CHAPTER 5
GLOBAL JOB SHOP SCHEDULING

In this chapter we describe a global job shop scheduling procedure that uses a genetic algorithm to find a good schedule. We have implemented the scheduling system in a semiconductor test area. The test area is a job shop and has sequence-dependent setup times at some operations. The concern of management is to meet their customer due dates and to increase throughput. This requires the coordination of many resources, a task beyond the ability of simple dispatching rules. We discuss a centralized procedure that can find a good schedule through the use of a detailed scheduling model and a genetic algorithm that searches over combinations of dispatching rules. We discuss our effort in developing a system that models the shop, creates schedules for the test area personnel, and contributes to test area management.

5.1 Introduction

In many areas of manufacturing, the ability of a facility to meet its objectives depends upon the close coordination of resources. This coordination must occur on different levels: capacity planning, release planning, and lot dispatching. Effective approaches exist (and new techniques are being developed) for the first two levels. The third level, meanwhile, continues to pose very difficult scheduling problems. These are the problems that we address. Like other researchers, we are interested in creating systems to better schedule resources in a manufacturing process, since effective scheduling can lead to improvements in throughput, customer satisfaction (measured by meeting due dates), and other performance measures.

We are concerned with the effective scheduling of a semiconductor test area, a job shop environment. In this facility, a lot is a number of identical semiconductor devices. Each lot must undergo a number of tests (electrical and physical) and other operations before the product can be

shipped to the customer. Associated with the lot is the due date of the customer order it will be used to fill. Because each product has a unique route, different lots take different paths through the test area.

A number of characteristics make the semiconductor test area difficult to control. These include the conflicting goals of management (balancing increased throughput against meeting due dates), work centers with sequence-dependent setup times, operations where multiple lots can be processed simultaneously, and the relationships between operations.

The purpose of this chapter is to describe the global job shop scheduling system developed for semiconductor testing. We will examine the needs of the semiconductor test area, their requirements for a scheduling system, and our approach to this problem.

The primary goal of the global job shop scheduling system is to use information about the current status of the shop, the jobs to be manufactured, and the production process in order to create a schedule of activities for each work center in the shop over a fixed time period. By using a centralized procedure with global information, the system can search for a schedule better than that computed by making nearsighted, local decisions.

The system finds good solutions with a genetic algorithm, a type of heuristic search. One interesting characteristic of this search is that it looks for a good combination of dispatching rules to find an efficient schedule. The use of dispatching rules is an effective local procedure to create a job shop schedule; such techniques do not, however, use any global information. Our procedure addresses this shortcoming. It goes beyond simply finding good local schedules; it executes a search that looks for a better global schedule. While global scheduling techniques have been proposed before, the important contributions of our work are the development of a genetic algorithm for job shop scheduling and the implementation of a scheduling system that uses such an advanced solution procedure.

The next section of this chapter will review some background and related research on job shop scheduling. The genetic algorithm for global scheduling is introduced in Section 5.3, where we also discuss a small example of our search space. In Section 5.4 we describe the scheduling

problem in the semiconductor test area under consideration and the global job shop scheduling system that we developed. We summarize our report in Section 5.5 and describe how a similar system may be useful in other manufacturing environments.

5.2 Job Shop Scheduling

As mentioned in the introduction, the problem of coordinating resources to ensure efficient manufacturing is a difficult problem. When the production process is fairly straightforward, techniques such as Just-In-Time (JIT) manufacturing result in efficient scheduling. In many situations, however, the process is much more complicated, and finding an optimal or near-optimal schedule is an impossible task. In this section we will review some of the approaches to job shop scheduling. More details on these papers can be found in Chapter 2.

The traditional method of controlling job shops is the use of dispatching rules. A dispatching rule is a sequencing policy that orders the jobs waiting for processing at a machine; the ordering depends upon the particular dispatching rule used. Common rules include the Shortest Processing Time (SPT) and Earliest Due Date (EDD) rules.

While such rules have been the subject of much research, standard dispatching rules have a narrow perspective on the scheduling problem, since they ignore information about other jobs and other resources in the shop. More advanced look-ahead and look-behind rules attempt to include more information, but their reach is still limited. Effective job shop scheduling depends upon the interaction of a number of factors. The complexity of these interactions makes the development of a global scheduling system a difficult task.

A number of researchers have looked at semiconductor manufacturing at all levels, from production planning to scheduling. Approaches to shop floor control include lot release policies, dispatching rules, deterministic scheduling, control-theoretic approaches, knowledge-based approaches, and simulation. Uzsoy *et al.* (1992a, 1993) review a substantial number of papers that consider these approaches to semiconductor scheduling.

This work in semiconductor manufacturing includes corporate-wide production planning that uses a rate-based model of production and linear programming (Leachman, 1993, and Hackman and Leachman, 1989). Such a global planning system has been developed under the name IMPReSS at Harris Semiconductor. Hung and Leachman (1992) have developed an iterative method that uses a linear program and a discrete-event simulation to develop long-term production plans for a wafer fab.

