Journal of Manufacturing Systems
Vol. 19/No. 4
2000

Predicting the Performance of Rescheduling Strategies for Parallel Machine Systems

Guilherme E. Vieira (gev@engineer.com), **Jeffrey W. Herrmann** (jwh2@eng.umd.edu), and **Edward Lin** (lin@eng.umd.edu), Institute for Systems Research, University of Maryland, College Park, Maryland, USA

Abstract

In dynamic, stochastic manufacturing systems, production planners and manufacturing engineers can benefit from understanding how rescheduling strategies affect system performance. This knowledge will help these experts design and operate better manufacturing planning and control systems. This paper presents new analytical models that can predict the performance of rescheduling strategies and quantify the trade-offs between different performance measures. In the parallel machine systems under consideration, jobs of different types arrive dynamically, and setups occur when production changes from one job type to another. Three rescheduling strategies are studied: periodic, hybrid, and event-driven based on the queue size. The scheduling algorithm groups jobs of the same type in batches to eliminate unnecessary setups. The analytical models require less computational effort than simulation models, and experimental results show that they accurately estimate important performance measures like average flow time, machine utilization, and setup frequency.

Keywords: *Scheduling/Rescheduling, Reactive Scheduling, Analytical Modeling*

1. Introduction

In dynamic, stochastic manufacturing systems, unpredictable events like breakdowns, expedite orders, quality problems, and material shortages occur during processing. Although careful scheduling coordinates activities to maintain productivity, these disruptions can render the desired schedule infeasible. Rescheduling attempts to diminish the loss by creating a new schedule that more accurately reflects the current state of the production system.

Many rescheduling strategies are used in practice, and many others have been proposed. Unfortunately, the impact of these strategies is not well understood. Few workers have tried to model analytically the performance of rescheduling strategies. However, many have used simulation. One advantage of using analytical models instead of simulation is the ability to estimate the system performance very quickly.

The performance of a rescheduling strategy is the performance of the manufacturing system when that strategy is used to control production. The main performance measures of interest in this work are the average flow time (or cycle time), machine utilization, and setup frequency. This paper presents analytical models that estimate system performance quickly without having to construct simulation models or spending time running experiments. One can use these models to make important operational decisions like selecting a rescheduling strategy or defining the rescheduling frequency. These models also quantify the trade-offs between different performance measures.

This paper describes analytical models that predict the performance of rescheduling strategies for parallel machine systems operating in an environment where jobs of different types arrive dynamically for processing and setups occur when production changes from one job type to another. (A job is equivalent to a production order. It contains a number of parts—its size—and can be grouped with jobs of the same type to form a batch of jobs.)

Three rescheduling strategies are considered in this study: periodic, hybrid, and event-driven based on the queue size. Under the periodic strategy, rescheduling occurs regularly with a constant time interval (the rescheduling period) between consecutive rescheduling events. No other events trigger rescheduling. Under the hybrid strategy, rescheduling occurs not only periodically but also whenever a machine failure or repair occurs. Under the event-driven based on the queue size (EDQS) strategy, rescheduling occurs when the number of jobs that have arrived since the last rescheduling event reaches a specific threshold. No other events trigger rescheduling.

The rescheduling strategies use a straightforward, heuristic scheduling algorithm that groups jobs of the same type into a batch to eliminate unnecessary setups. It sequences the jobs within each batch, prioritizes the batches, and assigns each batch to the first available machine. Although this paper presents a specific algorithm, the analytical models would apply to other scheduling algorithms that eliminate unnecessary setups (see Section 6).

2. Background

There is considerable variation, from one source to the next, in the definitions of terms related to rescheduling. Thus, it is necessary to define and use the terms precisely. Scheduling assigns times and resources to the tasks of a plan that must satisfy a set of domain constraints. Depending on the availability of the jobs prior to the schedule creation, scheduling systems can be classified as static or dynamic. In static scheduling, all jobs are identified when creating the schedule, and the production sequence does not change during processing. In dynamic scheduling, jobs continue to arrive dynamically, and scheduling must occur repeatedly.[1] Dynamic scheduling contains two general approaches: dispatching rules and rescheduling (reactive scheduling).

In a dispatching rule system, jobs are not scheduled for processing. Each resource has a queue of jobs waiting for processing. When the resource becomes available, the dispatching rule prioritizes these jobs, and the job with the highest priority leaves the queue and begins processing. When finished, the job moves to the queue of the next required resource. Because of its relatively limited scope, a dispatching rule cannot guarantee that the system will operate at a good performance level.[2] On the other hand, no effort is spent creating schedules, the resources are able to work independently, and the dispatching rules react quickly. Thus, unexpected events cause limited disruptions. In some cases, queueing theory can be used to analyze production systems controlled by dispatching rules. For example, a single-server system with multiple job types and a nonzero setup time can be modeled as a polling system[3] if the server rotates among job types and serves all of those jobs before going to the next. Unfortunately, these techniques cannot be applied to rescheduling strategies.

