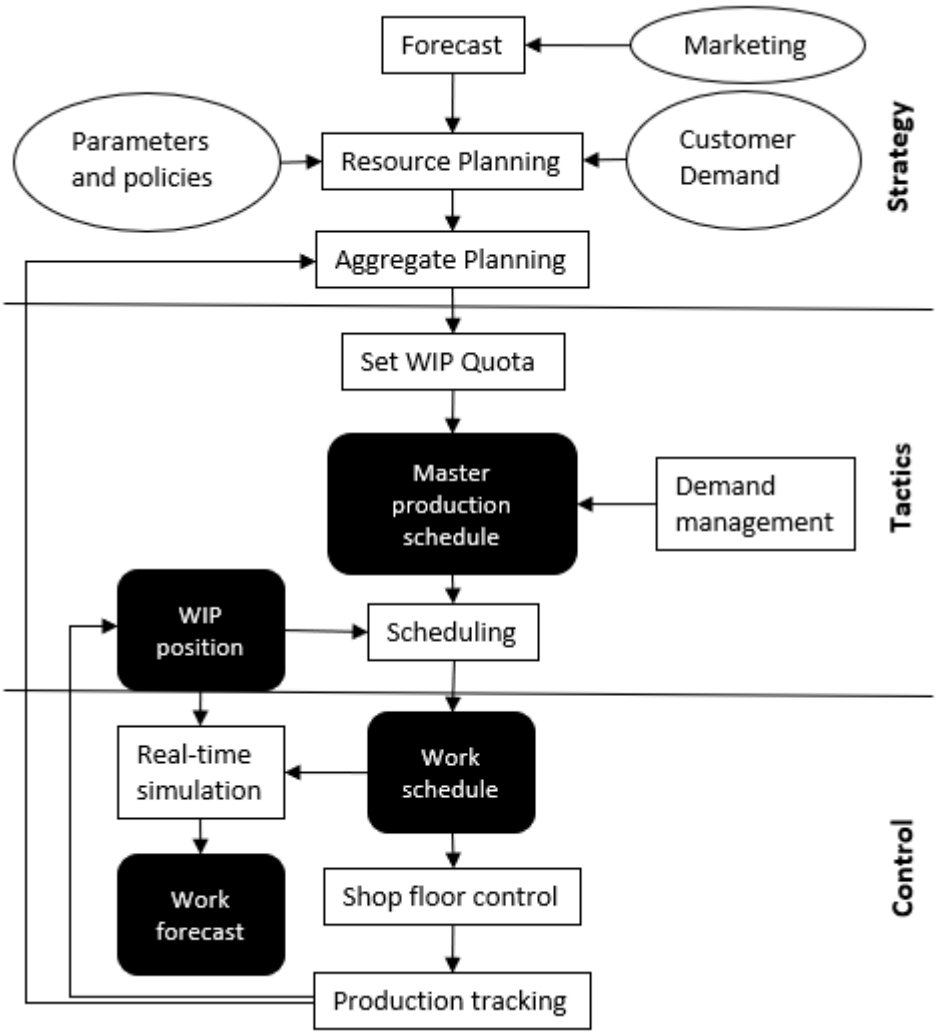


Figure 3-1: Production Planning and Control Hierarchy (Hopp and Spearman., 2011)

"Sequencing and scheduling are forms of decision-making that play crucial roles in manufacturing and service industries. In the current competitive environment effective sequencing and scheduling has become a necessity for survival in the market-place" (Pinedo, 2016). Not only does this study draws ideas and principles from academic scheduling theory, but also from the perspective of decision support software for the semiconductor industry. For instance, software vendor AMAT offers a predictive

scheduling solution that improves productivity of key bottleneck areas in the factory. According to a commercial software vendor (Applied Materials, 2012) the scheduling solution offers the following features: Co-optimize cycle time, throughput and yield; Incorporation of WIP management best-know-methods from industry; combination and integration of multiple data sources in real time and an open solutions users can customize objective functions and constraints by themselves among other features allowing users to conduct comprehensive experimentation and what-if analysis capability. Up to this point, a brief background into semiconductor and traditional scheduling has been provided in order to justify the need for more effective systems and algorithms. Now, we move to describe in greater detail the underlying hypotheses of this dissertation.

We hypothesize that the judicious combination of exact methods (optimization models) with heuristics can be used to solve practical scheduling problems. An effective combination of these methods can provide the basis for the development of an integrated decision support system that generates schedules that consistently outperform the dispatching rules currently used in the semiconductor industry. We further hypothesize that this combination of methods can render even better results in areas considered as the bottleneck of the production line, where shared auxiliary resources are required to process a job in a machine, such as photolithography. We also hypothesize that Integer Programming (IP) mathematical formulations can find solutions capable of outperforming any rule-based heuristic. We anticipate that a set of algorithms and

practices resulting from this dissertation will reduce manufacturing system inefficiencies by reducing setups and changeovers, therefore shortening cycle times.

3.2. Envisioned Framework of Study

In this dissertation, we seek to provide a manufacturing scheduling framework and develop mathematical optimization models, heuristics, and short-time scheduling plans for tactical and strategic scheduling planning purposes. A conceptual framework, depicted in Figure 3-3, is the result of bringing together several scheduling functions in the form of module in order to solve the problem at hand. The proposed framework is an integrated approach that enables development and testing of several independent modules as needed. A framework is key for a structured systems development and also serves as the foundation of the overall scheduling system.

We divide the study in three phases, Phase I and II aim to generate efficient schedules for small, medium and large problem instances under one hour and phase III aims to generate efficient solutions for multi-objective optimization (MOO) models for the offline analysis of the problem. This section presents the basic framework and the interaction among different modules that will result in a proof of concept scheduling system. We anticipate a few scenarios will arise in the real-world that our dissertation will address:

3.2.1. Possible Tactical Scenarios

We consider a complex manufacturing area in the semiconductor industry, such as photolithography, which is typically considered a factory constraint. We deal with the

case where we have incoming jobs and ready jobs. Often, incoming jobs are dynamic and may not arrive on the expected time. Each job may have a different processing time, depending on the operation, and may be restricted to run only on certain machines, further complicating the scheduling problem. Given the dynamic nature of the manufacturing environment being considered, it is extremely important to develop practical scheduling models capable of running in short periods (under 5 minutes) to be useful for real-time dispatching. The proposed study can also generate practical models capable of running under 15 minutes to be used as hybrid scheduling systems in which an existing real-time heuristic is combined with our proposed optimization models.

3.2.2. Near Optimal Allocation of Resources

The outcome of phase I models may result in tangible action plans or schedules in which a user can manually influence resource assignments to supplement existing rule-based heuristics.

Semiconductor factories experience high variability from different sources, as previously stated in chapter 1. Some of these sources of variability come from process variability, machine breakdown, customer demand fluctuations, and tighter on-time delivery requirements, among others. It is important to design a flexible framework and include multiple model parameters that will allow the decision maker to simulate the scenarios via optimization scheduling.

3.2.3. Possible Strategic Scenario

This scenario is likely to occur when the decision maker is not sure how to prioritize resource allocation given a factory being under-loaded or over-loaded, customer commits

performance (on-time delivery) and cycle time. For this possible scenario, we offer an interactive approach that allows the decision maker to assign weights in the objective function, set thresholds for a given objective, and optimize other objectives. A pareto efficient frontier is proposed for this strategic offline modeling. In this scenario, it is important that the model runs fast, but it is more important to provide insight to the decision maker, even if the model needs to run for an extended period of time to find a solution. It is important to mention that the outcome of this model is not meant to affect in real-time the schedules being used.

3.2.4. Overall Hierarchical Plan

It is extremely important to build a modeling framework that allows developers and decision makers to continuously learn and improve. The framework should also allow the decision makers and experts to customize objective functions and constraints and enable them to conduct comprehensive experimentation with what-if analysis capability. In order to develop a system capable of accommodating the previous scenarios, this dissertation uses a three-phase approach to solve the problems previously defined as $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$ and $P_m\left|s_{jk}, r_j, d_j, M_j, aux\ 1\right|\sum w_j C_j, \sum T_j, T_{max}$. An interactive scheduling system is envisioned and proposed. The envisioned system is motivated by the research carried out by Xue and Villalobos (2012) aiming to incorporate new information available from the decision maker to generate new solutions iteratively and by semiconductor software vendor AMAT (Marteney, 2011). The overall envisioned system architecture is presented in Figure 3-2.

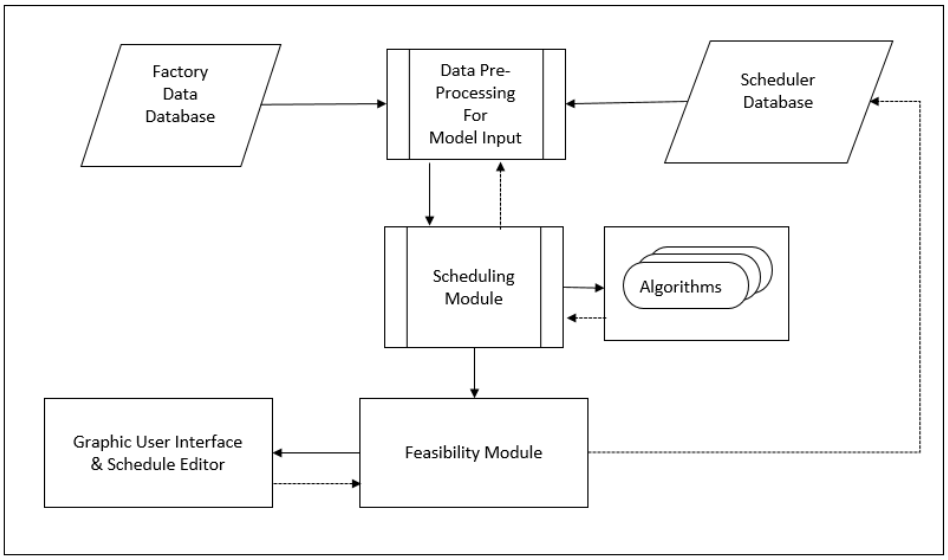


Figure 3-2: Flexible and Adaptive Manufacturing Scheduling System

A similar scheduling architecture is presented in Pinedo (2016), but has been modified to fit our proposal. The proposed system architecture envisions a real-time factory database interacting with a scheduling database through the data pre-processing module which takes all data inputs and generates input data files to feed the scheduling module. The scheduling module reads the optimization model(s) and the heuristic(s) from the algorithm module. This study also envisions a feasibility module with a graphical user interface to enable user interaction. It is worth noting that the feasibility module and graphic user interface will not be fully developed in this study. The development of these modules will be left as future research.

The focus of this dissertation lies in the scheduling and the algorithm modules and focuses in the development of IP models interacting with a heuristic in order to obtain a feasible solution. The objectives of this research are presented in the next section.

3.3. Objectives of the Research

In this dissertation, we aim to develop a modeling framework and a set of exact methods (optimization models) and heuristics that attempt to produce near-optimal scheduling solutions for highly complex and constrained areas. In order to achieve this, we target three main objectives:

1. Development of a practical scheduling framework that is compatible with semiconductor manufacturing systems. The goal of this framework is to enable algorithm design flexibility and user interaction capability. The architectural design must be highly modular in order to facilitate development, testing, and debugging.
2. Development mathematical models and heuristics capable of solving practical problems with instances ranging from small to large in order to provide the decision maker with near optimal job-machine-resource schedules for the manufacturing system outlined in section 1.3. The job-machine-resource schedules should be generated in 15 minutes or less, including time to generate input files. The aim of Phase I and II models is to generate schedules for real-time dispatching. We will also explore heuristic approaches to prime the model and reduce the time horizon
3. Develop a multi-objective optimization model with user interaction capability, seeking to assist management and decision makers with strategic decision making with respect to multiple conflicting objectives. The model should allow the decision maker set upper bounds for one or more objectives (i.e. maximum

tardiness < *t* units) and enable the decision maker to override existing optimal schedules as needed. In this research effort, a time-indexed IP model coupled combined with a MOO mathematical formulation is proposed in order to addresses each of the objectives described above. In order to validate our models, a set of randomized data that resembles real-world semiconductor lines needs to be created. The envisioned use of this model is an offline module capable of generating an entire or partial efficient frontier. The overall research flow diagram for each phase is presented in Figure 3-3.

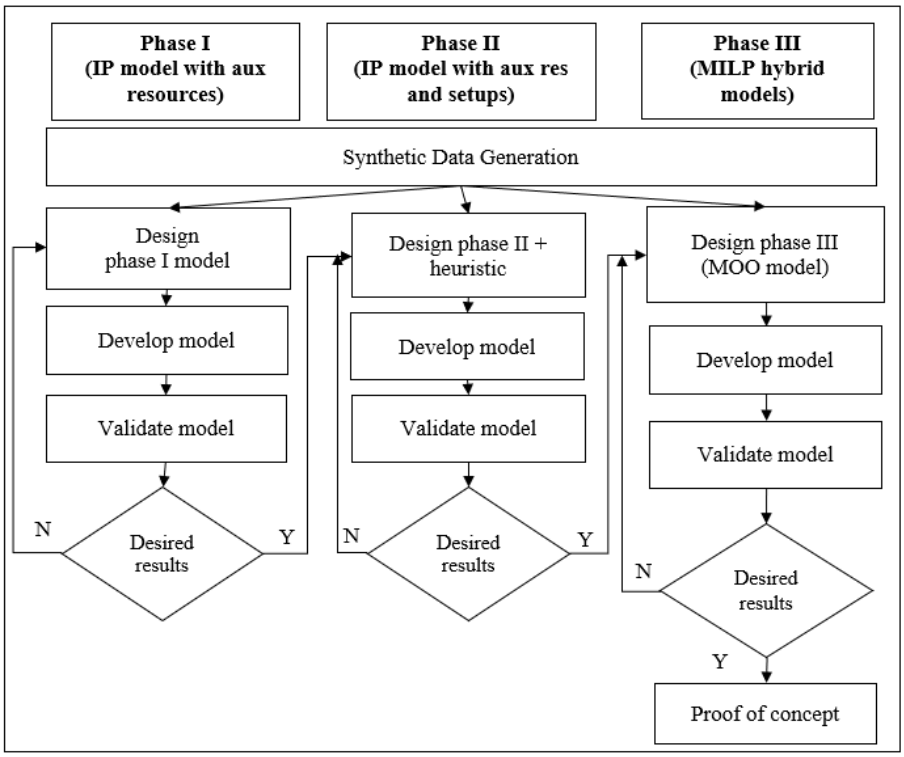


Figure 3-3: Research Phase Diagram

The plan is to formulate the mathematical model, program it in an optimization programing langue and validate the results as depicted in Figure 3-3. The model is deemed feasible if no job overlaps with each other in the same machine. Similarly, no job

should overlap on the same auxiliary resource as described in chapter 1. A Gantt chart is used to visually validate and inspect the solutions rendered by the model. Each phase is described in detail in the next section.

3.4. Phase I Optimization Model $P_m|r_j, M_j, aux\ 1|\sum C_j$

The first phase of the study consists of building optimization models to characterize the problem and to establish boundary conditions for the problem to be tractable when setups are not considered. In this case, we propose the development of optimization models that consider real-world variables in order to allow users represent real semiconductor problems. This study aims to incorporate as many model parameters as possible to enable the flexibility to schedule complex practical problems. Ready times, variable process times and job-machine restrictions are incorporated.

There will be a total of 45 problem instances to be explored as a result of the three factors considered: problem instance size, problem instance complexity, and machine eligibility restrictions (aka sharing density matrices) at five, three, and three levels, respectively. Problem instance size is a function of the number of jobs-machines-resources. Problem instance complexity is a function of job process times, ready times, and setup times. Sharing density matrices of 100% indicate that each job can run on every machine while a density matrix of 33% indicates that a job can run on 1/3 of the machine available. A table with the exact parameters used to generate each problem instance is presented in chapter 4.

3.5. Phase II Optimization Model and Heuristic $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$

Even though the outcome of phase I consists of the development of optimization models that ignored setups, the learnings in the model development will be invaluable. The objective of the second phase is to attempt to minimize total completion time by including setups which makes this approach attractive to solve real-world practical problems. The first phase is less complex as setups are not included. The model can still be used to determine rough cut capacity metrics and determine jobs to machine assignment. However, the model in phase I is not complex enough to create a real schedule that can be used in a wafer fabrication line. Hence, phase II focuses on creating efficient solutions for large models and optimal or near optimal solutions for small to medium problem instances. Phase II, similar to phase I, minimizes total completion time for the resource constrained parallel machine scheduling including release dates, job-machine restrictions and auxiliary resources including setup times $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$ for five problem instance sizes, three complexity levels, and three different levels of machine restrictions as depicted in Table 4-1.

A heuristic is also developed in phase II in order to interact with the optimization model in order to reduce the time horizon and provide an initial feasible solution.

3.6. Phase III MOO Optimization Model

$$P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|\sum w_j C_j, \sum T_j, T_{max}$$

In this phase, the key component of the problem formulation is the set of multiple competing objectives: (1) the minimization of weighted completion times, (2) the

minimization of total tardiness and (3) minimization of the maximum tardiness. To address this problem, we propose the use of a MOO IP in the form of a hypothetical case study with model-user interaction.

As previously stated in the literature review, this section provides a framework for practical scheduling system and user-model interaction. “While the literature on manufacturing scheduling models and solution procedures is extensive, very little has been written on how to bring these models and procedures into practice. This has given rise to the so-called ‘‘gap” between the theory and practice of scheduling“ (Framinan and Ruiz, 2010). Our goal with this section is to demonstrate how interactive optimization models can aid close the gap between theory and practice. We anticipate practitioners and decision makers will see the benefit of having multiple optimal alternative schedules and gaining insight in the model objectives, constraints and parameters used to generate it.

Phase III is a combination of the scheduling models developed in phase I and II with the addition of a lexicographic MOO model for resource constrained parallel machine scheduling including release dates, job-machine restrictions, auxiliary resources with and without setup times $P_m|s, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$. Setup times are included for the reduced small problem instance (i.e. 10 jobs, 2 machines, 4 resources) with no job-machine restrictions in order to keep run times tractable since the model must run *n* times to find the entire efficient frontier. For the medium size problem instance, setup times were removed for the reduced medium problem instance (i.e. 50 jobs, 5 machines, 12 resources with 100% machine eligibility). The decision to remove

setups was made to enable larger models to find *n* optimal solutions in the form of the entire efficient frontier.

3.7. Solution Approach

Linear Optimization, specifically Integer Programming (IP) is the exact method we selected to solve a highly complex problem previously introduced in chapter 1. These exact methods have been successfully applied across multiple industries with positive results. Practical problems can be modeled with a MILP or IP formulation in the form of:

$$\min\{cx : Ax \leq b, x \geq 0\}$$

Where A is an *m* by *n* matrix, *c* is an *n*-dimensional row vector, *b* is an *m*-dimensional column vector, and *x* an *n*-dimensional column vector of variables or unknowns. A binary problem is obtained if the variable *x* is restricted to be 0 or 1 and an integer program (IP) is obtained if the variable is restricted to non-negative integers (Wolsey, 1998). A know issue with mathematical IP models is that they require a significant amount of time to find an optimal solution as the problem instance size grows. Thus, exact heuristics need to be applied to find optimal solutions more efficiently. Branch and bound (B&B) is a widely known method also known as divide and conquer methodology. That is, a problem is divided into a series of smaller problems that are easier to solve. The smaller problems need to be put together in order to solve the original problem, the reader is referred to Wolsey (1998) for a detailed explanation on this method. The proposed problem will be solved with a commercial solver that applies a series of cuts and a branch-and-bound method to solve the IP/MILP models.

3.7.1. Phase I and Phase II IP Models

One of the main difficulties in solving the proposed models (in both phases) is dealing with problem instance size and computational complexity. This is due to the large number of decision variables and constraints involved in the formulations. There are multiple ways to reduce complexity by using a judicious combination the exact methods and heuristics. Transforming job's actual process times into a different time scale that allows the time horizon to be reduced is imperative to keep models tractable and render practical results. Clement et al. (2016) proposed a big bucket time indexed formulation for non-preemptive single machine scheduling problems. "The length of each period can be as large as the processing time of the shortest job. For larger minimum processing times the big bucket model can have significantly fewer variables and non-zeros than the time indexed model at the expense of a greater number of constraints" (Clement et al., 2016).

Another approach is to relax the setup restriction but respect resource allocation. This relaxation provides a lower boundary for all the objectives involved in this dissertation. That is, we are guaranteed to find better optimal solutions if setups are removed; however, those plans may not be feasible or easily implementable. These models without setup can be used to estimate the capacity of the area and the maximum throughput that can be achieved to establish goals.

For phase II, we plan to expand the model developed in phase I and add setups to create feasible schedules. The Phase II model is more computationally intensive. Our

approach to reduce complexity as previously mentioned, is to prime the model with an initial heuristic that is provided as initial solution. The hybrid model is anticipated to achieve better results since the search space is reduced.

3.7.2. Phase III MOO Model with User Interaction

The basic premise of this phase is to generate an efficient frontier using MOO models. This is a slightly different modeling approach from that used in phase I and II in which scheduling solutions had to run fast to accommodate changes in the environment and machines going up and down. This phase is more strategic in nature and the aim here is to allow the decision maker to interact with an offline model. A hypothetical case study is simulated that mimics real-world what-if scenarios As previously introduced in chapter 2, we plan to apply the MOO DMA approach presented by Masin and Bukchin, (2008). The generic MOO model assumes the minimization of multiple objective functions presented by the set *K* and each objective function has its own weight defined as w_k:

$$\min Z = \sum_{k \in K} w_k f^{(k)}(x)$$

s.t $x \in X \text{ where } X \text{ is the set of feasible solutions}$

The three objective functions to be minimized in chapter 6 are the following: weighted completion time, total tardiness and max tardiness. Refer to section 2.2.3 for a detailed explanation on the diversity maximization approach (DMA) we plan to use in chapter 6.

Prior to moving the chapter 4, we present a short list of resources required to complete this dissertation. Since this dissertation is a proof of concept of a scheduling

system, the key requirements are only related to software, hardware, and data generation. The software requirements are IBM Cplex Commercial Solver (full license installed and available) and Python 3.5+ to build data sets and heuristic. The hardware requirements are a Laptop with 16GB of RAM for model development and virtual server with Intel ® Xeon® CPU E5-2650 0 @ 2.00 GHz (2 processors) with 64GB RAM, with Windows server 2016 operating system.

