A Practical Scheduling Method for Multiclass Production Systems with Setups

Tava Lennon Olsen
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109-2117

Consider a multiclass production system where many job classes share a single server and a setup time is incurred whenever the server changes class. This paper presents a simple method for scheduling these systems that performs well, not only with respect to mean waiting time, but also with respect to waiting-time variance and the outer percentiles of waiting time. The scheduling method is dynamic and uses the ages of items in each queue, as well as the queue statistics, to decide which queue to service next.
(*Heuristics; Changeovers; Queueing; Polling; Manufacturing*)

Manufacturing systems where machines process a number of different products and require time to set up (or change over) between products are extremely common in industry. For example, most automakers run stamping plants where the setup time between parts involves changing heavy dies and is necessarily large. This paper presents a practical method for dynamically scheduling products in such environments.

The specific environment considered here is a manufacturing work center where a single machine (or server) processes a variety of different types (or classes) of items. If the item to be produced on the machine is of the same type as the item previously produced, then no machine setup is required. On the other hand, if the item is of a different type, then a (possibly lengthy) machine setup is required before it can be processed. We assume that the length of the setup time does not depend on the sequence of product types produced (i.e., the setup times are "sequence independent"). The canonical example for this type of setup is a die change operation where the setup time does not depend on which die was previously in place.

There are two main bodies of literature where scheduling multiple products requiring setups is fairly well understood. The first seeks to optimally schedule a fixed set of jobs or customer orders, all of which are present at time zero (see, for example, Pinedo 1995 or Webster and Baker 1995). The second is an inventory setting where inventory is depleted at a fixed rate (see, for example, Nahmias 1993). However, in a typical manufacturing environment, orders continue to arrive in an unpredictable pattern over time, which is highly relevant in deciding which product class to process next, how to sequence orders within a class, and so forth. We are interested in studying such dynamic systems.

Unfortunately, there appear to be few effective tools available for dynamically scheduling stochastic manufacturing systems with significant setup times. For example, Buzacott and Shanthikumar's (1993) seminal book, *Stochastic Models of Manufacturing Systems*, says effectively nothing about systems with setups. The goal of this paper is to present an easily implementable dynamic scheduling method that works well for systems with parameters that are realistic for a manufacturing environment.

Much of the relevant literature for dynamic scheduling of multiclass production systems with setups

0025-1909/99/4501/0116$05.00

can be found in the literature for what are known as "polling models" (see Takagi 1990 for a survey). A polling model is a queueing system with multiple customer classes requiring setups. A number of standard scheduling methods have been proposed for polling models. Perhaps the most studied are cyclic serve-to-exhaustion (CSTE) and cyclic gated service (CGS). Under both these methods the queues are visited in a fixed cyclical manner (see Takagi 1990 for further details). However, there has also been significant study of models where the queues are visited in a fixed noncyclic manner. The order in which queues are visited is determined by what is known as a "polling table." Queues with high priority are placed several times on the table. There has been significant work done on finding optimal sequences for polling tables based on objectives such as minimizing expected waiting cost where costs are assumed to be linear (see, for example, Boxma et al.). However, these types of systems still require that a strict pattern for visiting queues is followed and that a setup is incurred even if the queue is empty. While this might be realistic in a telecommunications setting, it is probably highly unrealistic in a manufacturing setting. Although Cooper et al. (1997) recently have shown that in CSTE models it is sometimes optimal to set up at a queue even if it is empty, our intuition is that this less likely to be the case under dynamic policies.

Very little has been done on policies that are neither fixed nor cyclical. The two-station model has been analyzed by Hofri and Ross (1987) and Reiman and Wein (1994). For the case where there are more than two stations, Liu et al. (1992) and Markowitz (1996) provide some optimality results that are described in §2. Browne and Yechiali (1989) derive quasi-dynamic index policies where each queue is visited exactly once per cycle but the order in which queues are visited within a cycle changes dynamically. Duenyas and Van Oyen (1996) provide a heuristic for scheduling systems with setups that is based on a dynamic programming formulation. Section 3 compares our heuristic to those of both Browne and Yechiali (1989) and Duenyas and Van Oyen (1996).

We study what is a "good" way to dynamically schedule multiclass production systems with significant setup times. The question, of course, is what is meant by a "good" scheduling method? The traditional performance measures used for such environments have been mean waiting time, or a weighted average of mean wait. However, this is probably due more to tractability issues rather than motivation from actual manufacturing environments. Other than Liu et al. (1992), all of the above work seeks to minimize various types of means. Nothing is known about the relative performance of different policies with respect to the variance or outer percentiles of waiting time. Furthermore, little is known even on minimum variance policies for systems without setups. When setups are zero and all queues are identical, it is well known that First-In-First-Out (FIFO) service stochastically minimizes waiting time. For asymmetric systems with zero setups, it is known that choosing the job with the shortest processing time minimizes mean wait; however, it does not stochastically minimize waiting time. Ayhan and Olsen (1997) have presented a heuristic for reducing the variance in systems without setup times which will be discussed in §2.1.

In our experience, mean waiting time may not be as relevant to a manufacturer as the outer percentiles of waiting time (e.g., 95th percentile) or the variance of waiting time. We therefore seek a scheduling heuristic that performs well, not only with respect to mean waiting time, but also with respect to the variance and 95th percentile of waiting time. In other words, we seek a scheduling method that produces a waiting time distribution with a moderate mean and a light tail. This paper does not seek an optimal policy for either of the previous two objectives. Rather, it uses the dual objective of moderate mean and light tail to present a policy that is designed to be the "best available" practical policy for manufacturing systems with multiple classes and significant setup times.

This paper is organized as follows. Section 1 presents the problem formulation and introduces notation. Section 2 describes our heuristic policy. Using simulation, §3 tests our heuristic against previously suggested policies from the literature. Lastly, §4 concludes the paper and describes possible extensions to our work.

1. Problem Formulation and Notation

This section describes the model further, outlines the notation used, and presents the assumptions made. We will consider an environment where production is make-to-order and the server has knowledge of how many customers are present in each class and when each customer arrived. We seek a scheduling method that produces a waiting time distribution with a moderate mean and a light tail.

Jobs arrive to each of n job classes according to independent Poisson processes of rate λ_i at class i, $1 \le i \le n$. The service times for customers in class i, $1 \le i \le n$, are independent and distributed according to the random variable B_i with mean b_i and second moment $b_i^{(2)}$. Let $\mu_i = 1/b_i$, $1 \le i \le n$. As mentioned previously, we assume that the duration of each setup depends only on the class being switched to and not on which class was served previously. The setup times for class i, $1 \le i \le n$, are independent and distributed according to the random variable S_i with mean s_i. The only assumptions made regarding the distribution of the service and setup times are that the means and variances are finite and that all distributions are independent. We also assume that all steady-state limits exist and are well-defined. Simulation results verify that such assumptions are valid for the parameter values of interest to us.

Let $\rho_i = \lambda_i b_i$ be the utilization at queue i, $1 \le i \le n$, and let ρ be the system utilization excluding setups (i.e., $\rho = \sum_{i=1}^{n} \rho_i$). We assume that $\rho < 1$. This assures that the system is stable under CSTE and CGS scheduling disciplines (see Takagi 1986). Takagi (1986) also shows that for a stable system, a CSTE system is regenerative, and hence steady-state limiting distributions exist.

