Pergamon

0305-0548(95)00003-8

Computers Ops Res. Vol. 23, No. 1, pp. 37–51, 1996

0305-0548/96 $9.50 + 0.00

PERFORMANCE ANALYSIS OF SCHEDULING POLICIES IN RE-ENTRANT MANUFACTURING SYSTEMS

Y. Narahari† and L. M. Khan‡

Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India

(Received June 1994; in revised form January 1995)

Scope and Purpose—Distinct multiple visits to machine centers are a distinguishing feature of semiconductor manufacturing systems and some flexible manufacturing systems. Re-entrant lines are queueing network models that are congenial for the modeling of such systems. Scheduling policies play an important role in deciding the performance of re-entrant lines. In this article, we develop an efficient computation methodology for predicting the performance of scheduling policies in re-entrant lines. The methodology is based on *mean value analysis*, a well-known queueing network analysis technique, and we believe this is the first analytical methodology for analysis of re-entrant lines. The proposed methodology predicts cycle times and throughputs accurately and is overwhelmingly efficient compared to simulation. The results will be useful in the rapid performance analysis and design of semiconductor fabrication systems and flexible manufacturing systems.

Abstract—Re-entrant lines are a class of non-traditional queueing network models that are congenial for the modeling of manufacturing systems with distinct multiple visits to work centers. Analyzing the performance of scheduling policies in re-entrant lines is a problem of significant research interest. Re-entrant lines are non-product form owing to priority scheduling, and all the existing performance studies have used simulation for analysis. In this paper we present an approximate technique for analytical performance prediction of re-entrant lines. The technique is based on MVA (Mean Value Analysis). The running time of the algorithm is linear in the product of the system population and the number of operations, which makes it overwhelmingly efficient compared to simulation. A detailed comparison of performance values obtained through simulation and the proposed technique shows that the analytical estimates are quite accurate.

1. INTRODUCTION

In this paper, we consider a type of non-traditional queueing models called re-entrant lines and provide an efficient and accurate analytical methodology for evaluating their performance. Re-entrant lines [1] are appropriate for modeling manufacturing systems with distinct multiple job visits to work centers. Examples of such manufacturing systems include semiconductor fabrication facilities, thin film lines, and systems with rework tasks. The proposed method, based on mean value analysis (MVA) [2] facilitates explicit modeling of scheduling policies used in re-entrant lines. We provide detailed numerical results which show the accuracy of the proposed analytical technique.

In a re-entrant line, the parts visit the same machine several times, at different stages of processing, before exiting the system, thus making the flow *non-acyclic*. A re-entrant line can be described as follows. There is a set of *service centers* $\{1, 2, \ldots, m\}$. Service center $i \in \{1, 2, \ldots, m\}$

† Y. Narahari is an Assistant Professor at the Indian Institute of Science, Department of Computer Science and Automation. His teaching and research interests are in the areas of Performance Modeling and Evaluation, Dynamic and Stochastic Scheduling, and Performance Analysis of Computer Systems and Manufacturing Systems. He has a Ph.D. from the Indian Institute of Science. He has written several papers on Petri net models of manufacturing systems, deadlock analysis in manufacturing systems, and performance analysis of discrete event systems. He has recently co-authored a book entitled Performance Modeling of Automated Manufacturing Systems. He was also a INDO-US visiting scientist at the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology.

‡ L. M. Khan received the B.E. degree in Electronics from the Aligarh Muslim University, Aligarh, India, and the M.S. degree in Computer Science from the Indian Institute of Science, Bangalore. Since 1991, he has been a Doctoral Student at the Department of Computer Science and Automation, Indian Institute of Science. His research interests include performance evaluation of manufacturing systems and data networks, and dynamic and stochastic scheduling.

has n_i logical or physical buffers, $b_{i1}, b_{i2}, \ldots, b_{in_i}$. For $j \in \{1, 2, \ldots, n_i\}$, the buffer b_{ij} contains parts visiting service center i for the jth time. A part visits these buffers in a given sequence and any service center is typically visited several times in the route of a part.

Figure 1 shows a typical re-entrant line with 3 service centers and 11 buffers. Parts enter the system at buffer b_{11} and visit the centers according to a deterministic route as shown. Finished parts emerge from center 3 after undergoing processing following a wait in b_{33}. Note that every part in this example line visits center 1 three times, center 2 five times, and center 3 three times.

1.1. Scheduling in re-entrant lines

There are two important decisions that have significant effect on the performance of a re-entrant manufacturing system. These are:

1. *Input release policies*, that decide when to release fresh jobs into the system.
2. *Dispatching or scheduling policies*, that decide which job to process next when a processing equipment becomes available.

Several factors, such as the availability of raw material and the current demand of the product influence the choice of input release policy in any manufacturing facility. A very popular policy is Fixed WIP (Work in Progress) policy. In this policy a fresh job is released into the system only when a finished job emerges from the system, thus the number of jobs in the system (WIP) is always fixed. When such a policy is used the stability of the system is guaranteed, and the system can be modeled as a closed queueing network.

The scheduling or dispatching problem in a re-entrant line becomes interesting because several parts at different stages of processing may be in contention with one another for service at the same machine. Several researchers have focused on the issue of scheduling in re-entrant lines [1, 3, 4, 5, 6, 7, 8]. Glassey and Resende [5] have considered the performance of four input release policies: Uniform, Fixed Work in Process, Workload Regulating, and Starvation Avoidance. Wein [6] has investigated the effect of both input release policies and dispatching policies, in a detailed simulation study. Bai and Gershwin [7] have made an excellent study of all issues to be considered in the scheduling of re-entrant lines arising in semiconductor manufacturing systems. For the same class of systems, Srivatsan *et al.* [8] have recently developed a software testbed for experimenting with hierarchical scheduling algorithms.

Distributed scheduling policies based on buffer priorities and due dates have been formulated and investigated by Kumar [1], Lu and Kumar [3], and Lu *et al.* [4]. Lu and Kumar [3] have investigated, among others, the FCFS (First Come First Serve) policy and the following fixed buffer priority policies:

- FBFS (First Buffer First Serve)
- LBFS (Last Buffer First Serve).