CHAPTER 4 PHASE I RESOURCE CONSTRAINED PARALLEL MACHINE SCHEDULING MODEL

To formulate a scheduling system framework compatible with semiconductor manufacturing systems, the phase I model was developed. The phase I model consists to minimize the total completion time for the parallel machine scheduling problem with job-machine restrictions and auxiliary resources without setup $P_m|r_j, M_j, aux\ 1|\sum C_j$ for five problem sizes, three complexity levels, and three machine eligibility restriction levels, as presented in chapter 3.

A time-indexed IP model is proposed to address the problem with *m* parallel machines and *j* jobs, ignoring setup time. The objectives of phase I are to verify and validate our mathematical model is efficient, and additionally, establish lower bounds for computational time and optimality gap when sequence-dependent setups are added in phase II. Phase I will effectively set the stage for phase II. A key contribution of the phase I model is the combination of medium and large model instances with identical parallel machines that include release times and auxiliary resources with an objective function that aims to minimize cycle time.

4.1. Phase I Mathematical Model Formulation

The purpose of phase I is to explore relatively simple, yet practical, IP models capable of scheduling *n* jobs in *m* machines for small-to-large job sizes for tactical purposes.

Indices and Sets

$j\ index\ for\ jobs, j \in J$

$r\ index\ for\ auxiliary\ resource, r\ \in R$

$m\ index\ for\ machines, m \in M$

$t\ index\ for\ time, t \in H$

Parameters:

$r_j\ release\ date\ for\ job\ j$

$p_j\ process\ time\ for\ job\ j$

$H\ Set\ of\ time\ periods\ where\ |H| \geq \sum_{j \in J} p_j + \max\{r_j\}$

Decision variables:

x_{jmrt}

$= \begin{cases} 1\ if\ job\ j\ is\ assigned\ to\ machine\ m, resource\ r\ to\ start\ processing\ at\ time\ t \\ 0\ otherwise \end{cases}$

Formulation

The objective function (eq 4.1) aims to minimize total sum of completion times (C_j). That is, if x_{jrmt} = 1, the process time for job *j* is added to start time *t* which determines completion time of job *j*. The sum of the job's completion time makes up the objective function which must be minimized. It is noteworthy that each job-machine combination has a unique auxiliary resource, but these resources are shared among multiple jobs-machines according to the job-machine eligibility restrictions.

$$min \sum_{j \in J} C_j \qquad (4.1)$$

Subject To:

$$\sum_{m\in M}\sum_{r\in R}\sum_{t=r_j}^{H-p_j+1}(t+p_j-1)\,x_{jmrt}\ \le\ C_j \forall\, j\in J \tag{4.2}$$

$$\sum_{m\in M}\sum_{r\in R}\sum_{t=r_j}^{H-p_j+1}x_{jmrt}=1\ \forall\, j\in J \tag{4.3}$$

$$\sum_{j\in J}\sum_{r\in R}\sum_{t'=\max(r_j,t-p_j+1)}^{\min(t,H-p_j+1)}x_{jmrt'}\le 1\ \forall\, m\in M,t \tag{4.4}$$

$$\in\{r_j\ldots H-p_j+1\}$$

$$\sum_{j\in J}\sum_{m\in M}\sum_{t'=\max(r_j,t-p_j+1)}^{\min(t,H-p_j+1)}x_{jmrt'}\le 1\ \forall\, r\in R,t \tag{4.5}$$

$$\in\{r_j\ldots H-p_j+1\}$$

$$x_{jrmt}\in\{0,1\} \tag{4.6}$$

The first constraint (eq. 4.2) calculates the expected completion time for each job *j*, and the expression in parentheses serves to offset the process time for each job. The second constraint (eq. 4.3) forces all jobs to be processed at one period *t*. Constraints (eq. 4.4) and (eq. 4.5) enforce that at any given time *t* only one job can be processed at most on a given machine and resource, respectively. Note that setups are not incorporated in this formulation; Sequence-dependent setups are incorporated in phase II, which is presented in chapter 5. Equation (4.6) defines the binary variable that indicates job *j* to start processing at the beginning of period *t* in machine *m* with resource *r*. At time *t* =1, the model assumes a resource (randomly selected) was previously loaded in the machine and aims to assign a job that is compatible with that resource to minimize setups. This

formulation can be applied to any practical, real-world manufacturing environments in the semiconductor and plastic molding injection industry.

4.2. Experimental Design

The experimental design adopted in this dissertation aims to represent real-world practical problems. Hence, the following three factors were selected: problem instance size, problem instance complexity and job-machine sharing density (also known as machine eligibility restrictions in the parallel machine scheduling literature). Problem instance size is defined as a function of the number of jobs, machines, and auxiliary resources. For instance, a small problem instance of 10|2|4 is defined as a combination of 10 jobs, 2 machines, and 4 auxiliary resources with problem instance complexity defined as a function of process times, release dates, and resource setup/delivery times (*cf*. Sousa and Wolsey, 1992). We define sharing density as the percent of job-machine combinations (i.e. 100%, 66% and 33%), which are used for each scenario described in Table 4-1 Based on personal experience in large scale manufacturing, real-world problem instances resemble more of a sparse matrix than a dense one. It must be noted that jobs were randomly assigned to families; hence, the number of families per problem instance is depicted in table below. It is worth mentioning that chapter 4 and chapter 5 will use the same problem instances.

Table 4-1: Model Parameter Summary for Model with Objective Function $\sum C_j$

Instance Size Jobs|Mach|Res (families)	**Instance Complexity**	**Sharing Density**
• 10 |2| 4 (2 families) • 25 |3| 6 (3 families) • 50|5|12 (6 families) • 75|7|18 (9 families) • 100|10|24 (12 families)	• Reduced • Moderate • Complex	• 33% • 66% • 100%

A total of 45 scenarios resulted from running 3 factors (size, complexity and sharing density) at 5, 3, 3 levels, respectively. We introduced three model complexities: reduced, medium, and hard complexity. Reduced complexity was defined as solving a problem with process times (PT) and ready times (RT) uniformly distributed between 1-5 units of time. Setup time is 1 unit and resource travel time between machines (RTTBM) is between 1-2 units of time. The process time and ready time lower bound was 1 unit of time, the upper bounds can be seen in Table 4-2, including the detailed parameters utilized for the medium and hard complexity cases.

Table 4-2: Complexity Design Parameters

Complexity	**PT UB**	**RT UB**	**RTTBM LB**	**RTTBM UB**
reduced	5	5	1	2
medium	10	10	3	4
hard	20	20	5	9

We believe these levels of problem instance size, complexity, and sharing density should provide insight to practitioners on the proposed models capability. Next, we present the experiment results and conclusions.

4.3. Phase I Experimental Results and Conclusions

The computational results for the time-indexed IP model are depicted in Table 4-3 for the 45 test problem instances which resulted from the combination of 3 complexity levels, 5 job-machine-resource combinations, and 3 job-machine density levels. The Gap (%) provided by Cplex is the difference between the best integer solution and the best bound. Solution time (seconds) includes the model build and solve times. The number of model variables (Vars) and constraints (Cons) generated by the model are also presented to provide insight into the Gap % and run times.

Optimal solutions were found for 45 out of the 45 scenarios in less than one hour for all the combinations. It must be highlighted that several instances of 100 jobs with reduced and medium complexity found optimal solutions under 150 seconds (2.5 minutes), up to the "MED|J100|M10|R24|0.33" instance. Optimal solutions were found in less than 600 seconds (10 minutes) for 41 out of the 45 scenarios, including the medium complexity J100 instance.

Table 4-3: Phase I IP Model Results

Com|Jobs|Mach|Res|Den	Obj Funct	Best Bound	Sol Time (sec)	Gap (%)	Vars	Cons
RED|J10|M2|R4|0.33	111	111	0.13	0	487	283
RED|J10|M2|R4|0.66	93	93	0.13	0	479	217
RED|J10|M2|R4|1.0	100	100	0.28	0	654	236
RED|J25|M3|R6|0.33	399	399	0.41	0	2098	619
RED|J25|M3|R6|0.66	267	267	0.38	0	2181	477
RED|J25|M3|R6|1.0	304	304	0.53	0	3760	535
RED|J50|M5|R12|0.33	739	739	1.78	0	4691	1150
RED|J50|M5|R12|0.66	793	793	3.19	0	9866	1289
RED|J50|M5|R12|1.0	692	692	3.44	0	14520	1149
RED|J75|M7|R18|0.33	1090	1090	2.94	0	10979	1746
RED|J75|M7|R18|0.66	899	899	3.53	0	19773	1594
RED|J75|M7|R18|1.0	1123	1123	9.77	0	33535	1846
RED|J100|M10|R24|0.33	1261	1261	21.92	0	17871	2162
RED|J100|M10|R24|0.66	1283	1283	13.6	0	37070	2163
RED|J100|M10|R24|1.0	1250	1250	26.38	0	54310	2172
MED|J10|M2|R4|0.33	211	211	0.25	0	893	514
MED|J10|M2|R4|0.66	181	181	0.31	0	867	387
MED|J10|M2|R4|1.0	192	192	0.25	0	1196	424
MED|J25|M3|R6|0.33	741	741	1.73	0	3927	1133
MED|J25|M3|R6|0.66	478	478	1.08	0	3800	809
MED|J25|M3|R6|1.0	574	574	1.81	0	6946	964
MED|J50|M5|R12|0.33	1382	1382	3.98	0	8447	2026
MED|J50|M5|R12|0.66	1444	1444	331.93	0	18041	2305
MED|J50|M5|R12|1.0	1266	1266	9.92	0	26260	2024
MED|J75|M7|R18|0.33	2007	2007	13.16	0	19868	3079
MED|J75|M7|R18|0.66	1634	1634	39.55	0	35475	2782
MED|J75|M7|R18|1.0	2103	2103	104.43	0	62228	3342
MED|J100|M10|R24|0.33	2336	2336	32.1	0	32038	3775
MED|J100|M10|R24|0.66	2313	2313	263.4	0	66521	3777
MED|J100|M10|R24|1.0	2206	2206	291.93	0	97520	3798
HAR|J10|M2|R4|0.33	411	411	0.27	0	1716	982
HAR|J10|M2|R4|0.66	358	358	0.3	0	1627	719
HAR|J10|M2|R4|1.0	383	383	0.41	0	2298	805
HAR|J25|M3|R6|0.33	1435	1435	6.31	0	7447	2120
HAR|J25|M3|R6|0.66	929	929	1.2	0	7316	1530
HAR|J25|M3|R6|1.0	1096	1096	5.14	0	13039	1785
HAR|J50|M5|R12|0.33	2607	2607	20.11	0	15970	3777
HAR|J50|M5|R12|0.66	2798	2798	123.09	0	34402	4338
HAR|J50|M5|R12|1.0	2402	2402	222.89	0	50215	3807
HAR|J75|M7|R18|0.33	3849	3849	336.99	0	37998	5793
HAR|J75|M7|R18|0.66	3138	3138	162.31	0	66891	5158
HAR|J75|M7|R18|1.0	4008	4008	741.87	0	119607	6322
HAR|J100|M10|R24|0.33	4427	4427	771.7	0	60364	7000
HAR|J100|M10|R24|0.66	4327	4327	1944.18	0	125540	7006
HAR|J100|M10|R24|1.0	4215	4215	3132.99	0	187890	7186

In conclusion, this formulation rendered positive results since optimal solutions were found for 45 out of the 45 scenarios in less than one hour. This chapter excluded setups to determine lower bounds on run times when setups are excluded. The emphasis of the subsequent phase is to reduce run-time and complexity when setups are included, which is studied in chapter 5.

These experimental results provide successful insight into how the time-indexed IP model alone could be used in a tactical space for practical problems where setups are not required, and resources are needed to process a job.

4.4. Phase I Conclusions

A time-indexed IP parallel machine scheduling problem with dual resources and ready times was studied. The proposed $P_m|r_j, aux\ 1|\sum C_j$ model is an expansion of the model originally proposed by (Sousa and Wolsey, 1992) known as the time-index formulation. The original model was expanded to include multiple machines and auxiliary resources. To our knowledge, the proposed IP formulation has not been published before nor have the following results been previously reported. Optimal solutions were found for 45 out of the 45 scenarios in less than one hour for all the combinations. Several instances of 100 jobs with reduced and medium complexity found optimal solutions under 150 seconds (2.5 minutes) up to the "MED|J100|M10|R24|0.33" instance. Optimal solutions were found in less than 600 seconds (10 minutes) for 41 out of the 45 scenarios including the medium complexity J100 instance. Phase I formulation is recommended to run longer time horizons in order to determine area capacity. This formulation can also be used to establish the objective function lower bound since a

model without setups is going to allow more jobs in the schedule seeing that there is no idle time changing auxiliary resources. Finally, this model can be used to determine the lower bound for run time and to decide complexity as compared with a model that includes setups.

4.5. Phase I Summary

This chapter has addressed the minimization of total completion time for resource constrained parallel machine scheduling problems with job-machine eligibility restrictions with release dates denoted $P_m|r_j, M_j, aux\ 1|\sum C_j$. To our knowledge, this exact formulation has not been used to solve the photolithography scheduling problems with release dates without setups.

A time-indexed IP optimization model has been formulated to solve this problem for 45 test problem instances resulting from five problem sizes, three complexity levels and three machine eligibility restriction levels as presented in chapter 3. We vary the number of jobs, machines, resources, process times, release times, and job-machine restrictions to determine how this model would behave in a real-world environment. IBM Cplex optimization engine was used to solve the problem with default settings.

These experimental results provide insight on how the time-indexed IP model alone could be used to solve real-world practical resource constrained parallel machine scheduling problems where setups are not required. Process times of real-world practical systems also need to be compared to those used in this study in order to gain more insight into the proposed model. The results of phase I modeling lay the foundation for the phase

II (Chapter 5) and Phase III (Chapter 6), where we will incorporate sequence-dependent setups, which further complicates the scheduling problem. These results could also be used to determine objective function bounds for large size problem instances where optimization model alone could not find a reasonable result in one hour.

CHAPTER 5 PHASE II RESOURCE CONSTRAINED PARALLEL MACHINE SCHEDULING WITH SETUPS MODEL

In Phase II, we expand the overall complexity of the models by adding sequence-dependent setup times and resource transportation time between machines. An NP-hard problem becomes even more difficult to solve with the addition of sequence-dependent setups.

The objective function aims to minimize total completion time for resource constrained parallel machine scheduling model with sequence-dependent setup times denoted as $P_m|s_{jk}, r_j, M_j, aux\ 1|\sum C_j$. A pre-processing heuristic is proposed in order to keep the solution time tractable according to the problem instance size and complexity previously depicted in Table 4-1. The main objective of this phase is to verify and validate our mathematical model is efficient with auxiliary resources and sequence-dependent setup times. A key contribution of this phase is the combination of medium and large size model instances for identical parallel machines, shared auxiliary resources with sequence-dependent setup time including release dates which also complicates the problem as opposed to the assumption that all jobs are available at time t=1.

We first describe an IP model to find exact optimal solutions for this problem. We then introduce a heuristic algorithm to find feasible solutions within a reasonable amount of time. This method aims to find feasible solutions to be provided to the timed-indexed IP model as a starting solution. The objective of this method is to attempt to improve the solution found by the heuristic using branch and bound algorithms provided by the IP solver. The manufacturing problem we are addressing considers jobs that must be

processed by one machine and one auxiliary resource simultaneously. Also, we assume that the auxiliary resource cannot be assigned to more than one machine at the same time. The environment has a set of jobs *J* either incoming to the system ("ready dates") or available for immediate scheduling. These jobs can be processed in one or more of identical parallel machines if fitted with one of the auxiliary resources *r* in set *R.* These scarce resources are shared among some or all machines. Incoming jobs require a release date to prevent assignment before they become available. Once a job is assigned to machine-auxiliary resource pair, that job must complete processing. That is, preemption is not allowed. Once a job has been assigned to a combination of resources, that job may need a setup time. A setup time is incurred when job *j* is not compatible with the previous job and a different auxiliary resource is required to be added to the machine to run the job. Thus, the machine must temporarily idle for the auxiliary resource setup. If jobs are compatible with each other they belong to the same family since no setup is required. In practice, a family consists of a combination of specific product and an operation; however, for this study a family is created by randomly assigning jobs to family sets forcing each set to have a similar number of jobs. If a resource is transferred from one machine to another one, an additional setup time (travel time between machines) is required. Inefficient resource assignments may cause a machine to be temporarily idle until the proper auxiliary resource is installed. Typically, auxiliary resources are finite and expensive that need to be shared among different machines, thus if a resource is being used in one machine, another one may be idle waiting for this resource. Thus, generating efficient schedules not only optimizes resource assignment and utilization, but

also overall factory throughput resulting in lower cycle times. We assume that a random auxiliary resource was previously loaded in a machine at time 0. Hence, the model must respect an initial setup during period t=1.

One of the goals of this study is to find near-optimal solutions for the manufacturing system previously described by applying exact optimization methods, such as the time-indexed IP model introduced in chapter 4. The second goal is to develop a heuristic capable of generating fast and feasible initial solutions when IP model is not capable of finding one. The third goal is to generate a hybrid model between the IP model and the proposed heuristic. That is, the solution found by the heuristic is given to the IP model aiming to reduce the time horizon and provide an initial solution to the IP model. We believe a combination of an exact and a heuristic algorithm allow practitioners tackle medium and large size problem instances where setups are required for machine and auxiliary resources in complex manufacturing systems such as photolithography.

We now turn to defining the time-indexed IP model including the indices, sets, objective function, and constraints.

5.1. Phase II Mathematical Model Formulation

Indices and Sets

$j\ index\ for\ jobs, j \in J$

$r\ index\ for\ auxiliary\ resource, r\ \in R$

$m\ index\ for\ machines, m \in M$

$t\ index\ for\ time, t \in |H'|$

Parameters:

r_j *release date for job j*

p_j *process time for job j*

$s_{r',r}$ *setup time between resource r′ and resource r*

$tt_{m',m}$ *auxiliary resource transfer time*

between machine m′ and machine m

H *Set of time periods where* $|H| \geq \sum_{j \in J} p_j + \max\{r_j\}$

H'*Prime set of periods where* $|H'| \geq \Theta * \left(\frac{\sum_{j \in J} p_j + \max\{r_j\}}{m}\right)$ *and* $2 \leq \Theta \leq m$ *for tighter formulation.*

H should be used for single machine scheduling or when model parameters are not well understood. Otherwise, use $|H'|$ formulation and a machine factor (Θ) of 2 or greater to tighten the formulation. This paper used Θ =2 for 45 problem instances without any infeasibility issues.

Decision variables:

x_{jmrt}

$= \begin{cases} 1 \text{ if job } j \text{ is assigned to machine } m, \text{resource } r \text{ to start processing at time } t; \\ 0 \text{ otherwise} \end{cases}$

Formulation:

The objective function aims to minimize total sum of completion times (C_j). That is, if x_{jrmt} = 1, the process time for job *j* is added to start time *t* which determines completion time of job *j*. The total sum of all the job's completion time makes up the

objective function which must be minimized. It is noteworthy that each job-machine combination has a unique auxiliary resource, but these resources are shared among multiple jobs-machines according to the density matrix.

$$min \sum_{j \in J} C_j \tag{5.1}$$

Subject To:

$$\sum_{m \in M} \sum_{r \in R} \sum_{t=r_j}^{H'-p_j+1} (t + p_j - 1)\, x_{jmrt} \;=\; C_j \forall\, j \in J \tag{5.2}$$

$$\sum_{m \in M} \sum_{r \in R} \sum_{t=r_j}^{H'-p_j+1} x_{jmrt} = 1 \;\; \forall\, j \in J \tag{5.3}$$

$$x_{jmrt} + \sum_{r' \in R} \sum_{t'=\max\left(r_j, t-p_i-tt_{m',m}-s_{r',r}+1\right)}^{\min\left(t+p_j+tt_{m,m'}+s_{r,r'}-1, H'-p_j+1\right)} x_{im'r't'} \leq 1 \tag{5.4}$$

$$\forall\, i, j \in J: i > j; m, m' \in M: m = m';\; r \in R;\; t \;\in \{r_j \ldots H' - p_j + 1\}$$

$$x_{jmrt} + \sum_{m' \in M} \sum_{t'=\max\left(r_j, t-p_i-tt_{m',m}-s_{r',r}+1\right)}^{\min\left(t+p_j+tt_{m,m'}+s_{r,r'}-1, H'-p_j+1\right)} x_{im'r't'} \leq 1 \tag{5.5}$$

$$\forall\, i, j \in J: i > j; m \in M;\; r, r' \in R: r = r';\; t \;\in \{r_j \ldots H' - p_j + 1\}$$

$$x_{jrmt} \in \{0,1\},\; C_j \;\in Z^+ \tag{5.6}$$

The first constraint set (eq. 5.2) calculates the expected completion time for each job *j*, the expression in parentheses serves to offset the process time for each job. The second constraint set (eq. 5.3) forces all jobs to start processing at only one period *t* between the ready time and the latest time it can start (and still finish by the end of the

horizon) per machine-resource combination. The third constraint set (eq. 5.4) prevents job-machine-period overlap and incorporates two different setups: *i*) resource-sequence dependent setup and *ii*) a machine-sequence dependent setup which is incorporated in the form of resource travel time between machines. It is worth pausing for a moment to highlight the resulting computational complexity of constraint (eq. 5.4), since all combinations between job *j* and *i* where $i \neq j$ must be explored. All combinations between machine m and m' where $m = m'$, also need to be explored. Similarly, all resource combinations must be explored between r and $r'$where $r = r'$. We estimate the model will generate $O(jmrt)$ variables and $O(jmt^2r)$ constraints. The constraint sets (5.4) and (5.5) are the two that generate the most constraints in this formulation up to 16.9 million among all the constraints in our experimentation.

Similarly, constraint set (eq. 5.5) enforces that at any given time *t* at most one job can be processed on a given machine and resource respectively This formulation resembles real-world complex manufacturing environments in the semiconductor industry, mold injection operations, and the PCB industry. Equation (5.6) defines a binary decision variable *x* and a non-negative integer variable for completion time.

5.2. Phase II Heuristic Pseudocode

This study proposes a heuristic to find fast feasible solutions that can be used to prime the proposed IP model with an initial feasible solution. The proposed heuristic aims to minimize weighted completion time by reducing auxiliary resource setups. The objective is accomplished by scheduling as many jobs as possible of the same family to the same machine and resource. That is, all jobs that can run on the same

machine/resources and do not require a setup (i.e. jobs are compatible due to same resource type) are said to be part of the same family. Jobs in the same family are scheduled and/or sequenced together in order to reduce setups and minimize auxiliary resource travel time when moved between machines when possible. That is, ready times may not allow all jobs of the same family to be scheduled back to back.

