CHAPTER 1
INTRODUCTION

Why investigate production scheduling problems motivated by semiconductor manufacturing? Because scheduling works. Scheduling, the term, refers to the process of assigning tasks to resources and determining when each task will be done. Scheduling is not, however, limited to manufacturing, since this type of problem occurs in other activities. Scheduling, the science, embodies knowledge about models, techniques, and insights related to actual systems (Baker, 1992). This field is an important part of operations research. In order to compete effectively in the marketplace, firms have used operations research as a tool for understanding and improving their manufacturing systems.

This is especially true in the manufacturing of high-technology products, where the new, complicated processes lead to production scheduling problems that are not well-solved. The post-assembly testing of semiconductors, for instance, is a complicated job shop scheduling problem; among the difficulties is the presence of sequence-dependent setups at certain operations. Moreover, scheduling, which has been studied from the earliest days of operations research, has received new attention due to the successful implementation of just-in-time systems that emphasize the close coordination of resources and the maintenance of low work-in-process inventory levels.

Although the research in production scheduling has yielded a large body of knowledge which has been successfully applied to a number of areas, there still remain problems to be solved. The problems previously studied have always simplified reality in some way. Adding realism creates problems that resist simple solution techniques. Further research into these types of problems is necessary in order to increase our ability to control manufacturing processes effectively.

The research described in this dissertation falls into this category. It includes factors that model reality more closely and studies a number of different problems in an effort to gain insight into how improved job shop scheduling may be achieved. This insight will be applied to the semiconductor test area being studied.

The problems that this research investigates are motivated by a high-technology field where much time has been spent investigating manufacturing systems, namely the semiconductor industry. The research is particularly motivated by the operations of semiconductor testing, although the problems can also be found in the processes of other manufacturing industries.

This introduction includes a number of sections about topics that will arise in the scheduling of semiconductor test operations. These topics are introduced here in order to describe how they are related to this research. The first of these is an overview of semiconductor manufacturing.

1.1 Semiconductor Manufacturing

In order to give some context to the problems being studied, this section describes the general flow of semiconductor manufacturing.

The manufacturing of semiconductors consists of many complex steps. The first process is that of wafer fabrication, done in a super-clean environment in order to protect the delicate structures on the wafers from contaminants. The wafers of silicon undergo repeated applications of the cycle of photolithography, etching, and diffusion in order to build, layer by layer, the different electronic devices. A wafer will contain a number of identical circuits laid out in a grid pattern on the wafer. In the next process, probe, these individual circuits are inspected with a wafer probe and marked if defective. In the assembly process, another clean area, the wafers are cut into the separate circuits. Good circuits are placed into the packages that protect them from the environment and that provide electrical connections to allow them to interact with the world. The identical devices from a lot of identical wafers become a lot that then moves into the test

facility, an area where environmental safeguards protect the devices from static electricity that could still damage the circuits.

The test process consists of initial testing, burn-in, and final testing. The testing includes electrical testing of each device and other testing of the device packages. Burn-in consists of subjecting the semiconductor chips to extreme thermal stresses. Other non-test operations, such as serialization and brand, take place during the process.

Not all of the lots follow the same route through the test floor. Different products are tested on different machines and undergo a different set of tests. Thus, the test area is a job shop, and the scheduling problem is a job shop scheduling problem. This job shop of semiconductor test operations has sequence-dependent setup times, re-entrant flows, batch processing, routes that temporarily leave the test area, and other complications that lead to many interesting scheduling questions.

The primary management objectives for this area are customer satisfaction and making profit. Because testing is the last process in the manufacturing flow, the output of this area affects how well the company can get orders to its customers on time and how much product the company can ship to generate revenue.

The particular semiconductor test area being studied has implemented a decision support system to assist in the scheduling of lots through the area. The programs in the system can order the lots waiting for an workstation with priority rules that include information about lateness, priority, and setups. As part of the facility-wide computer-integrated manufacturing system, it is able to access real-time information about the lots. The system is used to create schedules for a short time period (two to four hours), since dispatching more often would require excessive computer resources. Lots that are tardy and lots that have some externally-imposed priority are expedited whenever possible.

1.2 Job Shop Scheduling

The semiconductor test area is a job shop, and optimizing the scheduling of the area is a job shop scheduling problem. This section discusses job shop scheduling.

Job shop scheduling consists of those scheduling problems in which different jobs may follow different routes through the shop. A job consists of a number of operations or tasks, which must be processed in the given order. There exists a specified machine (or workstation) that can perform each operation. These problems are generally the hardest to solve optimally, since few properties of optimal schedules are known and the number of possible solutions explodes as the problems increase in size. However, in many cases, the job shop is a better model of reality than one-machine, parallel-machine, or flow shop problems.