In all models, we define a *cycle* at queue i, $1 \le i \le n$, as the time between two successive arrivals of the server to queue i. Let C_i be the random variable that has steady-state cycle time distribution for queue i, $1 \le i \le n$. It is also useful to define *intervisit time* at queue i, $1 \le i \le n$, as the time that elapses from the server leaving queue i to its next commencement of service at queue i. Thus the intervisit time for queue i equals the cycle time for queue i minus the time spent serving queue i. Steady-state intervisit time for queue i, $1 \le i \le n$, is represented by I_i. Let f_{ij} be the expected number of visits to queue j for every visit to queue i, $1 \le i, j \le n$, where $f_{ii} = 1$. We refer to the f_{ij} as the *relative queue visit frequencies*. Clearly, $f_{ij} = E[C_i]/E[C_j]$, $1 \le i, j \le n$.

Steady-state waiting time at queue i, W_i, is defined as the time from a job's arrival to queue i to when it commences service in steady-state, $1 \le i \le n$. Total waiting time W is the waiting time experienced by an arbitrary customer. Thus $E[W] = \sum_{i=1}^{n} \lambda_i E[W_i]/\lambda$. The pth percentile for the total waiting time distribution is defined as the number $P(p)$ such that $\text{Prob}(W \le P(p)) = p/100$. In our experience, it is usually the total wait W that is of interest to a manufacturer. However, §4 discusses the case where different queues have different priorities.

Lastly, we define a *symmetric* system as one where all queues have identical arrival rates, setup time distributions, and service time distributions. Conversely, an asymmetric system allows for differences among the queues.

2. Heuristic Procedure

This section presents our heuristic procedure for dynamically scheduling multiclass production systems with setups. The novelty of the procedure is in using customer ages multiplied by a "scale factor" to determine which queue to serve next. Section 2.1 describes this procedure in detail and provides motivation for using customer ages to select queues. Section 2.2 describes the selection of scale factors. These scale factors rely on desired queue visit frequencies that are derived in §2.3.

Our heuristic procedure is completely defined by the following policies:

Queue Greedy: The server never idles at a queue while there is still work present at that queue.

System Greedy: The server never idles while there is any work in the system.

Exhaustive: The server never switches from a queue that still contains work.

Patient: If the system is empty then the server idles at the most recently served queue.

FIFO Service: Service within a class is in the same order in which the jobs arrived (i.e., First-In-First-Out service).

Greatest Scaled Age: The next queue to service will be chosen as that with the most scaled age (defined in §2.1).

As described in detail below, these policy types are chosen both because of previously derived optimality results and because of general system intuition. In §3 we test our heuristic against six heuristic procedures that we found in the literature. Of these, all use FIFO service within a class and are queue greedy. All but one are system greedy and patient. All but two are exhaustive. However, the Duenyas and Van Oyen (1996) heuristic relies only upon FIFO and queue greedy service, and therefore, in §3, it will serve as a test of the above policies.

Queue and System Greedy Policies. If a policy is system greedy then it is also queue greedy, but not vice versa. Liu et al. (1992) show that (under the given independence assumptions) queue greedy policies stochastically minimize the unfinished work and the number of customers in the system at any time. Hence, for symmetric systems, queue greedy policies also minimize the expected waiting time. These results do not carry over to system greedy policies. While system greedy policies are not generally optimal, we choose to concentrate on them for the following three main reasons:

1. We believe it would be hard to persuade most manufacturers that they shouldn't be working even though there is work in the system.
2. Idling with work in the system could add another source of variability to the system.
3. It seems to work well in practice.

Exhaustive Service. Liu et al. (1992) also show that, under the given independence assumptions, exhaustive policies stochastically minimize the unfinished work and the number of customers in the system at any time. Hence, for symmetric systems, exhaustive policies also minimize the expected waiting time. However, this is not necessarily true for asymmetric systems. For example, in systems with no setup times it is well known (see, for example, Wolff 1989) that serving the queue with the smallest expected service time ("largest $c\mu$") minimizes the total expected wait. In systems with setup times, Duenyas and Van Oyen (1996) show that in order to minimize total expected wait it is optimal to serve exhaustively at the queue with the smallest service time. Markowitz (1996) has shown that under heavy-traffic, in order to asymptotically (as $\rho \uparrow 1$) minimize the expected waiting time in cyclic service systems, all but the queue with the smallest $c\mu$ should be served to exhaustion. (This result was previously shown in Reiman and Wein (1994) for the two queue case.) If all queues have identical costs (as in our case) Markowitz (1996) also shows that even the lowest priority queue should be served exhaustively. While exhaustive service might not be generally optimal, we choose to concentrate on it due to the above observations and for the following five additional reasons:

1. It conserves time spent in setups (as compared to other system greedy policies), which could be important if there is cost as well as time associated with setups.
2. Exhaustive service is self compensating, in that if the system is busy then less time is spent in setups than when the system is not busy.
3. Most reasonable exhaustive service policies are stable when $\rho < 1$. This cannot be said for policies that place a fixed limit on the number served at each visit.
4. Not serving exhaustively would seem to lead to another source of variability. An arrival is no longer guaranteed to wait at most a single cycle before receiving service.
5. It seems to work well in practice.

Patient and FIFO Service. Liu et al. (1992) have shown that patient service minimizes mean waiting time in a symmetric system. Switching to the queue with the highest arrival rate when the whole system is empty may slightly reduce waiting time. However, if either the number of queues or the setup times are relatively large then the difference between this strategy and patient service is not likely to be big because the percentage of idle time will be very small. In our simulation study we consider only patient service but leave the implementation of what to do when idle up to individual plant managers. In systems where setups

require considerable labor it is likely that they will adopt a patient system.

Lastly, because jobs within a class are stochastically identical, serving other than FIFO service would seem to only increase the variance of the waiting time. Therefore, only a FIFO queue discipline is considered.

2.1. Greatest Scaled Age Queue Selection Procedure

We wish to find a scheduling method that performs well not only with respect to mean waiting time but also with respect to the variance and outer percentiles of waiting time. As shown in this section, such a policy must take into account the age of current items when selecting a queue. This section describes the *greatest scaled age* procedure for choosing the next queue.

Let T_i be the total age currently present in queue i (sum over the time each customer present has been waiting) upon completing service at some other queue, $1 \le i \le n$. Then, after setting up for queue i, the expected total age is

$$A_i = \lambda_i s_i^2/2 + s_i N_i + T_i,$$

where N_i is the number of customers present in queue i prior to the setup. The greatest scaled age procedure chooses the next queue as the one with the most expected age A_i multiplied by a scale factor w_i, $1 \le i \le n$. Under extremely light traffic each queue will typically contain only one customer and therefore $w_i = 1\ \forall i$ is almost equivalent to FIFO service. However, it is easy to see that this could produce very long waits, particularly if the system is highly asymmetric. In particular, a successful scheduling method should assign a low priority to those queues with very long setup times. How these weights are assigned is discussed further in §2.2. We first motivate this procedure.

For most systems, in order to minimize mean waiting time, the age of the items present is not relevant. However, when considering higher moments of waiting time a scheduling method must take the age of the items into account. For example, FIFO service minimizes all moments of waiting time in $GI/G/1$ queues and chooses the next customer as that with the greatest age. Ayhan and Olsen (1997) propose a heuristic with the same objectives as this paper for multiproduct systems with no setups. This heuristic is based on heavy-traffic theory of Van Mieghem (1995) and asymptotically (as $\rho \uparrow 1$) minimizes the second moment of waiting time. The heuristic chooses the next job as that with the greatest value of age divided by expected service time. The idea of multiplying total age by a scale factor is clearly similar to this idea. However, instead of considering only the ages of individual items, the procedure here adds up the ages of all customers in a queue.