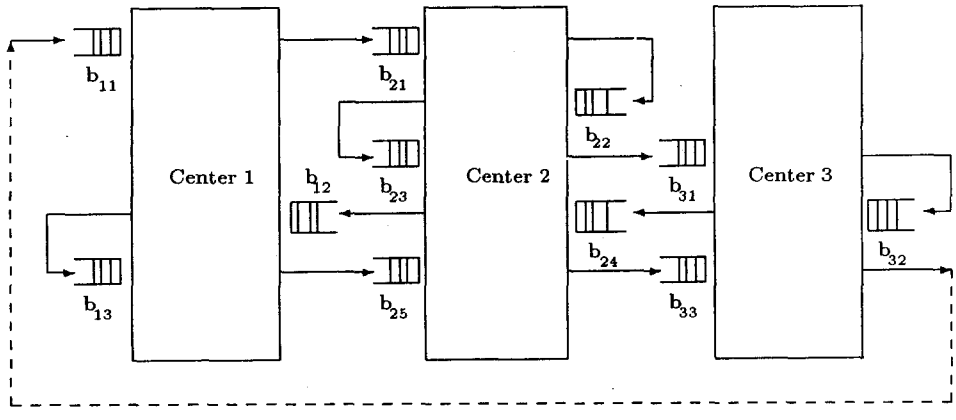

Fig. 1. A re-entrant line with 3 stations and 11 buffers.

For example, in the case of LBFS, we order the n_i buffers of processing center i as b_{in_i}, $b_{i,(n_i-1)}, \ldots, b_{i2}, b_{i1}$ in decreasing order of priority. Note that when a processing center finishes processing a part, it selects, from among the parts contending for that processing center, the one that has finished most of its processing, and hence has the least remaining processing. Thus we may say that each processing center *myopically* tries to clear parts from the system as fast as possible.

The above papers have also investigated the following due-date based policies:

- EDD (Earliest Due Date first)
- LS (Least Slack first).

More recently, Lu *et al.* [4] have proposed a class of *fluctuation smoothing* scheduling policies that aim at minimizing the mean or variance of total delay in the system. Extensive simulation results have been provided for these scheduling policies.

Recently, Connors *et al.* [9] have presented an open queueing network model designed for rapid performance analysis of semiconductor manufacturing facilities. In this paper, they assume an FCFS scheduling policy at all nodes of the queueing network and use a decomposition approach to analyze the queueing network model. Their model captures reworking and scrapping of jobs, and different types of incapacitation of events, but does not model scheduling policies other than FCFS.

1.2. Contributions of the paper

Existing results on the performance of re-entrant lines [1, 3, 4, 5, 6] are mostly based on simulation modeling. The main reason for this is the non-product form nature of re-entrant lines [1]. The non-product form features in re-entrant lines include:

- Priority scheduling among the buffers at a work center
- The processing times on different visits are different in general.

Schweitzer and Seidmann [10] have earlier considered the analysis of manufacturing systems with distinct multiple job visits to work centers. However they do not consider priority scheduling in their analysis. Priority scheduling has been considered by Shalev Oren *et al.* [11], however not in the context of re-entrant lines. In a recent paper, Kumar and Kumar [12] have presented an efficient linear programming based approach to obtain bounds on steady state performance measures for re-entrant lines and in general multiclass queueing networks. Their approach however does not yield mean values.

In this paper we propose an efficient method for approximate analysis of re-entrant lines employing *non-preemptive fixed buffer priorities*, and the fixed WIP input release policy. The method is based on mean value analysis (MVA) [2,13,14,11]. An important object of the analysis method is the ability to model scheduling policies such as LBFS, FBFS, and FCFS. The efficiency of the technique arises due to its iterative nature. The time complexity of the algorithm is $O(nb)$, where n is the population of the system, and b is the number of buffers in the system. This makes the analytical method overwhelmingly efficient compared to simulation. Also the method is quite accurate. A description of the proposed analytical method constitutes the subject of Section 2.

In Section 3, we present detailed numerical results obtained, through analytical and simulation experiments on some illustrative re-entrant lines. The performance indices considered are:

- mean steady-state cycle time or the mean steady-state total delay in the system
- mean steady-state throughput rate for a given fixed WIP in the system.

The simulation results have been found to validate the analytical method proposed to a fair degree of accuracy.

2. AN APPROXIMATE ANALYSIS METHODOLOGY

MVA yields expressions for mean values of performance measures such as steady-state queue lengths, delays, and throughputs. Two versions of MVA exist, namely, the *exact* MVA for product form queueing networks [2] and *approximate* MVA for non-product from networks [15]. Exact

MVA is based on the *Arrival Theorem*, which states that, in the steady state of a closed product form network with population k, the distribution of the network state seen by a job arriving at any node in the network is the same as the distribution of the network state a random observer would see with $(k-1)$ jobs circulating in the network.

In the literature, several extensions have been proposed to MVA to account for non-product form features [15, 11, 16, 17, 18, 19, 20, 21, 22]. Of special interest here are the MVA extensions for handling priority scheduling. Most of these approaches consider only preemptive priorities [23, 24, 16, 20, 19, 18].

The MVA extension proposed in this paper is unique in the sense that it takes into account in a natural way the following features of a re-entrant line:

- Deterministic route of parts in a re-entrant line.
- Multiple job visits to the same work center.
- Priority scheduling based on buffer priorities.

2.1. Assumptions and notation

The proposed analytical technique assumes that when a processing center i finishes servicing a part, it selects the next part for processing from among the buffers $b_{i1}, b_{i2}, \ldots, b_{in_i}$ in a fixed priority order, which is independent of the state of the system. We shall assume that the priorities accorded are non-preemptive. Further, parts in any given buffer are assumed to be processed in FCFS fashion.

We shall illustrate the formulation of MVA equations by assuming the LBFS scheduling policy. We assume that each processing center has exactly one machine and that the processing time of a job visiting center i on its jth visit is an independent exponentially distributed random variable with rate μ_{ij}. In the LBFS scheduling policy, parts visiting center i for the jth time get priority over parts visiting this center for the rth time where $r = 1, \ldots, j-1$. For example, in center 2 of Fig. 1, buffer b_{25} would get priority over b_{24}, b_{23}, b_{22} and b_{21}; buffer b_{24} would get priority over b_{23}, b_{22}, and b_{21}; and so on.

To apply MVA, we have to assume that the re-entrant line is a closed queueing network. This assumption is valid if the input release policy is a fixed-work-in-process policy (a fresh job is released into the network as soon as a finished job leaves the system) [5, 6]. Let N be the total number of jobs in the system. We shall use the following indices: i denotes a processing center; j denotes a buffer at a given processing center; k denotes a current job population and has the range, $1, \ldots, N$. Let state (i,j) correspond to the waiting or the processing of a job visiting center i for the jth time.

Let the performance measures of the network be denoted as follows.

$L_{ij}(k)$: mean steady-state number of jobs in stage (i,j) when the network has k jobs.
$W_{ij}(k)$: mean steady-state delay for jobs in stage (i,j) (mean waiting time in buffer b_{ij} + mean processing time)
$\lambda(k)$: mean steady-state throughput rate of jobs when the network has k jobs.

If $W(k)$ denotes the mean total delay (mean cycle time) in the entire network, we immediately have

$$W(k) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} W_{ij}(k). \tag{1}$$

Using MVA, we compute $W(N)$, and $\lambda(N)$ in a recursive way.