Rescheduling alters the schedule being used and creates a new schedule that reflects the new shop status and production requirements. In dynamic, stochastic manufacturing systems, rescheduling is practically mandatory.[1,2,4-9] Compared to dispatching rules, rescheduling approaches can create excellent schedules that can generate better system performance. One shortcoming is the computational effort required to execute scheduling algorithms (which are often attempting to solve NP-complete problems). However, as more computational power becomes widely available, generating excellent schedules is quite feasible. This was certainly not true in the past, when computers were much slower and more expensive.[10]

Many events can disrupt a production schedule and trigger rescheduling[5]: breakdowns, job arrival or cancellation, job priority (or due date) changes, quality problems, over- or under-estimating process times, material shortages, and being behind or beyond the schedule because of transportation, tools, or personnel delays. Wu, Storer, and Chang[4] and Abumaizar and Svestka[8] have used a generalized approach that considers machine breakdown as the only rescheduling trigger. They claim that most rescheduling triggers can be modeled as machine breakdowns since they disrupt the processing of jobs on one or more machines for some period of time. They use this observation to simplify their studies of rescheduling systems.

A number of authors have proposed rescheduling strategies for a variety of scheduling environments. Church and Uzsoy[2] and Wu, Storer, and Chang[4] studied, for instance, the single-machine problem. Church and Uzsoy also considered the parallel-machine case. The general job shop problem was studied by Dhingra, Musser, and Blankenship,[7] Li, Shyu, and Adiga,[5] Kim and Kim,[11] Abumaizar and Svestka,[8] Jain and ElMaraghy,[6] and Mehta and Uzsoy.[12] Zweben et al.[13] described rescheduling problems in space shuttle ground processing, a project scheduling problem. Church and Uzsoy developed a hybrid event-driven rescheduling strategy for single- and parallel-machine models with dynamic job arrivals. Their strategy reschedules the facility periodically, taking into account work that is already in the system. Regular events occurring between routine rescheduling are ignored until the next rescheduling moment. However, when an event is classified as an exception, immediate action is taken, with the entire facility being rescheduled and the resulting schedule implemented until the next schedule generation point. To create a schedule, the system uses the Earliest Due Date (EDD) rule to minimize maximum lateness. Church and Uzsoy's models do not consider other system performance measures, different job types, or setups.

Most of these studies used simulation experiments to evaluate the effectiveness of the proposed approaches. For example, Sabuncuoglu and Karabuk[14] used simulation to evaluate rescheduling strategies that address the effects of machine breakdowns and processing time variations. Their results indicated that periodic rescheduling with an appropriate period would be sufficient to cope with interruptions. They use mean tardiness and makespan to measure system performance but do not try to model these measures analytically. Like many others, they also ignored the effect of setups.

Vieira, Herrmann, and Lin[3] have studied a single-machine system and developed analytical models to estimate system performance. In that work, two rescheduling strategies were considered: periodic and event driven based on queue size. Their results show that the analytical models can accurately predict the performance of a single-machine system operating under those rescheduling strategies. This paper extends that study by investigating parallel machine systems, which have more complex rescheduling strategies.

3. Parallel Machine System

This section introduces the notation needed to model the parallel machine system and describes the scheduling algorithms that the rescheduling strategies use.

Jobs of different types and sizes arrive dynamically for processing. There are J job types. The arrival process for job type j is a Poisson process with an arrival rate of θ_j jobs per time unit, $j = 1, ..., J$. The size of a type j job is a random integer uniformly distributed between 1 and $2\overline{Z_j} - 1$, so the average equals $\overline{Z_j}$.

Each job requires processing on any one of the N_m parallel machines. The machine speed M_n is the number of parts that machine n can process per time unit, $n = 1, ..., N_m$. Therefore, the processing time is the job size divided by the machine speed. Note that this does not depend directly on the job type.

A machine requires a setup to switch it from one job type to another job type. Typically, this is needed to prepare the machine for different tools, chucks, or fixtures, or to execute other changeover activities like cleaning. A setup on machine n requires S_n time units.

Machines fail unexpectedly. For machine n, the time to failure is an exponentially distributed random variable with mean MTTF_n (Mean Time To Failure). The time to repair is an exponentially distributed random variable with mean MTTR_n (Mean Time To Repair).

Performance Measures

Of the many performance measures relevant when scheduling manufacturing systems, this paper focuses on the following measures:

- The average flow time $\overline{F_t}$ is the average flow time of all jobs. The flow time (or cycle time) of a job is the length of the interval between the time the job arrived and the time the job completes processing.
- The rescheduling frequency $\overline{F_r}$ is the average number of rescheduling events per time unit.
- The average setup frequency $\overline{F_s}$ is the average number of setups per time unit per machine.
- The average schedule execution time $\overline{E_t}$ is the average length of the interval between the time the first job or setup in a schedule begins and the time the last job (in that schedule) completes.
- The average machine utilization $\overline{U_m}$ is the average fraction of time that a machine is undergoing a setup, processing a job, or being repaired.
- The average setup time percentage β_s is the average percentage of time that a machine is undergoing a setup.
- The average processing time percentage β_p is the average percentage of time that a machine is processing a job.
- The average repair time percentage β_r is the average percentage of time that a machine is being repaired.
- The average idle time percentage β_i is the average percentage of time that a machine is idle.

Scheduling Algorithms

A schedule is a set of the sequences (of setups and jobs) that each machine will follow. Thus, for a system with N_m machines, the schedule has N_m sequences. The rescheduling strategies require two similar, but different, scheduling algorithms: the *normal* scheduling algorithm and the *emergency* scheduling algorithm. The periodic and EDQS rescheduling strategies use only the normal scheduling algorithm. The hybrid rescheduling strategy uses the normal scheduling algorithm at the beginning of each rescheduling period and the emergency scheduling algorithm when a machine fails or is repaired.