In order to motivate the use of total age in our heuristic we present the following example and lemma, which show why age is important for minimizing higher moments (in particular the second moment) of waiting time.

EXAMPLE 1. Consider a two-queue system where there are no arrivals. At time zero queue 1 has n_1 customers with ages $a_{11}, a_{12}, \ldots, a_{1n_1}$ and queue 2 has n_2 customers with ages $a_{21}, a_{22}, \ldots, a_{2n_2}$. Let $T_1 = \sum_{i=1}^{n_1} a_{1i}$ and $T_2 = \sum_{i=1}^{n_2} a_{2i}$ be the total ages in each queue. Suppose both queues have identically distributed service times with mean b and second moment $b^{(2)}$. Setup times at both queues are deterministic with mean s. Furthermore, suppose that neither queue is set up for at time 0. We will serve each queue exhaustively. Let W_{jk}^i be the waiting time of the kth customer at queue j when queue i is the first queue to be served, $1 \le i, j \le 2$ and $1 \le k \le n_j$. Let w_{12} be the difference in total wait between serving queue 1 first and serving queue 2 first. Then

$$\begin{aligned} w_{12} &= \sum_{k=1}^{n_1} EW_{1k}^1 + \sum_{k=1}^{n_2} EW_{2k}^1 - \sum_{k=1}^{n_1} EW_{1k}^2 - \sum_{k=1}^{n_2} EW_{2k}^2 \\ &= \sum_{k=1}^{n_1} (s + a_{1k} + kb) + \sum_{k=1}^{n_2} (2s + a_{2k} + (n_1 + k)b) \\ &\quad - \sum_{k=1}^{n_1} (2s + a_{1k} + (n_2 + k)b) - \sum_{k=1}^{n_2} (s + a_{2k} + kb) \\ &= (n_2 - n_1)s. \end{aligned}$$

As expected, to minimize the mean waiting time the first queue to be served should be the longest queue. Furthermore, if setup times are zero then the system is

work conserving and serving either queue results in the same total expected wait.

Let $w_{12}^{(2)}$ be the difference in the sum of the second moments of waiting time between serving queue 1 first and serving queue 2 first. Then it is easy to show that

$$w_{12}^{(2)} = 2T_2(s + n_1 b) - 2T_1(s + n_2 b) + 3(n_2 - n_1)s^2 + [n_2(n_2 + 1) - n_1(n_1 + 1)]bs.$$

Notice that age enters into this equation only as the sum over all customers in a queue. If $n_1 = n_2$ then the policy that minimizes the second moment of waiting time chooses the first queue as that with the most total age. However, if $n_1 \neq n_2$ then in order to minimize second moment, the optimal queue choice depends both on the total ages and on the number of customers present in each queue.

LEMMA 1. *Let π be a stationary exhaustive service policy where the decision at each time instant depends on the queue lengths and the ages of the customers in each queue. Let γ be an identical policy to π except that if two queues are of identical length with identical distributional properties then the queue with the most total age is preferred. Let $\Lambda_i(t)$ be the number of arrivals to queue i by time t, $1 \leq i \leq n$. Let W_{ik}^{π} (W_{ik}^{γ}) be the waiting time for the kth departing customer from queue i, $1 \leq i \leq n$, under policy π (γ). Then for fixed $T > 0$*

$$\text{(i)} \quad E\left[\sum_{i=1}^{n}\sum_{k=1}^{\Lambda_i(T)} W_{ik}^{\pi}\right] = E\left[\sum_{i=1}^{n}\sum_{k=1}^{\Lambda_i(T)} W_{ik}^{\gamma}\right],$$

and

$$\text{(ii)} \quad E\left[\sum_{i=1}^{n}\sum_{k=1}^{\Lambda(T)} (W_{ik}^{\pi})^2\right] \geq E\left[\sum_{i=1}^{n}\sum_{k=1}^{\Lambda(T)} (W_{ik}^{\gamma})^2\right].$$

PROOF. Part (i) is immediate from the definition of the policies. For (ii), let

$$V^{\pi}(T) = \sum_{i=1}^{n}\sum_{k=1}^{\Lambda_i(T)} (W_{ik}^{\pi})^2$$

and

$$V^{\gamma}(T) = \sum_{i=1}^{n}\sum_{k=1}^{\Lambda_i(T)} (W_{ik}^{\gamma})^2.$$

Let $n_i^{\pi}(t)$ be the number of customers in queue i at time t under policy π, $1 \leq i \leq n$. Also, let $A_i^{\pi}(t)$ be the total age in queue i at time t under policy π, $1 \leq i \leq n$. Consider π on some sample path ω. We will consider γ along a new sample path ω' where ω' has the same law as ω.

Suppose that some queues i and j have identical distributional properties, $1 \leq i, j \leq n$. Suppose at time t_1 queue i begins a setup under π, $n_i^{\pi}(t_1) = n_j^{\pi}(t_1)$, and $A_i^{\pi}(t) < A_j^{\pi}(t)$. Let ω' follow ω up to time t_1. Then under γ queue j commences a setup at time t_1. Let t_2 be the first time after time t_1 that queue j is setup for under π and let t_3 be the first time that queue j is exhausted after time t_2 under π. Without loss of generality we may take $T = t_3$ (because the following argument can be repeated as many times as necessary). Let the arrival epochs, setup times, and service time sequences under ω' be the same as ω for all queues other than queues i and j. Interchange the arrival, service, and setup processes for queues i and j between times t_1 and t_3. For example, the arrival times seen at queue i (j) between t_1 and t_3 under ω are those seen at queue j (i) under ω'. The interchange of arrival processes is allowable since the interarrival times are i.i.d. exponential (memoryless) random variables. The interchange of service and setup time processes is allowable since the service and setup times are i.i.d. and no service or setup is in progress at times t_1 or t_3. Because of this interchange, the time to exhaust queue i under π is the same as the time to exhaust queue j under γ. Therefore, the waiting time for all customers in these two systems, except those present in queues i and j at time t_1, is identical (the new arrivals to queue i (j) under π experience the same wait as the new arrivals to queue j (i) under γ). Let a_{ik} (a_{jk}) be the age of the kth customer at queue i (j) at time t_1, $1 \leq k \leq n_i^{\pi}(t_1)$ ($n_j^{\pi}(t_1)$). Let s_i^{π} (s_i^{γ}) be the first setup time after time t_1 at queue i under policy π (γ). Define s_j^{π} and s_j^{γ} similarly. Then $s_i^{\pi} = s_j^{\gamma}$ and $s_j^{\pi} = s_i^{\gamma}$. Let b_{il}^{π} (b_{il}^{γ}) be the lth service time after time t_1 at queue i under policy π (γ), $1 \leq l \leq n_i^{\pi}(t_1)$ ($n_i^{\gamma}(t_1)$). Similarly define b_{jl}^{π} and b_{jl}^{γ} so that $b_{il}^{\pi} = b_{jl}^{\gamma}$, $1 \leq l \leq n_i^{\pi}(t_1)$, and $b_{jl}^{\pi} = b_{il}^{\gamma}$, $1 \leq l \leq n_j^{\pi}(t_1)$. Therefore, we have that