We also distinguish between *external* and *internal* buffers. We call a buffer b_{ij} external if the buffer feeding b_{ij} is connected to a center different from center i, and buffer b_{ij} is called internal if the buffer feeding b_{ij} is connected to center i itself. For example, in the re-entrant line of Fig. 1, consider center 2. The buffers b_{21}, b_{24}, and b_{25} are external, since arrivals into these buffers come from center 1, center 3, and center 1, respectively. The buffers b_{22} and b_{23} are internal because they are directly fed by outputs from center 2 itself.

At the processing center i, let us denote by SE_i and SI_i, the sets of external and internal buffers

respectively. Thus for the center 2 in the Fig. 1 we have

$$SE_2 = \{b_{21}, b_{24}, b_{25}\}$$

$$SI_2 = \{b_{22}, b_{23}\}.$$

Using the notion of internal buffers, we introduce another notion, that of a *chain of buffers*. An ordered set of consecutive internal buffers at a center, each feeding the next one, together with the external buffer, which feeds all these internal buffers, is known as a *chain of buffers*. For example, at processing center 2, the ordered set (b_{21}, b_{22}, b_{23}) forms a chain of buffers. We shall call the next first buffer of a chain (the external one) as the *head of the chain*. Each isolated external buffer may also be considered as a chain consisting of the head buffer alone, e.g. b_{25} may be thought of as a chain of buffers, even though jobs from this buffer immediately leave center 2.

This way we can partition the ordered set of all buffers at any center into chains. For example the ordered set of buffers at center 2 can be partitioned into chains as follows

$$[(b_{21}, b_{22}, b_{23}), (b_{24}), (b_{25})].$$

In general let C_i denote the ordered set of chains at center i. Then

$$C_i = (C_{i1}, C_{i2}, \ldots, C_{ik_i})$$

where k_i is the number of chains at center i and $C_{i1}, C_{i2}, \ldots, C_{ik_i}$ are the individual chains.

It will also be useful to define H_i, the ordered set of head of chain buffers for each center i. For example,

$$H_2 = (b_{21}, b_{24}, b_{25}).$$

2.2. Computation of performance measures

We consider the calculation of $W(N)$ and $\lambda(N)$. It would be helpful to consider the scenario a job would see upon its arrival at a certain buffer of a machine, and the sequence of events that occur while it is waiting there.

When a job (we shall call it a distinguished job) arrives at a buffer, say b_{ij}, it sees a certain number of jobs in various buffers in the system, the ordered set of these integers forms the *state of the system* at the arrival instant of the job. Let S be the set of jobs, currently at center i and having higher priority than the distinguished job. Note that S will include all jobs that are ahead of the distinguished job in b_{ij} and all jobs in all buffers having higher priority than b_{ij}. The distinguished job must first wait until all jobs in S are serviced and leave the center i. Also, it must wait for the service completion of these jobs which arrive in higher priority buffers, during its wait in buffer b_{ij}. And finally it has to get processed before it enters the next buffer.

Hence, the mean total waiting time of a job at any buffer b_{ij} is seen as the sum of three components, let us call them Term 1, Term 2, and Term 3, defined as follows:

- Term 1: Mean total time until all jobs in the set S are serviced and leave center i.
- Term 2: Mean total time required to process all higher priority jobs which arrive during the stay of the distinguished job in the queue at b_{ij}.
- Term 3: Mean processing time of the distinguished part itself.

We now describe how Terms 1, 2, and 3 may be computed, by presuming that the arrival theorem is valid in the given network. In fact, the arrival theorem is not valid for this network since the network is not product form. However, since we are only seeking an approximate analysis, we assume the arrival theorem to be valid for this network and verify the accuracy of the approximation using detailed simulation results.

As the expressions for external buffers are quite different from those of internal buffers, we first describe the method for re-entrant lines which have no internal buffers, and later describe the modifications to be made in the case where internal buffers are present.

2.2.1. Analysis of lines without internal buffers

Computation of Term 1. Consider the buffer b_{ij}. In this case, an arriving job, according to arrival

theorem, would see $L_{it}(k-1)$ jobs in the buffers b_{it} where $t = 1, 2, \ldots, n_i$. Since LBFS scheduling policy is being used, the arriving job needs only to wait for the processing of jobs ahead of it in buffers b_{it} where $t = j, j, +1, \ldots, n_i$. Thus

$$\text{Term } 1 = \sum_{t=j}^{n_i} \frac{L_{it}(k-1)}{\mu_{it}}. \tag{2}$$

Computation of Term 2. The mean waiting time (*excluding processing time*) of a job in buffer b_{ij} is

$$W_{ij}(k) - \frac{1}{\mu_{ij}}.$$

During this waiting, parts may arrive into higher priority buffers at center i. Term 2 is the mean total time required to process all such parts. Since all the buffers in the model are external, then during the waiting, parts may arrive into any of the higher priority buffers (from other machines). By assuming the arrival theorem, $\lambda(k-1)$ can be taken as the rate at which the jobs are flowing in the network and therefore

$$\text{Term } 2 = \left(W_{ij}(k) - \frac{1}{\mu_{ij}}\right)\lambda(k-1)\left(\sum_{t=j+1}^{n_i} \frac{1}{\mu_{it}}\right). \tag{3}$$

Computation of Term 3. The mean processing time required for the service of distinguished part itself is of course $\frac{1}{\mu_{ij}}$. Thus Term $3 = \frac{1}{\mu_{ij}}$.

The total waiting time $W_{ij}(k)$ is now given by

$$W_{ij}(k) = \text{Term } 1 + \text{Term } 2 + \text{Term } 3. \tag{4}$$

Now using (1), $W(k)$ can be computed.

Applying Little's Law [25] for the job population in the network, we obtain

$$\lambda(k) = \frac{k}{W(k)}. \tag{5}$$

We can again use Little's Law to obtain

$$L_{ij}(k) = \lambda(k) W_{ij}(k). \tag{6}$$

Consider the following initial conditions

$$L_{ij} = (0); \quad i = 1, \ldots, m \qquad j = 1, \ldots, n_i. \tag{7}$$

$$\lambda(0) = 0. \tag{8}$$

Using the initial conditions above and the recurrence relations defined by (4) to (6), and the initial values (7) and (8), we can compute $W_{ij}(k)$, $L_{ij}(k)$, and $\lambda(k)$ for $k = 1, 2, \ldots, N$. Thus $W(N)$ and $\lambda(N)$ can be computed.