While most of the research in semiconductor scheduling has concentrated on the fabrication of semiconductor wafers, a number of other papers have addressed the problems of semiconductor test. These include Lee, Uzsoy, and Martin-Vega (1992), and Uzsoy *et al.* (1991a, 1991b, and 1992b). Lee *et al.* (1993) report on the implementation of a decision support system for the dispatching of lots in a semiconductor test area. Our work is the natural extension of this system.

Previous approaches to the production of detailed shop schedules for planning and shop floor control include expert systems like ISIS (Fox and Smith, 1984), OPIS (Smith, Fox, and Ow, 1986, and Ow and Smith, 1988), MICRO-BOSS (Sadeh, 1991), and OPAL (Bensana, Bel, and Dubois, 1988); cost-based procedures such as OPT (Optimized Production Technology, reviewed by a number of authors, including Jacobs, 1984), Faaland and Schmitt (1993), and the bottleneck dynamics of SCHED-STAR (Morton *et al.*, 1988); simulation (Leachman and Sohoni, Najmi and Lozinski); and leitstands (Adelsberger and Kanet, 1991). Adler *et al.* (1993) describe the implementation of a bottleneck-based scheduling support system for a paper bag production flexible flow shop. While these approaches have been developed for a number of different manufacturing processes, the complications of the semiconductor test area forced us to consider a new design.

Job shop scheduling, as one of the most difficult scheduling problems, has attracted a lot of attention from researchers. Techniques such as the shifting bottleneck algorithm (Adams, Balas, and Zawack, 1988) or bottleneck dynamics (see Morton, 1992, for example) concentrate on solving the problem at one machine at a time. More work has gone into the development and

evaluation of various dispatching rules. Panwalkar and Iskander (1977) present a list of over 100 rules. Recent studies include Fry, Philipoom, and Blackstone (1988), Vepsalainen and Morton (1988), and Bhasakaran and Pinedo (1991).

More sophisticated *look-ahead* and *look-behind* rules have also been discussed. Look-behind rules (called *x-dispatch* by Morton, 1992) consider the jobs that will be arriving soon from upstream machines. Look-ahead rules consider information about the downstream machines. This includes the work-in-next-queue and the number-in-next-queue rules of Panwalkar and Iskander (1977), bottleneck starvation avoidance (Glassey and Petrakian, 1989), and lot release policies that look-ahead to the bottleneck (Wein, 1988; Glassey and Resende, 1988; and Leachman, Solorzano, and Glassey, 1988). Look-ahead and look-behind scheduling problems have been studied in this dissertation and by Lee and Herrmann (1993).

Finally, heuristic searches have also been developed for job shop scheduling, and a number of these are discussed in Section 2.8 of this dissertation.

5.3 A Genetic Algorithm for Job Shop Scheduling

In this section we will describe how a genetic algorithm can be used to find good schedules for the job shop scheduling problem.

One approach to difficult scheduling problems such as job shop scheduling is local search, an iterative procedure that moves from solution to solution until it finds a local optimum. *Smart-and-lucky searches* (or heuristic searches, or probabilistic search heuristics) attempt to overcome the primary problem of these simple searches: convergence to local optima. These more complex searches are smart enough to escape from most local optima; they still must be lucky, however, in order to find the global optimum.

In previous sections we reviewed the basic concepts of genetic algorithms and mention some application of smart-and-lucky techniques for job shop scheduling. In this section we describe our application of these ideas.

5.3.1 The Heuristic Space

Many heuristic searches for the job shop scheduling problem have been considered. One recently-introduced idea is to search the problem and heuristic spaces, since a solution is the result of applying a heuristic to a problem. A change to the parameters of a heuristic or a problem yields a slightly different solution. This section will describe how a heuristic space can be used for job shop scheduling. We will show why a genetic algorithm is especially suited for searching this space.

The idea of searching heuristic and problem spaces was reported by Storer, Wu, and Vaccari (1992). In their paper, the authors examine the general job shop scheduling problem. They define a heuristic space composed of vectors of dispatching rules. Each vector in the space can be used to determine a schedule. Each rule in the vector is used for a fixed number of dispatching decisions, regardless of the machine being scheduled. That is, all of the machines use the same dispatching rule at the same time until the next rule replaces it.

Our approach uses a different perspective. Since we will be working in a dynamic job shop scheduling environment, we may not know how many operations will be scheduled. Thus, it is impossible to divide the scheduling horizon by allocating each rules to a fixed number of operations. Instead, it seems more fair to assign a dispatching rule to each machine. The system can evaluate this combination of dispatching rules (which we call a *policy*) by measuring the performance of the schedule that is created by using this policy. Changing the policy by modifying the dispatching rule on one or more machines changes the schedule created.

In order to demonstrate this idea, consider the following four-job, three-machine problem and two policies used to dispatch the jobs waiting for processing (see Figure 5.1 for the task processing times).