The heuristic aims to reduce completion time and accomplishes its objective through a series of sorts and assignments as depicted in Figure 5-1 : *i*) a set of pre-assigned jobs randomly selected (i.e. set name jmrt_pre-assigned) are given to the model to simulate that all machines start with a pre-assigned job in the busy state. *ii*) then, the pre-assigned jobs are removed from the "Full_Set" which contains all unscheduled jobs in addition to other attributes to run the heuristic. The pre-assigned jobs are added to the "jmrtf_scheduled" set which contains all jobs that have been scheduled including the following attributes: job id, machine id, resource id, family id, process time, ready time, start time and end time. Model variables and attributes are updated to reflect the pre-assigned jobs. iii) An initialization sort logic consisting of four attributes (i.e. mach_score, job process_time, job family and mach_res_score) is executed. The four attributes are sorted ascending or descending in order to select the machine to be loaded next. It is worth noting that four attributes sorted in two different ways (ascending or descending), similar to the levels of a factorial experiment design resulting in the generation of 16 combinations used to implement the heuristic. Then, a post-initialization sort is executed by sorting 3 attributes (i.e. res_last_completion_time, job_ready_time, job_fam) either all ascending or all descending resulting in two additional combinations.

Hence, a total of 32 sorting variations are executed for each problem instance based on the pre-initialization and post-initialization logic. iv) After the initialization logic step, all future sorts for the job-machine-resource combinations aim to select the least utilized machine-resource combination. Machine-resource utilization is tracked using the "mach_current_load" and "res_current_load" attributes that track total time assigned to those resources. The "Least Utilized Mach Sort" logic consists of sorting "mach_current_load" attribute ascending to select the least utilized machine as shown in Figure 5-1. Then, the job-resource are selected and evaluated to make sure they are available. That is, the resource must be readily available. The process it is repeated until all jobs are assigned and until all the scenarios are generated as outlined in Figure 5-1 The full set of attributes for jobs, machines (mach) and resources (res) that are used to keep track of time variable and the history of jobs assigned to machine and resources is shown in Figure 5-4.

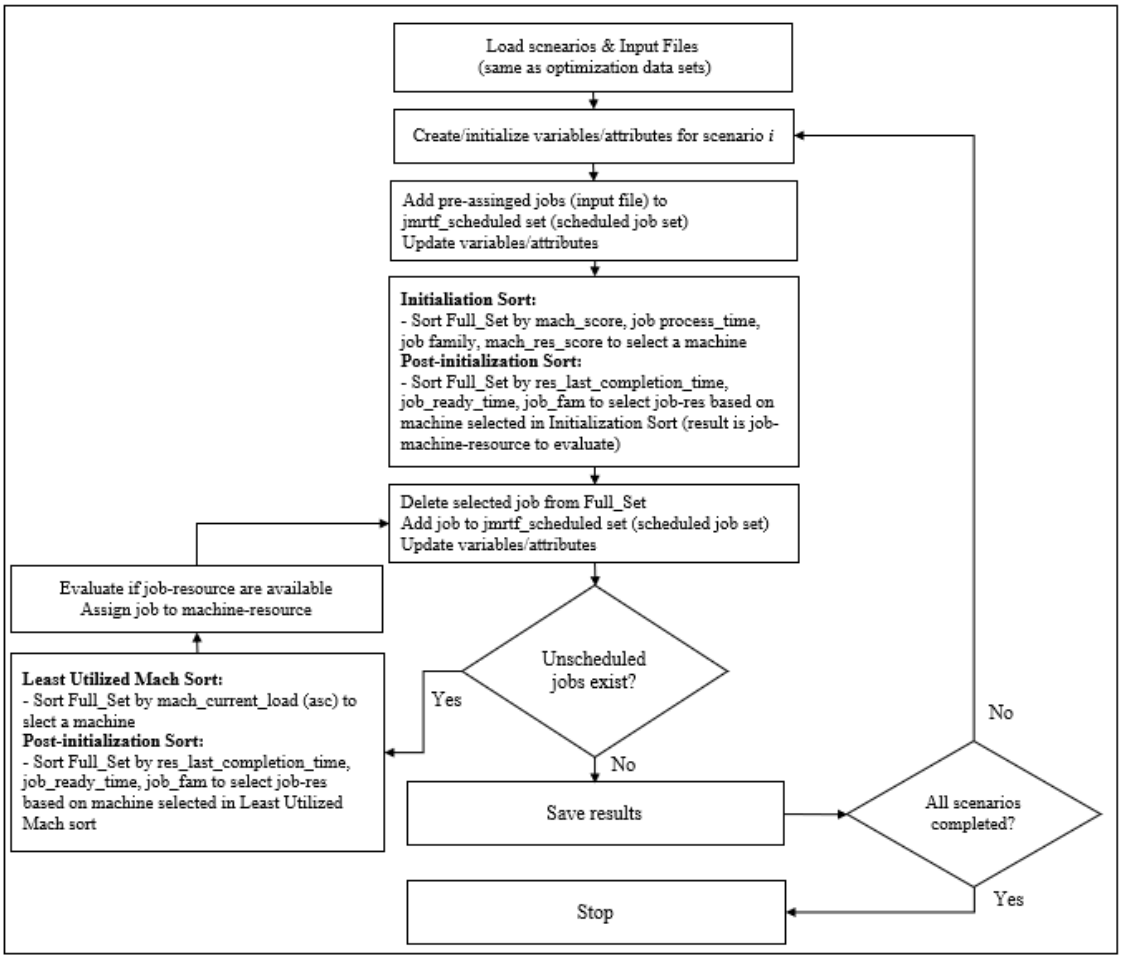
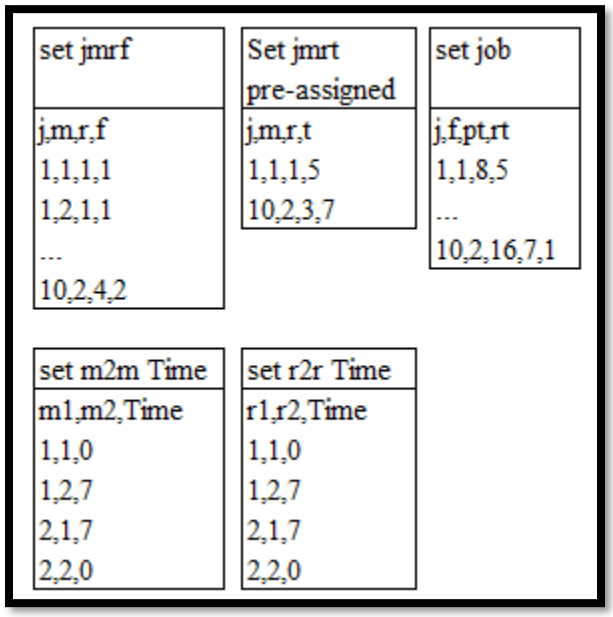


Figure 5-1 Heuristic Process Flow

A full set of input files and variables is presented next in Figure 5-2. These input sets are given to the optimization IP model and the heuristic. Set “jmrf” represents the job-machine-resource-family combinations and the sample shown represents the job id, machine id, resource id and family id. Similarly, the set “jmrt pre-assigned” represents the job-machine-resource-family randomly pre-assigned (part of the input files) where the numbers shown in the Figure 5-2 represent job id, machine id, resource id and job’s start time. The set “job” provides the job-family-process time (pt)- ready times (rt) combinations. Set “m2m Time” provides the resource travel time from m1 id to m2 id.

The “Set r2r Time” provides the setup time when swapping resource r1 id to r2 id. It must be noted that in this study the start time is set at t=1.



set jmrf
j,m,r,f
1,1,1,1
1,2,1,1
...
10,2,4,2

Set jmrt pre-assigned
j,m,r,t
1,1,1,5
10,2,3,7

set job
j,f,pt,rt
1,1,8,5
...
10,2,16,7,1

set m2m Time
m1,m2,Time
1,1,0
1,2,7
2,1,7
2,2,0

set r2r Time
r1,r2,Time
1,1,0
1,2,7
2,1,7
2,2,0

Figure 5-2 Input Sets

An explanation on how the attributes are initialized is provided next.

- mach_score = sum of process time of all jobs capable of running on a given machine.
- mach_last_res = 0 unless a job-resource was pre-assigned.
- mach_current_load = sum of the process time of the jobs assigned to it.
- mach_last_completion_time = t_start+ process time -1 for the last job assigned.
    - mach_res_score = 1 for all machine-resource combinations during initialization. Then, set = 0 for the machine-resources combinations that received a pre-assigned job. Sorting by mach_res_score (asc) after the job’s first pass allows the heuristic to select the next job of the same family aiming to eliminate a setup.

- res_last_machine= 0 unless a job-machine was pre-assigned.
- res_current_load = sum of the process times of the jobs assigned.
- res_last_completion_time = t_start+ process time -1 for the last job assigned.

The pseudo code associated with the heuristic is presented in Figure 5-3.

```
Load all scenario names //45 instances to run
For scenario i do
        Initialize variables (job_counter=0)
        Read data sets (See Figure 5-2)
        Create & Initialize machine-resource attributes (See Full_Set in Figure 5-4)
        Add pre-assigned jobs to jmrtf_scheduled set (initial solution given)
        Delete pre-assigned jobs from the Full_Set
      While unscheduled_jobs exist do
            If job_counter ==0 or job_first_pass = True //Initialization Sorting
                job_counter = job_counter +1 & job_first_pass = False
                 Full_Set.Sort by mach_score (**ascending/ descending)
                  Full_Set.Sort by job process_time (**descending/ ascending)
                    Full_Set.Sort by job family (**ascending/descending)
                      Full_Set.Sort by mach_res_score (**descending/ ascending)
              Else //after the first job pass, all future job sorting end up here!
                 Full_Set.Sort mach_current_load (ascending)
               top_mach = Full_Set.machine row1(least loaded mach_current_load)
            For jmrt_m in Full_Set where mach= top_mach sorted by **res_last_completion_time
                    (asc), job_ready_time (asc), job_fam (asc)
                 top_res = jmrt_m.resource row1(least loaded res_current_load)
                     For jmrt_mr in Full_Set where mach= top_mach & res=top_res
                     fam_filter= jmrt_mr.family id (row 1)
                        For jmrf_mrf  in Full_Set where mach= top_mach & res=top_res & Fam =
                              fam_filter sorted by job_ready_time (asc)
                     IF job_avail = True & Res_available=True (feasible schedule)
                             Add job to jmrtf_scheduled set
                           Update Full_Set attributes
                             Delete jmrt combination from Full_Set
                        ELSE
                            Do nothing, move to next row
                     End for jmrt_mr
                  End For jmrt_m
          If no jobs scheduled
                mach_last_completion_time = mach_last_completion_time+1
             End If
        End While unscheduled_jobs exist
       Save results (output) after all jobs have been scheduled for scenario i
End For scenario i do
```

Figure 5-3 Heuristic Pseudocode

The decision variable x_{jmrt} (heuristic output) is generated using the attributes presented in Figure 5-4 which allow the model keep track of all job-machine-resource assignments and time.

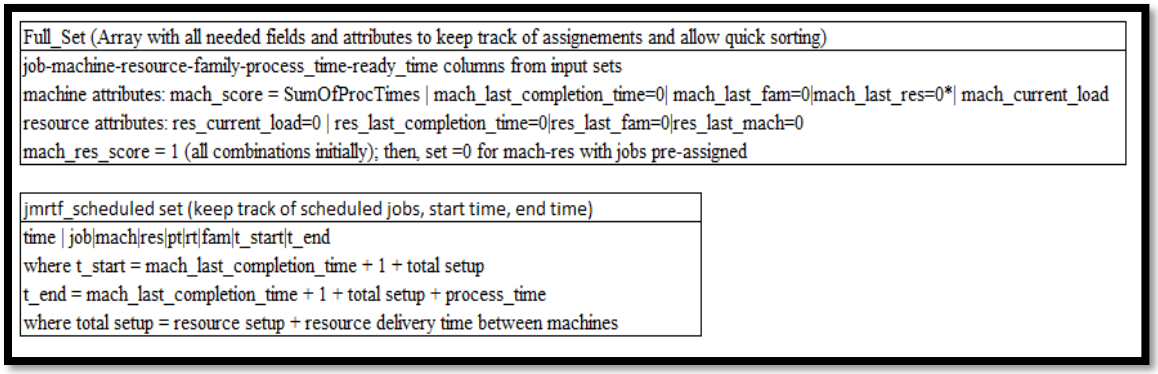


Figure 5-4 Heuristic Sets for Scheduled and Unscheduled jobs

A job can be scheduled if it is readily available (i.e. variable job_avail = True). That is, the job's ready time must be equal or less the machine's last completion time attribute denoted by mach_last_completion_time + 1 unit. A resource can be scheduled with a job-machine if the resource is available. That is, the variable Res_avail=True if resource is not being utilized by other machine or the resource is not being transferred between machines based on the set m2m

5.3. IP Model Experimental Results

The computational results for the time-indexed IP model are depicted in Table 5-1 for the 45 problem instances. The first column gives the problem instance as denoted by the combination of complexity, number of jobs-machines-resources and sharing density—Com|Jobs|Mach|Res|Den, The second column gives the objective function value, the third column provides the best bound generated by the solver, the fourth column shows the solution time in minutes. The fifth column gives the Gap (%) provided by Cplex as the difference between the best integer solution and the best bound. The last two columns give the number of variables (Vars) and constraints (Cons) generated by the

model. As previously stated, a total of 45 scenarios were run as a result of the combinations among 3 complexity levels, 5 job-machine-resource combinations and 3 job-machine density levels.

Optimal solutions were found for 16 out of the 45 scenarios in less than one hour for several combinations of 10/25 job sizes. A Gap of 10% or less was found for 24 out of the 45 scenarios ranging from 10 jobs up to 75 jobs (i.e. MED|J75|M7|R18|0.33). Optimal solutions were found for 15 out of the 45 scenarios for the small instances (10/25 job) in less than 600 seconds (10 minutes). No results are reported for one out of the 45 instances (HAR|J100|M10|R24|1.0) due to constraints equation (4) and (5) making the model excessively large. It must be noted that prior to using H' time horizon formulation the H formulation rendered at least 5 instances out of memory. Hence, it was decided to tighten the model using the H' formulation described in section 4.

Table 5-1: Model Results for Optimization Model $P_m|s_{jk}, r_j, aux|\sum C_j$

Com|Jobs|Mach|Res|Den	Obj Funct	Best Bound	Sol Time (sec)	Gap (%)	Vars	Cons
RED|J10|M2|R4|0.33	128	128	1	0%	542	1740
RED|J10|M2|R4|0.66	110	110	0	0%	556	2362
RED|J10|M2|R4|1.0	118	118	1	0%	756	4207
RED|J25|M3|R6|0.33	464	464	46	0%	2282	17521
RED|J25|M3|R6|0.66	322	322	43	0%	2409	24466
RED|J25|M3|R6|1.0	341	341	693	0%	4139	66275
RED|J50|M5|R12|0.33	991	957	3601	3%	5086	51560
RED|J50|M5|R12|0.66	937	873	3602	7%	10622	200233
RED|J50|M5|R12|1.0	1660	62	3601	96%	15794	500112
RED|J75|M7|R18|0.33	1451	1292	3602	11%	11891	191405
RED|J75|M7|R18|0.66	2231	85	3602	96%	21361	689285
RED|J75|M7|R18|1.0	2545	88	3603	97%	36290	1719167
RED|J100|M10|R24|0.33	3072	118	3602	96%	19385	409063
RED|J100|M10|R24|0.66	3087	125	3604	96%	40324	1769745
RED|J100|M10|R24|1.0	3063	122	3610	96%	59085	3793645
MED|J10|M2|R4|0.33	232	232	3	0%	1002	3235
MED|J10|M2|R4|0.66	197	197	1	0%	1020	4337
MED|J10|M2|R4|1.0	214	214	2	0%	1403	7803
MED|J25|M3|R6|0.33	824	824	185	0%	4297	33201
MED|J25|M3|R6|0.66	548	548	116	0%	4249	43286
MED|J25|M3|R6|1.0	619	609	3601	2%	7730	123864
MED|J50|M5|R12|0.33	1718	1655	3601	4%	9240	93821
MED|J50|M5|R12|0.66	1704	1581	3603	7%	19552	369559
MED|J50|M5|R12|1.0	2813	79	3602	97%	28786	913568
MED|J75|M7|R18|0.33	2449	2249	3601	8%	21707	349682
MED|J75|M7|R18|0.66	4398	107	3603	98%	38683	1248299
MED|J75|M7|R18|1.0	4897	114	3605	98%	67784	3214873
MED|J100|M10|R24|0.33	5683	155	3601	97%	35072	740634
MED|J100|M10|R24|0.66	5583	165	3606	97%	73046	3204091
MED|J100|M10|R24|1.0	5461	144	3615	97%	107061	6875867
HAR|J10|M2|R4|0.33	432	432	1	0%	1935	6277
HAR|J10|M2|R4|0.66	376	376	2	0%	1935	8239
HAR|J10|M2|R4|1.0	406	406	9	0%	2720	15157
HAR|J25|M3|R6|0.33	1524	1524	349	0%	8182	63302
HAR|J25|M3|R6|0.66	1016	1016	328	0%	8225	84006
HAR|J25|M3|R6|1.0	1154	1146	3602	1%	14610	233934
HAR|J50|M5|R12|0.33	3095	2940	3602	5%	17546	178486
HAR|J50|M5|R12|0.66	3109	2702	3602	13%	37414	708309
HAR|J50|M5|R12|1.0	2846	2001	3603	30%	55270	1756920
HAR|J75|M7|R18|0.33	4660	3974	3602	15%	41696	672674
HAR|J75|M7|R18|0.66	7792	160	3605	98%	73320	2366947
HAR|J75|M7|R18|1.0	9639	160	3620	98%	130768	6207290
HAR|J100|M10|R24|0.33	9582	2670	3602	72%	66446	1403993
HAR|J100|M10|R24|0.66	10442	239	3611	98%	138488	6080351
HAR|J100|M10|R24|1.0	-	-	-	-	-	-
Avg	2589	769	2334	37%	28402	1055745
Std dev	2722	939	1704	45%	34524	1743657

It is evident that there is a positive correlation (factor = 0.68) between Gap % and the number of constraints generated as depicted in Figure 5-5. It is also evident that the relationship is not linear.

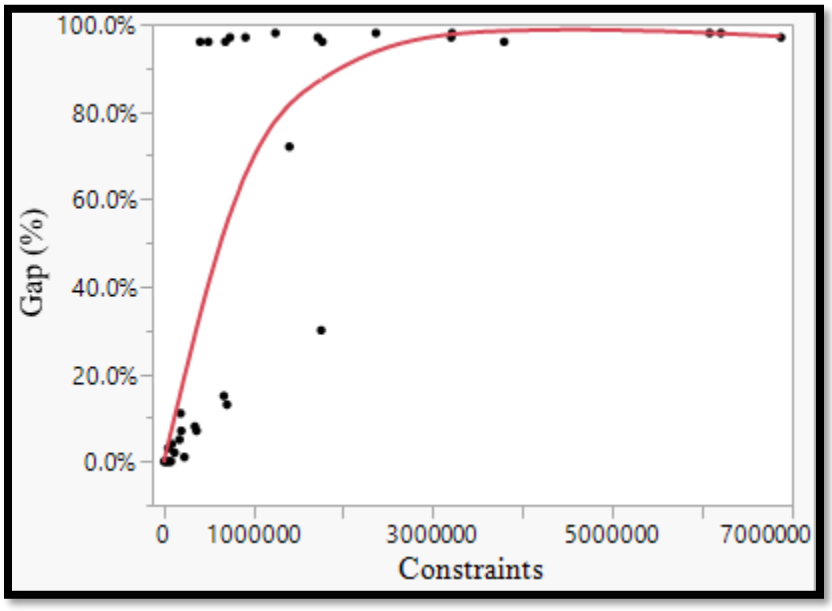


Figure 5-5 IP Model Constraints vs Gap (%) Correlation

The Gap % summary in Table 5.2 shows the IP formulation presents higher Gaps for the instances with 100% sharing which indicates the solution space is larger; hence, the model iterated through more combinations. Table 5-2 also shows that Gaps get larger as the number of Jobs increases. However, the Gap for the complexity instances did not follow a pattern despite the fact that the hard instances have more variables and constraints than the reduced ones. We believe, the Gap is not impacted due to factors such as multiple machines used and time horizon |H'| provided a tighter formulation potentially neutralizing the effect of process times. The average for the hard complexity includes a gap of 100% for the HAR|J100|M10|R24|1.0 combination which was not

solved by the optimization model. The table below shows the average gap by number of jobs, density sharing (DS %) and by complexity. The complexity table has two additional columns to show the average number of variables and average number of constraints.

Table 5-2 IP Model Gap % Summary

Jobs	Avg Gap %
10	0%
25	0%
50	29%
75	69%
100	94%

DS (%)	Avg Gap%
0.33	20.8%
0.66	40.6%
1.0	54.1%

Complexity	Avg Gap%	Avg Vars	Avg Const
reduced	39.9%	15368	629386
medium	40.3%	28042	1148408
hard	35.3%	42754	1413278
Overall Gap	37%		

In order to better understand the behavior of this model with respect to run time, we selected the J50|M5|R12|0.66 scenario and ran it for 30, 60, 90 and 120 minutes for the three complexity levels (reduced, medium, hard). Table 5-3 presents the results. Prior to discussing the results, a striking observation that emerged from this experiment needs to be discussed. The initial models ran on a virtual server and the results obtained became slightly inferior as the model ran longer than 60 minutes. That is, Cplex found slightly better solutions for the 60-minute run than for the 90- and 120-minute runs. Upon investigation in technical forums and by analyzing solver logs, it was apparent this behavior has been reported out by other researchers and Cplex logs show Cplex inducing what it seems to be a smart heuristic allowing the model to find a small improvement (less than 1%) minutes before model reached maximum run time. In order to fix this behavior, the model ran in a laptop to minimize the impact of a virtual server balancing

computing loads and the results are depicted in table 5-3. The laptop has an Intel Core i7-8665U CPU @ 1.9 GHz, 4 Core(s), 8 Logical Processors(s) and 32 GB RAM.