2.2.2. *Analysis of lines with internal buffers*

Computation of Term 1 for an external buffer. For the computation of Term 1, notice that, if there are any jobs in the high priority head of chain buffers when the distinguished job arrived at b_{ij}, then all of them must leave center i before a job from b_{ij} can be taken up for service. The same is true for all high priority internal buffers. So

$$\text{Term } 1 = \sum_{\substack{t \geq j \\ b_{it} \in H_i}} L_{it}(k-1)\left(\sum_{b_{ik} \in C_{it}} \frac{1}{\mu_{ik}}\right) \tag{9}$$

$$+ \sum_{\substack{t > j \\ b_{it} \in SI_i}} L_{it}(k-1)\left(\sum_{\substack{k \geq t \\ b_{ik} \in C_{it}}} \frac{1}{\mu_{ik}}\right). \tag{10}$$

Where C_{it} is the chain of buffers to which b_{it} belongs.

For example, consider buffer b_{21} in Fig. 1. Term 1 in this case can be seen to be

$$\text{Term } 1 = L_{21}(k-1)\left(\frac{1}{\mu_{21}}+\frac{1}{\mu_{22}}+\frac{1}{\mu_{23}}\right)+L_{24}(k-1)\left(\frac{1}{\mu_{24}}\right)+L_{25}(k-1)\left(\frac{1}{\mu_{25}}\right)$$
$$+L_{22}(k-1)\left(\frac{1}{\mu_{22}}\right)+L_{23}(k-1)\left(\frac{1}{\mu_{23}}\right).$$

Computation of Term 2 for an external buffer. While the distinguished job waits in b_{ij} for processing, some new jobs will arrive into external buffers of center i. Of these, the jobs which go to higher priority buffers must be processed before the distinguished job could be taken up. Hence

$$\text{Term } 2 = \left[W_{ij}(k)-\frac{1}{\mu_{ij}}\right]\lambda(k-1)\sum_{\substack{t>j\\ b_{it}\in H_i}}\sum_{b_{ik}\in C_{it}}\frac{1}{\mu_{ik}}. \tag{11}$$

Again C_{it} is the chain of buffers to which b_{it} belongs.

For example, again consider buffer b_{21} in Fig. 1. The Term 2 in this case is

$$\text{Term } 2 = \left[W_{21}(k)-\frac{1}{\mu_{21}}\right]\lambda(k-1)\left(\frac{1}{\mu_{24}}+\frac{1}{\mu_{25}}\right).$$

Computation of Term 1 for an internal buffer. If b_{ij} is an internal buffer, then the distinguished job which arrives in b_{ij} must have come from $b_{i,j-1}$. And the machine would have taken up the service at $b_{i,j-1}$, only when all the buffers have priority higher than b_{ij} were empty (LBFS policy requires this). Therefore any jobs which the distinguished job sees resident in higher priority (external) buffers upon its arrival at b_{ij}, must have come there while the distinguished job was undergoing processing at buffer $b_{i,j-1}$. Since the mean processing time at buffer $b_{i,j-1}$ is $\frac{1}{\mu_{i,j-1}}$, we have

$$\text{Term } 1 = \frac{1}{\mu_{i,j-1}}\lambda(k-1)\sum_{\substack{t>j\\ b_{it}\in H_i}}\sum_{b_{ik}\in C_{it}}\frac{1}{\mu_{ik}}. \tag{12}$$

For example, consider buffer b_{22} in Fig. 1. Here,

$$\text{Term } 1 = \frac{1}{\mu_{21}}\lambda(k-1)\left(\frac{1}{\mu_{24}}+\frac{1}{\mu_{25}}\right).$$

Computation of Term 2 for an internal buffer. It can be easily seen that Term 2 for an internal buffer is the same as given by (11).

Computation of Term 3. Clearly, Term 3 for any buffer b_{ij} (whether external or internal) is $\frac{1}{\mu_{ij}}$.

The recursion relations given by (9), (10) and (11) can be unfolded using initial values given by (7), and (8). Thus $W(N)$ and $\lambda(N)$ can be computed as before.

2.2.3. *Complexity of the methodology*

For a re-entrant line with a total of b buffers and having population N, the algorithm does N iterations. Each iterative step involves computations for each of b buffers. Hence time complexity of the algorithm is $O(Nb)$.

As each iteration needs only the values of mean queue lengths and mean throughput rate computed in the previous iteration, the space complexity of the algorithm is $O(b)$.

If we want to calculate the performance parameters for a set of populations, a simulation has to be run for each value of the population, whereas the proposed algorithm need only be executed once, for the maximum value of population. The performance values for all lower populations are produced in the intermediate iterations automatically.

3. VALIDATION OF THE PROPOSED TECHNIQUE

To see how far the performance predictions afforded by this technique tally with the actual performance of re-entrant lines, a large number of re-entrant lines were simulated using CACI SIMSCRIPT II.5 and performance measures obtained. The same lines were then analysed for

performance measures by the analytical technique described in this paper. The final results obtained through these two methods were found to be very close to each other.

In particular we present results obtained for two re-entrant lines shown in Fig. 1 and Fig. 2. Note that while re-entrant line of Fig. 1 has many internal buffers and chains, there are no internal buffers in the line of Fig. 2. These two cases have been carefully chosen to facilitate experimenting with all the MVA equations developed for all the specific cases.

3.1. Re-entrant line 1

Consider the system in Fig. 1. The mean processing time at each buffer are assumed as given below.

$$\frac{1}{\mu_{11}} = \frac{1}{\mu_{12}} = \frac{1}{\mu_{13}} = \frac{1}{3}$$

Mean steady-state cycle time

The mean cycle time for the re-entrant line 1 obtained through analytical technique as well as simulation are plotted in Fig. 3 as a function of the system population. It can be seen from the graph that there is a close agreement in the results obtained by the two methods. The maximum discrepancy is about -5.1% at a population of 4.

Throughput rate

The throughput rate of the re-entrant line 1 obtained through analytical technique as well as simulation are plotted in Fig. 4 as a function of the system population. Again there is a close agreement between the results. The maximum discrepancy is about $+5.4\%$ at a population of 4.

3.2. Re-entrant line 2

This line is shown in Fig. 2 and does not have any internal buffers. The mean processing times at each buffer are assumed as given below.

$$\frac{1}{\mu_{11}} = \frac{1}{\mu_{12}} = \frac{1}{\mu_{13}} = \frac{1}{\mu_{14}} = \frac{1}{3}$$

$$\frac{1}{\mu_{21}} = \frac{1}{\mu_{22}} = \frac{1}{\mu_{23}} = \frac{1}{2}$$

$$\frac{1}{\mu_{31}} = \frac{1}{\mu_{32}} = 1.$$

Various performance measures for this re-entrant line obtained through both the methods are summarized in Table 1, along with percentage discrepancies between the results. The close agreement between the results is apparent from the last two columns of the table.