Job	Due Date	Sequence of Machines to Visit:
J_1	8	Machine 1, Machine 2, Machine 3
J_2	12	Machine 1, Machine 2
J_3	17	Machine 2, Machine 1, Machine 3
J_4	15	Machine 2, Machine 3

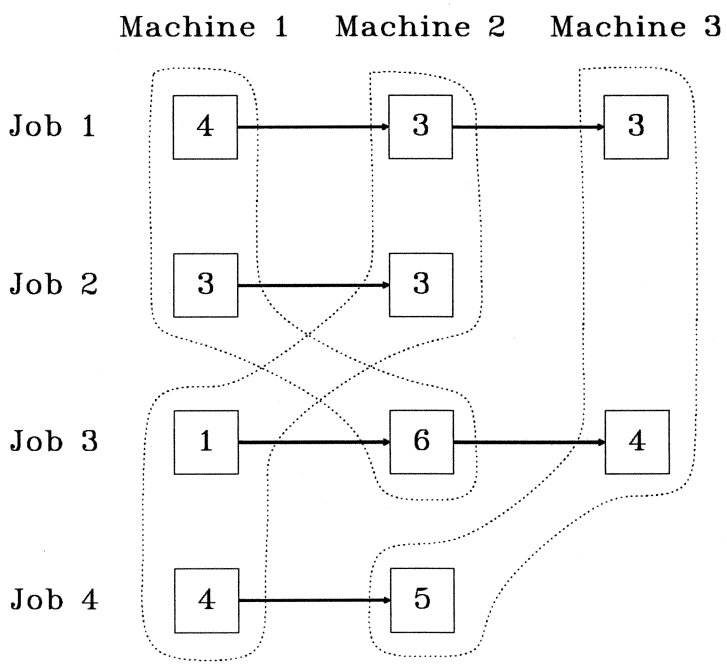
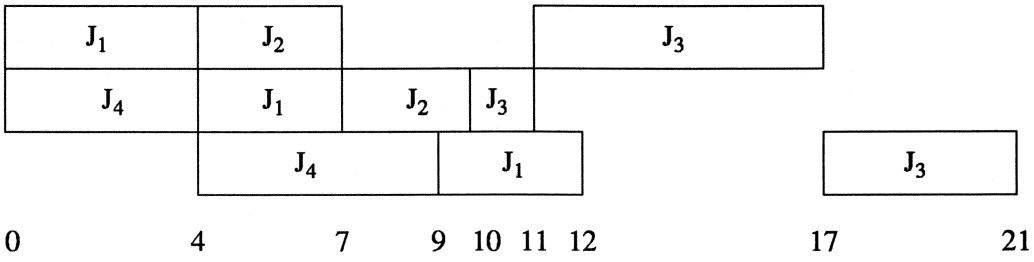


Figure 5.1. Task processing times for each job.
Tasks grouped by a dotted line are on the same machine.

Policy 1 = [EDD, EDD, EDD]. Under this policy, the tasks waiting for processing at M_1 are sequenced by earliest due date first. Thus, when both J_1 and J_2 are at M_1, J_1 is sequenced first since its due date is 8.

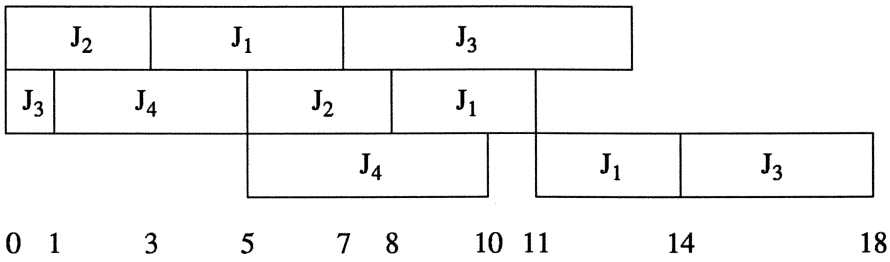
The dispatching rule for M_2 is EDD, so the tasks are sequenced by the job due date. When J_3 and J_4 are at M_2, J_4 is selected. After this task is completed, J_1 has arrived and its due date is also smaller than J_3. The same thing occurs when J_1 finishes and J_2 simultaneously arrives.

Policy 1 : [EDD, EDD, EDD]
Schedule: Makespan = 21.



In Policy 2, the altered dispatching rules, [SPT, SPT, SPT], change the selection of jobs. On M_1 the job J_2 has the shortest task processing time and is thus preferred. On M_2 the job J_3 has the shorter task processing time and is scheduled first.

Policy 2 : [SPT, SPT, SPT]
Schedule: Makespan = 18.



Obviously, the application of a different policy creates a different schedule with a different evaluation on any objective we could wish to measure (makespan, defined as the maximum job completion time, was chosen here only as an example).

These policies can be easily manipulated by a genetic algorithm. Traditional genetic algorithms must be modified for machine scheduling problems since the components of the strings (the jobs in the sequence) are not independent of each other. The heuristic space described above, however, consists of vectors that have independent elements. That is, the dispatching rule for the first machine does not affect what values the other elements can have. Thus, a crossover operation that breaks two strings (vectors) and joins the separate pieces yields offspring that are valid points in the search space.

5.3.2 A Genetic Algorithm for Global Scheduling

We will describe in this section the Genetic Algorithm for Global Scheduling (GAGS). This search procedure is the engine that finds a good schedule. After discussing it in this section, we will turn away and address the scheduling system that it drives.

GAGS consists of two primary components: the genetic search and the model of the shop floor. The interaction between these two functions is outlined in Figure 5.2. The genetic algorithm starts with a number of policies. (Each policy is a combination of dispatching rules, one rule for each machine. See Table 5.1 for a list of the dispatching rules that were used in the global scheduling procedure.) Each policy is evaluated by the schedule that is created if the jobs in the shop are dispatched according to this policy. The model of the shop floor is employed to build this schedule from the set of shop, job, and process information.