Table 5-3: J50|M5|R12|0.66 Results for 30 to 120 min. Run Time

Complexity	Stop Criteria (min)	OF	BB	Sol Time (sec)	Gap (%)	Var	Con
Reduced	30	1047	785	1802	24.9%	10622	200233
	60	950	876	3601	7.8%		
	90	948	876	5400	7.6%		
	120	947	876	7201	7.5%		
Medium	30	1681	1579	1801	6.1%	19552	369559
	60	1681	1579	3601	6.1%		
	90	1671	1579	5401	5.5%		
	120	1671	1579	7201	5.5%		
Hard	30	3144	2404	1801	23.5%	37414	708309
	60	3099	2931	3601	5.4%		
	90	3099	2931	5401	5.4%		
	120	3091	2949	8121	4.6%		

It is evident that 30 minutes is not enough time to get a near optimal solution for the reduced and hard problem instances. However, the results are the same for the medium 30- and 60-minute instance. The most noticeable improvement is from 30 to 60 minutes, running the model for 90 and 120 minutes did not have a significant effect for this problem instance.

Additionally, the model ran on a laptop with 3 parameters disabled as suggested in technical forums in order to remove the unexpected behavior of the solver. The results presented in Table 5.4 were obtained by applying the following settings: MIP.Strategy.RINSHeur = -1; MIP.Strategy.FPHeur= -1), and MIP.Strategy.LBHeur = 0

which disabled Cplex built-in heuristics. The columns of table below present the objective function, best bound, solution time, Gap %, number of variables and constraints grouped by complexity and maximum run time (stop criteria) given in minutes.

Table 5-4 J50|M5|R12|0.66 results Cplex heuristic disabled 30 to 120 min run time

Complexity	Stop Criteria (min)	Obj Funt	Best Bound	Sol Time (sec)	Gap (%)	Var	Con
Reduced	30	-1	872.7	1800	-	10622	200233
	60	989	876	3601	11.5%		
	90	989	876	5400	11.5%		
	120	964	876	7201	9.1%		
Medium	30	-1	1578	1801	-	19552	369559
	60	1738	1578	3601	9.2%		
	90	1734	1579	5401	9.0%		
	120	1734	1582	7201	8.7%		
Hard	30	-	2934.53	1800	-	37414	708309
	60	-	2946.69	3602	-		
	90	-	2947	5402	-		
	120	-	2947	7202	-		

Results in Tables 5-3 and 5-4 show that Cplex was consistent in finding either the same or better result as the model ran longer. However, disabling Cplex heuristics is not an option as many scenarios did not find a feasible solution ("-") and the results were significantly inferior

In summary, running the model longer than 60-minutes may not be the right decision for practical problems that resemble the characteristics of the J50 size with 66% job-machine sharing density. Next, we present the results obtained by the proposed heuristic.

5.4. Heuristic Experimental Results

The heuristic results for the 45 problem instances are presented in Table 5-5. The objective function value was calculated by adding up the completion time for each job (total completion time). It is evident that the heuristic run times are fast with a range of 0.4 seconds for the "RED|J10|M2|R4|D0.33" and 47 seconds for the RED|J100|M10|R24|1.0 scenario. The table below shows the minimum objective found by the heuristic, the average objective found for the 32 combinations, the standard deviation of the objective function. The last two columns provide the average solution time and the standard deviation (in seconds).

It is evident from Table 5-5 that the average objective function provided by the heuristic (2212) is lower than the average obj function provided by the IP model (2589). Section 5.6 shows the breakdown by scenario for the IP model, heuristic and the hybrid (heuristic + IP model). A total of 32 sorting variations were run per problem instance with the best solution given to the IP model in order to prime it with an initial feasible solution.

Table 5-5 Heuristic Results

Com|Jobs|Mach|Res|Den	Min Obj Funct	Avg Obj Funct	Std Obj Funct	Avg Sol Time (sec)	Std Sol Time (sec)
RED|J10|M2|R4|0.33	141	151	12	0.4	0.1
RED|J10|M2|R4|0.66	137	146	7	0.5	0.1
RED|J10|M2|R4|1.0	127	135	7	0.4	0.0
RED|J25|M3|R6|0.33	556	577	28	1.4	0.1
RED|J25|M3|R6|0.66	418	441	15	1.8	0.2
RED|J25|M3|R6|1.0	427	440	8	2.8	0.5
RED|J50|M5|R12|0.33	1400	1445	35	3.4	0.3
RED|J50|M5|R12|0.66	1062	1175	91	4.9	0.4
RED|J50|M5|R12|1.0	932	987	47	11.6	1.8
RED|J75|M7|R18|0.33	1947	2010	51	6.7	0.5
RED|J75|M7|R18|0.66	1452	1509	41	15.0	1.5
RED|J75|M7|R18|1.0	1593	1660	34	29.6	6.4
RED|J100|M10|R24|0.33	2366	2519	75	10.2	0.7
RED|J100|M10|R24|0.66	2010	2155	79	28.3	5.6
RED|J100|M10|R24|1.0	1930	1993	30	47.5	7.3
MED|J10|M2|R4|0.33	263	274	16	0.4	0.1
MED|J10|M2|R4|0.66	227	246	15	0.5	0.1
MED|J10|M2|R4|1.0	244	250	5	0.5	0.1
MED|J25|M3|R6|0.33	985	1060	75	1.4	0.2
MED|J25|M3|R6|0.66	738	762	15	1.8	0.1
MED|J25|M3|R6|1.0	721	752	28	3.6	0.4
MED|J50|M5|R12|0.33	2457	2521	44	3.0	0.2
MED|J50|M5|R12|0.66	2110	2209	104	5.8	0.7
MED|J50|M5|R12|1.0	1651	1752	84	13.3	2.8
MED|J75|M7|R18|0.33	3333	3435	57	6.0	0.3
MED|J75|M7|R18|0.66	2723	2760	35	14.4	0.9
MED|J75|M7|R18|1.0	2720	2878	158	22.9	1.8
MED|J100|M10|R24|0.33	4183	4287	63	9.9	0.7
MED|J100|M10|R24|0.66	3544	3667	71	29.9	7.7
MED|J100|M10|R24|1.0	3267	3418	122	42.2	2.3
HAR|J10|M2|R4|0.33	497	516	30	0.5	0.2
HAR|J10|M2|R4|0.66	437	463	19	0.4	0.1
HAR|J10|M2|R4|1.0	477	481	4	0.5	0.1
HAR|J25|M3|R6|0.33	1865	1954	116	1.7	0.2
HAR|J25|M3|R6|0.66	1336	1509	101	3.6	1.3
HAR|J25|M3|R6|1.0	1349	1387	23	3.9	0.4
HAR|J50|M5|R12|0.33	4612	4665	53	4.0	0.4
HAR|J50|M5|R12|0.66	3994	4073	51	5.9	0.2
HAR|J50|M5|R12|1.0	3295	3334	37	12.2	2.4
HAR|J75|M7|R18|0.33	6244	6452	136	7.4	0.7
HAR|J75|M7|R18|0.66	4757	5032	146	15.2	1.1
HAR|J75|M7|R18|1.0	5187	5338	110	24.3	1.7
HAR|J100|M10|R24|0.33	7590	7689	63	12.0	1.5
HAR|J100|M10|R24|0.66	6213	6780	232	25.5	4.2
HAR|J100|M10|R24|1.0	6039	6271	143	42.2	3.3
Avg	2212	2301	60	11	1
Std dev	1938	2006	-	12	-

A short summary characterizing the sorting results are provided for small, medium and large problem sizes. It is evident form the results that for the problem size instances with 10 jobs all instances achieved the best results when post initialization sort was: last_completion_time (desc), job ready_time (desc), family (desc). For 4 out of 9 combinations for job size =10 the best results were obtained when the initialization sorting logic was the following: m_score (asc), process time (desc), family (asc), mr_score (asc). For problem instances with 25 jobs 4 out of 9 scenarios obtained the best results when the initialization sort logic was the following: m_score (asc), process time (asc), family (desc), mr_score (desc). Similarly, for J50 problem instances 6 out of 9 scenarios obtained best results when initialization sort logic was -> m_score (asc), process time (asc), family (desc), mr_score (desc). For problem instances with 75 and 100 jobs no sorting logic dominated the best results. In summary, the post-initialization sorting logic had more effect on the smaller problem instances than the larger ones.

5.5. Hybrid Model Experimental Results (IP/Heuristic)

Table 5-6 presents the results for the hybrid model consisting of the heuristic feasible solution provided to the IP model by the heuristic. As previously stated, the heuristic solutions were provided to the IP model as "MIP start" in order to reduce time horizon and to provide the IP model with an initial solution which could be further improved. Reducing the time-indexed IP formulation horizon results in a smaller solution space which is likely to provide better results in less time. It is also important to highlight that providing an initial solution from the heuristic reduced time horizon by ~50% when H formulation was used instead of the H' presented in section 4.

Table 5-6 Hybrid IP/Heuristic Results for $P_m|s_{jk}, r_j, aux|\sum C_j$

Com|Jobs|Mach|Res|Den	Obj Funct	Best Bound	Sol Time (sec)	Gap (%)	Vars	Cons
RED|J10|M2|R4|0.33	128	128	0.5	0%	542	1740
RED|J10|M2|R4|0.66	110	110	0.3	0%	556	2362
RED|J10|M2|R4|1.0	118	118	0.5	0%	756	4207
RED|J25|M3|R6|0.33	464	464	43.3	0%	2072	15715
RED|J25|M3|R6|0.66	322	322	29.6	0%	2409	24466
RED|J25|M3|R6|1.0	341	341	608.5	0%	4064	64915
RED|J50|M5|R12|0.33	992	957	3601.8	4%	7406	77776
RED|J50|M5|R12|0.66	1062	63	3600.7	94%	9877	184813
RED|J50|M5|R12|1.0	932	62	3601.3	93%	15794	500112
RED|J75|M7|R18|0.33	1454	1293	3601.9	11%	11891	191405
RED|J75|M7|R18|0.66	1452	85	3602.2	94%	21361	689285
RED|J75|M7|R18|1.0	1593	88	3605.2	94%	31565	1472594
RED|J100|M10|R24|0.33	2366	118	3601.9	95%	19385	409063
RED|J100|M10|R24|0.66	2010	125	3605.3	94%	40324	1769745
RED|J100|M10|R24|1.0	1930	122	3611.4	94%	55085	3507813
MED|J10|M2|R4|0.33	232	232	0.5	0%	942	3010
MED|J10|M2|R4|0.66	197	197	0.4	0%	972	4085
MED|J10|M2|R4|1.0	214	214	1.7	0%	1243	6715
MED|J25|M3|R6|0.33	824	824	231.5	0%	4157	31997
MED|J25|M3|R6|0.66	548	548	77.6	0%	3955	39872
MED|J25|M3|R6|1.0	616	616	3045.4	0%	6305	98024
MED|J50|M5|R12|0.33	1724	1655	3601.8	4%	12280	128173
MED|J50|M5|R12|0.66	1696	1578	3603.8	7%	17764	332551
MED|J50|M5|R12|1.0	1492	1202	3602.3	19%	25786	808508
MED|J75|M7|R18|0.33	2477	2250	3602.2	9%	21707	349682
MED|J75|M7|R18|0.66	2723	107	3603.9	96%	35434	1132478
MED|J75|M7|R18|1.0	2720	114	3607.9	96%	56759	2639536
MED|J100|M10|R24|0.33	3861	2189	3605.4	43%	35072	740634
MED|J100|M10|R24|0.66	3544	165	3610.1	95%	71000	3104173
MED|J100|M10|R24|1.0	3267	144	3622.1	96%	86061	5375249
HAR|J10|M2|R4|0.33	432	432	1.2	0%	1647	5197
HAR|J10|M2|R4|0.66	376	376	0.9	0%	1647	6727
HAR|J10|M2|R4|1.0	406	406	3.3	0%	2120	11077
HAR|J25|M3|R6|0.33	1524	1524	390.5	0%	6467	48553
HAR|J25|M3|R6|0.66	1016	1016	95.0	0%	5873	56694
HAR|J25|M3|R6|1.0	1154	1154	3124.5	0%	9810	146894
HAR|J50|M5|R12|0.33	3097	2936	3604.1	5%	20106	207414
HAR|J50|M5|R12|0.66	3146	2942	3603.5	6%	29517	544857
HAR|J50|M5|R12|1.0	2817	2191	3605.7	22%	41020	1257885
HAR|J75|M7|R18|0.33	4618	3972	3603.5	14%	40786	656449
HAR|J75|M7|R18|0.66	4664	2497	3609.4	46%	60324	1903663
HAR|J75|M7|R18|1.0	5187	2371	3628.1	54%	93493	4262103
HAR|J100|M10|R24|0.33	7575	3197	3606.9	58%	63176	1327463
HAR|J100|M10|R24|0.66	6213	239	3634.6	96%	111208	4748111
HAR|J100|M10|R24|1.0	6039	198	3645.5	97%	153025	9458057
Avg	1993	931	2335	0	27617	1074485
Std dev	1838	1039	1691	0	33552	1863529

The feasible solution found by the heuristic for problem instance "HAR|J100|M10|R24|D1.0" was not further improved by the hybrid model in one hour due to the large number of constraints generated (9,458,057).

It is evident From Table 5-7 that the hybrid model Gap % follows a similar pattern to the optimization model where the gap grows as there are more jobs and the sharing density grows. The process times used to simulate the complexity do not make the gap worse since we used H' making the model tighter.

Table 5-7 Hybrid Model Gap% Summary

Jobs	Avg Gap %	DS (%)	Avg Gap%	Complexity	Avg Gap%
10	0%	0.33	19.7%	reduced	44.9%
25	0%	0.66	42.0%	medium	34.6%
50	28%	1.0	44.4%	hard	26.6%
75	57%				
100	91%	Total Gap 34%			

Optimal solutions were found for 18 out of the 45 scenarios in less than one hour for all the combinations of 10/25 job sizes. A Gap of 10% or less was found for 24 out of the 45 scenarios ranging from 10 jobs up to 75 jobs (i.e. MED|J75|M7|R18|0.33). Optimal solutions were found for 15 out of the 45 scenarios for the 10/25 job size instances in less than 600 seconds (10 minutes).

5.6. Best Integer Solution Comparison

Table 5-8 shows the best integer solution found for smaller size problem instances and the comparison among them. Optimal solutions were obtained in one hour or less for all the scenarios executed by the hybrid model. The optimization IP model alone shows a

Gap of 2% and 1% for scenarios “MED|J25|M3|R6|1.0” and “HAR|J25|M3|R6|1.0” when stopped after one hour. The best integer solution % delta was calculated as follows:

$1 - \frac{Opt}{Heur}$; $1 - \frac{Hybr}{Heur}$; $1 - \frac{Hybr}{Opt}$, respectively.

In summary, the heuristic does not add a lot of value for the small scenarios since the IP model is faster and obtains the same results.

Table 5-8 Best Solution Small Scenarios (10/25 Jobs)

	Best Integer Solution			**Solution Time (sec)**			**Best Integer Solution % Delta**		
Com|Jobs|Mach|Res|Den	Opt	Heur	Hybr	Opt	Heur	Hybr	Opt vs Heur	Hybr vs Heu	Hybr vs Opt
RED|J10|M2|R4|0.33	128	141	128	1	0	0	9%	9%	0%
RED|J10|M2|R4|0.66	110	137	110	0	0	0	20%	20%	0%
RED|J10|M2|R4|1.0	118	127	118	1	0	0	7%	7%	0%
RED|J25|M3|R6|0.33	464	556	464	46	1	43	17%	17%	0%
RED|J25|M3|R6|0.66	322	418	322	43	2	30	23%	23%	0%
RED|J25|M3|R6|1.0	341	427	341	693	3	609	20%	20%	0%
MED|J10|M2|R4|0.33	232	263	232	3	0	1	12%	12%	0%
MED|J10|M2|R4|0.66	197	227	197	1	0	0	13%	13%	0%
MED|J10|M2|R4|1.0	214	244	214	2	1	2	12%	12%	0%
MED|J25|M3|R6|0.33	824	985	824	185	1	232	16%	16%	0%
MED|J25|M3|R6|0.66	548	738	548	116	2	78	26%	26%	0%
MED|J25|M3|R6|1.0	619	721	616	3601	4	3045	14%	15%	0%
HAR|J10|M2|R4|0.33	432	497	432	1	1	1	13%	13%	0%
HAR|J10|M2|R4|0.66	376	437	376	2	0	1	14%	14%	0%
HAR|J10|M2|R4|1.0	406	477	406	9	1	3	15%	15%	0%
HAR|J25|M3|R6|0.33	1524	1865	1524	349	2	391	18%	18%	0%
HAR|J25|M3|R6|0.66	1016	1336	1016	328	4	95	24%	24%	0%
HAR|J25|M3|R6|1.0	1154	1349	1154	3602	4	3124	14%	14%	0%
Average							16%	16%	0%
Std Dev							5%	5%	0%

Table 5-9 shows the best integer solution found for medium size problem instances and the comparison among them. As expected, the hybrid model outperformed

the IP model when stopped after one hour except for three instances: RED|J50|M5|R12|0.66, MED|J75|M7|R18|0.33, HAR|J50|M5|R12|0.66. The IP model found a better solution due to the reduced time horizon provided by H' formulation and the heuristic initial solution not being as efficient. Additionally, four instances *|J50|M5|R12|0.33 and MED|J75|M7|R18|0.33 did not accept the initial solution of the heuristic since the H' formulation was more efficient than the C_{max} provided by the heuristic. Hence, the H' formulation was modified for these instances from $\theta = 2$ to $\theta = 3$ in order to extend the time horizon and accept the initial solution from the heuristic. The average for the best integer solution delta (%) is provided at the bottom of the table. The results show that the heuristic adds more value for the medium size scenarios than for small size scenarios.

Table 5-9 Best Solution Medium Scenarios (50/75 Jobs)

	Best Integer Solution			**Solution Time (sec)**			**Best Integer Solution % Delta**		
Com|Jobs|Mach|Res|Den	Opt	Heur	Hybr	Opt	Heur	Hybr	Opt vs Heur	Hybr vs Heu	Hybr vs Opt
RED|J50|M5|R12|0.33	991	1400	992	3601	3	3602	29%	29%	0%
RED|J50|M5|R12|0.66	937	1062	1062	3602	5	3601	12%	0%	-13%
RED|J50|M5|R12|1.0	1660	932	932	3601	12	3601	-78%	0%	44%
RED|J75|M7|R18|0.33	1451	1947	1454	3602	7	3602	25%	25%	0%
RED|J75|M7|R18|0.66	2231	1452	1452	3602	15	3602	-54%	0%	35%
RED|J75|M7|R18|1.0	2545	1593	1593	3603	30	3605	-60%	0%	37%
MED|J50|M5|R12|0.33	1718	2457	1724	3601	3	3602	30%	30%	0%
MED|J50|M5|R12|0.66	1704	2110	1696	3603	6	3604	19%	20%	0%
MED|J50|M5|R12|1.0	2813	1651	1492	3602	13	3602	-70%	10%	47%
MED|J75|M7|R18|0.33	2449	3333	2477	3601	6	3602	27%	26%	-1%
MED|J75|M7|R18|0.66	4398	2723	2723	3603	14	3604	-62%	0%	38%
MED|J75|M7|R18|1.0	4897	2720	2720	3605	23	3608	-80%	0%	44%
HAR|J50|M5|R12|0.33	3095	4612	3097	3602	4	3604	33%	33%	0%
HAR|J50|M5|R12|0.66	3109	3994	3146	3602	6	3604	22%	21%	-1%
HAR|J50|M5|R12|1.0	2846	3295	2817	3603	12	3606	14%	15%	1%
HAR|J75|M7|R18|0.33	4660	6244	4618	3602	7	3604	25%	26%	1%
HAR|J75|M7|R18|0.66	7792	4757	4664	3605	15	3609	-64%	2%	40%
HAR|J75|M7|R18|1.0	9639	5187	5187	3620	24	3628	-86%	0%	46%
Average							-18%	13%	18%
Std Dev							48%	13%	22%

Table 5-10 shows that for the large size (100 jobs) the heuristic initial solution was further improved by the hybrid model for only two out of 9 scenarios. That shows the heuristic becomes more attractive as the number of jobs increases above 100 jobs. In summary, the average best integer solution of the optimization IP model vs heuristic is -50% showing the heuristic adds more value as feasible solutions were found and the IP formulation failed to find efficient solutions within one hour.

Table 5-10 Best Solution Large Scenario (100 Jobs)

	Best Integer Solution			**Solution Time (sec)**			**Best Integer Solution % Delta**		
Com|Jobs|Mach|Res|Den	Opt	Heur	Hybr	Opt	Heur	Hybr	Opt vs Heur	Hybr vs Heu	Hybr vs Opt
RED|J100|M10|R24|0.33	3072	2366	2366	3602	10	3602	-30%	0%	23%
RED|J100|M10|R24|0.66	3087	2010	2010	3604	28	3605	-54%	0%	35%
RED|J100|M10|R24|1.0	3063	1930	1930	3610	48	3611	-59%	0%	37%
MED|J100|M10|R24|0.33	5683	4183	**3861**	3601	10	3605	-36%	8%	32%
MED|J100|M10|R24|0.66	5583	3544	3544	3606	30	3610	-58%	0%	37%
MED|J100|M10|R24|1.0	5461	3267	3267	3615	42	3622	-67%	0%	40%
HAR|J100|M10|R24|0.33	9582	7590	**7575**	3602	12	3607	-26%	0%	21%
HAR|J100|M10|R24|0.66	10442	6213	6213	3611	26	3635	-68%	0%	40%
HAR|J100|M10|R24|1.0	-	6039	6039	-	42	3645	-	0%	-
Average							-50%	1%	33%
Std Dev							17%	3%	7%

It is evident From Table 5-10 that the solutions found by the hybrid model outperformed the IP model for the large instances when a stop limit of one hour is used (i.e. optimization IP model has an average Gap of ~33% if the hybrid solution is used as the lower bound).