$$
\begin{aligned}
&V^{\pi}(T)(\omega) - V^{\gamma}(T)(\omega') \\
&= \sum_{k=1}^{n_i^{\pi}(t_1)} \left(s_i^{\pi} + a_{ik} + \sum_{l=1}^{k} b_{il}^{\pi} \right)^2 \\
&\quad + \left(t_2 - t_1 + s_j^{\pi} + a_{jk} + \sum_{l=1}^{k} b_{jl}^{\pi} \right)^2 \\
&\quad - \left(t_2 - t_1 + s_i^{\gamma} + a_{ik} + \sum_{l=1}^{k} b_{il}^{\gamma} \right)^2 \\
&\quad - \left(s_j^{\gamma} + a_{jk} + \sum_{l=1}^{k} b_{jl}^{\gamma} \right)^2 \\
&= \sum_{k=1}^{n_i^{\pi}(t_1)} \left[2(a_{ik} - a_{jk})(t_1 - t_2) \right. \\
&\quad \left. + 2(a_{ik} - a_{jk}) \left(s_i^{\pi} - s_j^{\pi} + \sum_{l=1}^{k} (b_{il}^{\pi} - b_{jl}^{\pi}) \right) \right] \\
&> \sum_{k=1}^{n_i^{\pi}(t_1)} 2(a_{ik} - a_{jk})(s_i^{\pi} - s_j^{\pi} + \sum_{l=1}^{k} (b_{il}^{\pi} - b_{jl}^{\pi})) .
\end{aligned}
$$

The right-hand side has expectation equal to zero. Therefore taking expectations (i.e., unconditioning on ω and ω'), we have proved the desired result.

The previous example and lemma have motivated (1) why we use total age as opposed to some other metric involving age, and (2) why total age is important when considering the second moment of waiting time. We use this motivation and general system intuition to hypothesize that a procedure using total age in queue selection should have lighter tails than one that ignores age.

2.2. Choosing Scale Factors for the Greatest Scaled Age Procedure

Under the greatest scaled age procedure the next queue to be served is selected as that with the greatest expected total age multiplied by a scale factor. We will choose the scale factors to reduce mean waiting time. In particular, if f_{ij}, $1 \le i, j \le n$, are desired relative queue visit frequencies, chosen so as to reduce mean waiting time, then we seek to choose the scale factors so that these visit frequencies are approximately achieved. The selection of appropriate f_{ij} is discussed in §2.3.

Let w_i be the scale factor for queue i, $1 \le i \le n$. Then using the following two approximations (to be discussed in detail later), and given relative queue visit frequencies f_{ij}, $1 \le i, j \le n$, the w_i can be chosen to approximately achieve these frequencies. Recall that I_i represents the steady-state intervisit time for queue i, $1 \le i \le n$.

APPROXIMATION 1. *The average scaled age upon commencement of service at queue i is approximately $w_i\lambda_i E[I_i]^2/2$, $1 \le i \le n$.*

APPROXIMATION 2. *The average scaled age any queue when it commences service is approximately the same for all queues.*

Under these two approximations, for $1 \le i, j \le n$,

$$\frac{w_i\lambda_i E[I_i]^2/2}{w_j\lambda_j E[I_j]^2/2} \approx 1.$$

Therefore, since $E[I_i] = (1 - \rho_i)E[C_i]$,

$$f_{ij} = \frac{E[C_i]}{E[C_j]} \approx \sqrt{\frac{w_j\lambda_j(1-\rho_j)^2}{w_i\lambda_i(1-\rho_i)^2}}, \tag{2.1}$$

which may be solved to find the w_i (once the f_{ij} have been defined), $1 \le i, j \le n$.

Discussion of Approximation 1. If customers were to arrive according to a deterministic flow (rather than Poisson arrivals), then the approximation would be exact. This approximation may therefore be viewed as a "fluid" approximation.

The number of customers that arrived to queue i in a given intervisit period I_i is Poisson($\lambda_i I_i$), $1 \le i \le n$. Suppose the distribution of the number of these customers within I_i is independent of the magnitude of I_i. Then there will be on average $\lambda_i E[I_i]$ customers at queue i upon commencement of service at queue i. Also, each customer will have an average scaled age of $w_i E[I_i]/2$. This yields a total expected scaled age of $w_i\lambda_i E[I_i]^2/2$ (as in Approximation 1). This is an approximation because if no customers arrive at the beginning of an intervisit period, then that would tend to make the intervisit period longer than if many

customers arrive at the beginning. Therefore, the distribution of the number of customers within an intervisit time is not completely independent of the length of that period.

In §3.1 we present numerical results from simulating 18 different 10-queue systems. These systems were designed to test the effect of different system parameters on the heuristic. In order to get a feel for this approximation, for each test case, and for each queue i, $1 \le i \le n$, we computed the simulated values for $w_i\lambda_i E[I_i]^2/2$ (call this x_i) and the average scaled age upon commencement of queue i (call this y_i). We then computed $\Sigma_{i=1}^n (x_i - y_i)^2/(\Sigma_{i=1}^n x_i)^2$. Over all 18 test cases this metric had an average value of 0.016, a minimum value of 0.0002, and a maximum value of 0.215. The only trend we observed in this metric was that as the system became less symmetric the value increased. However, simulation is not good at accurately estimating such small values, and hence this trend might be more a result of the natural variability caused by the added asymmetry rather than because of a real trend. The real test of this approximation will come in the performance of the heuristic.

Discussion of Approximation 2. If the system is symmetric, then Approximation 2 is exact. It will become less exact as the system becomes more unbalanced. If the system is made up of a large number of moderately similar queues, then we would expect this approximation to perform well. For each of the eighteen test cases of §3.1 and each queue i, $1 \le i \le n$, we let y_i be average scaled age upon commencement of queue i. We then computed $\Sigma_{i=1}^n (y_i - \bar{y})^2/((n-1)\bar{y}^2)$ where $\bar{y} = \Sigma_{i=1}^n y_i/n$. Over all 18 test cases this metric had an average value of 0.012, a minimum value of 0.000002, and a maximum value of 0.053. As expected, as the system became less symmetric this metric increased. The one symmetric example in the test cases (case 1) had the minimum value of 0.000002. As the theoretical value for this case is zero, this serves as an illustration of the accuracy of the simulation. Again, the real test of this approximation will come in the performance of the heuristic.

2.3. Queue Visit Frequencies

As described in the previous section, once we have derived desired queue visit frequencies, the heuristic policy is completely defined. This section discusses the selection of queue visit frequencies that approximately minimize the expected waiting time. These queue visit frequencies lead to the following scale factors:

$$w_i = \frac{1}{s_i(1-\rho_i)}, \quad 1 \le i \le n. \tag{2.2}$$

We do not assume any methodology for deciding which queue to visit next but merely assume that such a methodology results in the desired visit frequency at each queue. The derived visit frequencies turn out to be the same as those found by Boxma et al. (1991) for the special case of queues that are visited in a fixed order according to a polling table. In order to derive these frequencies we make the following two assumptions that are discussed further below.