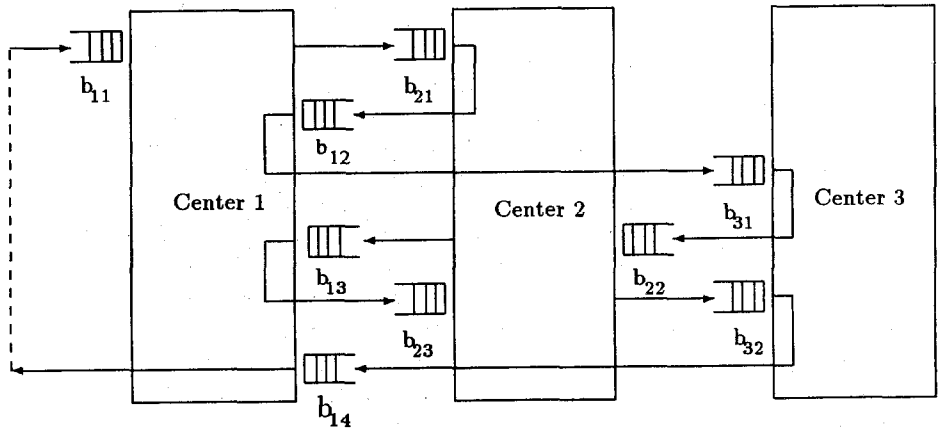

Fig. 2. A re-entrant line with no internal buffers.

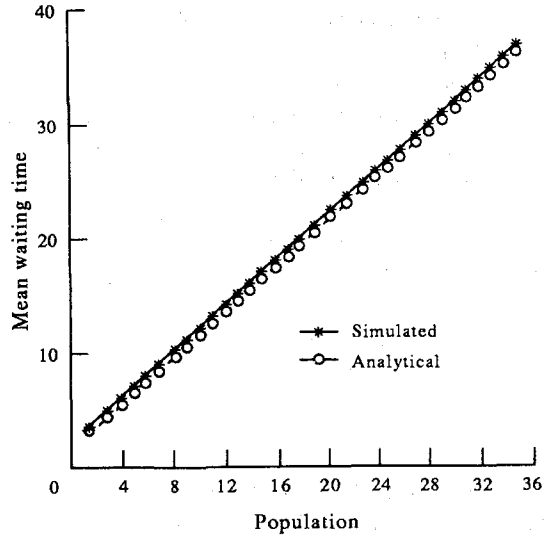

Fig. 3. Mean cycle time of re-entrant line 1 for different populations.

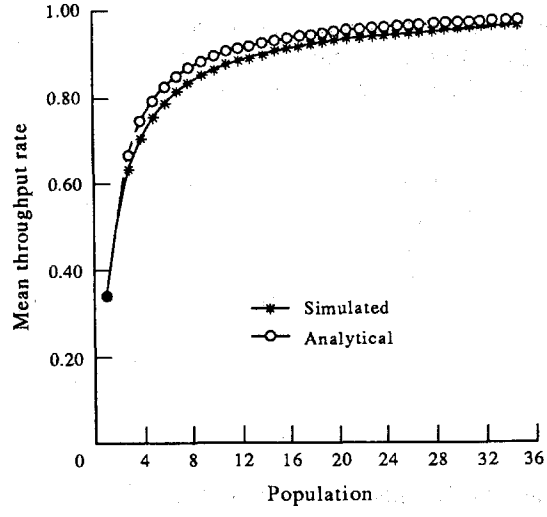

Fig. 4. Throughput rate of re-entrant line 1 for different populations.

3.3. Re-entrant line 3

Consider again the line of Fig. 2, with the following mean processing times at various stages of manufacturing.

$$\frac{1}{\mu_{11}} = 4 \quad \frac{1}{\mu_{12}} = 3 \quad \frac{1}{\mu_{13}} = 2 \quad \frac{1}{\mu_{14}} = 1$$

$$\frac{1}{\mu_{21}} = 10 \quad \frac{1}{\mu_{22}} = 5 \quad \frac{1}{\mu_{23}} = 2$$

$$\frac{1}{\mu_{31}} = 5 \quad \frac{1}{\mu_{32}} = 10.$$

Note that the mean processing times are different for different visits to the same processing center, which is a well known non-product form feature, even under a naive FCFS scheduling policy without any priorities. Figures 5 and 6 show the mean total cycle time and the throughput rate for

Table 1. Simulation and analytical results for the re-entrant line of Fig. 2

Popln of system	Cycle time (SIM)	Cycle time (MVA)	Throughput (SIM)	Throughput (MVA)	Pct Error (%)	Pct Error (%)
1	4.844	4.833	0.206	0.207	−0.23	+0.49
2	6.499	6.193	0.308	0.323	−4.71	+4.87
3	8.197	7.839	0.366	0.383	−4.56	+4.64
4	9.944	9.562	0.402	0.418	−3.84	+3.98
5	11.689	11.357	0.428	0.440	−2.84	+2.80
6	13.504	13.195	0.444	0.455	−2.29	+2.48
7	15.312	15.067	0.457	0.465	−1.60	+1.75
8	17.135	16.966	0.467	0.472	−0.99	+1.07
9	18.979	18.888	0.474	0.477	−0.48	+0.63
10	20.830	20.827	0.480	0.480	−0.01	+0.00
11	22.690	22.779	0.485	0.483	+0.39	−0.41
12	24.575	24.742	0.488	0.485	+0.68	−0.61
13	26.479	26.712	0.491	0.487	+0.88	−0.81
14	28.390	28.689	0.493	0.488	+0.05	−1.01
15	30.300	30.670	0.495	0.489	+1.22	−1.01
16	32.239	32.654	0.496	0.490	+1.29	−1.21
17	34.177	34.641	0.497	0.491	+1.36	−1.21
18	36.119	36.630	0.498	0.491	+1.41	−1.41
19	38.072	38.621	0.499	0.492	+1.44	−1.41
20	40.057	40.613	0.499	0.492	+1.39	−1.40
21	42.018	42.606	0.500	0.493	+1.40	−1.40
22	43.990	44.600	0.500	0.493	+1.39	−1.40
23	45.950	46.594	0.500	0.494	+1.40	−1.20
24	47.933	48.589	0.500	0.494	+1.37	−1.20
25	49.908	50.585	0.500	0.494	+1.36	−1.40
26	51.893	52.581	0.500	0.494	+1.33	−1.40
27	53.885	54.577	0.500	0.495	+1.28	−1.20
28	55.877	56.574	0.500	0.495	+1.25	−1.20
29	57.867	58.571	0.500	0.495	+1.22	−1.20
30	59.855	60.568	0.500	0.495	+1.19	−1.20
31	61.851	62.566	0.500	0.495	+1.16	−1.20
32	63.840	64.563	0.500	0.496	+1.13	−1.00
33	65.835	66.561	0.500	0.496	+1.10	−1.00
34	67.828	68.559	0.500	0.496	+1.08	−1.00
35	69.820	70.557	0.500	0.496	+1.06	−1.00