Because of the complexity of job shop scheduling, algorithms to find the optimal solution (in a reasonable amount of time) for any arbitrary objective function do not exist. Recent research has shown that the *shifting bottleneck* heuristic is successful at finding good solutions for a simple job shop scheduling problem. Traditionally, however, researchers have studied and schedulers have used dispatching rules to order the jobs waiting for processing at a machine. When a machine becomes available, it chooses from among the jobs in its queue by using a dispatching rule. Common dispatching rules employ processing times and due dates in both simple rules and complex combinations. These dispatching rules are often extensions of the algorithms used to solve one-machine problems.

In any shop, there may exist a bottleneck machine whose throughput is a limiting factor on the capacity of the shop. If so, the improved scheduling of this one machine becomes an important objective.

1.3 Look-ahead and Look-behind Scheduling

This section addresses a weakness of the traditional dispatching rule approach to scheduling job shops and defines two types of rules that overcome this weakness.

Traditional dispatching rules are myopic; they are concerned only with the machine to be scheduled and the jobs waiting for that machine. Improved scheduling may be realized by using rules that can consider more information, in order to see that critical lots are arriving soon or that a certain machine has an excessive queue. Look-ahead and look-behind scheduling includes procedures that look around the shop for more information to use in making a scheduling decision. With this additional information, they can hopefully produce better decisions. Look-ahead models consider the machines where the jobs will be headed after this stage. Look-behind models consider the jobs that will be arriving at this machine soon.

The terms *look-ahead* and *look-behind* are used to designate scheduling procedures that do more than consider just the state of one machine. Both types of models look into the future (where jobs will go and what jobs will be arriving). The difference is what part of the future they consider.

1.4 Setups

One of the complications of scheduling the operations in semiconductor testing is the presence of different product types that require different configurations on the same machine. The task of configuring a machine in order to process a job is a *setup*.

If the setup for a job is independent of the job that was scheduled before it on the same machine, the setup time can be included in the processing requirements of that job. If the setup for a job is *sequence-dependent*, that is, the time of the setup depends upon the immediate predecessor job, the scheduling problem becomes quite difficult.

Because the problems with sequence-dependent setups are quite difficult, researchers have examined special cases of the general problem. The most common problem is the *class scheduling problem*, where the jobs to be processed are grouped into a number of job *classes*. There exists no sequence-dependent setup between jobs from the same class, although there does exist a special setup, the *class setup*, when a job from one class is processed after one from another class. There may also exist a class setup for the first job processed. These class setups

may be sequence-dependent in that they depend upon the class that was last processed. Class scheduling is still difficult, though: the class scheduling extensions of one-machine problems that are easy to solve are usually NP-complete problems.

An example of class setups occurs in the electrical testing of assembled semiconductor devices on a machine which can test a number of different types of semiconductors. If a machine is scheduled to test a lot consisting of devices that are different from the devices tested in the previous lot, various setup tasks are required. These tasks may include changing a handler and load board that can process only certain types of semiconductor packages or loading a new test program on the tester. However, if the new lot consists of devices that are the same as the previous type, none of this setup is required. This is a class scheduling problem.

1.5 Smart-and-lucky Searches

As mentioned before, algorithms to solve the job shop scheduling problem optimally in reasonable time do not exist. In addition to the shifting bottleneck heuristic and the use of dispatching rules, one way to find good solutions is to search for them. This section will discuss standard local searches and new, more sophisticated heuristic searches.

One method of finding good solutions to hard optimization problems has been local search. A local search begins with some initial solution and moves from an incumbent solution to a better neighboring solution, ending when no improvement can be found. At this point, the search has reached a local optimum. Two common local searches are *hillclimbing* and *steepest descent.* In a hillclimbing search, the neighbors are chosen at random, and the first better neighbor found is chosen as the new solution. In a steepest descent search, the entire neighborhood of the incumbent solution is searched and the neighbor that has the best performance improvement is selected. In order to overcome the fact that these searches converge only to local optima, new heuristic searches have been developed. These include *simulated annealing*, *tabu search*, and *genetic algorithms.*

These heuristic searches are complex searches that are *smart* enough to escape from most local optima and are *lucky* enough to generally find good solutions.

Simulated annealing (SA) is a variant of the hillclimbing algorithm. Simulated annealing is so called because the algorithm views optimization as a process analogous to physical annealing, the cooling of a system until it reaches a low-energy state. In the same way that a system may pass through higher-energy states during the cooling process, the simulated annealing search occasionally moves to worse neighbors before settling into a good solution.