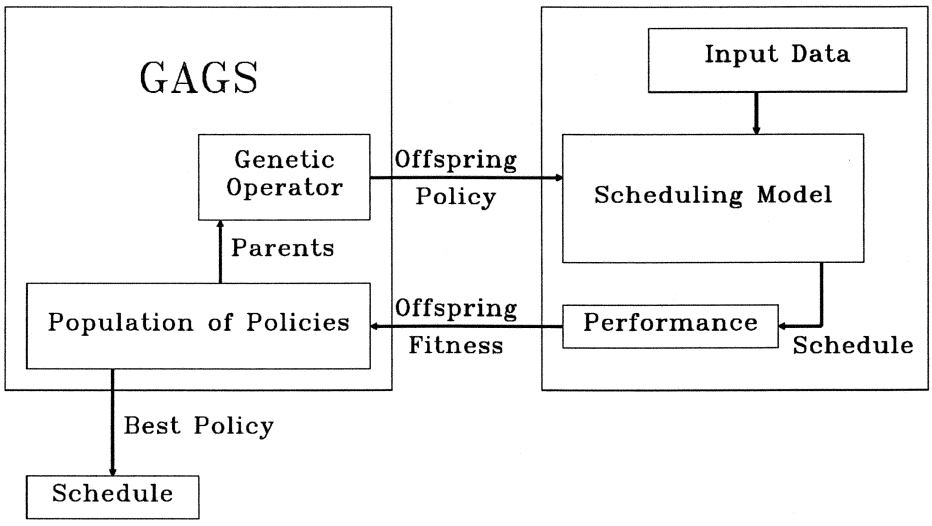


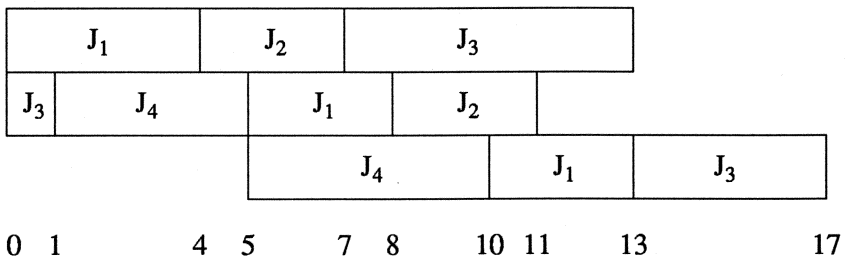
Figure 5.2. GAGS - Scheduling Model Interface

Table 5.1. List of dispatching rules considered.

SPT
Minimum Setup
EDD
Shop: Late-Setup Match-EDD
Slack per remaining operation
Modified Due Date
Longest Processing Time
Shortest Remaining Processing Time
Earliest Finish Time
First In First Out
Least work in next queue
EDD Look-ahead to burn-in
SPT Look-ahead to burn-in
Look-behind from burn-in

The genetic algorithm creates new offspring policies by combining two policies in the population (crossover) or changing a solitary parent (mutation). For instance, consider again the above three-machine problem and the two policies [EDD, EDD, EDD] and [SPT, SPT, SPT]. These policies were evaluated by creating schedules for all three machines. Their evaluations (on the makespan objective function) were 21 and 18. If we split each policy after the first rule and link the beginning of the first policy with the end of the second policy, we create the offspring policy [EDD, SPT, SPT], which the reader can confirm creates a schedule with a makespan of 17.

Policy 3 : [EDD, SPT, SPT]
Schedule: Makespan = 17.



These offspring policies are evaluated using the scheduling model. The genetic algorithm converges to a number of good policies, and the best is used to create a final schedule.

The model of the shop floor is the problem to be solved, and the genetic algorithm provides different heuristics. The system depends upon this model, therefore: only with a valid model can

a scheduling system create plans that match reality. For our project, the model was a deterministic simulation of the shop. It uses as input a set of resources that correspond to the equipment and staffing of the test area, a set of jobs that correspond to the lots that need processing in the test area, and expert knowledge about the production processes. The policy provided by the genetic algorithm sequences the jobs at each resource. Events in the simulation correspond to resources beginning work on a task and resources finishing tasks. Other events are added as necessary to control the simulation of the production process.

Note that the search is not affected by the complexity of the scheduling problem being solved. The genetic algorithm can find solutions to classical problems and to problems with previously-unconsidered characteristics. The genetic algorithm can take advantage of complexity by including dispatching rules designed for use in that environment.

5.4 Global Job Shop Scheduling

In this section we will discuss the manufacturing process and scheduling needs of the semiconductor test area for which we are trying to create good schedules. As we will see, the problem is quite difficult. This will lead us into the description of the design, implementation, and contributions of our global job shop scheduling system.

5.4.1 The Semiconductor Test Process

The manufacturing of semiconductors consists of many complex steps. This includes four primary activities: wafer fabrication, probe, assembly, and test. This research is concerned with the last facility. Although the routes of lots through the test process vary significantly over the many different types of products, general trends can be described (see Section 2.1 of this dissertation).