Overall, we believe these results should provide insight to industry practitioners to estimate solution time feasibility for real world complex problems requiring multiple setups and including ready times. These results provide insight to how the proposed time-indexed IP model, heuristic and hybrid models behaved for the 45 different problem instances. We believe the proposed models should allow practitioners to find optimal solutions for small and medium problem instances. This study also provides insight on how the time-index IP formulation can be applied to larger models if practitioners can transform 1 unit of time into y units of time per period t where $y > 1$. For example, if

processing times take anywhere from 100 to 150 minutes, these process times can be divided by 5 minutes to find a reduced time horizon which in turns reduces the computational complexity of the model. That is, this transformation allows the modeler to control computational complexity by keeping the time horizon from growing extremely large; hence, preventing the model from becoming increasingly complex. Lastly, practitioners may also modify time-indexed IP model constraint (Eq. 3) from an "=" sign to a "≤" sign as shown below in order to keep the model from growing extremely large and complex.

$$\sum_{m\in M}\sum_{r\in R}\sum_{t=r_j}^{H-p_j+1} x_{jmrt} \leq 1 \quad \forall\, j \in J$$

Additionally, a new constraint must be added in order to force at least *n* lots to be scheduled where *n* is approximately 50 jobs in order to obtain near optimal solutions in relatively short time by controlling the time horizon.

5.7. Chapter 5 Conclusions

This study proposed an exact time-indexed IP model and a heuristic algorithm (hybrid) for parallel machine scheduling with dual resources, ready times, and multiple setups. The proposed IP $P_m|s_{jk}, r_j, aux|\sum C_j$ model is an expansion of the model originally proposed by (Sousa and Wolsey, 1992) known as the time-index formulation. The original model was expanded to include multiple machines, auxiliary resources, and sequence-dependent setups (i.e. resource and resource travel time between machines). The heuristic was developed with the objective to find fast, feasible solutions that can be

provided to prime the time-indexed IP model in order to reduce the time horizon and provide an initial solution that could be further improved. Generating 32 sort variations per problem instance for the heuristic resulted in superior integer solutions than the IP time-bound model for medium and large problem instances.

The hybrid model (IP model with initial solution given from heuristic) found optimal solutions for 18 out of the 45 scenarios in less than one hour for all the problem instances of 10/25 jobs. A Gap of 10% or less was found for 24 out of the 45 scenarios ranging from 10 jobs up to 75 jobs (i.e. MED|J75|M7|R18|0.33). Optimal solutions were found for 15 out of the 45 scenarios for the 10/25 job size instances in less than 600 seconds (10 minutes). The heuristic initial solutions for the 45 problem instances were improved by the hybrid model by 2% to 33%. As expected, the "hard" scenario with 100 jobs was not solved in less than one hour due to the excessive number of constraints (9,458,057) which are mainly generated by equations (3) and (4). These equations prevent overlap among jobs, machines, and resources for each period and include both setups (i.e. resources and resource travel time). To our knowledge, the proposed IP formulation has not been published before nor has solutions for small (10/25 Jobs) size problems been solved to optimality in less than 10 minutes with this formulation.

5.8. Chapter 5 Summary

This chapter has addressed the minimization of total completion time for resource constrained parallel machine scheduling problem with job-machine eligibility restrictions with release dates denoted $P_m|r_j, M_j, aux\ 1|\sum C_j$. To our knowledge, this exact

formulation has not used to solve the photolithography scheduling problem with release dates with setups.

A time-indexed IP optimization model has been formulated to solve this problem for 45 test problem instances resulting from five problem sizes, three complexity levels and three machine eligibility restriction levels as presented in chapter 3. We vary the number of jobs, machines, resources, process times, release times and job-machine restrictions to determine how this model would behave in a real-world environment. IBM Cplex optimization engine was used to solve the problem with default settings.

This study proposes an exact time-indexed IP model and a heuristic algorithm (hybrid) for parallel machine scheduling with dual resources, ready times, and multiple setups. The proposed IP $P_m|s_{jk}, r_j, aux|\sum C_j$ model is an expansion of the model originally proposed by (Sousa and Wolsey, 1992) known as the time-index formulation. The original model was expanded to include multiple machines, auxiliary resources, and sequence-dependent setups (i.e. resource and resource travel time between machines). The heuristic was developed with the objective to find fast, feasible solutions that can be provided to prime the time-indexed IP model in order to reduce the time horizon and provide an initial solution that could be further improved.

The hybrid model (IP model with initial solution given from heuristic) found optimal solutions for 18 out of the 45 scenarios in less than one hour for all the problem instances of 10/25 jobs. A Gap of 10% or less was found for 24 out of the 45 scenarios ranging from 10 jobs up to 75 jobs (i.e. MED|J75|M7|R18|0.33). Optimal solutions were found for 15 out of the 45 scenarios for the 10/25 job size instances in less than 600 seconds (10

minutes). The heuristic initial solutions for the 45 problem instances were improved by the hybrid model by 2% to 33%. As expected, the "hard" scenario with 100 jobs was not solved in less than one hour due to the excessive number of constraints (9,458,057) which are mainly generated by equations (5.3) and (5.4). These equations prevent overlap among jobs, machines, and resources for each period and include both setups (i.e. resources and resource travel time). To our knowledge, the proposed IP formulation has not been published before nor has solutions for small (10/25 Jobs) size problems been solved to optimality in less than 10 minutes with this formulation.

In the future, we plan to further improve the heuristic that is currently providing "good" feasible solutions in a relative short amount of time by testing more variations beyond the 32 combinations per instance. We also plan to incorporate multiple objectives (multi-objective) allowing practitioners to use this formulation for real-world problems where there is a need to model machines with resources and setups. If setups were omitted, the run time would be reduced significantly as the number of variables and constraints would be reduced. We believe it would be interesting to solve the time-index IP formulation using a rolling horizon approach to maintain the time horizon short and efficient.

In conclusion, a time-index IP model with heuristic (hybrid) can add value to practitioners looking to solve practical scheduling problems with parallel machines, dual resources and sequence-dependent setups. It is extremely important to control problem instance size and time horizon to obtain near-optimal solutions within acceptable run

times. The proposed heuristic primes with the IP model to reduce the space search by reducing the time horizon and provide an initial solution to the optimization model.

CHAPTER 6 MOO RESOURCE CONSTRAINED PARALLEL MACHINE SCHEDULING MODEL

The focus of this study is to develop efficient model-based procedures for the scheduling of identical parallel machines with shared, constrained, auxiliary resources with sequence-dependent setups, machine eligibility restrictions, job release and due dates with single and multiple objectives. As previously stated, the need to effectively schedule finite resources exist across multiple industries including semiconductor fabrication and assembly, mold injection, and printed circuit board (PBC) assembly, amongst others. These industries often are subject to manufacturing bottlenecks, which reduce throughput and overall profit. Thus, improvements in modeling to get efficient scheduling solutions will have a large impact on those manufacturing systems subject to capacity constraints, especially if the system being optimized is the overall factory constraint.

This chapter explores the multi-objective resource constrained parallel machine scheduling model with sequence-dependent setups, machine eligibility restrictions, release and due dates with user interaction: $P_m|S_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$. This environment has a set of jobs *J* either ready to be processed ("ready jobs") or incoming to the system. These jobs can be processed in any of the unrestricted *M* parallel machines, if they are fitted with an instance *r* of the set of scarce auxiliary resources *R*, which are shared with other machines.

More specifically, the time indexed Integer programing (IP) multi-objective optimization (MOO) model is used to solve small problem instances with setups and medium size problem instances without setups. Prior to re-introducing key terms and formulations, it is noteworthy to highlight that to our knowledge, this work is the first one combining aforementioned parameters to solve problems found in the semiconductor industry.

Multi-objective optimization (MOO), also known as multi-criteria optimization, is concerned with decision making based on multiple criteria. Typically, mathematical optimization models involving more than one objective function are used in order to optimize *k* objective functions simultaneously. A key difference between MOO and single-objective problem is that MOO does not have a single optimal (best) solution. The Pareto method, which is one way to implement MOO, finds a feasible schedule that minimizes several objectives in such a way that no improvement can be made on one objective without degrading the other objective metric in the objective vector (Suresh and Mohanasundaram, 2006). According to Suresh et al. (2006) the user may generate a schedule with a weighted combination of several scheduling objectives as the performance measure. MOO models allow experts to choose a Pareto optimal solution according to the existing priorities when a decision needs to be made. In this case, a Pareto optimal set is to be found as a family of best trade-off schedules. The set of Pareto solutions is called the Pareto frontier (Suresh et al, 2006). In this chapter, we refer to the Pareto frontier as the partial or entire efficient frontier.

Phase III consists of a combination of scheduling models developed in phase I, phase II, and the addition of new multiple objectives using the Diversity Maximization Approach (DMA) model proposed by (Masin and Bukchin, 2008) to solve the MOO model $P_m|S_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$. The reader is referred to section 2.2.3 for detailed MOO and DMA definitions. However, a brief re-introduction to DMA MOO is provided next.

Masin and Bukchin (2008) define [P1**]** which finds the initial point $Y_e \in E$**:**

$$\textbf{[P1]}\ \min Z = \sum_{k\in K} w_k f^{(k)}(x)\ \text{ s.t } x \in X$$

where W_k *k = 1...W*are positive weights for k^{th} objective function. The subset *E* to be used to build the efficient frontier starts empty, then one point per iteration is added until a full efficient frontier is obtained or a pre-defined ε is provided.

$$\textbf{[P1']}\ \min Z = lex\min\left(\alpha, \sum_{k\in K} w_k f^{(k)}(x)\right)$$

$$\text{s.t } \min \alpha = max_{y_e \in E}\left(min_{1\le k\le W} \frac{f^{(k)}(x) - y_e^{(k)}}{\Delta_{ke}}\right)\ x \in X.$$

Then, the non-linear constraint α used as constraint for **P1'** is linearized using binary variables.

$$[\textbf{P2}]\ \min Z = lex\min\left(\alpha, \sum_{k\in K} w_k f^{(k)}(x)\right)$$

$$\text{s.t. } \alpha \ge \beta_e \forall y_e \in E,$$

$$\beta_e \le \frac{f^{(k)}(x) - y_e^{(k)}}{\Delta_{ek}} \forall\ y_e \in E, k = 1 \dots W,,$$

$$\beta_e = \sum_{k=1}^{W} \frac{\gamma_{2ke} - y_e^{(k)}\gamma_{1ke}}{\Delta_{ek}} \forall\ y_e \in E,$$

$$\sum_{k=1}^{W} \gamma_{1ke} = 1 \; \forall e \in E$$

$$\gamma_{2ke} \geq f^{(k)}(x) + (\gamma_{1ke} - 1)M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \leq f^{(k)}(x) + (1 - \gamma_{1ke})M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \leq \gamma_{1ke} M \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{1ke} \in \{0,1\} \; \forall y_e \in E, k = 1, \dots, W$$

$$\gamma_{2ke} \geq 0 \; \forall y_e \in E, k = 1, \dots, W$$

$$x \in X.$$

For this phase we study a pareto efficient frontier obtained by solving *k* objectives simultaneously which adds a layer of complexity to what is already a complex scheduling optimization model (NP-hard). The objective functions for this phase aim to minimize total weighted completion time $\sum w_j C_j$, total tardiness $[\sum T_j]$ and maximum tardiness T_{max} which allows manufacturers to target meeting on-time delivery customer commitments. The objective of this phase is to find optimal solutions for small instances and enable decision makers to interact with the model by editing or proposing schedules and/or add new constraints to drive the solutions according to his/her needs.

The model and the framework presented in this chapter represents a conceptual prototype, tested using simulated data since real-world data is not publicly available. Similarly, certain assumptions about the decision maker are made in finding a solution for the problem, such as the quality of the solution and when to stop the algorithm. Nonetheless, this model and framework mimic real world requirements which can easily be replicated using real data.

The framework proposed in this dissertation also aims to demonstrate how decision makers can interact with the model and learn from it, ultimately gaining trust and insight to make better strategic decisions with respect to policies bounding the options of these decision makers, i.e. multiple conflicting objectives).

As previously indicated, enabling human-model interaction has potential benefits. Ahn (2008) stated "human computation problems" are those large-scale computational problems that often cannot be solved by either computer or humans alone. The goal of this work is to harnesses human brainpower to solve complex problems that computers alone may not be able to solve in relatively short periods of time. That is, computers should enhance human intelligence instead of replacing it. Reinschmidt et al. (1990) and Framinan and Ruiz, (2010) state that there are potential benefits to the overall scheduling system if human experts are able to interact with the model, especially if the system is programmed in such a way that it is open and flexible so that expert knowledge can be captured and translated into objectives and constraints.

Another objective of this chapter is to show how to provide feedback to the decision maker, in terms of the associated consequences, if a solution—schedule in the form of a Gantt chart—is changed or overridden. Here, feedback will be given to the decision makers in the form of the Euclidian distance between every point in the efficient frontier, in terms of time units, and the new point associated with the new solution. A key contribution of this phase is the user-model interaction for the medium problem size instances.

Two user-model interaction scenarios are evaluated in this chapter using the RED|J50|M5|R12|1.0 problem instance. The first scenario aims to find an alternate schedule that reduces the weighted completion time $\sum w_j C_j$ for a given schedule that has one of the lowest maximum tardiness (T_{max}) while maintaining the best possible total tardiness [$\sum T_j$]. The second scenario consists of investigating how far the decision maker's edited schedule is (new point) from the efficient frontier using the Euclidian distance as the standard metric.

The rest of this chapter is subdivided as follows. Section 6.1 presents the MDA MOO mathematical model, section 6.2 presents the case study hierarchical model and process flow, section 6.3 present the experimental results for RED|J10|M2|R4|1.0, section 6.4 presents the experiment results for RED|J50|M5|R12|1.0, section 6.5 presents chapter conclusions, and section 6.6 provides a summary of this chapter. We proceed to describe the MOO resource constrained PMS mathematical formulation.

6.1. MOO Resource Constrained PMS Mathematical

$$\boldsymbol{P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})}$$ **Model**

The purpose of this model is to find multiple job-machine-resource schedules that are optimal with respect to the three objective functions previously described. The indices and sets along with the mathematical model are defined next.

Indices and Sets:

$j\ index\ for\ jobs, j \in J$

$m\ index\ for\ machines, m \in M$

$r\ index\ for\ auxiliary\ resource, r\ \in R$

$t\ index\ for\ time, t \in H$

Parameters:

$d_j\ due\ date\ for\ job\ j$

$r_j\ release\ date\ for\ job\ j$

$w_j\ priority\ weight\ for\ job\ j$

$W_k\ priority\ weight\ for$ obj function k

$p_j\ process\ time\ for\ job\ j$

$s_{ij}\ setup\ time\ between\ job\ i\ and\ j$

$tt_{m',m}\ auxiliary\ resource\ transfer\ time$

$between\ machine\ m'\ and\ machine\ m$

$H\ Set\ of\ time\ periods\ (total\ processing\ time\ plus\ maximum\ release\ date)$

$where\ H \geq \sum_{j \in J} p_j + \max\{r_j\}$

Decision variables:

x_{jrmt}

$= \begin{cases} 1\ if\ job\ j\ is\ assigned\ to\ machine\ m, resource\ r\ to\ start\ processing\ at\ time\ t \\ 0\ otherwise \end{cases};$

$C_j\ Completion\ time\ job\ j;\ to\ minimize\ weighted\ completion\ time \sum_{j \in J} w_j\ C_j$

$T_j\ Tardiness\ of\ job\ j\ where\ T_j = \max\{0, C_j - d_j\};\ to\ minimize\ total\ tardiness$

$\sum_{j \in J} T_j$

T_{max} Maximum $Tardiness$ among all $jobs$

Then, [P1] becomes the model used to find the initial point $Y_e \in E$ as follows**:**

[P1] $\min Z = W_1 f_1 + W_2 f_2 + W_3 f_3$ (6.1)

Subject To:

$$\sum_{j \in J} w_j C_j = f_1 \quad (6.2)$$

$$\sum_{j \in J} T_j = f_2 \quad (6.3)$$

$$T_{max} = f_3 \quad (6.4)$$

$$C_j - d_j \leq T_j \; \forall j \in J \quad (6.5)$$

$$T_j \leq T_{max} \; \forall j \in J \quad (6.6)$$

$$\sum_{m \in M} \sum_{r \in R} \sum_{t=r_j}^{H-p_j+1} (t + p_j - 1)\, x_{jmrt} \; = \; C_j \; \forall \, j \in J \quad (6.7)$$

$$\sum_{m \in M} \sum_{r \in R} \sum_{t=r_j}^{H-p_j+1} x_{jmrt} = 1 \; \forall \, j \in J \quad (6.8)$$

$$x_{jmrt} + \sum_{r' \in R} \sum_{t'=\max\left(r_i, t-p_i-tt_{m',m}+s_{r',r}+1\right)}^{\min(t, H-p_j)} x_{im'r't'} \leq 1 \quad (6.9)$$

$\forall \, i, j \in J \; i \neq j; m, m' \in M: m = m'; \; r \in R; \; t \; \in \{r_j \ldots H - p_j\}$

$$x_{jmrt} + \sum_{m' \in M} \sum_{t'=\max\left(r_i, t-p_i-tt_{m',m}+s_{r',r}+1\right)}^{\min(t, H-p_j)} x_{im'r't'} \leq 1 \quad (6.10)$$

$\forall \, i, j \in J \; i \neq j; m \in M; \; r, r' \in R: r = r'; \; t \; \in \{r_j \ldots H - p_j\}$

$x_{jrmt} \in \{0,1\}$, $C_j, T_j, T_{max}, f_k \; \in Z^+$. (6.11)

Equation (eq. 6.1) calculates the objective function to be minimized which is the sum of weighted function f_1, f_2, f_3 where these objective functions are defined as the first three constraints (eq. 6.2, 6.3 and 6.4), respectively. The first constraint eq. 6.2 calculates total weighted completion time. Equation 6.3 calculates total tardiness and equation 6.4 calculates maximum tardiness for all jobs. Equation 6.5 calculates tardiness for each job, equation 6.6 calculates maximum tardiness (T_{max}). Equation 6.7 calculates completion time for every job. Equation 6.8 enforces all jobs to be processed at one period *t*. Equation 6.9 and 6.10 enforce that at any given time *t* at most one job can be processed on a given machine and resource, respectively. Please note that in chapter 6 we plan to run one small instance including resource sequence-dependent setups and a medium size instance excluding setups due to the high complexity of the model that needs to run multiple times in order to find the full efficient frontier. The last constraint (eq 6.11) defines the binary variable indicating job *j* to start processing in period *t* at machine *m* with resource *r* and the non-zero integer variables needed to estimate completion time, tardiness, maximum tardiness, and each function. Final note, in order to run this model without setups equations 6.9 and 6.10 need to be replaced with Equations 4.4 and 4.5 previously introduced in chapter 4:

$$\sum_{j\in J}\sum_{r\in R}\sum_{t'=\max(r_j,t-p_j+1)}^{\min(t,H-p_j+1)} x_{jmrt'} \leq 1 \;\; \forall\, m \in M, t \;\in \{r_j \ldots H-p_j+1\}$$

$$\sum_{j\in J}\sum_{m\in M}\sum_{t'=\max(r_j,t-p_j+1)}^{\min(t,H-p_j+1)} x_{jmrt'} \leq 1 \;\; \forall\, r \in R, t \;\in \{r_j \ldots H-p_j+1\}$$

After P1 is solved the first efficient point is found; then, we proceed to iteratively solve model [P2] via the lexicographic MOO method as per Table 6-1

$$[\text{P2}]\ \min Z = lex\min(\alpha, W_1 f_1 + W_2 f_2 + W_3 f_3) \tag{6.12}$$

Subject To:

[P1] constraints Eq. 6.2-6.11

$$\alpha \geq \beta_e \forall y_e \in E \tag{6.13}$$

$$\beta_e \leq \frac{f^{(k)}(x) - y_e^{(k)}}{\Delta_{ek}} \forall\ y_e \in E, k = 1 \ldots W \tag{6.14}$$

$$\beta_e = \sum_{k=1}^{W} \frac{\gamma_{2ke} - y_e^{(k)} \gamma_{1ke}}{\Delta_{ek}} \forall\ y_e \in E, \tag{6.15}$$

$$\sum\nolimits_{k=1}^{W} \gamma_{1ke} = 1\ \forall e \in E \tag{6.16}$$

$$\gamma_{2ke} \geq f^{(k)}(x) + (\gamma_{1ke} - 1)M\ \forall y_e \in E, k = 1, \ldots, W \tag{6.17}$$

$$\gamma_{2ke} \leq f^{(k)}(x) + (1 - \gamma_{1ke})M\ \forall y_e \in E, k = 1, \ldots, W \tag{6.18}$$

$$\gamma_{2ke} \leq \gamma_{1ke} M\ \forall y_e \in E, k = 1, \ldots, W \tag{6.19}$$

$$\gamma_{1ke}\ \in \{0,1\}\ \forall y_e \in E, k = 1, \ldots, W \tag{6.20}$$

$$\gamma_{2ke} \geq 0\ \forall y_e \in E, k = 1, \ldots, W \tag{6.21}$$

Where M is a large number, equations (6.14-6.16) ensure that $\beta_{e=} min_{1 \leq k \leq W} \left(f^{(k)}(x) - y_e^{(k)} / \Delta_{ke} \right)$ and equations (6.17-6.19) ensure that $\gamma_{2ke} = f^{(k)}(x) \gamma_{1ke}$.as defined by Masin and Bukchin (2008). The A-DMA algorithm is used to find the efficient frontier.

Table 6-1: A-DMA Algorithm (Masin and Bukchin (2008))

Step	Action
	Solve [P1]
	Let $y^* = f(x^*)$ be optimal value
1	Set $E=\{y^*\}$
	Select $\varepsilon \geq 0$ and/or $sample\ size$
	Solve [P3]
2	Let $y^* = f(x^*)$ be optimal value
	IF $\alpha(y^*) < -\varepsilon$ or $Npoints \leq Threshold$ THEN
	$E = E \cup y^*$
	Go to step 2**
3	ELSE
	Stop;
	END IF

Selecting the right value of ε may not be straightforward in practice; hence, we propose to select an acceptable run time or number of efficient points as stopping criteria. The main advantage of the DMA algorithm is that the first few points are computational friendly and relatively easy to obtain (run time is fast) and those points are guaranteed to be as diverse as possible which provides a good representation of the extreme points in the efficient frontier. Thus, the entire efficient frontier may not be needed in practice nor practical for the decision maker to test different hypothesis. The following section describes the process flow to be used in this chapter for the model-user interaction case study.

6.2. Case Study Hierarchical Model and Process Flow

This subsection aims to explain the proposed process flow to show how a decision maker may interact with the model and find the best schedule that meets the decision maker's objectives

1. Solve job, machine, and auxiliary resource schedules using DMA-MOO IP for small and medium size problem instances.
2. Let the decision maker review all schedules.
3. The preferred schedule is selected, and the decision maker changes some job sequences.
4. Re-run MOO model incorporating decision maker's desires in the form of new constraint(s).
5. A new schedule and the distance of the new point to the efficient frontier is provided to the decision maker.