APPROXIMATION 3. *Suppose that the squared coefficient of variation of the intervisit time at each queue is approximately the same for all queues and does not significantly change with different policy types.*

APPROXIMATION 4. *Suppose the system is never idle.*

LEMMA 2. *Under Approximations* 3 *and* 4 *the queue visit frequencies that minimize mean waiting time for dynamic exhaustive service policies are*:

$$f_{ij} = \sqrt{\frac{s_i\lambda_j(1-\rho_j)}{s_j\lambda_i(1-\rho_i)}}, \quad 1 \le i, j \le n.$$

PROOF. Let X_i be the waiting time in an $M/G/1$ queue with arrival rate λ_i and service time B_i, $1 \le i \le n$. Then the *decomposition property* for $M/G/1$ queues with generalized vacations of Fuhrmann and Cooper (1985) implies that

$$E[W_i] = \frac{E[I_i^2]}{2E[I_i]} + E[X_i], \tag{2.3}$$

where I_i is the intervisit time for queue i, $1 \le i \le n$. Rearranging yields

$$E[W_i] = E[I_i](c_{I_i}^2+1)/2 + E[X_i], \tag{2.4}$$

where $c_{I_i}^2$ is the squared coefficient of variation for the intervisit time at queue i, $1 \le i \le n$. Therefore, under Approximation 3, choosing a scheduling policy which

minimizes $\sum_{i=1}^{n} \lambda_i E[W_i]$ is equivalent to choosing one which minimizes $\sum_{i=1}^{n} \lambda_i E[I_i]$ such that the $E[I_i]$ are feasible intervisit times for an exhaustive service system. Under Approximation 4, work balancing arguments (i.e., work flow into the system equals work flow out of the system) imply that (in steady-state), for $1 \le i \le n$,

$$E[C_i] = \sum_{j=1}^{n} f_{ij} s_j + \rho E[C_i],$$

and hence

$$E[C_i] = \frac{\sum_{j=1}^{n} f_{ij} s_j}{1 - \rho}.$$

However, $f_{ij} = E[C_i]/E[C_j]$, $1 \le i, j \le n$, so that

$$\sum_{j=1}^{n} s_j / E[C_j] = 1 - \rho.$$

But $E[I_i] = (1 - \rho_i) E[C_i]$, $1 \le i \le n$, and thus

$$\sum_{j=1}^{n} (1 - \rho_j) s_j / E[I_j] = 1 - \rho. \tag{2.5}$$

Minimizing $\sum_{i=1}^{n} \lambda_i E[I_i]$ subject to (2.5) yields

$$f_{ij} = \sqrt{\frac{s_i \lambda_j (1 - \rho_j)}{s_j \lambda_i (1 - \rho_i)}}, \quad 1 \le i, j \le n.$$

Notice that these visit frequencies imply that as some s_i increases the server is less likely to visit queue i (relative to other queues). In other words, it switches less often to queues with a high setup overhead. Also, as some λ_j increases (with ρ_j constant), f_{ij} increases. Therefore the server is more likely to go to a queue where serving to exhaustion will cause a large number of customers to be served.

Solving (2.1) with the above visit frequencies yields the scale factors given in (2.2).

Discussion of Approximation 3. Approximation 3 is similar to ones made by previous authors. Browne and Yechiali (1989) replace the objective of minimizing $\sum_{i=1}^{n} \lambda_i E[W_i]$ (which is "computationally hard") by that of minimizing $\sum_{i=1}^{n} E[C_i]$ ("a greedy objective"). Also Boxma et al. (1991) show that waiting time is minimized when $c_{I_i}^2$ is taken to be zero, $1 \le i \le n$, and perform their heuristic analysis assuming that this is the case. In simulation tests we found that $c_{I_i}^2$ is significantly less than one, and hence most of the magnitude of $E[W_i]$ arises from $E[I_i]$, $1 \le i \le n$. Olsen (1996) shows that $c_{I_i}^2$ in CSTE queues converges to zero for all queues i, $1 \le i \le n$, as both $\rho \uparrow 1$ and $s \uparrow \infty$, where s is total mean setup in a cycle. This is a result of a strong law of large numbers effect where the intervisit time is a sum of a large number of service times. In simulation tests we found that this effect appears to carry over to these more dynamic system. For example, in Table 2 in §3.1 cases 12, 13, and 14 have increasing setup time magnitude. The simulated values for $\sum_{i=1}^{n} c_{I_i}^2/n$ for these three cases (under our heuristic) are 0.19, 0.11, and 0.07. Similarly, for cases 4, 5, and 6 in Table 2, where system load is increased, the simulated values of $\sum_{i=1}^{n} c_{I_i}^2/n$ for the three cases are 0.37, 0.29, and 0.26, which can be seen to be decreasing. For each of the 18 test cases in §3.1 we computed the average value of $\sum_{i=1}^{n} c_{I_i}^2/n$ for each of the 7 given heuristics. For each test case, the mean and standard deviation for $\sum_{i=1}^{n} c_{I_i}^2/n$ were computed over the 7 heuristics. The averages of these values were 0.295 and 0.079, respectively, for the mean and standard deviation. The maximum difference in $\sum_{i=1}^{n} c_{I_i}^2/n$ among the 7 heuristics across any one test example was 0.33.

Discussion of Approximation 4. Approximation 4 is equivalent to approximating the system by one where, if the system is empty, the server randomly selects the next queue to visit as that where the visit frequencies for individual queues remain in proportion to the original system. Notice that if setups are significant, then this approximation is unlikely to make a large difference as percentage idle time will be very small. The average idleness seen across the eighteen test cases simulated in §3.1 was 0.5%. The maximum idleness seen was 5%, and the minimum idleness seen was 0%.

3. Testing

In order to evaluate the assumptions made in §2, this section compares our suggested heuristic with other

known heuristic scheduling methods. The heuristics are compared using a very general simulation program that was written in the C programming language. The simulation run length was at least 5,000,000 times the average (over all queues) expected service time. The system is originally started empty. To avoid statistical problems due to "initial transience" (i.e., the system is not in steady-state when it is started), the first 10% of data is deleted. The statistics are collected using the method of "batch-means" (see, for example, Law and Kelton 1991) with 10 batches. The confidence intervals for the percentile statistics are calculated using the batch-mean related method developed in Mũnoz (1991). Because of space considerations we do not quote confidence intervals. However, in general, they were less than 5% of the estimated value. We also use common random numbers across the simulations so that the estimates are positively correlated. Our simulation was validated using the numerical examples provided in the cited papers.[1]

Our scheduling policy (denoted by HEUR) will be compared to the following alternative policies:

CSTE: Cyclic-Serve-To-Exhaustion (e.g., Takagi 1990)

CGS: Cyclic Gated Service (e.g., Takagi 1990)

MW: Most Work

BLW: Boxma et al.'s (1991) exhaustive service heuristic

BY: The exhaustive service heuristic of Browne and Yechiali (1989).

DVO: The heuristic suggested in Duenyas and Van Oyen (1996).

The MW scheduling method chooses the next queue to be served as that with the most expected work and serves exhaustively. This was shown by Liu et al. (1992) to minimize mean waiting time in symmetric systems. When setup times are zero the Duenyas and Van Oyen (1996) heuristic reduces to the $c\mu$ rule. Also, when all queues are symmetric, apart from its idling procedure, it is identical to MW.

After validating these heuristics, we then altered them in a couple of ways to make them more comparable to our heuristic HEUR. First, we are assuming that the server has knowledge of the system state, therefore it seems reasonable to modify traditionally static heuristics as follows:

1. No setups are done at empty queues;
2. If the system is empty, then the server idles at the most recently served queue (patient service); and
3. An arrival to an empty system causes the server to immediately jump to the new arrival's queue and begin a setup (if necessary).