The area under study tests commercial-use semiconductor devices and consists of two domains: one dedicated to Digital products and another to Analog products. We directed our work toward the Digital side, where there are over 1400 product lines. The test area has three

shifts per day, five days a week. The resources in the Digital side include nearly sixteen electrical test heads and a dozen branders. The staffing on a normal shift consisted of nearly fifteen personnel in eight areas. The product mix changes continuously, and staffing and machine resources often change to reflect this.

Although a typical product may have a route that consists of over 30 numbered operations, only a fraction of those are steps where significant time and resources are required. In our job shop model of the test area, we concentrated on the following operations: electrical test (room, low, and high temperature), brand, burn-in, burn-in load and unload, visual-mechanical test, and document review. Due to the re-entrant nature of the process, an average lot will have twelve steps that need processing.

Most of the operations in the semiconductor test area are electrical or physical tests, performed in an automated fashion or by hand. During these tests, each device in a lot must be examined. Thus, the processing times depend directly upon the size of the lot. Also, the products in the test area have different package types. A package is the shell in which the semiconductor chip resides. These packages can be plastic or ceramic and come in many different sizes and styles. The package of a product also influences how the devices are tested.

In the job shop model, a work center for each of the resources in the shop (except the burn-in ovens) is a tester, machine, or operator that needs to be scheduled. Each type of work center has a list of the operations that can be performed at the station. Distinct resources that perform the same operations are modelled as separate stations, each with a queue of jobs that next need processing at that station. These queues are sequenced by the dispatching rule for that station.

Two unique operations in semiconductor testing are electrical test and burn-in. Electrical test operations have sequence-dependent setups, and burn-in is a bulk-service process.

The sequence-dependent setups at electrical test are a result of the test requirements. First, a handler must be attached to the test head to automatically feed the devices in the lot. However, due to the physical attributes of the packages, different handlers are required for different packages. Testing also occurs at various temperatures, which requires additional equipment. The

amount of setup is thus affected by the setup of the previous test, since it will be necessary to put into place new equipment only when the previous setup was different.

The other unique operation is burn-in. At this operation, the devices in the lot are placed into one of a number of ovens for a fixed period of time. These times range from 24 hours to over 4 days. Many lots can be burned-in at once, and the capacity constraint is more often on the availability of burn-in boards than on the space in the ovens. Additionally, after the boards are removed from the oven, the devices are unloaded from the boards, and the lot must undergo an electrical test within 96 hours to locate any faulty devices. The combination of bulk service, secondary capacity constraints, and operation deadlines requires special modeling.

Access to burn-in is controlled by the burn-in load work center, since each lot must be loaded onto burn-in boards before being placed in the oven. If sufficient boards are not available, the lot cannot be processed. After burn-in load, the lot moves directly to the burn-in oven and begins the burn-in period. The factory control system provides information on which day the lot should be removed from burn-in. On that date, the lot is scheduled for unload, and the burn-in boards become available for another lot.

As Lee *et al.* (1993) mention, the test area has five features that distinguish it from classical job shop scheduling:

1. Sequence-dependent setup times and re-entrant product flows,
2. Machines with different scheduling characteristics,
3. Complex interactions between machines,
4. Dynamic production environment, and
5. Multiple, conflicting objectives.

These characteristics make effective scheduling of the semiconductor test area a difficult task. They also encourage us to design a global scheduling system that can search for good solutions without being obstructed by complexity.

5.4.2 The Previous Scheduling System

In this subsection we briefly discuss how scheduling was done previously in the test area using dispatching rules and some drawbacks to this system. (The development of this system is described in Lee *et al.* 1993.)

The test area was using a set of dispatching stations to sequence the lots awaiting processing at each work center at the beginning of the shift. During the shift, this ordered list of lots was updated by using the dispatch station to resequence the lots in the queue, since more lots may have arrived. The dispatching stations and rules are part of a software module called Short-Interval Scheduling (SIS).

SIS is a component of WORKSTREAM, a product of Consilium, Inc. (Consilium, 1988). WORKSTREAM is the test area's computer-integrated manufacturing (CIM) system and is implemented on the corporate VAX mainframe. Transactions such as processing a lot or beginning a setup are logged into WORKSTREAM, which maintains information about the current status of each lot and each machine. This information is used by SIS to sequence the lots waiting for processing.

In addition to WORKSTREAM, the company uses higher-level systems such as Activities Planning & Dispatching (AP/D) to match the production lots to customer orders and IMPReSS to determine the amount of product that each area of the company (including the test area) should be produced each week. All of these systems provide critical data about the lots to be processed and the characteristics of the test area. We will need this information to model the test area.

While SIS has led to better scheduling in the test area, it has a number of disadvantages related to the structure and capabilities of the CIM system. The primary obstacle is that dispatching rules are making local decisions (even when they attempt to look-ahead or look-behind). Thus they are unable to know how their decisions affect the work at other areas in the test area. And they are unable to advantage of information that may lead to better dispatching

decisions. In addition, the dispatching rules are unable to use information about processing and setup times.