Normal Scheduling Algorithm

The normal scheduling algorithm eliminates any unnecessary setups and attempts to balance the load among the machines. The algorithm follows these steps:

1. Consider the following jobs: those that were scheduled but have not begun processing and those that have arrived since the last normal rescheduling event. (The latter jobs have never been scheduled.)
2. For each job type present, form a batch of those jobs. Within each batch, sequence the jobs by FIFO; that is, a job with an earlier arrival time should precede a job with a later arrival time.
3. For each machine, calculate the expected available time. Unless that job type is absent, start each machine's sequence with the batch of jobs that have the same type as the last job processed (or the job currently being processed). Add the batch processing time to the machine's expected available time.
4. Sequence the remaining batches by FIFO; that is, a batch whose first job has an earlier arrival date should precede a batch whose first job has a later arrival date. Assign each batch, in turn, to the machine with the smallest expected available time. Add the machine setup time and batch processing time to the machine's expected available time.

For machine n, Step 3 calculates the expected available time, T_n, based on the machine state at t, the time at which the rescheduling event occurs. Let π_n be the time that the current job started processing, let ρ_n be the time that the current setup began, and let Z_n be the size of the current job. Note that if the machine is being set up, the job that requires the setup is not eligible for rescheduling. Also, the actual repair time is not known, so the algorithm uses the expected value (MTTR_n) to estimate when the machine will be available, as follows:

- If machine n is idle, $T_n = t$
- If machine n is being set up, $T_n = \rho_n + S_n + Z_n/M_n$
- If machine n is processing a job, $T_n = \pi_n + Z_n/M_n$
- If machine n is under repair, $T_n = \pi_n + \mathrm{MTTR}_n + Z_n/M_n$

When, in Steps 3 or 4, a batch is assigned to the machine, the expected available time T_n is increased. The batch processing time is the sum of the constituent jobs' processing times.

To illustrate how the normal scheduling algorithm works, consider the following two-machine example that has four job types: A, B, C, and D. Let each job be denoted by its type and arrival time as *type(arrival time)*. A rescheduling event occurs at time t = 22, and there are three jobs {A(10), C(8), B(12)} that were not processed in the last schedule. Eight new jobs {B(15), D(15.5), B(16), A(18), D(19), C(20), A(21), C(21.5)} have arrived since the last rescheduling event. Machine 1 is currently processing a job of type A. Machine 2 is idle but last completed a job of type D. The scheduling algorithm will then proceed as follows (see also *Figure 1*):

1. The following jobs need to be scheduled: {B(15), D(15.5), B(16), A(18), D(19), C(20), A(21), C(21.5), A(10), C(8), B(12)}.
2. Form four batches and sequence the jobs within each batch: {A(10), A(18), A(21)}, {B(12), B(15), B(16)}, {C(8), C(20), C(21.5)}, {D(15.5), D(19)}.
3. Assign the batch of type A jobs to Machine 1. Assign the batch of type D jobs to Machine 2.
4. Sequence the remaining batches: {C(8), C(20), C(21.5)}, {B(12), B(15), B(16)}. Assign the batch of type C jobs to Machine 2. Assign the batch of type B jobs to Machine 1.

Emergency Scheduling Algorithm

The emergency scheduling algorithm is similar to the normal scheduling algorithm. There are two primary differences. First, in Step 1, the only jobs con-

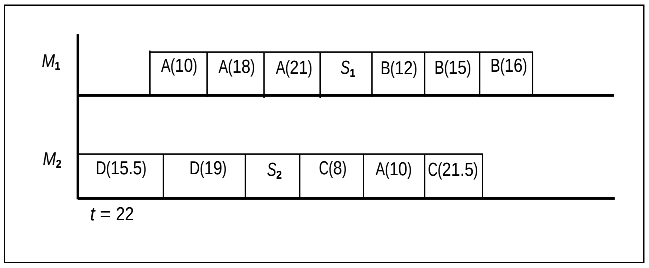

Figure 1
Example Schedule

sidered for rescheduling are those that were scheduled but have not begun processing. Second, in Step 4, the expected available time of any machine being repaired is set to some arbitrarily large value. Thus, the algorithm assigns no more batches to a down machine.

The reduced scope of the emergency scheduling algorithm reflects the fact that, in many production environments, two types of rescheduling occur. The first type, often done periodically (daily or weekly), releases new orders and involves documentation and other tasks associated with order release. The second type, done when needed, doesn't release new orders but instead reassigns work to offload a down machine or utilize a now-available one. The normal scheduling algorithm corresponds to the first situation, while the emergency scheduling algorithm corresponds to the second.

To illustrate the emergency scheduling algorithm, consider the example. Suppose Machine 1 (M_1) fails while processing the third type A job. The emergency scheduling algorithm reassigns the type B jobs to Machine 2. If M_1 is repaired before Machine 2 begins the setup for the type B jobs, the emergency scheduling algorithm will return them to Machine 1.

4. Analytical Models for the Rescheduling Strategies

Before addressing the specifics of the rescheduling strategies, the following expressions are presented to define quantities that will be used in the analytical models.