Figure 6-1 presents the proposed flow diagram of the decision methodology just described for the decision maker to interact with the model and obtain quantitative feedback on the quality of the new solution (job-machine-resource schedule) by providing the minimum distance between the new point and each efficient point in the Pareto frontier.

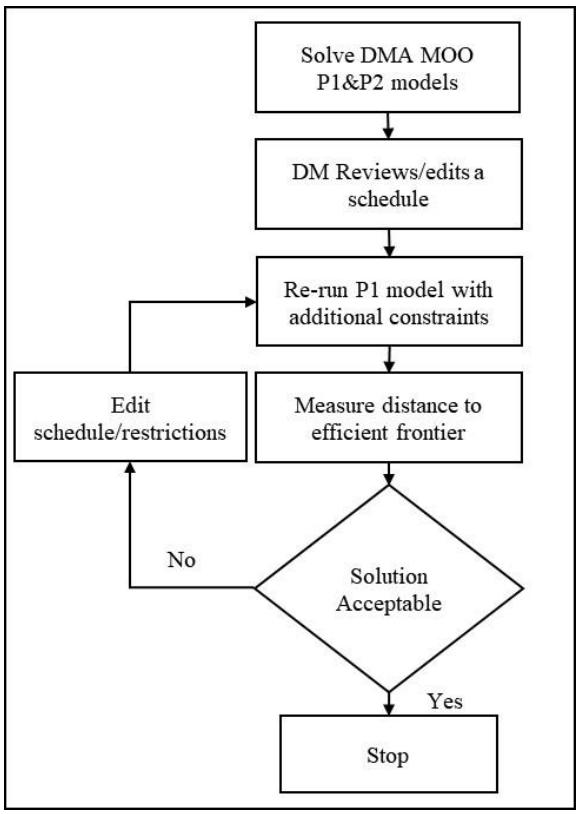


Figure 6-1: DMA MOO Flow Diagram

The proposed framework is strategic at this point since medium or large problem instances may run for several hours or days in order to run [P1] model followed by multiple [P2] models in order to find *n* efficient points using DMA MOO algorithm. The run time is directly related to the problem instance complexity and size. The algorithm is storing the efficient point and the job-machine-resource schedule presented in the form of a Gant Chart.

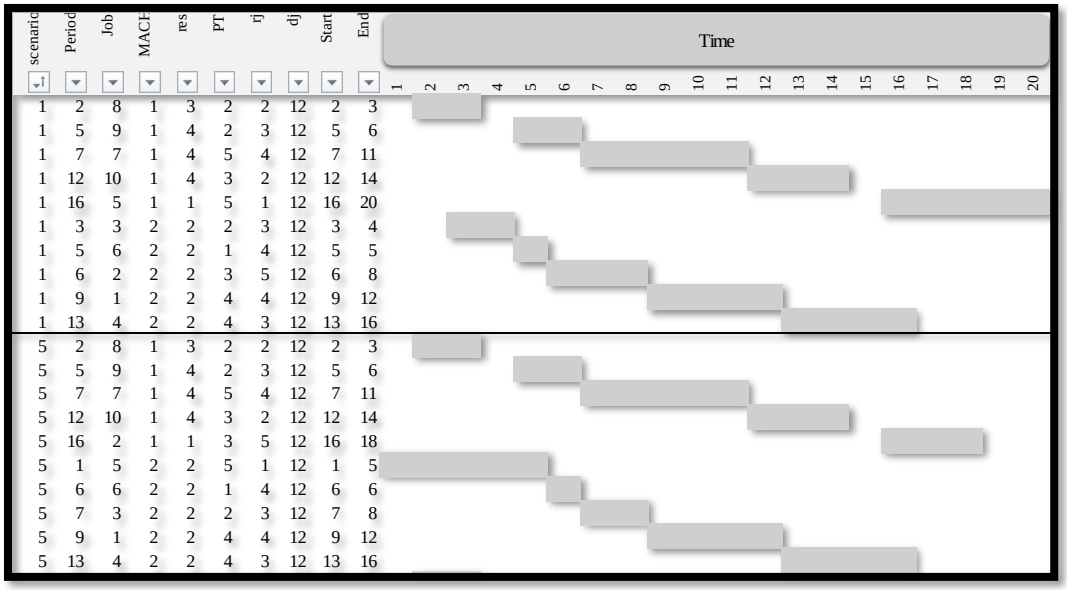


Figure 6-2: Example of Gantt for Job-Machine-Resource Schedule

The example depicted in Figure 6-2 shows two schedules labeled scenario 1 and 5, each job is assigned to start running at period *t* in machine *m* and resource *r*. The decision maker reviews the different schedules and selects one. The preferred schedule is modified by removing a partial set of jobs without impacting the rest of the jobs). Then, [P1] model runs again considering a new constraint (eq. 6.22) which ensures that a set of pre-assigned jobs (solutions) denoted by φ are enforced by the model.

$$x_{\text{jmrt}} = 1 \; \forall \; \text{jmrt} \in \varphi \qquad (6.22)$$

Then, the decision maker obtains an edited schedule and gets feedback on how far from optimal it is using the Euclidian distance for this point with respect to all other points in the efficient frontier. In this simulated case study, T_{max} is not acceptable to the decision maker. Hence, equation 6.4 $T_{max} = f_3$is modified as follows:

$$T_{max} \leq \tau \qquad (6.23)$$

Where τ is a user-defined threshold (i.e. same unit times as per the objective function). Then, the model runs for a second time with user defined parameters.

Let us assume that the new optimal solution is acceptable to the decision maker. Since there are many possible ways on how the decision maker can interact with a complex scheduling system, we limit our case study to "keep/delete" jobs from a previously generated optimal schedule. The experimental results are presented next.

6.3. Phase III Experimental Results for RED|J10|M2|R4|1.0

The experimental results for the $P_m|S_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$ model with a small problem instance RED|J10|M2|R4|1.0 is presented in Table 6-2. The smallest problem instance was selected in order to understand how the MOO-MDA model would respond to the resource constrained parallel machine scheduling with setups. The entire efficient frontier is found in a relatively short period of time (~8 minutes) and only 9 points were enough to obtain the full frontier.

Table 6-2: MOO-MDA Efficient Frontier for RED|J10|M2|R4|1.0

Point	WCompTime	Tardiness	T_{max}	alpha
1	260	14	8	-
2	355	13	5	-3.0000
3	295	12	7	-1.0000
4	277	15	6	-0.3333
5	302	12	6	-0.3333
6	284	14	7	-0.1157
7	268	18	7	-0.0947
8	274	17	7	-0.0315
9	275	16	7	-0.0210

The same points are depicted in Figure 6-3 where a 3-D graph represents the entire efficient frontier for completion time, tardiness, and maximum tardiness. One can conclude that for this specific problem instance the completion time and T_{max} have a trade-off. The shorter the weighted completion time, the higher the T_{max}.

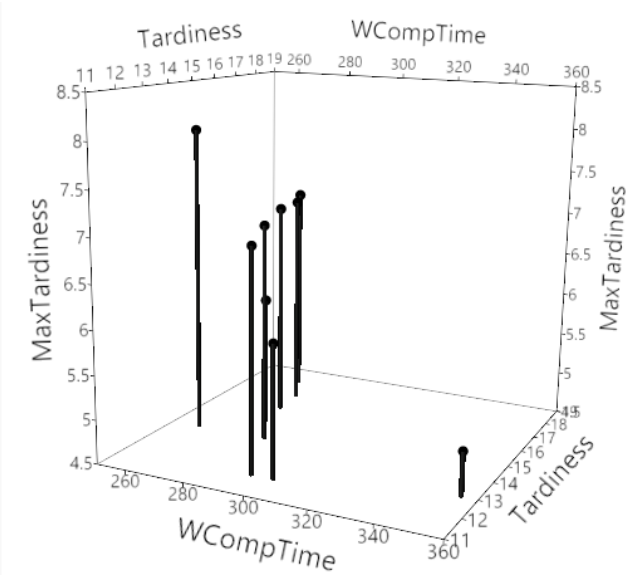


Figure 6-3: 3-D Plot for J10 Efficient Frontier

Figure 6-3 clearly depicts that the run time for a small instance problem the solution time grows linear with one point (i.e. 9) running slower than the rest (82 seconds).

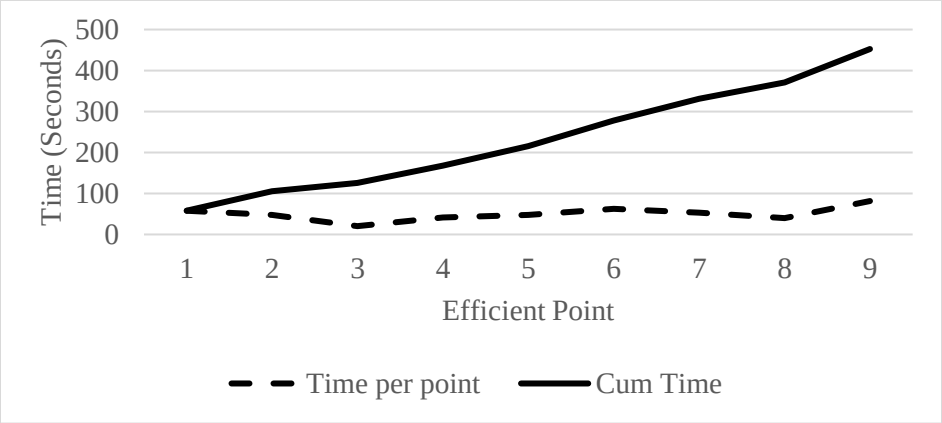


Figure 6-4: Solution Time for RED|J10|M2|R4|1.0

6.4. Experimental Resuts for RED|J50|M5|R12|1.0 with User Interaction

This subsection presents the experimental results for the medium size problem instance RED|J50|M5|R12|1.0 omitting sequence-dependent setups in order to obtain 30 or less points in approximately one week. A medium-size problem instance with multiple objectives and sequence dependent setups would not be able to run in one week which is an extremely long time to wait for results in tactical or strategic decision-making.

Previously described, two scenarios are evaluated in this subsection using the RED|J50|M5|R12|1.0 problem instance. The first scenario consists of investigating how far the decision maker's edited schedule is from the efficient frontier using the Euclidian distance for a standard metric. The second scenario aims to find an alternate schedule that reduces the maximum tardiness (T_{max}) while maintaining the best possible $\sum w_j C_j$ and total tardiness $[\sum T_j]$. Figure 6-1 shows the flow diagram.

The medium size problem instance was selected to better understand how the decision maker could learn how multiple conflicting objectives interact with each other and to demonstrate how a user can interact the model as previously stated.

For this case, even though setups were omitted, finding a solution for the model took long time (183 hrs.) and found 29 points of the efficient frontier. Table 6-3 shows the results found by the proposed model from chapter 5 combined with the MOO-MDA $P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j\ , \sum T_j, T_{max})$ model. In this chapter, weighted completion time was used instead of completion time in order to get a richer efficient frontier.

Table 6-3: Partial Efficient Frontier for RED|J50|M5|R12|1.0

Point	WCompTime	Tardiness	T_{max}	Alpha
1	3233	91	13	-
2	3885	22	7	-69.0000
3	3271	102	7	-0.9417
4	3420	45	8	-0.7125
5	3400	60	5	-0.5000
6	3612	30	5	-0.3750
7	3247	83	11	-0.2347
8	3461	41	6	-0.2316
9	3255	84	9	-0.2224
10	3278	68	16	-0.1871
11	3280	68	14	-0.1818
12	3284	68	12	-0.1779
13	3285	87	7	-0.1764
14	3287	69	10	-0.1733
15	3294	71	8	-0.1625
16	3317	58	14	-0.1250
17	3320	58	12	-0.1227
18	3322	59	10	-0.1196
19	3322	62	8	-0.1125
20	3532	32	13	-0.1125
21	3333	77	6	-0.1028
22	3395	51	6	-0.1012
23	3546	32	11	-0.1012
24	3528	33	9	-0.1000
25	3984	22	5	-0.1000
26	3543	33	8	-0.0919
27	3311	69	7	-0.0909
28	3421	52	5	-0.0909
29	3459	44	5	-0.0909

Figure 6-5 presents the partial efficient frontier for the 29 points (i.e. schedules) found by the MOO MDA algorithm.

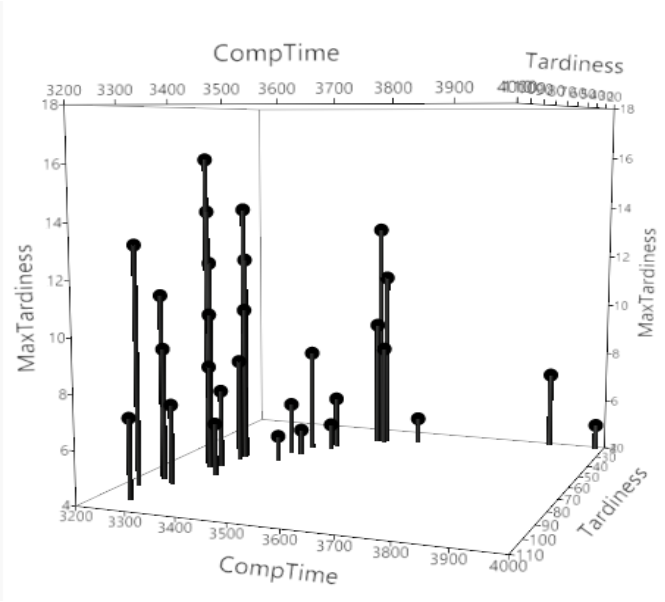


Figure 6-5: 3-D Plot for J50 Efficient Frontier

It is evident that the largest tradeoff this for this example is between completion time and tardiness. That is, schedules that end up with shorter completion time results in jobs with higher tardiness. The biggest drawback to obtain these results is that the model ran for over one week (183 hrs) which is not acceptable in industry to solve practical strategic problems.

Figure 6-6 shows the run time for each point in the efficient frontier and the cumulative run time required to find the full set of points. It is visible in the graph that those points after point 19 took significantly higher time to solve (i.e. 10+ hrs).

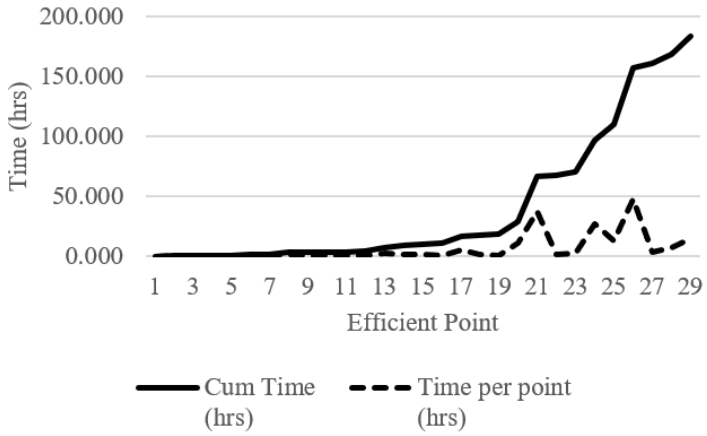


Figure 6-6: Solution Time for RED|J50|M5|R12|1.0

Table 6-4 presents run time in hours for each point and the cumulative run time graphed in Figure 6-6. As previously mentioned, not many practitioners would be willing to wait one week to interact with the model and make decisions. Since the algorithm used guarantees to find the most diverse frontier, we pose the following question: Is there a point where we can learn enough from the partial frontier that every point is adding less value and increasingly extending the run time?

Table 6-4: Run Time for RED|J50|M5|R12|1.0

Point	Time per point (hrs)	Cum Time (hrs)
1	0.0033	0.0033
2	0.0989	0.1022
3	0.0321	0.1343
4	0.2016	0.3359
5	0.3253	0.6612
6	0.6942	1.3555
7	0.4219	1.7774
8	1.1865	2.9639
9	0.1129	3.0767
10	0.1681	3.2448
11	0.4217	3.6666
12	0.6088	4.2754
13	2.6353	6.9107
14	1.7587	8.6694
15	1.4343	10.1038
16	0.5324	10.6362
17	5.5163	16.1525
18	1.7826	17.9351
19	0.4570	18.3921
20	10.8920	29.2841
21	36.8548	66.1389
22	1.7237	67.8626
23	2.2799	70.1425
24	26.8327	96.9752
25	13.0967	110.0720
26	47.4828	157.5547
27	3.7707	161.3254
28	7.4193	168.7447
29	14.4476	183.1922

The answer to question above is yes, the decision maker could obtain a good representation of the partial frontier with only 12 points as depicted in Figure 6-7 (right hand side). Table 6-4 shows that it took 4.2 hours to obtain the partial representaiton depicted in Figure 6-7 (right hand side) vs. 183 hrs for the 3-D plot depicted Figure 6-7 (left hand side).

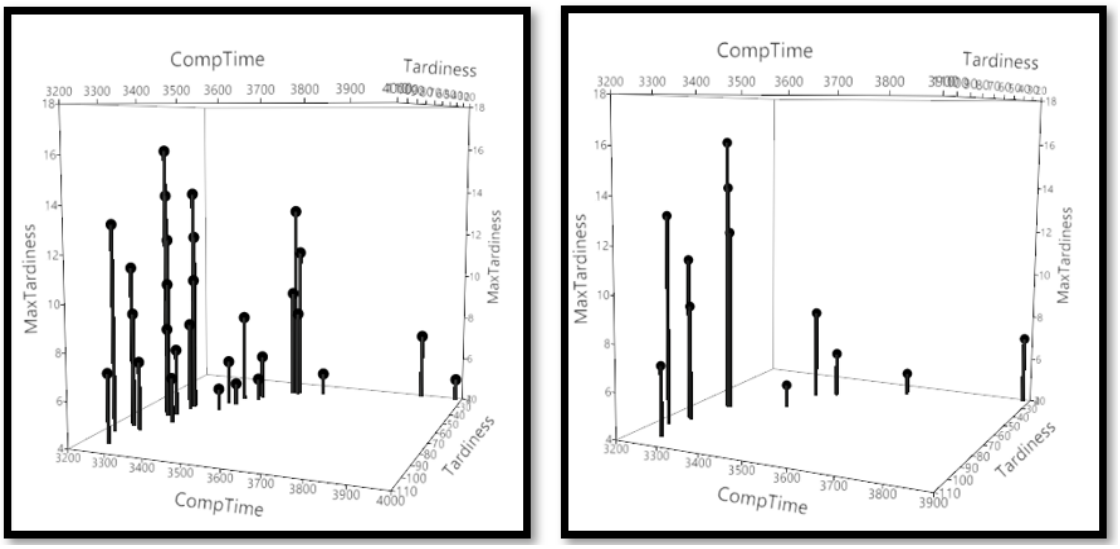
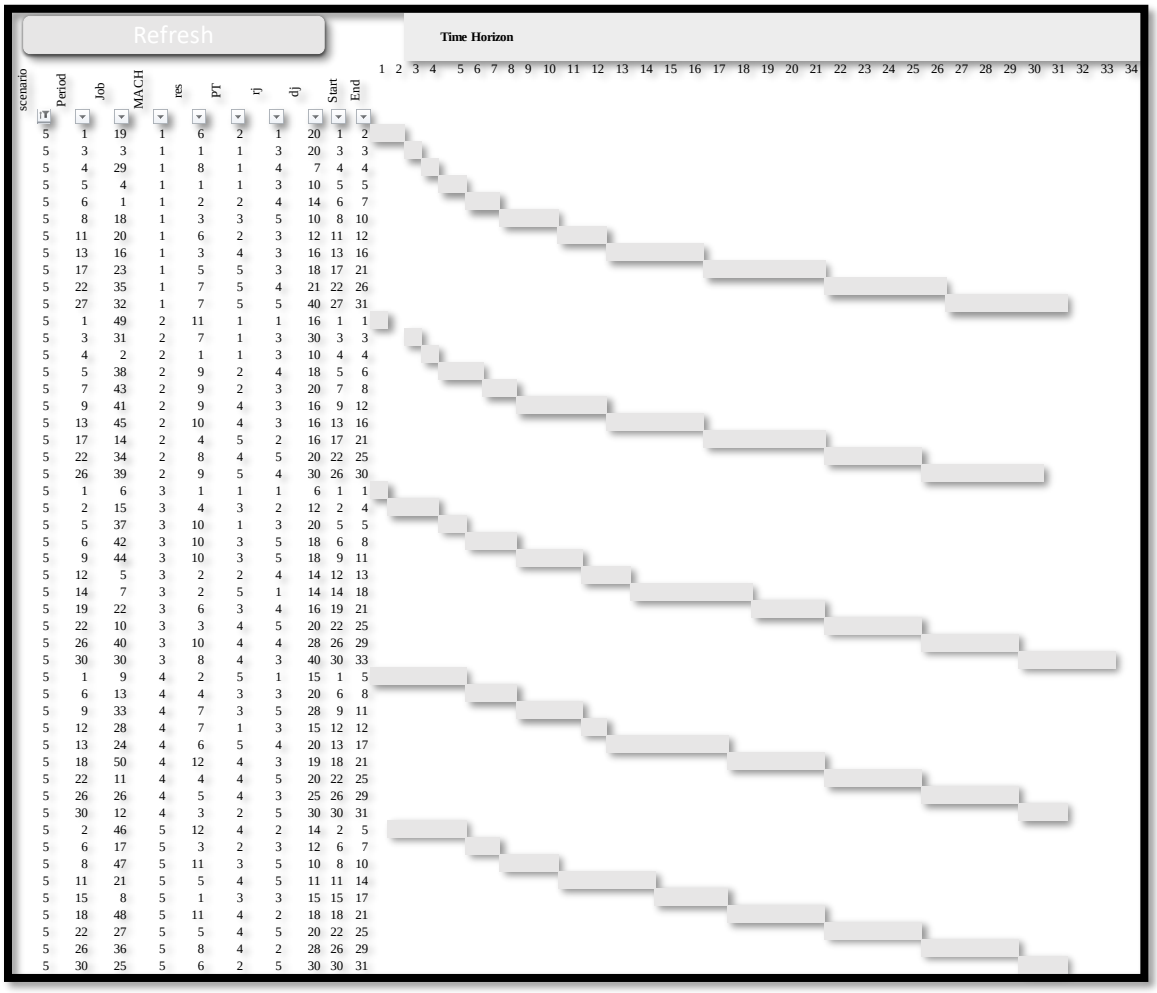


Figure 6-7: 183-hr Partial Efficient Frontier vs. 4-hr Partial Efficient Frontier

It is noteworthy to mention that each point of the pareto was validated by running the [P1] model and fixing two objectives (Tardiness and T_{max}) in order to solve for the weighted completion time. If the solution matched the values provided by the partial efficient frontier, we concluded that the model worked as intended.