Since the implemented dispatching rules cannot use global information, we began to consider job shop scheduling procedures. While a number of optimization procedures have been suggested (see Section 2), these have concentrated on the classical job shop problem of minimizing makespan. In particular, the more well-known procedures use the disjunctive graph to represent the problem. Unfortunately, manufacturing environments like semiconductor test area often have more complicated problems that require more complicated scheduling models.

We wanted to make use of global information in a complex production process and to search for a better schedule. Therefore, we decided to implement the genetic algorithm for global scheduling. We described the primary characteristics of this procedure in Section 3. In the remainder of Section 4 we will describe the development of our global job shop scheduling system.

5.4.3 Scheduling Needs

Over the course of a number of years, the scheduling system in the semiconductor test area progressed from manual dispatching to an integrated rule-based decision support system. As we mentioned above, this system had significant limitations, since it was based on dispatching rules. However, the planners in the area need to know in what order the lots waiting at a station should be processed, and this system gave them this information. They would also like to know when these lots will be completed.

In addition to these tactical decisions, the planners need to have some idea of how their limited resources (in both personnel and equipment) on the shop floor affect the performance of the shop in relation to meeting the shop goals. Finally, they need to measure the performance of the personnel on the floor.

The managers of this test area have two objectives. The first is to meet customer due dates. The company stresses customer satisfaction, and the responsibility of the test area is to ship the

required number of parts to the customer by the requested time. The test area is measured on the number of delinquent line items, the number of customer orders that are not shipped on-time. Thus, the most important of the two primary objectives is to minimize the number of tardy jobs.

The second objective is to test as much product as possible. This goal is both positively and negatively correlated with the first goal. If the area can test product more quickly, it is more likely to meet customer due dates. However, the push to increase the quantity tested can interfere with the need to process lots on-time (since the lots with the earliest due dates may be those with the largest processing times). At the time we were working on this project, the test area had extra capacity, so we considered the objective of minimizing flowtime to be a subordinate one.

5.4.4 Scheduling System Design

We have looked at some of the complexity that occurs in scheduling a semiconductor test area. These characteristics make the problem in the semiconductor test area much more difficult than the job shop problems previously considered. At the same time, however, the CIM systems in place are able to provide the type of area-wide information needed to create a schedule. Therefore, we are motivated to try a new approach: a global scheduling system that uses a genetic algorithm to find good schedules in the presence of shop floor complexity. In this section we will describe the basic design features.

This system includes a simulation model of the test floor and an optimization procedure that can search for schedules using a genetic algorithm. The search finds a schedule that is better than any schedule that a predefined set of dispatching rules could create. This schedule specifies the order in which the lots should be processed and the times at which the operations should be done.

The system can be used to react to unforeseen events that might occur during the course of a shift by including new information and producing a new schedule; for example, a machine may fail, or an important lot may arrive. The system also serves in a decision support manner, allowing the planners to determine the effects that changing resources has upon the schedule.

Through the use of fair processing times in an accurate model, the schedule can be used as a target for the shift; the work completed by the shift personnel can be compared to the work on the schedule.

The foundation for the system is the test area's extensive CIM system, which will provide the information necessary to model the test area. Finally, the global scheduling system is under the control of production planners, who use the computing power of the system to evaluate alternative plans and to create a detailed schedule for the shop floor.

The most important component of a global job shop scheduling system is the model of the shop floor. Only with a valid model can a scheduling system create plans that match reality. For our project, the model was a deterministic simulation of the test area. It uses as input a set of resources that correspond to the equipment and staffing of the test area, a set of jobs that correspond to the lots that need processing in the test area, and a set of dispatching rules that sequence the jobs at the resources. Thus, the model creates a schedule for the current scenario.

This is the central relationship in the system. Because the schedule depends upon the dispatching rules, the optimization procedure in this global scheduling system is a search of the combinations of dispatching rules in order to find a good schedule (one that meets management goals efficiently). A genetic algorithm is employed to search over the rules, evaluating each set by running the simulation with those rules and measuring how well the schedule performed at keeping jobs on-time. This part of the system is called the Genetic Algorithm for Global Scheduling, and it is discussed in Section 5.3 of this chapter.

5.4.5 Information Requirements

The scheduling system makes use of data from a number sources inside and outside the test area's CIM system. In this subsection we will mention the types of information that are used in the system. See Figure 4 for a diagram of how the primary data structures interact.