Total job arrival rate $\theta^T = \sum_{j=1}^{J} \theta_j$

Total part arrival rate $\lambda^T = \sum_{j=1}^{J} \theta_j \overline{Z_j}$

Average job size $\overline{Z} = \frac{\lambda^T}{\theta^T}$

Availability of machine n is A_n. $A_n = \frac{MTTF_n}{MTTF_n + MTTR_n}$

Capacity of machine n is $\mu_n = M_n A_n$

Average setup time $\overline{S} = \frac{1}{N_m} \sum_{n=1}^{N_m} S_n$

Average mean time to failure $\overline{MTTF} = \frac{1}{N_m} \sum_{n=1}^{N_m} MTTF_n$

Average mean time to repair $\overline{MTTR} = \frac{1}{N_m} \sum_{n=1}^{N_m} MTTR_n$

Total machine speed $\overline{MTTR} = \frac{1}{N_m} \sum_{n=1}^{N_m} MTTR_n$

Average machine speed $\overline{M} = \frac{M^T}{N_m}$

Total machine capacity $\mu^T = \sum_{n=1}^{N_m} \mu_n$

Average machine capacity $\overline{\mu} = \frac{\mu^T}{N_m}$

The analysis requires the following conditions. The number of different job types that the system can manufacture is greater than the number of parallel machines; that is, $J > N_m$. If not, then setups can be avoided entirely by assigning each job type to one (or more) dedicated machines. The resulting rescheduling strategies would be trivial.

The total part arrival rate must be less than the total machine capacity: $\lambda^T < \mu^T$. Otherwise, the system is not stable. In addition, the following condition must be true:

$$\max\left\{\theta_j \overline{Z_j} : j = 1, \ldots, J\right\} < \min\left\{\mu_n : n = 1, \ldots, N_m\right\}$$

Otherwise, a high-volume job type could be assigned to a machine that is not fast enough to process it.

Periodic Rescheduling Strategy

This section presents the analytical models for the periodic rescheduling strategy, which has a single parameter, the rescheduling period h. Rescheduling occurs every h time units. The following models give estimates for the system performance measures that Section 3 introduced.

Because the rescheduling period is constant, the average rescheduling period $\overline{P_r}$ equals h, and the average rescheduling frequency is its reciprocal, as follows:

$$\overline{F_r} = \frac{1}{h} \tag{1}$$

$$\overline{P_r} = h \tag{2}$$

The probability that at least one job of type j will arrive during a rescheduling period is $1-e^{-\theta_j h}$. Since there are J independent job types, and each one requires a setup, the expected gross number of setups is $\sum_{j=1}^{J}\left(1-e^{-\theta_j h}\right)$.

However, because the scheduling algorithm eliminates unnecessary setups, N_m job types will not require a setup. The identity of the unnecessary setups varies from one rescheduling event to the next. It is assumed that all job types have an equal chance ($1 - N_m/J$) to be one of the necessary ones. The expected total number of setups N_S depends on this probability and the probability of being present (given above), as follows:

$$N_S = \left(1-\frac{N_m}{J}\right)\sum_{j=1}^{J}\left(1-e^{-\theta_j h}\right) \tag{3}$$

Because of machine failures, some machines may not complete all scheduled setups and jobs; that is, some setups and jobs will be delayed until the next rescheduling period. N_s/N_m is adjusted based on the fact that the total repair time (for one machine during one rescheduling period) has a gamma distribution because the individual machine repair times are exponentially distributed.

Let

$$W=\frac{\lambda^T}{M^T}h,\ e_t=\frac{N_S}{N_m}\bar{S}+W, \text{ and } H=h-e_t$$

Assume that, if there are no failures, no setups are delayed. (In practice, delays could occur due to the random number of jobs and job sizes.) The expected delay caused by failures will be as follows:

$$D=\sum_{p=1}^{\infty}\left(E[D\,|\,p]P\{F=p\}\right)$$

The number of failures F depends on how much the average machine is working and the mean time to failure. Specifically, F is has a Poisson distribution. For $p \geq 1$,

$$P\{F=p\}=\frac{\left(\dfrac{W}{\overline{\overline{MTTF}}}\right)^p e^{-W/\overline{MTTF}}}{p!}$$

If there are p breakdowns, the total repair time x is then distributed with a gamma distribution with mean $p\overline{MTTR}$ and shape parameter p. Then, the probability density function

$$f(x)=\frac{x^{p-1}e^{-x/\overline{MTTR}}}{\Gamma(p)\overline{MTTR}^p}$$

The expected delay is the amount that x exceeds H. Thus, D = max $\{x - H, 0\}$. For p failures, the following can be derived:

$$E[\max\{x-H,0\}\,|\,p]=\int_H^{\infty}(x-H)f(x)dx$$

$$=\frac{1}{\Gamma(p)\overline{MTTR}^p}\left[\int_H^{\infty}x^p e^{-x/\overline{MTTR}}dx-H\int_H^{\infty}x^{p-1}e^{-x/\overline{MTTR}}dx\right]$$

$$=\frac{1}{\Gamma(p)\overline{MTTR}^p}\left[-\overline{MTTR}.\,H^p e^{-x/\overline{MTTR}}\Big|_H^{\infty}\right.$$

$$\left.+\left(p\overline{MTTR}-H\right)\int_H^{\infty}x^{p-1}e^{-x/\overline{MTTR}}dx\right]$$

$$=\frac{1}{\Gamma(p)\overline{MTTR}^p}\left[\overline{MTTR}.\,H^p e^{-H/\overline{MTTR}}+\left(p\overline{MTTR}-H\right)\right.$$

$$\left.\sum_{r=0}^{p-1}e^{-x/\overline{MTTR}}\frac{(p-1)!}{(p-1-r)!}x^{p-1-r}(-1)\overline{MTTR}\Big|_H^{\infty}\right]$$

$$=\frac{e^{-H/\overline{MTTR}}}{\Gamma(p)\overline{MTTR}^p}\left[\overline{MTTR}.\,H^p+\left(p.\,\overline{MTTR}-H\right)\right.$$

$$\left.\sum_{r=0}^{p-1}\frac{(p-1)!}{(p-1-r)!}H^{p-1-r}\,\overline{MTTR}^{r+1}\right]$$