Tabu search (TS) is a variant of steepest descent. Given an incumbent solution, a TS canvasses the neighborhood of this solution in order to find the best allowable move. If the search is at a local optimum, it is forced to move away, and the move that would return to the previous solution is temporarily prohibited (tabu) by adding it to the tabu list for a number of moves. In this way, TS may continue to move away from a local optimum and find an area of the space that leads to another local optimum. An aspiration level insures that the search will make a move that leads to a solution better than any yet found, even if tabu. Thus, a tabu search works because the tabu list forces the search to explore new areas of the solution space. The short-term aspect of the memory and the aspiration level allow, however, the search to get to a global optimum.

A genetic algorithm (GA) is a procedure that mimics the adaptation that nature uses to find an optimal state. In genetic algorithms, solutions are represented as strings (chromosomes) of alleles. An allele is a bit of information about the solution. The search performs operations on the population of solutions. These operations are 1) the evaluation of individual fitness, 2) the formation of a gene pool, and 3) the recombination and mutation of genes to form a new population. After a period of time, good strings dominate the population, providing a good solution.

When applied to scheduling problems, simulated annealing and tabu search easily search the complex solution spaces with simple types of moves, generally find good solutions fast, and are smart and lucky variations of standard searches such as hillclimbing and steepest descent.

Genetic algorithms are something completely different, work well by using good strings and implicit parallelism, and are harder to implement.

1.6 Problem Space and Heuristic Space

Typical problem-solving searches like those described in the above section have examined the solution space. A solution can be a vector of values or a sequence of objects. Applying a heuristic to a problem yields a point in the solution space. This implies that a search over all of the possible heuristics would find a solution that gives the optimal objective function value.

Moreover, applying a heuristic to another, similar problem also generates a point in the solution space which can be evaluated for the original problem using the objective function. Thus, a search over the new problem space provides a way to solve the problem.

An example of a heuristic space is the space of vectors of m dispatching rules for a n-job and m-machine job shop scheduling problem, where each dispatching rule is applied to a different machine. Searching over the vectors of dispatching rules is similar to machine learning. For a complicated one-machine problem, a sample problem space would be the space of n-element vectors, where each element corresponds to an alternative due date for a job. Sorting the jobs by these alternative due dates creates a new solution.

1.7 Objective Functions

The management of a manufacturing system is concerned with many different performance measures, including customer service, inventory holding costs, throughput, and machine utilization. This section describe how the objectives used in scheduling mirror these concerns.

Scheduling problems include many different objective functions that attempt to model the concerns of those who are running the system. Most objectives are functions of the completion times of the jobs to be scheduled. The *makespan* is the maximum completion time, which is when all of the jobs are finished, and is some measure of the throughput of the system. The *total flowtime* is the total time that the jobs spend in the system and is a measure of the work-in-

process inventory held while processing the jobs. Other objective functions include due dates for each job, where a job is tardy if it completes after its due date. These objective functions are concerned with customer satisfaction, and include *maximum lateness*, which is the maximum difference between a completion time and a due date, the *number of tardy jobs*, and the *total tardiness*. Another interesting objective function is *earliness*, which is a measure of the holding cost incurred by storing finished product until it is delivered at its due date.

While all of these objective functions (and many more) have been studied for one-machine problems, most analysis of flow shop and job shop scheduling has focused on the minimization of makespan. Since this objective is not appropriate in semiconductor manufacturing, this research is especially interested in solving job shop scheduling problems with other objectives.

1.8 Overview of Research

There exist different approaches to attacking the job shop scheduling problem. One can use the shifting bottleneck algorithm to minimize the makespan, use good dispatching rules in a dynamic environment, or search for good global schedules. This research is concerned with the last two methods (see Figure 1.1). This research investigates the use of smart-and-lucky searches over the new search spaces to find good solutions to the job shop scheduling problem. Also investigated are a number of interesting one-machine class scheduling problems and look-ahead models in order to devise techniques that can be used as good dispatching rules. Finally, this research studies how these techniques may be applied to the actual semiconductor test floor that is the motivation for this work.

The benefits of this work include three areas: 1. The addition of results to the body of knowledge about specific scheduling problems, including the difficult class scheduling problems and little-studied look-ahead models. 2. The construction of a robust genetic algorithm for one-machine class scheduling problems. 3. The development and implementation of a genetic algorithm for global job shop scheduling. Moreover, this work continues the investigation of scheduling semiconductor test operations that has only recently begun.