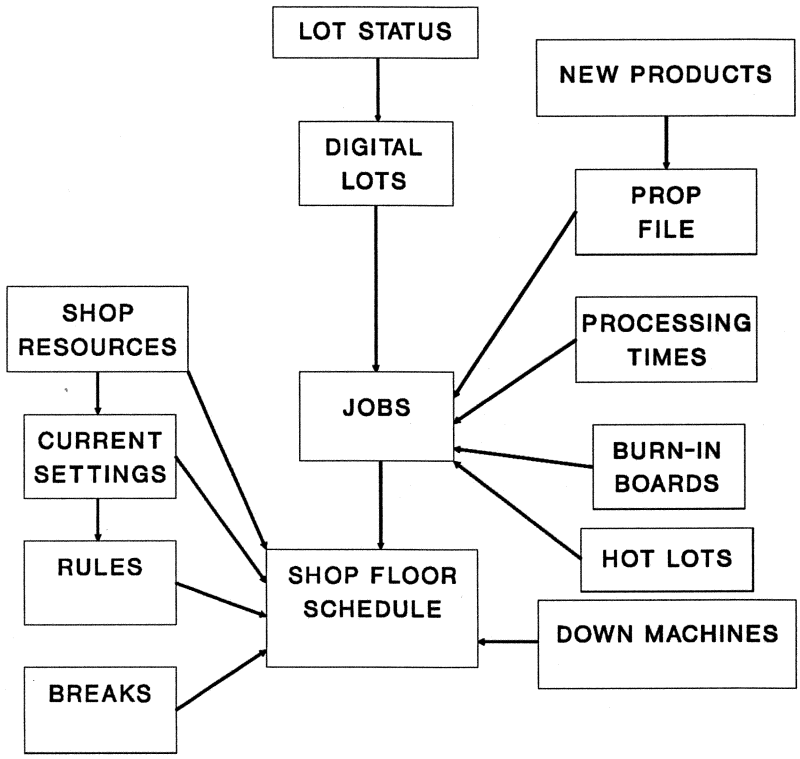


Figure 4. Data Structures

Three general types of files can be described: data extracted from the CIM, data collected by the project team, and data entered by the user. The first category includes data that is fairly stable and data that constantly changes. Stable data includes information about the product routes. This Product-Operation (PROP) file matches each product to the operations that need to be performed. This file is updated weekly as new products are added to the product mix. Other files of extracted information list package types, burn-in board requirements, and tester requirements.

The dynamic data is primarily the lot status information. The status of a lot changes as it undergoes different operations. Before each shift the CIM system takes a snapshot of the status for each lot in the test area. Our system extracts lot status information for each digital lot from this report. Although the status of a lot continues to change during the day, our system cannot see the changes until the next report.

The project team also collected data on processing times and test area resources. The collection of processing times will be discussed in Section 4.7. The resources files matches each type of resource (tester, brander) to the operations that it can process.

Finally, the user controls other information needed for the scheduling system, including the current test area resources (how many of each type), the dispatching rules to use, the breaks schedule, the status of any down machines, and the arrival of any new hot lots.

5.4.6 Implementation of Global Scheduling

The implementation of the global scheduling system followed the design of the system and the collection of data sources. In this subsection we will describe how we implemented the system and how the test area creates schedules.

After creating the scheduling model and the genetic algorithm off-line, we installed the basic programs on the corporate mainframe. We began testing the system and developing the utilities that collect the necessary job, shop, and process information. Working closely with the test area personnel, we began to run schedules each day. Feedback from these initial schedules led to improvements in all parts of the system. A technology transfer session formally introduced the system capabilities to the test area personnel, and soon they began to run the system themselves. The project team created user interface programs and documentation so that the test area would be able to use, understand, and maintain the system.

The scheduler performs shift scheduling for the Digital side of the test area. Output from the system includes the shift schedule, which lists each work center, the lots to be scheduled, and the approximate start and finish times for each operation. Also available are post-shift performance reports that compare the production of the test area to the schedule.

User interface programs make it easy for the users to collect and modify data, create a schedule, view the output, and create performance reports.

Creating a schedule consists of a number of steps. First, the current status of all of the lots in the test area is determined from the pre-shift lot status report. Non-digital lots and lots in a stores operation are ignored.

Second, the lot status information is combined with the PROP file, which converts lots into jobs. The job consists of relevant lot information and a number of operations. The lot information includes product class, due date, and lot quantity. For each operation a processing time is computed along with a remaining cycle time.

Third, the user must check the current shop settings on resources, dispatching rules, and breaks. The user is shown the default settings and has the opportunity to make any changes.

Fourth, the schedule is created using the job, shop, and process information collected to this point. This schedule can be created using the genetic algorithm, which searches over combinations of dispatching rules, or the dispatching rules which the user chose. In the latter case, no search is performed; the scheduling model uses the user's rules and outputs the schedule that these rules yield.

The genetic algorithm begins with a population of randomly-created policies. The genetic operators shift the population towards more successful policies (measured on their ability to create a schedule with more on-time jobs). The algorithm stops after a fixed period of time; this time was selected after experimentation into the trade-off between schedule quality and search effort.

In addition to creating a shift schedule, the system creates a number of other files that are used to drive performance reports. There are three performance reports: shift summary, daily summary, and daily detailed. The summary reports compare what was done with what was scheduled. The scheduled operations for each machine are compared to the operations that have been those lots. The detailed report compares processing times. The scheduled processing times are compared to the actual times that are derived from daily lot history files.

If the user makes no changes to the default shop settings, the entire process can take as little as ten minutes. This compares to the 15 minutes necessary in SIS to sequence just the lots

waiting at one work center. Therefore, the process can be done as part of the pre-shift planning, and the user has ample opportunity to update any data and to experiment with different shop settings.

The system is currently being used by the test area personnel, who will soon be expanding the system to include the Analog side of the floor. After discussing some of the issues that our implementation raised, we will describe the contributions of our system.

5.4.7 Implementation Issues

Researchers engaged in projects like this one often encounter difficulties that are considered in no classroom and occasionally learn things that are in no textbook.