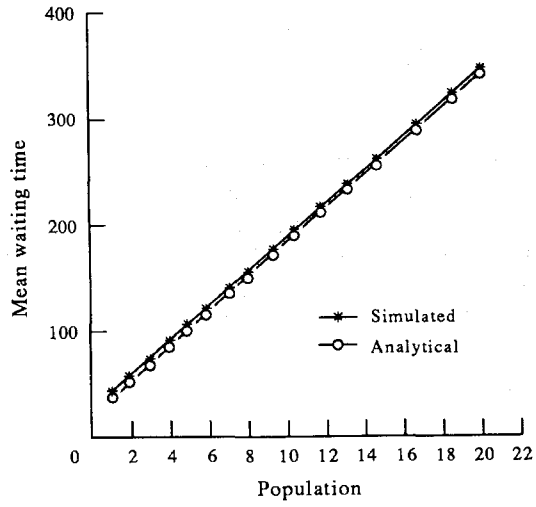

Fig. 5. Mean cycle time of re-entrant line 3 for different populations.

various populations. Both analytical and simulation results are shown and there is again a close agreement between these.

3.3.1. Waiting times at individual buffers

The objective of fixed buffer priority policies, such as FBFS or LBFS is to speed up the processing of some critical processing stages (with respect to some performance measures) by cutting down

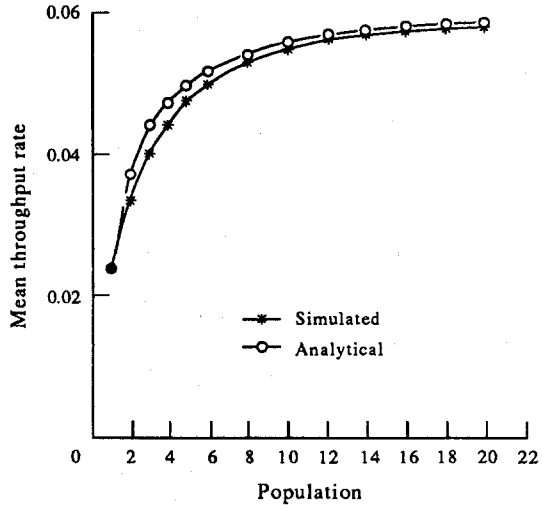

Fig. 6. Throughput rate of re-entrant line 3 for different populations.

Table 2. Mean waiting times at individual buffers in Re-entrant line 3

	W_{11}		W_{21}		W_{31}	
Population	SIM	MVA	SIM	MVA	SIM	MVA
1	4.04	4.00	10.16	10.00	9.98	10.00
2	4.28	4.83	12.48	13.68	13.66	12.38
4	5.29	6.29	21.01	23.80	16.66	16.35
6	6.44	7.17	34.87	35.70	19.13	18.86
8	7.37	7.73	51.86	49.19	21.43	20.49
10	8.03	8.10	71.58	64.92	23.10	21.71
12	8.67	8.37	93.03	81.51	24.20	22.80
14	8.83	8.56	116.55	100.55	24.59	22.98
16	9.03	8.70	141.08	121.98	25.40	23.39
18	9.06	8.80	168.73	144.60	25.72	23.70
20	9.24	8.87	198.01	169.49	26.00	23.99

queueing times at those stages. This is done at the cost of some other, less critical stages of processing. In our model, each stage of processing corresponds to a distinct (physical or logical) buffer. Any analytical technique for the analysis of re-entrant lines should not only be capable of predicting total cycle time, but it should also be able to predict waiting times at various stages of processing faithfully. A comparison of mean steady-state waiting times at individual buffers, obtained through simulation and those predicted by this algorithm is given in Table 2, for three buffers of re-entrant line 3. These buffers are (1, 1), (2, 1), and (3, 1), which are the lowest priority buffers at the respective processing centers under the LBFS policy. As can be seen, the agreement between the two is very good.

4. CASE STUDY OF A FULL SCALE RE-ENTRANT LINE

The re-entrant lines we have considered so far were academic in nature as they do not reflect the magnitude of a realistic re-entrant manufacturing system, say a semiconductor fab. In this section we apply our analysis technique to a more realistic re-entrant line. This line was considered by Lu *et al.* [4] as a model of a full scale semiconductor fab. The model, shown in Fig. 7, consists of 12 processing stations and a total of 60 buffers. It must be pointed out that, in the model considered in [4], some of the stations consisted of more than one identical machines, we have replaced such stations by a single, n times faster machine, where n is the number of identical machines in the corresponding station of the original model. Let $\frac{1}{\mu_i}$ be the mean processing time at center