Up to this point P1 and P2 models found 29 optimal alternate schedules with respect to the three objective functions previously introduced. For the first case study the decision maker selects one schedule that meets his/her objectives. Then, the decision maker deletes all jobs and leaves the top 10 jobs in the original position. Then, the model runs again. Assuming the decision maker is not happy with the results, a new set of restrictions (Eq 6.22) are added to the model in order to guide the new solution. Once the new optimal schedule is obtained, feedback is provided to the decision maker in the form of Euclidean distance between the new point and the efficient frontier.

Figure 6-8 shows one of 29 schedules found. Let us assume the decision maker selected this scenario (i.e. schedule) as the best solution since T_{max} =5 is one of the lowest.



scenario	Period	Job	MACH	res	PT	rj	dj	Start	End
5	1	19	1	6	2	1	20	1	2
5	3	3	1	1	1	3	20	3	3
5	4	29	1	8	1	4	7	4	4
5	5	4	1	1	1	3	10	5	5
5	6	1	1	2	2	4	14	6	7
5	8	18	1	3	3	5	10	8	10
5	11	20	1	6	2	3	12	11	12
5	13	16	1	3	4	3	16	13	16
5	17	23	1	5	5	3	18	17	21
5	22	35	1	7	5	4	21	22	26
5	27	32	1	7	5	5	40	27	31
5	1	49	2	11	1	1	16	1	1
5	3	31	2	7	1	3	30	3	3
5	4	2	2	1	1	3	10	4	4
5	5	38	2	9	2	4	18	5	6
5	7	43	2	9	2	3	20	7	8
5	9	41	2	9	4	3	16	9	12
5	13	45	2	10	4	3	16	13	16
5	17	14	2	4	5	2	16	17	21
5	22	34	2	8	4	5	20	22	25
5	26	39	2	9	5	4	30	26	30
5	1	6	3	1	1	1	6	1	1
5	2	15	3	4	3	2	12	2	4
5	5	37	3	10	1	3	20	5	5
5	6	42	3	10	3	5	18	6	8
5	9	44	3	10	3	5	18	9	11
5	12	5	3	2	2	4	14	12	13
5	14	7	3	2	5	1	14	14	18
5	19	22	3	6	3	4	16	19	21
5	22	10	3	3	4	5	20	22	25
5	26	40	3	10	4	4	28	26	29
5	30	30	3	8	4	3	40	30	33
5	1	9	4	2	5	1	15	1	5
5	6	13	4	4	3	3	20	6	8
5	9	33	4	7	3	5	28	9	11
5	12	28	4	7	1	3	15	12	12
5	13	24	4	6	5	4	20	13	17
5	18	50	4	12	4	3	19	18	21
5	22	11	4	4	4	5	20	22	25
5	26	26	4	5	4	3	25	26	29
5	30	12	4	3	2	5	30	30	31
5	2	46	5	12	4	2	14	2	5
5	6	17	5	3	2	3	12	6	7
5	8	47	5	11	3	5	10	8	10
5	11	21	5	5	4	5	11	11	14
5	15	8	5	1	3	3	15	15	17
5	18	48	5	11	4	2	18	18	21
5	22	27	5	5	4	5	20	22	25
5	26	36	5	8	4	2	28	26	29
5	30	25	5	6	2	5	30	30	31

Figure 6-8: Schedule for Efficient Point Five

However, the total weighted completion time for scenario 5 is high when compared to the other 29 scenarios. Thus, the decision maker removes all jobs except the first 19 jobs which are left in the same sequence aiming to find a new schedule with

lower weighted completion time. The 19 jobs left in the same sequence are depicted in Table 6-5.

Table 6-5: Set of Job-Machine-Resource Pre-assigned at Time *t*

Scenario	Time	Job	Mach	Res	p_j	r_j	d_j	Start	End
5	1	19	1	6	2	1	20	1	2
5	1	49	2	11	1	1	16	1	1
5	1	6	3	1	1	1	6	1	1
5	1	9	4	2	5	1	15	1	5
5	2	15	3	4	3	2	12	2	4
5	2	46	5	12	4	2	14	2	5
5	3	3	1	1	1	3	20	3	3
5	3	31	2	7	1	3	30	3	3
5	4	29	1	8	1	4	7	4	4
5	4	2	2	1	1	3	10	4	4
5	5	4	1	1	1	3	10	5	5
5	5	38	2	9	2	4	18	5	6
5	5	37	3	10	1	3	20	5	5
5	6	1	1	2	2	4	14	6	7
5	6	42	3	10	3	5	18	6	8
5	6	13	4	4	3	3	20	6	8
5	6	17	5	3	2	3	12	6	7
5	7	43	2	9	2	3	20	7	8
5	8	18	1	3	3	5	10	8	10

After running the model, the edited schedule provides an improved weighted completion time of 3235 units which is lower than the original 3400. Tardiness and T_{max} resulted in 91 and 14-time units, respectively. However, T_{max} is not acceptable to the decision maker since it exceeded 10-time units. Hence, a maximum threshold τ of 10 hrs. is imposed on the objective function T_{max} using the following equation:

$$T_{max} \leq 10$$

After the model runs again a new alternate schedule is obtained. Let us assume the decision maker is satisfied with the new results since weighted completion time is 3240-time units, Tardiness 94 and T_{max} 10. Figure 6-9 depicts the final edited schedule resulting from the user-model interaction.

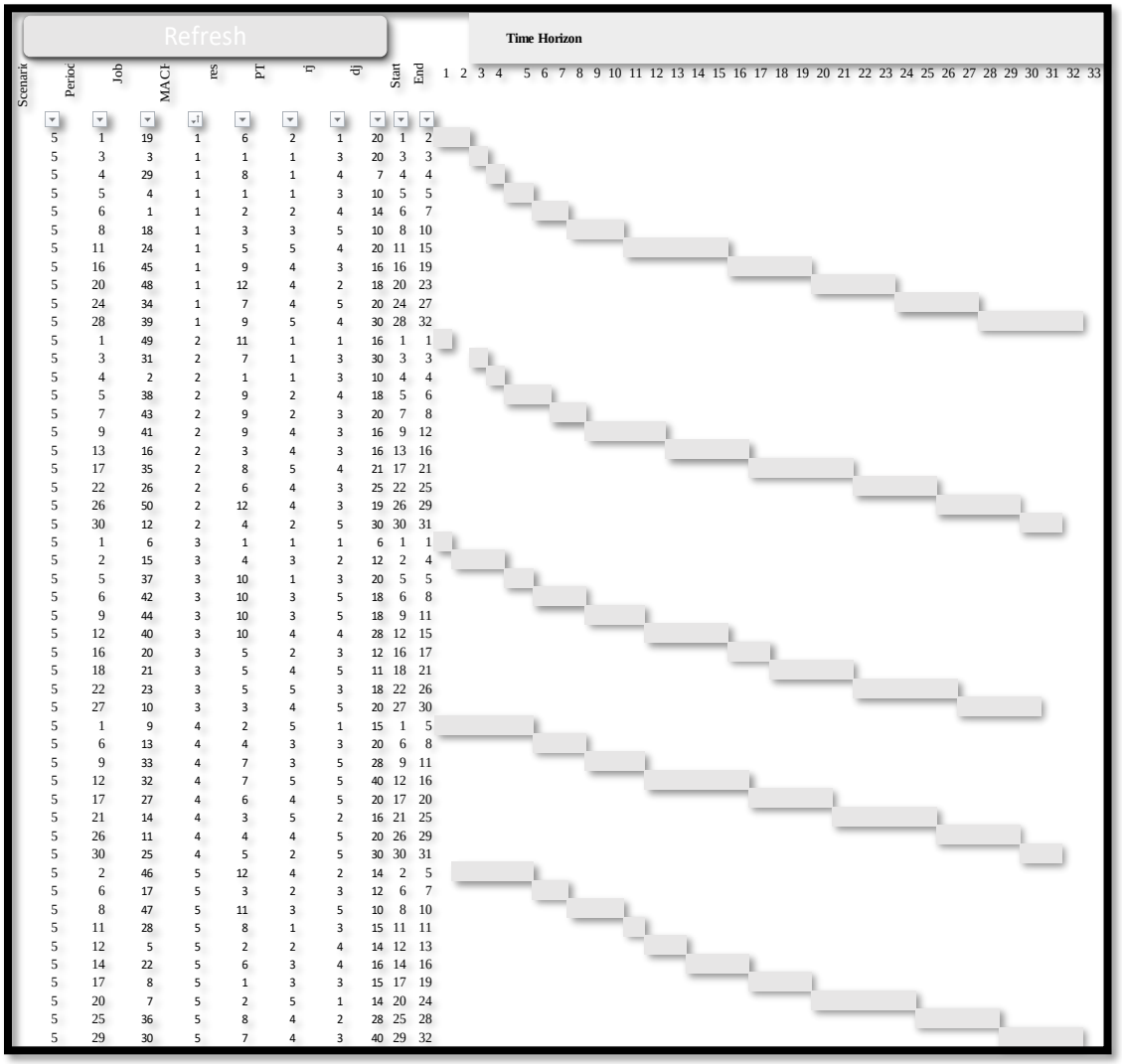


Figure 6-9: User-Model Interaction Resulting Schedule

Figure 6-10 depicts the partial efficient frontier and the new point (red asterisk) which is the resulting schedule obtained by the user (decision maker)-model interaction.

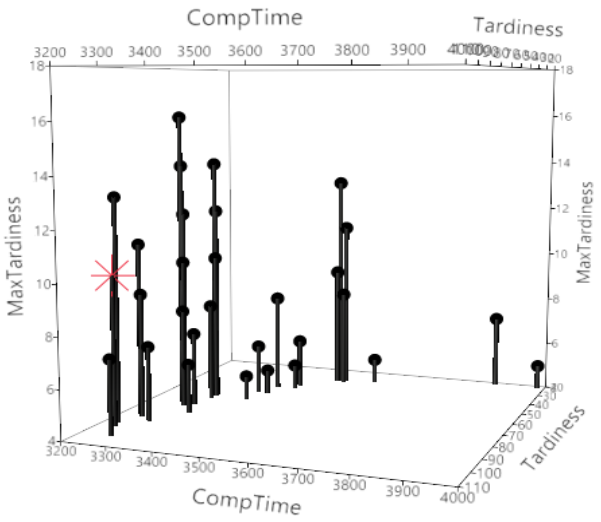


Figure 6-10: Efficient Frontier with New Point (edited solution)

For the second test case, the decision maker gets feedback on the quality of the solution that resulted from editing one of the schedules. As previously stated, the quality of the solution is measured by calculating the distance between the new point (edited schedule) and each point in the efficient frontier using the Euclidean distance. In this example, point 1 resulted to be the closest efficient point to the new point (red asterisk) based on the distance of 8.18-time units (See Table 6-6).

Table 6-6: Euclidian Distance Between New Point and Efficient Frontier

Point	WCompTime	Tardiness	T_{max}	alpha	Distance between Eff Point and New Point
1	**3233**	**91**	**13**	**-**	**8.18**
2	3885	22	7	69.0000	649.01
3	3271	102	7	-0.9417	32.16
4	3420	45	8	-0.7125	186.56
5	3400	60	5	-0.5000	163.65
6	3612	30	5	-0.3750	377.50
7	3247	83	11	-0.2347	13.08
8	3461	41	6	-0.2316	227.30
9	3255	84	9	-0.2224	18.06
10	3278	68	16	-0.1871	46.43
11	3280	68	14	-0.1818	47.87
12	3284	68	12	-0.1779	51.15
13	3285	87	7	-0.1764	45.64
14	3287	69	10	-0.1733	53.24
15	3294	71	8	-0.1625	58.73
16	3317	58	14	-0.1250	85.09
17	3320	58	12	-0.1227	87.75
18	3322	59	10	-0.1196	89.16
19	3322	62	8	-0.1125	88.05
20	3532	32	13	-0.1125	298.52
21	3333	77	6	-0.1028	94.63
22	3395	51	6	-0.1012	160.90
23	3546	32	11	-0.1012	312.22
24	3528	33	9	-0.1000	294.39
25	3984	22	5	-0.1000	747.49
26	3543	33	8	-0.0919	309.09
27	3311	69	7	-0.0909	75.33
28	3421	52	5	-0.0909	185.88
29	3459	44	5	-0.0909	224.69
New Point	**3240**	**94**	**10**	**-**	
Min Dist					**8.19**

At this point, let us assume the decision maker can set different weights and/or restriction to this model but using data, he/she can interactively learn about the trade-off between multiple competing objectives and the cost associated with user overrides.

6.5. Chapter 6 Conclusions

The MDA MOO is a powerful methodology that allows researchers find the most diverse Pareto frontier. If it is not practical to wait for all the points in the efficient frontier, the decision maker can stop the algorithm after *t* units of time or after alpha reaches a threshold. The most useful, key component of the MDA MOO algorithm is that the first few points are the most diverse, which represent the extreme points on the frontier. There is a potential to fit hyperplanes through the partial efficient frontier to project the full frontier, this proposal will be listed as future research.

On the other hand, one of the shortcomings of the proposed methodology is the long run time observed for medium size problem instance for the scheduling of identical parallel machines with shared, constrained, auxiliary resources, machine eligibility restrictions, job release and due dates with three objectives (weighted completion time, tardiness and maximum tardiness). The proposal is to find a partial efficient frontier and fix a model to stop after *n* hrs. since it was evident the complexity and run time grew exponentially as the number of points increased for the J50|M5|R12|1.0 problem instance.

6.6. Chapter 6 Summary

This chapter has addressed the minimization of weighted completion time, total tardiness, maximum tardiness (T_{max}) for the scheduling of identical parallel machines

with shared, constrained, auxiliary resources with sequence-dependent setups, machine eligibility restrictions, job release and due dates with single and multiple objectives denoted as $P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$. To our knowledge, this exact formulation has not been used to solve the photolithography scheduling problem in the semiconductor industry.

A time-indexed IP optimization model has been formulated to solve the scheduling problem for the small problem instance RED|J10|M2|R4|1.0 and the medium size problem instance RED|J50|M5|R12|1.0. It is noteworthy that for the medium size problem instance sequence-dependent setups were omitted to speed up the model which took 183 hrs. to find 29 efficient points (schedules).

In summary, this chapter resulted in an exact IP model capable of finding 9 optimal solutions (entire efficient frontier) for the small size problem instance in 8 minutes. The computational complexity of this model behaved in a linear fashion. For the medium size problem instance, the model found 29 points in 183 hrs. and the computational complexity of this model grew exponentially. The first 12 points were obtained in 4.2 hrs; however, after the 12th point the time per solution was as high as 47 hrs. for a single schedule.

The MOO MDA model can be used for strategic decision making for small size problem instances and for medium size problem instances with time-based stop criteria. We believe it would be interesting to solve the time-index IP formulation using a rolling horizon approach to maintain the time horizon short and efficient. For future research it

would be interesting to add constraint programming (CP) to MOO model and a heuristic to potentially speed up the results.

CHAPTER 7 CONCLUSIONS AND FUTURE RESEARCH

This chapter is subdivided in three sections, section 7.1 presents a brief dissertation summary, section 7.2 presents the conclusions and contributions, and section 7.3 presents future research.

7.1. Dissertation Summary

When manufacturing capacity becomes the lead constraint in revenue and profit generation, efficient execution of the production systems can make the difference between a thriving and a failing enterprise. This is especially true in the wafer fabrication lines of the semiconductor industry due to high equipment costs and the highly complex process.

In this dissertation, we advocate for the use of deterministic scheduling theory for the design and development of more efficient scheduling strategies to serve as alternatives to the current methods in order to increase the utilization and the capacity of manufacturing systems.

The focus of this dissertation is to develop efficient model-based procedures for the scheduling of identical parallel machines with shared, constrained, auxiliary resources with sequence-dependent setups, machine eligibility restrictions, job release and due dates with single and multiple objectives. Increasing throughput at the factory lead-constraint will increase the overall factory throughput dictated by the new constraint. Improved overall factory throughput will directly correlate to increased capacity of the manufacturing system, which is the objective of system optimization.

We developed a single objective time-indexed IP model and a multi-objective (MOO) time-indexed IP model that resemble practical manufacturing systems. We also

developed a heuristic capable of solving large problems to collaborate with the time-indexed IP models by providing feasible initial solution under one minute of run time. The proposed models can be defined as $P_m|S_{jk}, r_j, M_j, aux\ 1|\sum C_j$ and $P_m\left|S_{jk}, r_j, d_j, M_j, aux\ 1\right|\sum w_j C_j\ , \sum T_j, T_{max}$

7.2. Conclusions and Contributions

In this study, we discussed the advantages and disadvantages of various proposed modeling approaches for the multi-objective optimization (MOO) parallel machine scheduling problem with auxiliary resources, sequence-dependent setup times, job release dates, due dates, and machine restrictions.

In Phase I (Chapter 4) we addressed the minimization of total completion time for resource constrained parallel machine scheduling problems with job-machine eligibility restrictions with release dates denoted as $P_m|r_j, M_j, aux\ 1|\sum C_j$.

Optimal solutions were found for 45 out of the 45 scenarios in less than one hour of computing time for all the problem instance combinations. Several instances of 100 jobs with reduced and medium complexity found optimal solutions under 150 seconds (2.5 minutes) including the "MED|J100|M10|R24|0.33" scenario. Optimal solutions were found in less than 600 seconds (10 minutes) for 41 out of the 45 scenarios. Phase I formulation is recommended to run longer time horizons in order to determine area capacity. This formulation can also be used to establish objective function lower bound since a model without setups is going to allow more jobs in the schedule since there is not idle time changing auxiliary resources. Finally, this model can be used to determine

lower bound for run time and to determine complexity as compared with a model that includes setups.

In Phase II (Chapter 5) we addressed the minimization of total completion time for constrained parallel machine scheduling problem with sequence-dependent setups, job-machine eligibility restrictions and release dates denoted as $P_m|s_{jk}, r_j, aux\ 1|\sum C_j$. The original formulation proposed by Sousa and Wolsey (1992) known as the time-index formulation was expanded to include multiple machines, auxiliary resources, and sequence-dependent setups (i.e. resource and resource travel time between machines). Our work has successfully proofed a scheduling system that can work for increasingly complex systems than the original formulation in 1992. To our knowledge, this exact formulation has not used to solve the photolithography scheduling problem with respect to release dates and setups.

A heuristic for parallel machine scheduling problem with dual resources, ready times, and multi-setup was developed with the objective to find fast feasible solutions that can be provided to the time-indexed IP model in order to reduce the time horizon and provide an initial solution to the model that could be further improved. The heuristic initial solutions for the 45 problem instances were improved by the hybrid model by 2% to 33%. The hybrid model (IP model with initial solution given from heuristic) found optimal solutions for 18 out of the 45 scenarios in less than one hour for all the problem instances of 10/25 jobs. A Gap of 10% or less was found for 24 out of the 45 scenarios ranging from 10 jobs up to 75 jobs (i.e. MED|J75|M7|R18|0.33). Optimal solutions were

found for 15 out of the 45 scenarios for the 10/25 job size instances in less than 600 seconds (10 minutes).

As expected, the "hard" scenario with 100 jobs was not solved in less than one hour due to the excessive number of constraints (9,458,057) which are mainly generated by the most complex constraints. The hybrid model (time indexed IP and heuristic) has potential to be applied in real-world semiconductor environments to solve medium to large problem size instances for the photolithography area. The small and medium problem instances found optimality in one hour or less, but the large model with 100 jobs was only solved by the heuristic. The proposed models have potential to render better results that existing heuristics used in many manufacturing environments as stated by Ham (2018)

In Phase III (Chapter 6) we addressed the minimization of weighted completion time, total tardiness, maximum tardiness for the scheduling of identical parallel machines with shared, constrained, auxiliary resources with sequence-dependent setups, machine eligibility restrictions, job release and due dates with single and multiple objectives denoted as $P_m|s_{jk}, r_j, d_j, M_j, aux\ 1|lex(\alpha.\sum w_j C_j, \sum T_j, T_{max})$. To our knowledge, this exact formulation has not been used to find the Pareto frontier in the semiconductor Industry.

A time-indexed IP optimization model has been formulated to solve the scheduling problem for the small problem instance RED|J10|M2|R4|1.0 and the medium size problem instance RED|J50|M5|R12|1.0. It is noteworthy that for the medium size problem instance sequence-dependent setups were omitted to speed up the model that

ended up taking 183 hrs. to find 29 efficient points. Our model allows user-model interaction for the parallel machine scheduling with shared auxiliary resource.

In summary, this chapter resulted in an exact IP model capable of finding 9 optimal solutions (entire efficient frontier) for the small instance problem in 8 minutes. The small problem instance complexity grew linearly. For the medium problem size instance, the model was capable of finding 29 points in 183 hrs. of computation time since the model complexity grew exponentially. After the first 12 points the time per solution was as high as 47 hrs. for a single schedule. In summary, practitioners may not be able to find the entire efficient frontier for medium size problems. Hence, the proposed model will provide insight to the multiple objective functions but is not recommended for real-time dispatching.

The key contributions from this study to the literature and to practitioners looking to solve complex manufacturing problems are the following:

- According to (Allahverdi, 2015)"The research on scheduling problems with setup times/costs is still less than 10 percent of the available research on scheduling problems while most scheduling environments involve setup operations. Hence, more research on scheduling problems with explicit consideration of setup times/costs is needed". Our proposed phase II model includes sequence-dependent setup times and addresses this recommendation.
- According to (Allahverdi, 2015) "...for the single machine environment with family setup time case, about 75 percent of the papers addressed the sequence-independent problem. This indicates the need for addressing the sequence-

dependent scheduling problems in single machine". Our formulation addresses the multiple identical parallel machine with sequence-dependent scheduling problem; hence, setting m=1 aligns with the recommendation previously stated.