Processing times. One primary problem was a result of our attempt to model an uncertain process with a deterministic procedure. We wanted to use processing times that were realistic but that also set fair targets for the test floor. Thus, we needed good estimates of what the processing times should be. The collection of these estimates was an important concern. We had historical data from the factory control system, which monitors when each lot begins and ends each operation. We began with an average for each operation, but this average ignored the variability in processing time due to lot quantity. We then took the historical data for the subset of products and fitted a linear regression to the data in an effort to predict processing time from lot size. The regression was run for different package types in order to remove another source of variability.

Perfection. We learned that perfection is an unattainable goal in a setting as complex as semiconductor test. However, we also learned that falling short of perfection is sufficient if our system improves the ability of the facility to satisfy customer demand efficiently. From the beginning, management knew (better than the project team did) that we could not hope to capture all of the activities that occur. This attitude allowed us to build a significant model instead of being overwhelmed by the complexity of finding an optimal solution. With modeling and controlling the test area perfectly beyond our reach, we concentrated on developing a system

which meets the needs of the test area planners and provides them with a tool which they can use to intelligently manage their facility.

Simplicity. A significant feature of the implemented system is the ability of production planners to modify data related to resources and other factors. This called for a number of modules that could manipulate the data without requiring undue effort from the planners. These modules were user-friendly (easy to understand and fail-safe), flexible routines that the planners felt comfortable using. We feel that this has contributed to the success of the implementation.

Timeliness. Next to the estimates of processing times, accurate information about lot status was the hardest data to gather. Since there are hundreds of lots in the facility's inventory, determining which lots are waiting at which stations is not a trivial task. The system depends upon WIP extracts that were run from the factory control system before each shift. Although the planners have the ability through the factory control system to check on the status of individual lots, it was not possible to develop procedures to gather the status of the lots during the middle of the shift.

Because the system could not gather the current lot status during the shift, it could only react to unforeseen events by rebuilding the original schedule from the beginning of the shift and incorporating the new information about machine breakdowns and lot arrival. Searches to find good schedules were not possible.

Optimization procedure. A number of other researchers continue to work on ways to solve the job shop scheduling problem. Although we cannot guarantee that our heuristic space genetic algorithm is the best procedure for finding a good shop schedule, a search over dispatching rules seemed to be an ideal approach for the needs and requirements that we faced, however.

The system makes use of the scheduling model and a separate genetic algorithm program. The two procedures are distinct modules developed and modified independently. This gave us a great deal of flexibility as we began to test and implement the scheduling system. We chose the genetic algorithm to gain the power of its parallelism and its simplicity and to avoid the problems

of local searches. The test area personnel could understand how policies combined to form new policies and how these policies could be used to create schedules.

Due to the impossibility of finding an optimal schedule to the dynamic job shop scheduling problem, it is sufficient for our heuristic space search (an approximation algorithm) to determine a quality schedule quickly.

5.4.8 Contributions of Global Scheduling

In this section we will discuss the benefits of our implementation of the global scheduling system. While primarily unquantifiable, the gains are real.

The test area now has a tool that can substantially improve scheduling with a combination of global information, detailed scheduling model, genetic search for good schedules, reaction to unforeseen events, and short-term performance reports.

The global system has a number of strengths compared to SIS, the previous scheduling system. As a centralized procedure, it can make use of information from around the test area including processing times, queue lengths, current setups, resource availability, and job arrivals. The genetic algorithm searches for a schedule that has more on-time lots than a schedule created by any fixed set of dispatching rules.

As a shift scheduler, the system gives the test area a plan that shows how work at one area depends upon work at another and that can be used to measure the performance of that shift. In fact, the ability to accurately model the test area and compute a shift schedule was just as important to the test area as the ability to find better schedules. The simulation component of the system allows the test area planners to forecast how modifying the level of available resources will affect the output of the test area. The system can also react to certain unforeseen events that may necessitate a change in the schedule.

As mentioned earlier, the product mix in the semiconductor test area is constantly changing. This variety is a significant factor on the test area performance. Thus, we are unable to measure any long-term quantitative benefits. The test area managers have expressed their

confidence that the system will lead to improvements in the performance of the test area through improved schedules and planning tools.

5.5 Chapter Summary

In this chapter we have described the development of a global scheduling system that uses a genetic algorithm to find good schedules. This system has been successfully implemented in a semiconductor test area. Controlling the test area is a complex, dynamic job shop scheduling problem where it is difficult to meet the management objectives of satisfying customer demand on-time and increasing throughput. Our scheduling system uses the extensive data available in the CIM databases in order to simulate the operation of the test area. This system uses a genetic algorithm to search for combinations of dispatching rules that yield schedules that are better than the sequencing that could be done with fixed dispatching rules. The primary practical accomplishment of this research is the implementation of an advanced job shop scheduling system in a manufacturing environment. Moreover, this implementation makes use of a new heuristic procedure that searches the combinations of dispatching rules to find a good schedule. Thus it is able to adapt each shift to changing conditions in the jobs to be scheduled and the shop resources.

This approach could be extended to any manufacturing area that has a complicated shop scheduling problem and a computer-integrated factory control system that can supply the necessary data for such a global scheduling system. In fact, the semiconductor manufacturing firm where we have implemented the system is considering exporting this system to other areas. Additionally, other optimization techniques may be useful in finding good schedules.