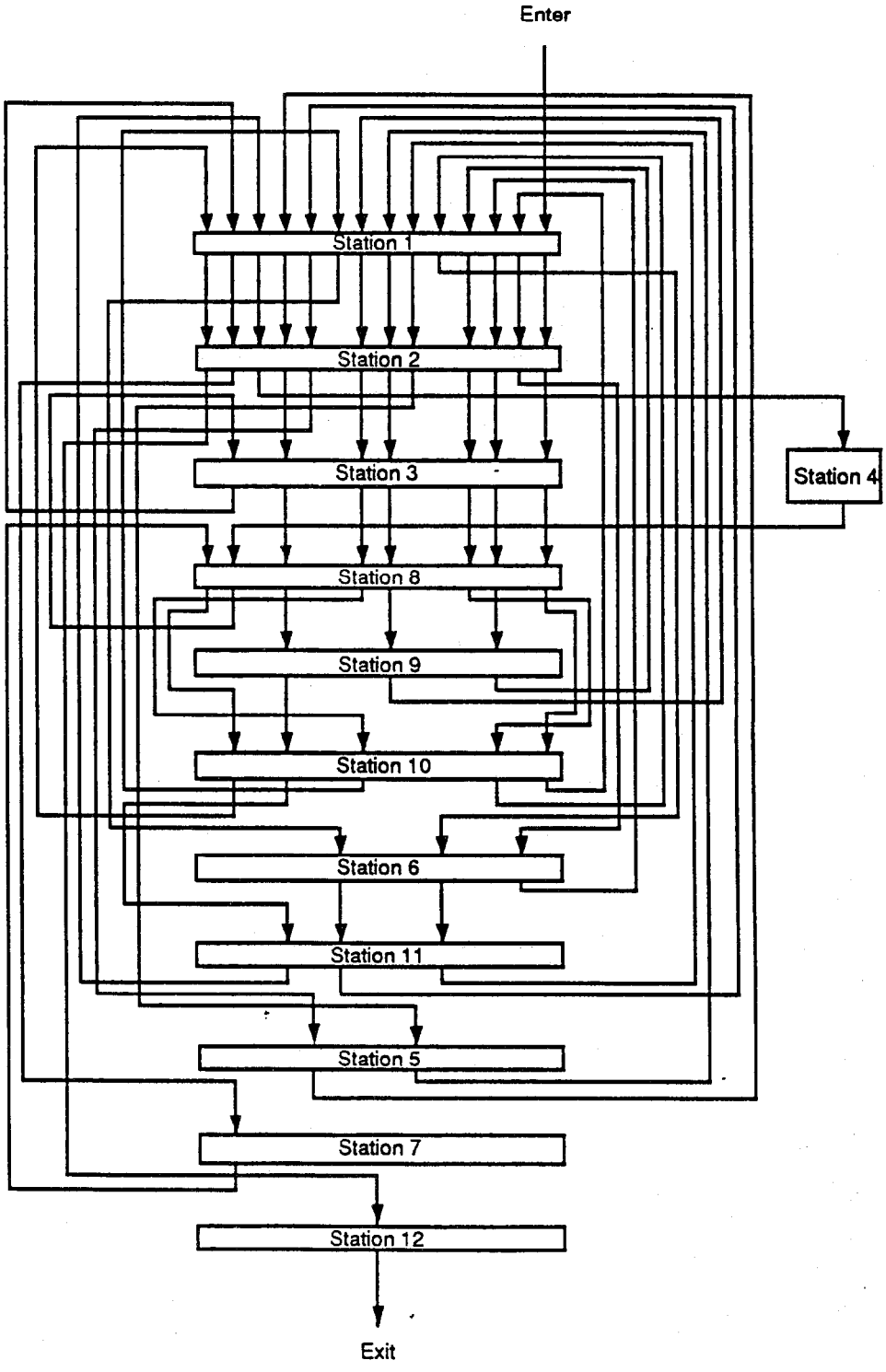

Fig. 7. A re-entrant line with 12 centers and 60 buffers.

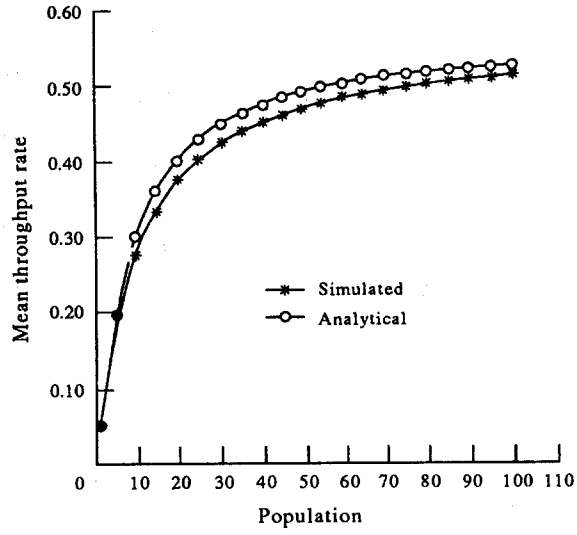

Fig. 8. Throughput rate under the LBFS policy.

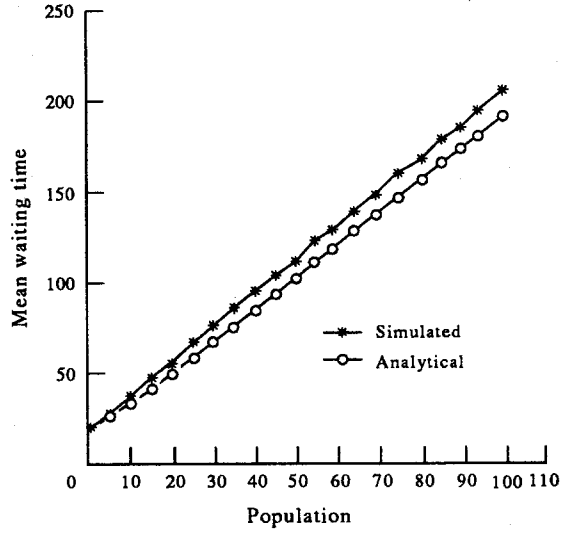

Fig. 9. Mean cycle time under the FBFS policy.

Table 3. Waiting times at individual buffers for the FBFS policy

	$W_{1,14}$		W_{88}		W_{37}	
Population	SIM	MVA	SIM	MVA	SIM	MVA
1	0.13	0.13	0.20	0.20	0.25	0.25
10	0.28	0.29	0.38	0.41	0.53	0.57
20	0.55	0.63	0.67	0.74	1.00	1.17
30	0.89	1.11	0.95	1.09	1.53	1.90
40	1.28	1.68	1.28	1.41	2.22	2.83
50	1.65	2.30	1.59	1.71	3.08	3.75
60	2.03	2.46	1.84	1.96	3.74	4.60
70	2.46	3.03	1.97	2.18	4.92	5.61
80	2.85	4.20	2.28	2.38	5.51	6.50
90	3.32	4.30	2.63	2.55	6.34	7.30
100	3.63	4.50	2.64	2.70	7.13	8.19

Table 4. Waiting times at individual buffers for the LBFS policy

	W_{11}		W_{81}		W_{31}	
Population	SIM	MVA	SIM	MVA	SIM	MVA
1	0.13	0.13	0.19	0.20	0.25	0.25
10	0.29	0.29	0.43	0.41	0.58	0.57
20	0.63	0.64	0.82	0.74	1.24	1.18
30	1.10	1.12	1.22	1.09	2.08	1.96
40	1.66	1.68	1.64	1.41	3.26	2.84
50	2.34	2.31	1.96	1.71	4.10	3.76
60	3.08	2.97	2.20	1.96	5.20	4.69
70	3.85	3.63	2.70	2.18	6.26	5.61
80	4.57	4.30	2.91	2.38	7.59	6.50
90	5.39	4.95	3.10	2.55	8.45	7.37
100	6.31	5.60	3.21	2.70	9.57	8.19

$i(i = 1, \ldots, 12)$ for each visit to that center. The parameters we assume are

$$\frac{1}{\mu_1} = 0.125 \quad \frac{1}{\mu_2} = 0.125 \quad \frac{1}{\mu_3} = 0.250$$

$$\frac{1}{\mu_4} = 1.800 \quad \frac{1}{\mu_5} = 0.900 \quad \frac{1}{\mu_6} = 0.600$$

$$\frac{1}{\mu_7} = 1.800 \quad \frac{1}{\mu_8} = 0.200 \quad \frac{1}{\mu_9} = 0.600$$

$$\frac{1}{\mu_{10}} = 0.333 \quad \frac{1}{\mu_{11}} = 0.600 \quad \frac{1}{\mu_{12}} = 1.250.$$

The above parameters are the same as in [4].