Second, in MW, BY, and DVO, where there might be queues with ties, the ties are broken by choosing the queue with the most scaled age (i.e., according to the same rule as our heuristic).

In this study, all products are assumed to have equal priority. Extensions to systems with different priorities among the classes are discussed in §4. The main statistics we choose to compare are total mean wait, standard deviation of the total wait, and the 95th percentile of waiting time. The variance statistic is of only limited value as it is computed across all items. Therefore, even if all items had deterministic waiting times, if the waiting times for different items differ then variance will be nonzero. Although percentiles further out than the 95th (e.g., 99th) could be of even greater value to a manufacturer, we found that simulating such values to within a reasonable confidence interval was sometimes beyond the memory capacity of the workstation we were using.

3.1. Designed Problems

We begin by considering a symmetric 10-queue system with exponential service and deterministic setup times, both with mean 1 and system load, ρ, equal to 0.8. These parameters seem to be reasonable for the manufacturing systems of interest to us. This base model will be modified to test the effect of differing system parameters on the performance of our heuristic relative to the other heuristics. In particular, Table 1 gives the parameters used in the 18 test cases that we designed. A polling table of size 50 was used for the BLW heuristic except for cases 7 and 8 (which have more than 10 queues) where polling tables of size 100 and 200 were used, respectively.

Table 2 shows the percentage difference between our heuristic and the six other heuristics listed above

[1] This testing revealed a small typographical error on page 148 of Boxma et al. (1991), where $c_i = 1.0$ should read $c_i = 0.02$.

Table 1 Parameters Used for Testing

			Odd Queues					Even Queues				
	n	ρ	λ_i/λ	b_i	B_i distn.	s_i	S_i distn.	λ_i/λ	b_i	B_i distn.	s_i	S_i distn.
1	10	0.8	10	1	exp.	1	det.	10	1	exp.	1	det.
2	10	0.8	15	1	exp.	1	det.	5	1	exp.	1	det.
3	10	0.8	19	1	exp.	1	det.	1	1	exp.	1	det.
4	10	0.7	10	1	exp.	1	det.	10	2	exp.	2	det.
5	10	0.8	10	1	exp.	1	det.	10	2	exp.	2	det.
6	10	0.9	10	1	exp.	1	det.	10	2	exp.	2	det.
7	20	0.8	10	1	exp.	1	det.	10	2	exp.	2	det.
8	40	0.8	10	1	exp.	1	det.	10	2	exp.	2	det.
9	10	0.8	10	1	exp.	1	det.	10	2	exp.	1	det.
10	10	0.8	10	1	exp.	1	det.	10	3	exp.	1	det.
11	10	0.8	10	1	exp.	1	det.	10	4	exp.	1	det.
12	10	0.8	10	1	exp.	1	det.	10	1	exp.	2	det.
13	10	0.8	10	1	exp.	1	det.	10	1	exp.	5	det.
14	10	0.8	10	1	exp.	1	det.	10	1	exp.	10	det.
15	10	0.8	10	1	det.	1	det.	10	2	det.	2	det.
16	10	0.8	10	1	unif.	1	det.	10	2	unif.	2	det.
17	10	0.8	10	1	exp.	1	unif.	10	2	exp.	2	unif.
18	10	0.8	10	1	exp.	1	exp.	10	2	exp.	2	exp.

for the test cases give in Table 1. For each heuristic, both the mean waiting time ($E[W]$) and the 95th percentile of waiting time ($P(95)$) are compared. If y is the value that is being compared (e.g., 95th percentile of throughput time for CSTE) and x is HEUR's value then the percentage difference is computed as $100(y - x)/x$. Where numbers are negative this illustrates that the given heuristic is outperforming HEUR. The actual values for HEUR in case 1 are $E[W] = 23.9$ and $P(95) = 61.7$. This should give the reader some indication of the magnitude of the differences between the heuristics.

Case 1 (the base case) is a completely symmetric model. MW is known to be optimal in this case, and therefore HEUR's value for $E[W]$ can be seen to be about 4% suboptimal. However, as the system becomes more asymmetric, HEUR can be seen to outperform both MW and DVO even in terms of mean wait. Cases 4, 5, and 6 show that as load increases, the different heuristics become more comparable. The relative performance of the DVO heuristic probably worsens because it is the only heuristic with forced idleness. As load increases this idleness becomes less effective. Cases 5, 7, and 8 show that as the number of queues increase, HEUR improves its performance in terms of mean wait over all heuristics except for DVO. Furthermore, HEUR's performance in terms of 95th percentile also improves over all heuristics except for CSTE and CGS. Both CSTE and CGS are very "fair" heuristics and will tend to lead to even performance over all queues.

Cases 12, 13, and 14 show that as the setup times become more asymmetric (and increase) the performance of HEUR in terms of mean wait improves with respect to all the other different heuristics except for BEA. Mean waiting time becomes closer to that of BEA and hence Approximations 1 and 2 must have tightened. As Approximations 3 and 4 become more accurate as setups increase (see §2.3), mean waiting time performance improvement over policies other than BEA can be expected. The "fair" policies of CSTE, CGS, and BY appear to do better with respect to the 95th percentile as the setups become more asymmetric and increase. The service and setup time distributions appear to have little effect on system performance. This robustness to changes in distribution was noted in Boxma et al. (1990) for systems with polling tables.

The last row of Table 2 computes the average of all

Table 2 Percentage Differences to HEUR

	CSTE		CGS		MW		BEA		BY		DVO	
	E[*W*]	*P*(95)	*E*[*W*]	*P*(95)	*E*[*W*]	*P*(95)	*E*[*W*]	*P*(95)	*E*[*W*]	*P*(95)	*E*[*W*]	*P*(95)
1	9.58	3.70	26.92	10.80	−3.99	7.32	9.58	3.70	0.63	4.90	−3.98	7.32
2	11.14	3.40	33.26	12.33	−2.61	12.12	17.93	21.71	6.63	5.92	0.76	24.99
3	5.95	5.55	33.96	16.51	2.64	1.04	11.87	13.63	2.72	9.90	12.30	6.48
4	14.50	9.56	31.78	16.34	10.45	32.42	14.25	18.60	4.51	9.78	−4.56	13.21
5	13.07	6.30	30.36	12.55	13.64	32.78	16.10	19.28	4.12	7.68	−1.59	11.93
6	8.36	3.83	25.62	9.63	17.77	32.16	13.66	18.45	3.39	5.86	3.61	7.79
7	17.11	6.01	26.62	9.88	14.62	41.11	21.33	25.59	4.49	8.59	−2.14	18.52
8	19.55	4.96	24.40	6.95	14.97	46.63	24.50	29.86	4.71	9.05	−3.66	21.78
9	8.89	4.55	24.93	10.01	2.05	19.93	8.89	4.55	0.79	5.75	−3.11	9.20
10	7.10	4.61	19.57	6.97	14.93	39.24	7.10	4.61	0.45	5.61	−3.32	7.52
11	5.68	4.08	14.63	3.61	28.87	58.15	5.68	4.08	−0.10	4.78	3.56	13.78
12	10.66	2.12	28.30	9.60	−1.56	7.13	17.51	24.21	2.95	4.83	−2.85	11.85
13	18.10	−1.99	37.51	5.84	9.67	3.29	11.05	16.68	13.84	0.44	1.60	24.46
14	27.79	−3.59	49.32	4.60	22.33	0.59	8.46	21.56	25.40	−1.40	3.26	23.17
15	14.33	5.69	33.41	13.31	13.26	37.63	17.97	23.45	4.20	8.39	−1.34	16.21
16	13.73	5.95	32.21	13.21	13.36	35.85	17.30	21.93	4.13	8.19	−1.47	14.51
17	13.06	6.31	30.07	12.34	13.57	32.33	15.64	18.77	4.01	7.47	−1.90	11.16
18	12.41	5.97	29.51	12.11	13.65	32.26	15.33	18.53	3.97	7.43	−2.07	10.98
avg.	12.87	4.57	29.69	10.68	11.36	27.16	14.40	17.48	4.92	6.48	−0.56	13.84

18 rows. It can be seen that, on average, HEUR outperforms all other tested heuristics in all categories except for DVO. DVO has a slight advantage over HEUR in terms of mean wait but performs very poorly with respect to the 95th percentile of wait. We were somewhat surprised by the excellent performance of HEUR with respect to mean waiting time. Clearly, the approximations have served to improve performance rather than to degrade it.