From this is obtained the following:

$$D=\sum_{p=1}^{\infty}\left\{\frac{\left(\dfrac{W}{\overline{\overline{MTTF}}}\right)^p e^{-W/\overline{MTTF}}}{p!}\ \frac{e^{-H/\overline{MTTR}}}{\Gamma(p)\overline{MTTR}^p}\right. \tag{4}$$

$$\left.\left[\overline{MTTR}.\,H^p+\left(p.\,\overline{MTTR}-H\right)\sum_{r=0}^{p-1}\frac{(p-1)!}{(p-1-r)!}H^{p-1-r}\,\overline{MTTR}^{r+1}\right]\right\}$$

Recall that, because p is a positive integer, Γ (p) = (p – 1)!

Finally, it is estimated that the average number of setups per machine is as follows:

$$\overline{N_S}=\left(1-\frac{D}{e_t}\right)\frac{N_S}{N_m} \tag{5}$$

The average number of setups per machine per time unit and the average setup time percentage follow immediately.

$$\overline{F_S} = \frac{\overline{N_S}}{h} = \overline{N_S}\,\overline{F_r} \tag{6}$$

$$\beta_S = \frac{\overline{N_S}\,\overline{S}}{h} = \overline{F_S}\,\overline{S} \tag{7}$$

Next can be estimated the average processing time percentage, average repair time percentage, average idle time percentage, average machine utilization, and average schedule execution time.

$$\beta_p = \frac{\lambda^T}{M^T} \tag{8}$$

$$\beta_r = \frac{\lambda^T}{M^T}\;\frac{\sum_{n=1}^{N_m}\left(MTTR_n \big/ MTTF_n\right)}{N_m} \tag{9}$$

$$\beta_i = 1 - \left(\beta_S + \beta_p + \beta_r\right) \tag{10}$$

$$\overline{U_m} = \beta_S + \beta_p + \beta_r \tag{11}$$

$$\overline{E_t} = \left(\beta_S + \beta_p + \beta_r\right)h \tag{12}$$

The average flow time is the sum of the average time spent waiting for the next rescheduling event ($h/2$) and the average time from the rescheduling event to job completion. The second term is estimated by averaging the expected time that the first job completes and the expected time that the last job completes. This yields the following expression:

$$\overline{F_t} = \frac{1}{2}\left(h + \alpha\overline{S} + \frac{\overline{Z}}{\overline{\overline{M}}} + \overline{E_t}\right) \tag{13}$$

where α, the probability that the first job requires a setup, equals the probability that the current job type is absent. This equals

$$1 - \frac{1}{J}\sum_{j=1}^{J}\left(1 - e^{-\theta_j H}\right)$$

Hybrid Rescheduling Strategy

This section presents analytical models for the hybrid rescheduling strategy, which has one parameter, the rescheduling period h. Rescheduling will occur every h time units, every time a machine fails, and every time a machine is repaired. The average rescheduling frequency depends on h and the total effective machine failure rate. Machines fail only during processing. Because, on average, the fraction of time that a machine is processing is β_p, it is estimated that the effective failure rate is

$$\beta_p \sum_{n=1}^{N_m} \frac{1}{MTTF_n}$$

There are two rescheduling events for every failure (one when the failure occurs and another when the machine is repaired). The number of rescheduling events per time unit (the rescheduling frequency) is estimated as follows:

$$\overline{F_r} = \frac{1}{h} + 2\beta_p \sum_{n=1}^{N_m} \frac{1}{MTTF_n} \tag{14}$$

Consequently, the average rescheduling period is estimated as follows:

$$\overline{P_r} = \frac{1}{\overline{F_r}} = \frac{1}{\dfrac{1}{h} + 2\beta_p \sum_{n=1}^{N_m} \dfrac{1}{MTTF_n}} \tag{15}$$

When a machine fails, the hybrid rescheduling strategy performs emergency scheduling to reassign the batches. In theory, unless machine utilization is very high, all setups or jobs should be performed within the rescheduling period. Equation (3) still holds as the estimate of the expected total number of setups. Because there are no delays, the average number of setups per machine can be estimated as follows:

$$\overline{N_S} = \frac{N_S}{N_m} \tag{16}$$

The analytical models for the remaining performance measures are the same as those presented in the previous subsection [Eqs. (6)-(13)].

EDQS Rescheduling Strategy

This section presents analytical models for the EDQS rescheduling strategy, which has one parameter, the queue size threshold qs. Rescheduling occurs when qs jobs have arrived since the last rescheduling event. Although the time between rescheduling events is a random variable, this strategy is otherwise very similar to the periodic rescheduling strategy.

The average rescheduling period is the mean time for qs jobs to arrive, and the average rescheduling frequency is its reciprocal:

$$\overline{F_r} = \frac{\theta^T}{qs} \tag{17}$$

$$\overline{P_r} = \frac{qs}{\theta^T} \tag{18}$$

After replacing the rescheduling period *h* by the average rescheduling period qs/θ^T, the remaining analytical models are the same as those presented in the last section [Eqs. (3)-(13)].