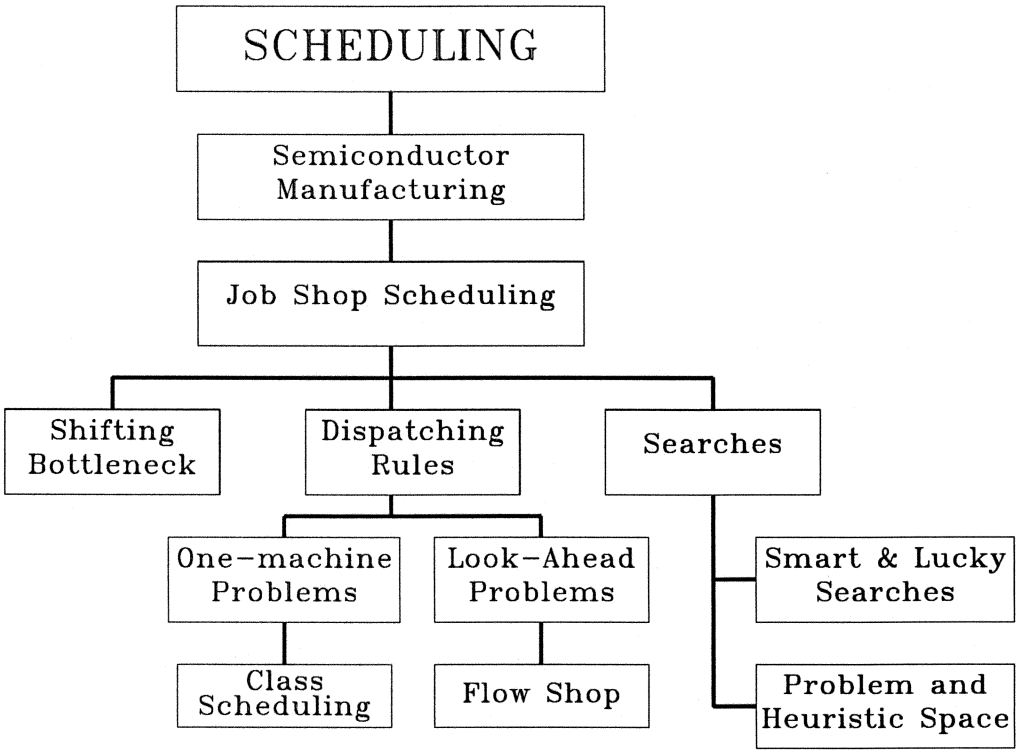

Figure 1.1. Relationship of dissertation topics.

1.9 Plan of Dissertation

This research investigates both one-machine and multi-machine problems. The one-machine problems studied are class scheduling problems that model the complicating factor of machine setups in the manufacturing process. This work also considers some three-machine general flow shop problems that are attempts to model how scheduling can be improved by considering the states of other machines. The research also investigates a general job shop scheduling problem in an attempt to see how new search spaces may be used for global job shop scheduling.

For these problems, the research has focused on finding good lower bounds, developing heuristics to find good solutions for the problems, and using genetic algorithms to find better solutions. Empirical testing of these bounds and heuristics serves as a way of evaluating the

heuristics and the searches. Also, good heuristics for the subproblems can be implemented as advanced dispatching rules for solving the job shop scheduling problem.

In addition, analytical results such as NP-completeness, lower bounds, error bounds, and optimality conditions are presented for the problems being studied.

The machine scheduling problems under investigation are as follows:

1. Constrained Flowtime with Setups (CFTS)
2. Class Scheduling with Release and Due Dates (CSRDD)
3. Flowtime with Setups and Release Dates (FTSRD)
4. Three-Machine Look-Ahead Scheduling: Makespan (3MLA-MS)
5. Three-Machine Look-Ahead Scheduling: Flowtime (3MLA-FT)
6. Three-Machine Look-Ahead Scheduling: Number of Tardy Jobs (3MNT)
7. Job shop scheduling

The first problem is the class scheduling extension of the one-machine problem of minimizing the total flowtime of a set of jobs that have deadlines. The second problem is the class scheduling problem where each job has a release date and a due date. The objective is to minimize the number of tardy jobs. The objective in the third problem is to minimize the total flowtime where each job has a release date.

The research next considers the three-machine look-ahead problems with the objectives of makespan, total flowtime, and number of tardy jobs.

Finally, this research investigates the general job shop scheduling problem and a heuristic space approach to finding good solutions with a genetic algorithm.

The dissertation consists of the following five chapters: the background; the research on one-machine class scheduling problems, on three-machine look-ahead problems, and on job shop scheduling; and the conclusions.

The background material contains detailed information about the problems and methods under investigation and summaries of a number of papers that are related to this work. The topics include semiconductor test operations and semiconductor scheduling, job shop scheduling

techniques, class scheduling, some one-machine problems, smart-and-lucky searches, problem and heuristic space, look-ahead and flow shop scheduling, burn-in scheduling, and NP-completeness.

The discussion of the research contains reports on the problems that have been investigated: three one-machine class scheduling problems, three three-machine general flow shop problems, and a heuristic search for use on the general job shop scheduling problem. In the conclusions we summarize this research and identify some directions for future research.