3.2. Comparisons Using Industry Data Sets

Thus far the examples presented have been deliberately designed. However, we have been fortunate enough to obtain three data sets from actual industrial batch manufacturing environments. While the actual situations are not exact matches for our model (for differing reasons), these data sets provide a better set of test cases than is possible from designed problems.

The first data set is from a fence-making operation. A setup time corresponds to the rethreading of the machine with differing widths between the wires. The actual operation was a make-to-stock system, and setups were sequence dependent. In order to use these data for our nonsequence dependent environment, we ran a traveling salesman routine and found the minimum sequence of setups over all 19 products. These setup times were then assigned to each product as their fixed (nonsequence dependent) mean setup time. This data set is the most interesting of the three data sets because the setup and service times are different at different queues.

The data set consists of 19 different products, with an average mean service time of 0.1 hours and a range from 0.02 to 0.27. The average mean setup time over all products is 2 hours with a range from 0.17 to 11.3. The arrival rates at the different queues are very diverse and range from 0.17% of arrivals to 29.9%. We have the relative arrival rates along with the mean service and setup times but not the system utilization or the actual setup and service time distributions. We therefore run this data set for differing utilizations. Our testing showed that the results are robust to changes in service or setup distribution. The following table uses exponential service and setup times. When testing the BLW heuristic the polling table was taken to be of size 100.

Table 3 considers utilizations of $\rho = 0.5$, 0.7, and 0.9. The last row gives the actual values under HEUR and all previous rows give percentage difference to HEUR. HEUR is dominant both in terms of the mean and $P(95)$. The most competitive scheduling procedure is BLW. As in the designed examples, there is an apparent decrease in percentage differences as ρ increases. In the designed problems CSTE, CGS, and BY were frequently competitive. Here, where there is much higher variability in the test data set and thus serving each queue an equal number of times is not a good idea, they perform very poorly.

The next data set arises from a stamping operation of a major automobile manufacturer. This data set is the least interesting of the three because both mean service and mean setup times are identical at all queues. This is because neither the time to stamp one part nor the time to change a die depends noticeably on the particular die being used. The mean service time includes the average time spent servicing breakdowns and equals 0.24 minutes. Mean setup time is one hour. Percentage arrival rates at the 12 different queues range from 0.75% to 22.23% of all arrivals. The system utilization ρ is 0.76. Because the data set did not include distributions, we assume the setup times are deterministic and the service times are exponential. The polling table for the BLW heuristic is taken to be of size 51.

In the columns labeled "orig," Table 4 tests the various scheduling policies using the stamping data. The columns labeled "mod 1" and "mod 2" are designed to test the effect of asymmetry of setup times on the heuristic. For mod 1 we randomly generated setup times between 30 and 90 and then scaled the resulting setup times so that the mean setup was again 60. For mod 2 we subtracted 60 from the setup times in mod 1, doubled them, and then added 60. Therefore Table 4 shows the effect of increasing setup variability. The last row gives the actual values under HEUR, and all previous rows give percentage difference to HEUR.

Table 3 Effect of System Load in Fence-Making Data

	$E[W]$			σ_W			$P(95)$		
	$\rho = 0.5$	$\rho = 0.7$	$\rho = 0.9$	$\rho = 0.5$	$\rho = 0.7$	$\rho = 0.9$	$\rho = 0.5$	$\rho = 0.7$	$\rho = 0.9$
CSTE (%)	216.2	234.3	237.8	49.1	36.3	23.3	113.8	108.7	91.4
CGS (%)	249.5	290.8	314.7	46.9	37.4	23.8	122.7	125.1	113.0
MW (%)	61.9	68.0	67.9	37.7	40.5	37.3	43.2	43.9	36.5
BLW (%)	17.0	13.8	10.6	9.5	3.1	−2.4	29.3	27.5	24.8
BY (%)	226.5	236.3	233.2	52.7	39.2	21.6	120.1	111.6	88.5
DVO (%)	38.1	51.2	73.7	309.9	304.0	316.8	44.7	59.2	80.4
HEUR	12.0	18.7	52.7	17.8	29.7	87.8	40.6	64.2	187.4

Table 4 Effect of Asymmetric Setups on Stamping Data

	$E[W]$			σ_W			$P(95)$		
	orig	mod 1	mod 2	orig	mod 1	mod 2	orig	mod 1	mod 2
CSTE (%)	27.2	32.7	49.4	−15.7	−15.0	−13.3	−4.3	−2.9	2.3
CGS (%)	61.7	68.7	89.9	−15.6	−14.9	−13.2	8.9	10.7	16.5
MW (%)	17.8	16.0	22.7	142.2	130.2	121.9	35.1	30.5	28.6
BLW (%)	8.8	7.1	5.3	5.6	20.7	18.1	7.5	12.6	15.5
BY (%)	27.2	32.8	49.5	−15.6	−14.9	−13.2	−4.4	−2.8	2.2
DVO (%)	26.0	28.6	29.8	185.2	196.7	194.5	49.3	63.3	65.3
HEUR	17.3	16.6	14.8	15.2	15.1	14.8	44.0	43.3	41.1

Table 5 Effect of Setup Time for Thermofit Data

	$E[W]$			σ_W			$P(95)$		
	$s_i = 0.5$	$s_i = 1$	$s_i = 2$	$s_i = 0.5$	$s_i = 1$	$s_i = 2$	$s_i = 0.5$	$s_i = 1$	$s_i = 2$
CSTE (%)	18.2	19.5	20.8	−1.5	−7.6	−10.6	8.6	4.7	3.5
CGS (%)	20.8	22.8	24.5	−3.5	−8.6	−11.1	7.8	5.1	4.4
MW (%)	118.4	135.2	151.4	336.2	420.7	486.1	211.0	252.3	287.8
BLW (%)	16.2	13.4	10.1	14.3	12.2	8.8	15.7	15.0	13.9
BY (%)	12.4	16.1	19.5	7.7	−0.3	−5.4	13.2	8.8	7.2
DVO (%)	66.5	72.4	73.4	241.6	286.7	309.8	119.4	135.6	141.7
HEUR	120.6	246.9	500.0	119.5	221.2	429.6	341.9	195.5	1234.6

As would be expected, as setups become more variable the performance of the "fair heuristics" (CSTE, CGS, and BY) degrades. The trends with respect to the other heuristics do not appear to be linear.