Evaluation

To evaluate the analytical models and show that they accurately estimate the performance of a parallel machine system with the specified rescheduling strategies, a simulation model was developed of a five-machine system. The predicted performance (from the analytical models) was then compared with the performance obtained from the simulation experiments.

The system had the following characteristics. For all five machines, M_n = 1 parts/time unit, MTTR_n = 33.3333 time units, MTTR_n = 300 time units, and S_n = 2 time units. Thus, A_n = 0.90 for all five machines. The number of job types J = 100 jobs, with θ^T = 0.5641 jobs/time unit and λ^T = 2.3044 parts/time unit. The average job arrival rates (not given here) varied from 0.001 to 0.010 jobs/time unit. The average job size varied from 1 to 7 parts.

Each rescheduling strategy was evaluated under four scenarios. For the periodic and hybrid rescheduling strategies, *h* = 25, 100, 500, or 1000 time units. For the EDQS rescheduling strategy, *qs* = 14, 56, 282, or 546 jobs. These values of *qs* yielded rescheduling periods similar to those set for the other two strategies.

Five simulation runs were conducted for each scenario. Each simulation run was 250,000 time units long, with a warmup period of 25,000 time units.

Tables 1, *2*, and *3* show the results of the experiments. For the experimental results, the 95% confidence interval is *Average* ± *Half-length*. (For more on confidence intervals and analyzing simulation results, please see Law and Kelton.[15]) Also, *Table 3* lists the rescheduling period for each value of *qs*.

First, the predicted values were compared to the experimental results. In most cases, the differences were small. The notable exceptions are the average flow time predictions when the rescheduling period is small (*h* = 25, *qs* = 14) and the setup frequency for the hybrid strategy when *h* = 25. These results reflect the variability and high utilization that occur when the rescheduling period is small. Setups occur more frequently, which reduces the amount of idle time. At some rescheduling events, low-volume job types will appear, requiring a setup. Failures delay setups and jobs, which disrupt and postpone the beginning of the next schedule. However, as the rescheduling period increases, the predictions are more accurate. The amount of idle time increases, so failures are less disruptive because there is more time to recover. The job types are present more consistently, and variability decreases.

As the rescheduling period increases, more jobs are done per setup, and the setup frequency and utilization both decrease significantly. However, the average flow time increases because jobs must wait longer until the next rescheduling event, and the delay until job completion (after rescheduling) increases (see *Figure 2*). This trade-off, which should be considered when selecting rescheduling parameters, occurs under all three rescheduling strategies. Of course, if the rescheduling period is too small, there is not enough capacity to perform all of the required setups each period. The analytical models presented here do not apply in this case. See Vieira, Herrmann, and Lin[3] for more discussion of this phenomenon.

The experimental results show that the periodic and EDQS rescheduling strategies yield, for the same rescheduling period, system performance that is approximately equal. For a given value of *h*, the hybrid rescheduling strategy has, compared to the periodic rescheduling strategy, a higher rescheduling frequency, caused by the machine failures and repairs. Otherwise, the experimental results show that, except for a small decrease in average flow time, the hybrid rescheduling strategy yields the same system performance. (Experiments were also conducted with simulation models that used Weibull distributions for the time to failure and time to repair. The values predicted by the analytical models and the experimental results were again very similar, showing the robustness of the models, especially for non-ideal practical situations.)

6. Summary and Conclusions

Rescheduling is a necessary part of controlling many dynamic, stochastic manufacturing systems. This paper describes research that attempts to understand how rescheduling strategies affect the perfor-

Table 1
Performance of Periodic Rescheduling Strategy

h	Performance Measure	Predicted Value	Experimental Results		Difference (%)
			Average	Half-Length	
25	$\overline{F_r}$	0.0400	0.0400	0.0000	0.0
	$\overline{F_s}$	0.0967	0.0968	0.0002	-0.1
	$\overline{U_m}$	0.7054	0.7070	0.0023	-0.2
	$\overline{F_t}$	24.2309	25.9039	0.0940	-6.5
100	$\overline{F_r}$	0.0100	0.0100	0.0000	0.0
	$\overline{F_s}$	0.0778	0.0764	0.0001	1.8
	$\overline{U_m}$	0.6676	0.6654	0.0011	0.3
	$\overline{F_t}$	86.0141	87.1249	0.1760	-1.3
500	$\overline{F_r}$	0.0020	0.0020	0.0000	0.0
	$\overline{F_s}$	0.0331	0.0330	0.0001	0.4
	$\overline{U_m}$	0.5783	0.5783	0.0036	0.0
	$\overline{F_t}$	396.7472	394.3458	1.0332	0.6
1000	$\overline{F_r}$	0.0010	0.0010	0.0000	0.0
	$\overline{F_s}$	0.0183	0.0183	0.0000	0.0
	$\overline{U_m}$	0.5486	0.5495	0.0018	-0.2
	$\overline{F_t}$	776.3805	774.8005	0.2389	0.2