4.1. Cycle time and throughput rate

Figure 8 gives estimates of throughput rates obtained using simulation as well as by our analytical technique for different populations, assuming LBFS policy. Figure 9 gives the simulation and analytical estimates of mean cycle time, assuming FBFS policy. A close agreement between the two estimates is apparent from the figures.

4.2. Cycle times at individual buffers

Table 3 compares the waiting times at some buffers computed by the analytical and the simulation methods, for different system populations. The scheduling policy used is FBFS.

Table 4 is similar to Table 3 except that, here, LBFS scheduling policy is used.

5. CONCLUSIONS

The analytical method, based on MVA approximation, presented in this paper has been found to be efficient and quite accurate in predicting the performance of re-entrant lines. The advantages of the analytical method can be summarized as follows.

1. The method is much faster (by almost three to four orders of magnitude in observed experiments) than detailed simulation.
2. The method yields, as an attractive by-product, the performance measures for all intermediate populations.
3. Many scheduling policies, such as FBFS, LBFS, FCFS, and any fixed priority policy can be easily analyzed using the method.
4. Given the route of the jobs and the structure of the re-entrant line, the recursive equations can be automatically formulated and solved.

Topics for future work include: modeling of due date based policies and fluctuation smoothing policies [1,4] using the MVA approximation and comparison of performance of different scheduling policies using the analytical methodology presented in this paper.

Acknowledgement—This research was supported in part by the Office of Naval Research and the Department of Science and Technology grant N00014-93-1017. We would also like to acknowledge the excellent facilities at the Intelligent Systems Laboratory, Department of Computer Science and Automation, Indian Institute of Science. We also acknowledge several fruitful discussions with Professor P. R. Kumar.

REFERENCES

1. P. R. Kumar, Re-Entrant lines. *Queueing Systems: Theory and Applications* **13,** 87–110 (1993).
2. M. Reiser and S. S. Lavenberg, Mean value analysis of closed multichain queueing networks. *J. ACM* **27,** 313–322 (1980).
3. S. H. Lu and P. R. Kumar, Distributed scheduling based on due dates and buffer priorities. *IEEE Trans. autom control* **36,** 1406–1416.
4. S. H. Lu, Deepa Ramaswamy and P. R. Kumar, Efficient scheduling policies to reduce mean and variance of cycle-time in semiconductor manufacturing plants. *IEEE Trans semiconductor mfg* **7,** 374–388 (1994).
5. C. R. Glassey and M. G. C. Resende, Closed-loop job release control for VLSI circuit manufacturing. *IEEE Trans. semiconductor mfg* **1,** 36–46 (1988).
6. L. M. Wein, Scheduling semiconductor wafer fabrication. *IEEE Trans. semiconductor mfg* **1,** 115–130 (1988).
7. S. X. Bai and S. B. Gershwin, A manufacturing scheduler's perspective on semiconductor fabrication. Technical Report 89–518, MIT Microsystems Research Center, March 1989.
8. N. Srivatsan, S. X. Bai and S. B. Gershwin, Hierarchical real-time integrated scheduling of a semiconductor fabrication facility. Technical report, Laboratory for Manufacturing and Productivity, Massachusetts Institute of Technology, 1992.
9. D. Connors, G. Feigin and D. Yao, A queueing network model for semiconductor manufacturing. *IEEE Trans. semiconductor mfg* (To appear in 1994).
10. P. J. Schweitzer and A. Seidman, Processing rate optimization for FMSs with distinct multiple job visits to work centers. In K. E. Stecke and R. Suri, (eds), *Flexible Manufacturing Systems: Operations Research Models and Applications*, (pp. 79–84). Elsevier, Amsterdam (1989).
11. S. Shalev Oren, A. Seidmann and P. J. Schweitzer, Analysis of flexible manufacturing systems with priority scheduling: PMVA. *Ann. Ops. Res.* **3,** 115–139 (1985).
12. S. Kumar and P. R. Kumar, Performance bounds for queueing networks and performance policies. *IEEE Trans. autom control* **39,** 1600–1611 (1994).
13. R. Suri and R. R. Hildebrant, Modeling flexible manufacturing systems using mean value analysis. *J. mfg sys.* **31,** 27–38 (1984).
14. P. J. Schweitzer, A survey of mean value analysis, its generalizations, and applications for networks of queues. In *Second International Conference on Mathematics for Operations Research*, Amsterdam (1990).
15. Y. Bard, Some extensions to multiclass queueing network analysis. In M. Arato, A. Butrimenko and E. Gelenbe (eds), *Performance of Computer Systems*, (pp. 51–61). North-Holland, Amsterdam (1979).
16. C. S. Ramanjaneyulu and V. V. S. Sarma, Modeling server unreliability in closed queueing networks. *IEEE Trans. reliability* **38,** 90–95 (1989).
17. R. M. Bryant, A. E. Krzesinki and P. Teunissen, The MVA preemptive resume priority approximation. In *Proc. ACM SIGMETICS Conf. on Measurement and Modeling of Computer Systems*, pp. 12–27 (1983).
18. R. M. Bryant, A. E. Krzesinki, M. S. Lakshmi and K. M. Chandy, The MVA priority approximation. *ACM Trans. comput. Syst.* **2,** 335–359 (1984).
19. J. B. M. Doremalen Van, J. Wessels and R. J. Wijbrands, Approximate analysis of priority queueing networks. In O. J. Boxma, J. W. Cohen and H. C. Tijms (eds), *Teletraffic Analysis and Computer Performance Evaluation*, (pp. 117–131). North-Holland, Amsterdam (1986).
20. D. L. Eager and J. N. Lipscomb, The AMVA priority approximation. *Performance Evaluation* **8,** 173–193 (1988).
21. P. J. Schweitzer and A. Seidmann, Optimizing processing rates for flexible manufacturing systems. *Mgmt. Sci.* **37,** 454–466 (1991).
22. R. Suri, J. L. Sanders and M. Kamath, Performance evaluation of production networks. In S. C. Graves, A. G. Rinnoy Kan and P. Zipkin, (eds), *Handbooks in OR and MS*, (Vol. 4, pp. 199–286). Elsevier, Amsterdam (1993).
23. K. M. Chandy and M. S. Lakshmi, An Approximation technique for queueing networks with preemptive priority queues. Technical report, Department of Computer Science, University of Texas, Austin (1983).
24. A. B. Bondi and Y. M. Chung, A new MVA based approximation for closed queueing networks with a preemptive priority server. *Performance Evaluation*, **8,** 195–221 (1988).
25. J. D. C. Little, A proof of the queueing formula $l = \lambda w$. *Ops Res.* **9,** 383–385 (1961).