The last data set arises from Raychem Corporation's Thermofit division and is given in Lennon (1994). Thermofit produces a product known as heat-shrinkable tubing. Product families in this setting correspond to different diameters of tubing. The final step in the Thermofit production process is executed by machines called expanders and production is make-to-order. This data set arises from a single workcenter consisting of four identical expanders. We will conform this multi-server environment to our single-server model by scaling arrival rates down by 4. The workcenter processes 69 product families, and a setup time of mean 1 hour is incurred each time an expander switches from processing one product family to processing another. This setup consists of a number of die changes, as well as re-threading of the new diameter tubing. Although Thermofit is always striving to reduce setup times, the setup, by its very nature, must take a significant length of time. Product variety is Thermofit's competitive advantage and therefore this is a typical environment where we see our heuristic being of value.

Thermofit determined that service time was effectively deterministic for the first 55 queues and exponential for the remaining 14. This is because the first 55 queues corresponded to fixed order sizes, whereas order sizes for the remaining 14 queues was highly variable. Mean service time across all products was 1.76 hours with a range from 0.062 to 37.95. The setup time distribution was taken to be exponential at all queues. The percentage of arrivals to each queue ranged from 0.03% to 5.78%. System utilization was 0.89. For the BLW heuristic the polling table size was taken to be 205.

Table 5 tests the scheduling policies using the Thermofit data set. To test the effect of setup time magnitude the setup time at each queue is taken as 0.5, 1 (the actual value), and 2 hours. The last row gives the actual values under HEUR and all previous rows give percentage difference to HEUR. As for the designed cases, and as predicted by the approximations, performance in terms of mean wait decreases for all heuristics other than BLW relative to HEUR as setup time increases. Also, as previously noted, BLW becomes closer to HEUR as setup times increase.

4. Conclusions and Extensions

In summary, we found our scaled age heuristic HEUR to be extremely effective. Although some of the other heuristics were occasionally competitive there was no heuristic that was consistently competitive. We are especially encouraged by HEUR's excellent performance on the realistic data sets. Although the three data sets do not come from systems that are exact matches of our model, we feel that they are close enough to provide considerable insight. They provide significant evidence for our claim that HEUR is the "best available" heuristic for dynamically scheduling multi-product manufacturing systems with significant setups. Notice that our heuristic cannot be expected to

perform well when setups are not significant. In particular, the scale factor w_i is not defined for $s_i = 0$.

We are only considering environments where all items are of similar priority (which was indeed the case with the three companies we studied). However one can imagine situations where different classes have different priorities. In particular, many models (e.g., Duenyas and Van Oyen 1996, Boxma et al. 1991, etc.) have assigned a linear cost c_i to waiting at queue i, $1 \leq i \leq n$, and then attempted to minimize the expected cost per unit time. There is no reason why these costs cannot be included in the minimization in §2.3. If this is done, then the visit frequencies again correspond to those suggested by Boxma et al. (1991). This would lead to scale factors of

$$w_i = \frac{c_i}{s_i(1 - \rho_i)}, \quad 1 \leq i \leq n$$

for our scaled age heuristic.

It is not clear, however, how priorities should be used to weight waiting time percentiles. Such a study would depend greatly on the application of interest. One immediate way to alter priorities within the system is to serve high priority queues exhaustively and low priority queues with gated service. Again, the success of such a scheme would depend greatly on the application and is left until a suitable application is found.

The theory of §2 uses the fact that arrivals are Poisson. However, there is no reason the heuristic could not be used for system with general arrival patterns. However, since the theory behind it would no longer hold, it is not clear how well it would perform. The testing of the heuristic's robustness to changes in arrival distribution is left as the subject for future research.

References

Ayhan, H., T. L. Olsen. 1997. Scheduling of multi-class single-server queues under nontraditional performance measures. To appear *Oper. Res.*

Boxma, O., H. Levy, J. Weststrate. 1990. Optimization of polling systems. Performance '90. P. J. B. King, I. Mitrani, and R. J. Pooley, eds. Elsevier Science Publishers B.V., North-Holland, Amsterdam.

——, ——, ——. 1991. Efficient visit frequencies for polling tables: minimization of waiting cost. *Queueing Systems Theory Appl.* **9** 133–162.

Browne, S., U. Yechiali. 1989. Dynamic priority rules for cyclic type queues. *Adv. Appl. Probab.* **21** 432–450.

Buzacott, J. A., J. G. Shanthikumar. 1993. *Stochastic Models of Manufacturing Systems.* Prentice-Hall, Englewood Cliffs, NJ.

Cooper, R. B., S. Niu, M. M. Srinivasan. 1997. Setups in polling models: Does it make sense to set up if no work is waiting? To appear in *J. Appl. Probab.*

Duenyas, I., M. Van Oyen. 1996. Heuristic scheduling of parallel heterogeneous queues with set-ups. *Management Sci.* **42** 814–829.

Fuhrmann, S. W., R. B. Cooper. 1985. Stochastic decompositions in the $M/G/1$ queue with generalized vacations. *Oper. Res.* **33** 1117–1129.

Hofri, M., K. W. Ross. 1987. On the optimal control of two queues with server set-up times and its analysis. *SIAM J. Comput.* **16** 399–420.

Kekre, S., K. Srinivasan. 1990. Broader product line: A necessity to achieve success? *Management Sci.* **36** 1216–1231.

Law, A. M., W. D. Kelton. 1991. *Simulation Modeling & Analysis.* McGraw-Hill, New York.

Lennon, T. M. 1994. Response-Time Approximations for Multi-Server Polling Models, with Manufacturing Applications. PhD Thesis, Stanford University, Stanford, CA.

Liu, Z., P. Nain, D. Towsley. 1992. On optimal polling policies. *Queueing Systems Theory and Appl.* **11** 59–84.

Mũnoz, D. 1991. Cancellation methods in the analysis of simulation output. PhD Thesis, Stanford University, Stanford, CA.

Markowitz, D. M. 1996. A Unified Treatment of the Single Machine Scheduling Problem in a Dynamic Stochastic Environment. PhD Thesis. M.I.T., Cambridge, MA.

Nahmias, S. 1993. *Production and Operations Analysis.* 2nd ed. Irwin, Homewood, IL.

Olsen, T. L. 1996. Asymptotics for polling models with increasing setups. Under submission.

Pinedo, M. 1995. *Scheduling Theory, Algorithms, and Systems.* Prentice-Hall, Englewood Cliffs, NJ.

Reiman, M. I., L. M. Wein. 1998. Dynamic scheduling of a two-class queue with setups. *Oper. Res.* **4** 532–547.

Takagi, H. 1986. *Analysis of Polling Systems.* MIT Press, Cambridge, MA.

——, Ed. 1990. *Stochastic Analysis of Computer and Communication Systems.* Chapter 1. Elsevier Science Publishers B.V. North-Holland, Amsterdam.

Van Mieghem, J. A. 1995. Dynamic scheduling with convex delay costs: the generalized c-mu rule. *Ann. Appl. Prob.* **5** 808–833.

Webster, S., K. R. Baker. 1995. Scheduling groups of jobs on a single machine. *Oper. Res.* **43** 692–703.

Wolff, R. W. 1989. *Stochastic Modeling and the Theory of Queues.* Prentice-Hall, Englewood Cliffs, NJ.

Accepted by Hau L. Lee; received May 24, 1996. *This paper has been with the author* 6 *months for* 2 *revisions.*