Table 2
Performance of Hybrid Rescheduling Strategy

h	Performance Measure	Predicted Value	Experimental Results		Difference (%)
			Average	Half-Length	
25	$\overline{F_r}$	0.0554	0.0554	0.0004	0.0
	$\overline{F_s}$	0.1029	0.0970	0.0002	6.0
	$\overline{U_m}$	0.7178	0.7053	0.0019	1.8
	$\overline{F_t}$	24.3854	25.5464	0.0688	-4.5
100	$\overline{F_r}$	0.0254	0.0254	0.0004	0.0
	$\overline{F_s}$	0.0801	0.0772	0.0001	3.
	$\overline{U_m}$	0.6722	0.6669	0.0027	0.8
	$\overline{F_t}$	86.2439	85.2184	0.1347	1.2
500	$\overline{F_r}$	0.0174	0.0174	0.0003	0.0
	$\overline{F_s}$	0.0331	0.0330	0.0000	0.4
	$\overline{U_m}$	0.5783	0.5774	0.0020	0.2
	$\overline{F_t}$	396.7549	391.7543	0.8540	1.3
1000	$\overline{F_r}$	0.0164	0.0165	0.0005	-0.6
	$\overline{F_s}$	0.0183	0.0182	0.0000	0.0
	$\overline{U_m}$	0.5486	0.5479	0.0019	0.1
	$\overline{F_t}$	776.3805	771.7882	0.6272	0.6

mance of such manufacturing systems. This article presented analytical models that predict the performance of rescheduling strategies for parallel machine systems with multiple job types and setups between job types. The paper described three different rescheduling strategies: periodic, hybrid, and event-driven based on the queue size. The models predict the following performance measures: average flow time, rescheduling frequency, average setup frequency, average schedule execution time, average machine utilization, average setup time percentage, average processing time percentage, average repair time percentage, and average idle time percentage.

The paper presented the results of simulation experiments done to estimate the performance of the rescheduling strategies. These results show that the analytical models can accurately predict the performance measures, especially as the rescheduling period increases. In addition, there is a conflict between avoiding setups and reducing flow time, and the rescheduling period affects both objectives significantly. Finally, all three rescheduling strategies yield system performance that is approximately equal. The hybrid rescheduling strategy, however, performs rescheduling more often.

In this work, the rescheduling strategies used a heuristic scheduling algorithm that eliminates unnecessary setups. The analytical models rely on this setup-elimination aspect, not on the priority rules used to sequence the batches or the jobs within each batch. Thus, one could apply the models presented here with scheduling algorithms that used other priority rules (for example, sequence the jobs by earliest due date and sequence the batches by batch processing time).

More complex scheduling policies that combine setup reduction with other goals may be more appropriate in some settings (see, for example, Herrmann and Lee[16]). Analyzing these is beyond the scope of this paper and remains a topic for future research. In addition, research is needed to analyze flow shop and job shop production systems.

Table 3
Performance of EDQS Rescheduling Strategy

qs	Performance Measure	Predicted Value	Experimental Results		Difference (%)
			Average	Half-Length	
14	$\overline{F_r}$	0.0403	0.0403	0.0002	0.0
($\overline{P_r}$ =24.8)	$\overline{F_s}$	0.0967	0.0974	0.0005	-0.6
	$\overline{U_m}$	0.7055	0.7065	0.0048	-0.1
	$\overline{F_t}$	24.0777	24.5884	0.1682	-2.1
56	$\overline{F_r}$	0.0101	0.0101	0.0000	0.0
($\overline{P_r}$ =99.3)	$\overline{F_s}$	0.0779	0.0770	0.0003	1.2
	$\overline{U_m}$	0.6679	0.6671	0.0025	0.1
	$\overline{F_t}$	85.4241	85.0686	0.1881	0.4
282	$\overline{F_r}$	0.0020	0.0020	0.0000	0.0
($\overline{P_r}$ =499.9)	$\overline{F_s}$	0.0331	0.0329	0.0001	0.6
	$\overline{U_m}$	0.5783	0.5785	0.0020	0.0
	$\overline{F_t}$	396.6716	393.1822	0.6011	0.9
546	$\overline{F_r}$	0.0010	0.0010	0.0000	0.0
($\overline{P_r}$ =967.9)	$\overline{F_s}$	0.0188	0.0188	0.0001	0.0
	$\overline{U_m}$	0.5497	0.5483	0.0019	0.3
	$\overline{F_t}$	752.0581	747.6194	1.9611	0.6

This paper has three main contributions. First, it presents analytical models that can easily and quickly estimate important performance measures for rescheduling strategies in a dynamic, stochastic manufacturing system. Second, these models quantify the trade-offs between different objectives and allow one to select optimal rescheduling parameters quickly without the need to develop and run simulation experiments. Finally, this study shows that the rescheduling strategy can significantly impact system performance, and thus, one should consider and model the manufacturing planning and control strategies when designing a manufacturing system.

Acknowledgment

This research was performed with financial support from CAPES - Brazil (grant number 3135/95-3) and using the facilities of the Computer-Integrated Manufacturing Laboratory at the University of Maryland.