- According to (Allahverdi, 2015) "...in the current competitive work environment, firms strive to save in every possible way, and minimizing work-in process is a major cost saving. It has been observed that the total completion time performance measure has been addressed by a relatively smaller number of papers for the single machine, parallel machine, and job shop environments. It has been also observed that total tardiness performance measure has been only utilized in a few papers for the job shop environments. Therefore, there is a need to consider these performance measures in those scheduling environments". Our phase III objective function addresses this recommendation by minimizing total weighted completion time and total tardiness.
- For the Diversity Maximization Approach (DMA) for MOO presented by Masin and Bukchin, (2008) with MILP scheduling problem, our model allows user-model interaction for the parallel machine scheduling with shared auxiliary resource which addresses Masin and Bukchin's, (2008) recommendation to explore an interactive DMA with the decision maker resulting in a more effective method since the user could focus on the relevant parts of the efficient frontier. Our proposed research is going to address this recommendation since we plan to enable model-user interaction for the DMA MOO model in phase III.

- Ham (2018) suggested to apply the time-indexed MILP model instead of "positional & assignment" variables used in Ham's study. Similarly, Avella *et al*., (2017) successfully applied MILP model in runway scheduling problems reversing the opinion that time-indexed MILP models are unattractive for real-time applications or large scale models. Our formulations address these two recommendations by applying time-indexed IP and MILP models for the three phases.
- Edis (2009) claims that most of the research on parallel machine scheduling neglects machine eligibility restrictions. Our three phases apply job-machine eligibility restrictions to mimic real-world practical models.

7.3. Future Research

This section is further subdivided in two sections. Section 7.3.1 presents proposed research for methods and algorithms and section 7.3.2 provides future research related to the overall framework. We propose the following topics for future research on methods and algorithm:

7.3.1. Methods and Algorithms

The heuristic proposed in Chapter 5 is currently providing "good" feasible solutions in a relatively short amount of time. The model accomplishes its results by generating 32 sorting variations per problem instance with the lowest completion time selected as the final solution to prime the time-indexed IP model. The proposal for future research is to

further improve the heuristic that is currently providing "good" feasible solutions in a relative short amount of time by testing more sorting variations beyond the 32 combinations per instance. We also propose to incorporate multiple objectives (multi-objective) allowing practitioners to use this formulation for real-world problems where there is a need to model machines with resources and setups.

7.3.2. Overall Framework Future Research

The proposed research in this subsection deals with the integration of each component in the system architecture.

A scheduler database module deals with the storage of input data to the model and output from the model which may interact with the factory database. The proposed databased should be designed in a way that allows data input/output into the algorithms, input for machines and Gantt Charts. The database may be designed with pre-processing subroutines to allow faster model building.

The feasibility module's main objective is to determine if user-model interaction results in a feasible schedule. If the user overrides the model rendering an unfeasible instance this module should warn the user or potentially repair the change to make it feasible. This module could also help link two or more schedules in order to minimize completely different schedules. That is, the module should minimize schedules' changes especially if auxiliary resources were requested to be transferred between machines.

Finally, it is recommended that a user-friendly interface be developed that allows the decision maker to efficiently interact with the Gantt Charts in a way that he/she can

edit, add and delete job-machine-resource combination with minimal interface operations.

REFERENCES

Ahn, L. von, 2008. Human Computation.

Akcali, E., Uzsoy, R., 2000. A sequential solution methodology for capacity allocation and lot scheduling problems for photolithography, in: Twenty Sixth IEEE/CPMT International Electronics Manufacturing Technology Symposium (Cat. No.00CH37146). Santa Clara, CA, USA, pp. 374–381. https://doi.org/10.1109/IEMT.2000.910749

Akcalt, E., Nemoto, K., Uzsoy, R., 2001. Cycle-time improvements for photolithography process in semiconductor manufacturing. IEEE Trans. Semicond. Manuf. 14, 48–56. https://doi.org/10.1109/66.909654

Allahverdi, A., 2015. The third comprehensive survey on scheduling problems with setup times/costs. Eur. J. Oper. Res. https://doi.org/10.1016/j.ejor.2015.04.004

Applied Materials, 2012. A predictive scheduling solution that improves productivity of key bottleneck areas in the factory [WWW Document]. URL https://www.appliedmaterials.com/files/automation_software_resources/SmartFactory-Scheduling-Solution-Brief.pdf

Avella, P., Boccia, M., D'Auria, B., 2005. Near-optimal solutions of large-scale single machine scheduling problems. INFORMS Joural Comput. 17, 183–191.

Avella, P., Boccia, M., Mannino, C., Vasilyev, I., 2017. Time-Indexed Formulations for the Runway Scheduling Problem. Transp. Sci. 51, 1196–1209. https://doi.org/10.1287/trsc.2017.0750

Baker, K.R., 1974. Introduction to sequencing and scheduling. Wiley, New York, New York, USA.

Behnamian, J., Zandieh, M., Fatemi Ghomi, S.M.T., 2010. A multi-phase covering Pareto-optimal front method to multi-objective parallel machine scheduling. Int. J. Prod. Res. 48, 4949–4976. https://doi.org/10.1080/00207540902998349

Behnamian, J., Zandieh, M., Fatemi Ghomi, S.M.T., 2009a. Due window scheduling with sequence-dependent setup on parallel machines using three hybrid metaheuristic algorithms. Int. J. Adv. Manuf. Technol. 44, 795–808. https://doi.org/10.1007/s00170-008-1885-7

Behnamian, J., Zandieh, M., Fatemi Ghomi, S.M.T., 2009b. Parallel-machine scheduling problems with sequence-dependent setup times using an ACO, SA and VNS hybrid algorithm. Expert Syst. Appl. 36, 9637–9644. https://doi.org/10.1016/j.eswa.2008.10.007

Bettayeb, B., Kacem, I., Adjallah, K.H., 2008. An improved branch-and-bound algorithm to minimize the weighted flowtime on identical parallel machines with family setup times. J. Syst. Sci. Syst. Eng. 17, 446–459. https://doi.org/10.1007/s11518-008-5065-y

Bigras, L., Gamache, M., Savard, G., 2008. The time dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. Discret. Optim. 5, 685–699.

Bitar, A., Dauzère-Pérès, S., Yugma, C., Roussel, R., 2016. A memetic algorithm to solve an unrelated parallel machine scheduling problem with auxiliary resources in semiconductor manufacturing. J. Sched. 19, 367–376. https://doi.org/10.1007/s10951-014-0397-6

Blackstone, J.H., Phillips, D.T., Hogg, G.L., 1982. A state-of-the-art survey of dispatching rules for manufacturing job shop operations. Int. J. Prod. Res. 20, 27–45. https://doi.org/10.1080/00207548208947745

Blazewicz, J., Dror, M., Weglarz, J., 1991. Mathematical programming formulations for machine scheduling: A survey. Eur. J. Oper. Res. 51, 283–300.

Blazewicz, J., Lenstra, J.K., Rinnooy Kan, A.H.G., 1983. Scheduling subject to resource constraints: Classification and complexity. Discret. Appl. Math. 5, 11–24.

Bozorgirad, M.A., Logendran, R., 2012. Sequence-dependent group scheduling problem on unrelated parallel machines. Expert Syst. Appl. 39, 9021– 9030.

Brown, S.M., Hanschke, T., Meents, I., Wheeler, B.R., Zisgen, H., 2010. Queueing model improves IBM's semiconductor capacity and lead-time management. Interfaces (Providence). 40, 397–407. https://doi.org/10.1287/inte.1100.0516

Brucker, P., 2007. Scheduling algorithms, Springer. https://doi.org/10.1007/978-3-540-69516-5

Cakici, E., Mason, S.J., 2007. Parallel machine scheduling subject to auxiliary resource constraints. Prod. Plan. Control 18, 217–225.

Çaliş, B., Bulkan, S., 2015. A research survey: review of AI solution strategies of job shop scheduling problem. J. Intell. Manuf. 26, 961–973. https://doi.org/10.1007/s10845-013-0837-8

Chai, X., Vuong, B., Doan, A., Naughton, J.F., 2009. into Information Extraction and Integration Programs. Syntax Semant. 87–99.

Cheng, T.C.., Sin, C.C.S., 1990. A state-of-the-art review of parallel-machine scheduling

research. Eur. J. Oper. Res. 47, 271–292.

Chiandussi, G., Codegone, M., Ferrero, S., Varesio, F.E., 2012. Comparison of multi-objective optimization methodologies for engineering applications. Comput. Math. with Appl. 63, 912–942. https://doi.org/10.1016/j.camwa.2011.11.057

Chudak, F., Hochbaum, D., 1999. A half-integral linear programming relaxation for sceduling precedence-constrained jobs on a single machine. Oper. Res. Lett. 25, 199–204.

Chung, S.H., Pearn, W.L., Tai, Y.T., 2009. Fast and effective algorithms for the liquid crystal display module (LCM) scheduling problem with sequence-dependent setup time. J. Oper. Res. Soc. 60, 921–933. https://doi.org/10.1057/palgrave.jors.2602604

Clement, R., Boland, N., Waterer, H., 2016. A big bucket time indexed formulation for nonpreemptive single machine scheduling problems. INFORMS J. Comput. 28, 14–30.

Dios, M., Framinan, J.M., 2016. A review and classification of computer-based manufacturing scheduling tools. Comput. Ind. Eng. 99, 229–249. https://doi.org/10.1016/j.cie.2016.07.020

Dyer, M., Wosley, L., 1990. Formulating the single machine sequencing problem with release dates as a mixed integer program. Discret. Appl Math 26, 255–270.

Edis, E.B., 2009. Resource Constrained Parallel Machine Scheduling Problems with Machine Eligibility Restrictions: Mathematical and Constraint Programming Based Approaches.

Fan, B., Tang, G., 2006. A column generation for a parallel machine scheduling with sequence-dependent setup times. Tongji Daxue Xuebao/Journal Tongji Univ. 34, 680-683+693.

Fox, M.S., 1990. Constraint-guided scheduling-A short history of research at CMU. Comput. Ind. 14, 79–88. https://doi.org/10.1016/0166-3615(90)90107-Z

Framinan, J.M., Ruiz, R., 2010. Architecture of manufacturing scheduling systems: Literature review and an integrated proposal. Eur. J. Oper. Res. 205, 237–246. https://doi.org/10.1016/j.ejor.2009.09.026

Garey, M.R., Johnson, D.S., 1978. “Strong” NP-Completeness Results: Motivation, Examples, and Implications. J. ACM 25, 499–508.

Graham, R.., Lawler, E.L., Lenstra, J., Rinnooy Kan, A., 1979. Optimization and approximation in deterministic sequencing and scheduling a survey.pdf.

Grigoriev, A., Sviridenko, M., Uetz, M., 2005. Unrelated parallel machine scheduling with resource dependent processing times. Math. Program. Ser. B 110, 209–228.

Ham, A., 2018. Scheduling of Dual Resource Constrained Lithography Production : Using CP and MIP/CP. IEEE Trans. Semicond. Manuf. 31, 52–61.

Ham, A.M., Cho, M., 2015. A Practical Two-Phase Approach to Scheduling of Photolithography Production. IEEE Trans. Semicond. Manuf. 28, 367–373. https://doi.org/10.1109/TSM.2015.2451512

Hoitomt, D.J., Luh, P.B., Pattipati, K.R., 1993. A Practical Approach to Job-Shop Scheduling Problems. IEEE Trans. Robot. Autom. 9, 1–13. https://doi.org/10.1109/70.210791

Hopp, W., Spearman, M., 2011. Factory physics. Boston :Irwin/McGraw-Hill,.

Hou, Z.-L., Guo, X.-P., 2013. Parallel Machine Scheduling with Resources Constraint and Sequence Dependent Setup Times. R. Dou (ed.), Proc. 2012 3rd Int. Asia Conf. Ind. Eng. Manag. Innov. https://doi.org/10.1007/978-3-642-33012-4_80

Hu, D., Yao, Z., 2011. Parallel Machines Scheduling with Sequence-Dependent Setup Times Constraints. Adv. Sci. Lett. 4, 2528–2531. https://doi.org/10.1166/asl.2011.1551

Hunter, J., Delp, D., Collins, D., Si, J., 2002. Understanding a semiconductor process using a full-scale model. IEEE Trans. Semicond. Manuf. 15, 285–289.

Kanet, J.J., Adelsberger, H.H., 1987. Expert systems in production scheduling. Eur. J. Oper. Res. 29, 51–59. https://doi.org/10.1016/0377-2217(87)90192-5

Keha, A.B., Khowala, K., Fowler, J.W., 2009. Computers & Industrial Engineering Mixed integer programming formulations for single machine scheduling problems. Comput. Ind. Eng. 56, 357–367. https://doi.org/10.1016/j.cie.2008.06.008

Khowala, K., Keha, A., Fowler, J., 2005. A comparison of different formulations for the non-preemptive single machine total weighted tardiness scheduling problem, in: The Second Multidisciplinary International Conference on Scheduling: Theory & Application (MISTA).

Kovalyov, M.Y., Shafransky, Y.M., 1998. Uniform machine scheduling of unittime jobs subject to resource constraints. Discret. Appl. Math. 84, 253–257.

Lasserre, J., Queyranne, M., 1992. Generic scheduling polyhedral and a new mixed integer formulation for single-machine scheduling, in: Ln: Pittsburg: Carnegie Mellon University.

Leachman, R., 2013. Semiconductor Production Planning, in: Lecture Notes: University of California at Berkeley.

Loveland, J.L., Monkman, S.K., Morrice, D.J., 2007. Dell Uses a New Production-Scheduling Algorithm to Accommodate Increased Product Variety. Interfaces (Providence). 37, 209–219. https://doi.org/10.1287/inte.1060.0264

Manne, A.S., 1960. On the Job-Shop Scheduling Problem. Oper. Res. 8, 159–224.

Marteney, S., 2011. Predictive, short-interval scheduling improves litho utilization and cycle time. Solid State Technol. 54, 17–18.

Masin, M., Bukchin, Y., 2008. Diversity Maximization Approach for Multiobjective Optimization. Oper. Res. 56, 411–424. https://doi.org/10.1287/opre.1070.0413

McNaughton, R., 1959. Scheduling with Deadlines and Loss Functions. Manage. Sci. 6, 1–12.

Metta, H., 2008. Adaptive, multi-objective job shop scheduling using genetic algorithms.

Min, H.-S., Yih, Y., 2003. Selection of dispatching rules on multiple dispatching decision points in real-time scheduling of a semiconductor wafer fabrication system. Int. J. Prod. Res. 41, 3921–3941. https://doi.org/10.1080/0020754031000118099

Mönch, L., Fowler, J.W., Dauzère-Pérès, S., Mason, S.J., Rose, O., 2011. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. J. Sched. 14, 583–599. https://doi.org/10.1007/s10951-010-0222-9

Monkman, S.K., Morrice, D.J., Bard, J.F., 2008. A production scheduling heuristic for an electronics manufacturer with sequence-dependent setup costs. Eur. J. Oper. Res. 187, 1100–1114.

Nogueira, T., Carvalho, C., Ravetti, M., Souza, M., 2019. Analysis of mixed integer programming formulations for single machine scheduling problems with sequence dependent setup times and release dates. Pesqui. Operacional 39, 109–154.

Olsen, T.L., 1999. A practical scheduling method for multiclass production systems with setups. Manage. Sci. 45, 116–130. https://doi.org/10.1287/mnsc.45.1.116

Parameswaran, A., Garcia-molina, H., Park, H., Widom, J., 2012. Crowdscreen: algorithms for filtering data with humans, in: In: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. Scottsdale, AZ, USA., pp. 361–372.

Park, S., Fowler, J., Carlyle, M., Hickie, M., 1999. Assessment of potential gains in productivity due to proactive reticle management using discrete event simulation, in: Proc. 1999 Winter Simulation Conf. pp. 856–864.

Park, T., Lee, T., Kim, C.O., 2012. Due-date scheduling on parallel machines with job splitting and sequence-dependent major/minor setup times. Int. J. ofAdvanced Manuf. Technol. 59, 325–333.

Pessoa, A., Uchoa, E., Poggi, M., Rosiane, D.A., 2010. Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. Math. Program. Comput. 259–290. https://doi.org/10.1007/s12532-010-0019-z

Pinedo, M.L., 2016. Scheduling Theory, Algorithms, and Systems., 5th editio. ed. Springer.

Pinedo, M.L., 2008. Scheduling - Theory, Algorithms, and Systems, Igarss 2014. https://doi.org/10.1007/s13398-014-0173-7.2

Potts, C., 1980. An algorithm for the single-machine sequencing problem with precedence constraints, in: Lecture Notes: Combinatorial Optimization II, Springer, Pp 78-87.

Queyranne, M., 1993. Structure of a simple scheduling polyhedron. Math. Program. 58, 263–285.

Queyranne, M., Schultz, A., Universitat, T., 1994. Polyhedral approaches to machine scheduling, in: Tech. Rep. Berlin, Germany: Technical University of Berlin, Deprtment of Mathematics.

Queyranne, M., Wang, Y., 1991. Single-Machine Scheduling Polyhedra with Precedence Constraints. Math. Oper. Res. 16, 1–20.

Raghavan, H., Allan, J., 2007. An interactive algorithm for asking and incorporating feature feedback into support vector machines. Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. - SIGIR ’07 79. https://doi.org/10.1145/1277741.1277758

Razaq, T.A., Potts, C., Wassenhove, L.V., 1990. A survey of algorithms for the single machine total weighted tardiness scheduling problem. Discret. Appl. Math. 26, 235–253.

Reinschmidt, K.F., Slate, J.H., Finn, G.A., 1990. Expert systems for plant scheduling using linear programming. Proc. Fourth Int. Conf. Expert Syst. Prod. Oper. Manag. Head Island, USA.

Rocha, P.L., Ravetti, M.G., Mateus, G.R., Pardalos, P.M., 2008. Exact algorithms for a

scheduling problem with unrelated parallel machines and sequence and machine-dependent setup times. Comput. Oper. Res. 35, 1250–1264. https://doi.org/10.1016/j.cor.2006.07.015

Romero-Silva, R., Santos, J., Hurtado, M., 2015. A framework for studying practical production scheduling. Prod. Plan. Control 26, 438–450. https://doi.org/10.1080/09537287.2014.919413

Ruiz-Torres A.J., Lopez, F.J., Ho, J.C., 2007. Scheduling uniform parallel machines subject to a secondary resource to minimize the number of tardy jobs. Eur. J. Oper. Res. 179, 302–315.

Sadykov, R., 2006. Integer programming-based decomposition approaches for solving machine scheduling problems. Universite catholique de Louvain Faculte des Sciences Appliquees.

Sadykov, R., Vanderbeck, F., 2011. Column generation for extended formulations. Electron. Notes Discret. Math. 37, 357–362.

Schaller, J.E., 2014. Minimizing total tardiness for scheduling identical parallel machines with family setups. Comput. Ind. Eng. 72, 274–281. https://doi.org/10.1016/J.CIE.2014.04.001

Sourd, F., n.d. New exact algorithms for one machine earliness tardiness scheduling. INFORMS J. Comput. 21, 167–175.

Sousa, J.P., Wolsey, L.A., 1992. A time indexed formulation of non-preemptive single machine scheduling problems. Math. Program. 54, 353–367. https://doi.org/10.1007/BF01586059

Steffen, M.S., 1986. A survey of artificial intelligence-based scheduling systems. Proc. Fall Ind. Eng. Conf.

Stevenson, M., Huang, Y., Hendry, L.C., 2009. The development and application of an interactive end-user training tool: Part of an implementation strategy for workload control. Prod. Plan. Control 20, 622–635. https://doi.org/10.1080/09537280903034313

Suresh, R. k., Mohanasundaram, K. m., 2006. Pareto archived simulated annealing for job shop scheduling with multiple objectives. Int. J. Adv. Manuf. Technol. 29, 184–196. https://doi.org/10.1007/s00170-004-2492-x

Tanaka, S., Araki, M., 2013. An exact algorithm for the single-machine total weighted tardiness problem with sequence-dependent setup times. Comput. Oper. Res. 40, 344–352.

Tanaka, S., Fujikuma, S., Araki, M., 2009. An exact algorithm for single-machine scheduling without machine idle time. J. Sched. 12, 575–593.

Tavakkoli-Moghaddam, R., Mehdizadeh, E., 2007. A new ILP model for identical parallel-machine scheduling with family setup times minimizing the total weighted flow time by a genetic algorithm. Int. J. Eng. Trans. A 20, 183–194.

Toksarı, M.D., Güner, E., 2010. Parallel machine scheduling problem to minimize the earliness/tardiness costs with learning effect and deteriorating jobs. J. Intell. Manuf. 21, 843–851. https://doi.org/10.1007/s10845-009-0260-3

Unlu, Y., Mason, S.J., 2010. Computers & Industrial Engineering Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems q. Comput. Ind. Eng. 58, 785–800. https://doi.org/10.1016/j.cie.2010.02.012

Ventura, J.A., Kim, D., 2000. Parallel machine scheduling about an unrestricted due date and additional resource constraints. IIE Trans. 32, 147–153.

Wagner, H.M., 1959. An integer linear-programming model for machine scheduling. Nav. Res. Logist. 6, 131–140.

Wolsey, L.A., 1998. Integer Programming Volume 52 of Wiley Series in Discrete Mathematics and Optimization, illustrate. ed. John Wiley & Sons.

Xue, L., Villalobos, J.R., 2012. A multi-objective optimization primary planning model for a POE (Port-of-Entry) inspection. J. Transp. Secur. 5, 217–237. https://doi.org/10.1007/s12198-012-0093-8

Yan, B., Chen, H., Luh, P., Wang, S., Chang, J., 2013. Litho Machine Scheduling With Convex Hull Analyses. IEEE Trans. Autom. Sci. Eng. 10, 928–937. https://doi.org/10.1109/TASE.2013.2277812

Yan, B., Chen, H., Luh, P., Wang, S., Chang, J., 2012. Optimization-based Litho Machine Scheduling with Load Balancing and Reticle Expiration, in: 8th IEEE International Conference on Automation Science and Engineering. Aug 20-24, Seoul, Korea. IEEE, pp. 575–580. https://doi.org/10.1109/CoASE.2012.6386493

Yan, B., Chen, H., Luh, P., Wang, S., Chang, J., 2011. Optimization-based Litho Machine Scheduling with Multiple Reticles and Setups, in: 2011 IEEE International Conference on Automation Science and Engineering. pp. 114–119. https://doi.org/10.1109/CASE.2011.6042516

Zhang, J., Ding, G., Zou, Y., Qin, S., Fu, J., 2017. Review of job shop scheduling research and its new perspectives under Industry 4.0. J. Intell. Manuf.

https://doi.org/10.1007/s10845-017-1350-2