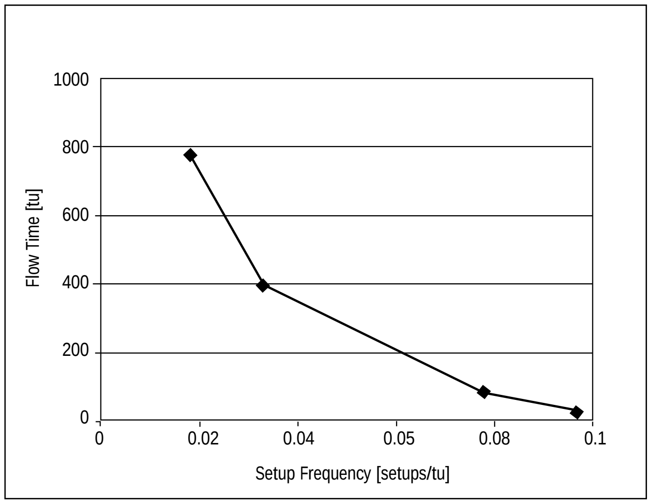

Figure 2
Flow Time and Setup Frequency for Different Rescheduling Periods

References

1. J. Fang and Y. Xi, "A Rolling Horizon Job Shop Rescheduling Strategy in the Dynamic Environment," *Int'l Journal of Advanced Mfg. Technology* (v13, 1997), pp227-232.
2. L.K. Church and R. Uzsoy, "Analysis of Periodic and Event-Driven Rescheduling Policies in Dynamic Shops," *Int'l Journal of Computer Integrated Mfg.* (v5, 1992), pp153-163.
3. Hideaki Takagi, *Analysis of Polling Systems* (Cambridge, MA: MIT Press, 1986).
4. S.D. Wu, R.H. Storer, and P.-C. Chang, "One-Machine Rescheduling Heuristics with Efficiency and Stability as Criteria," *Computers & Operations Research* (v20, 1993), pp1-14.
5. R-K. Li, Y.-T. Shyu, and S. Adiga, "A Heuristic Rescheduling Algorithm for Computer-Based Production Scheduling Systems," *Int'l Journal of Production Research* (v31, 1993), pp1815-1826.
6. A.K. Jain and H.A. ElMaraghy, "Production Scheduling/Rescheduling in Flexible Manufacturing," *Int'l Journal of Production Research* (v35, 1997), pp281-309.
7. J.S. Dhingra, K.L. Musser, and G.L. Blankenship, "Real-Time Operations Scheduling for Flexible Manufacturing Systems," *Proc.* of 1992 Winter Simulation Conf. (1992), pp849-855.
8. R.J. Abumaizar and J.A. Svestka, "Rescheduling Job Shops Under Disruptions," *Int'l Journal of Production Research* (v35, 1997), pp2065-2082.
9. G.E. Vieira, J.W. Herrmann, and E. Lin, "Rescheduling Manufacturing Systems: A Survey of Definitions, Strategies, Methods, and Techniques." This paper is being evaluated for publication in 2000 at the *Journal of Scheduling* (UK).
10. G.E. Vieira, J.W. Herrmann, and E. Lin, "Analytical Models to Predict the Performance of a Single-Machine System Under Periodic and Event-Driven Rescheduling Strategies," *Int'l Journal of Production Research* (v38, n8, 2000), pp1899-1915.
11. M.H. Kim and Y.-D. Kim, "Simulation-Based Real-Time Scheduling in a Flexible Manufacturing System," *Journal of Mfg. Systems* (v13, n2, 1994), pp85-93.
12. S.V. Mehta and R.M. Uzsoy, "Predictable Scheduling of a Job Shop Subject to Breakdowns," *IEEE Trans. on Robotics and Automation* (v14, 1998), pp365-378.
13. M. Zweben, E. Davis, B. Daun, and M.J. Deale, "Scheduling and Rescheduling with Iterative Repair," *IEEE Trans. on Systems, Man and Cybernetics* (v23, 1993), pp1588-1596.
14. I. Sabuncuoglu and S. Karabuk, "Rescheduling Frequency in an FMS with Uncertain Processing Times and Unreliable Machines," *Journal of Mfg. Systems* (v18, n4, 1999), pp268-283.
15. Averill M. Law and W. David Kelton, *Simulation Modeling and Analysis*

(New York: McGraw-Hill Int'l Editions - Industrial Engineering Series, 1991).
16. J.W. Herrmann and C.-Y. Lee, "Solving a Class Scheduling Problem with a Genetic Algorithm," *ORSA Journal of Computing* (v7, 1995), pp443-452.

Authors' Biographies

Guilherme E. Vieira is a PhD candidate at the University of Maryland at College Park. He holds a BS in control and industrial automation engineering (1994) and an MS in mechanical engineering (1996). He is a member of SME and APICS, and his work focuses on scheduling, planning, and control of manufacturing systems. Current research interests include: analytical and simulation modeling of manufacturing systems, logistics, master production scheduling, and supply chain management. For more information, refer to http://www.angelfire.com/md/gevieira/.

Jeffrey W. Herrmann is an assistant professor at the University of Maryland, where he holds a joint appointment with the Department of Mechanical Engineering and the Institute for Systems Research. He is the director of the Computer-Integrated Manufacturing Laboratory. He is a member of INFORMS and ASME. He earned his BS in applied mathematics from Georgia Institute of Technology. As a National Science Foundation Graduate Research Fellow from 1990 to 1993, he received his PhD in industrial and systems engineering from the University of Florida. His current research interests include the design and control of manufacturing systems and the integration of product design and manufacturing system design. For more information, see http://www.isr.umd.edu/~jwh2/jwh2.html.

Edward Lin is a research engineer at the Computer-Integrated Manufacturing Laboratory in the Institute for Systems Research at the University of Maryland. He received his PhD from the School of Industrial and Systems Engineering at the Georgia Institute of Technology in 1994. He has had five years of industrial experience in automation of manufacturing and production systems. He also has several years of experience in developing object-oriented databases, distributed manufacturing applications, and web-based manufacturing services for government and industrial projects. His research interests include adaptable manufacturing simulation, production planning and scheduling, and web-based design. For more information, see http://www.isr.umd.edu/Labs/CIM/profiles/lin.