Re-entrant lines *

P.R. Kumar

University of Illinois, Department of Electrical and Computer Engineering, and Coordinated Science Laboratory, 1308 West Main St., Urbana, IL 61801, USA

Received 1 November 1992; revised 30 June 1992

Traditionally, manufacturing systems have mainly been treated as either job shops or flow shops. In job shops, parts may arrive with random routes, with each route having a low volume. In flow shops, the routes are fixed and acyclic, as in assembly lines. With the advent of semiconductor manufacturing plants, and more recently, thin film lines, this dichotomy needs to be expanded to consider another class of systems, which we call "re-entrant lines". The distinguishing feature of these manufacturing systems is that parts visit some machines more than once at different stages of processing.

Scheduling problems arise because several parts at different stages of processing may be in contention with each other for service at the same machine. There may be uncertainties in the form of random service or set-up times, as well as random machine failures and repairs. The goal of scheduling is to improve performance measures such as mean sojourn time in the system, which is also known as the mean "cycle-time", or the variance of the cycle-time.

In this paper we provide a tutorial account of some recent results in this field. We describe several scheduling policies of interest, and provide some results concerning their stability and performance. Several open problems are suggested.

Keywords: Manufacturing systems, semiconductor manufacturing, thin film lines, re-entrant lines, scheduling policies, queueing networks, buffer priority policies, due date policies, stability, stochastic control, mean delay, variance of delay, machine failures, set-up times.

1. Introduction

In the traditional dichotomy, manufacturing systems are classified as either job shops or flow shops; see Graves [1]. Briefly, job shops are used for low volume production of parts, typically made to order. Each (possibly custom made) part may have a different route (i.e., sequence of processing steps) through the system, and so the part routes may be regarded as "random". In contrast, flow shops (such as assembly lines) are used for high volume dedicated production of a part-type, which is made to stock. The part route is fixed, and acyclic.

* The research reported here has been partially supported by the National Science Foundation under Grant No. ECS-90-25007, and the Joint Services Electronics Program under Contract No. N00014-84-C-0149.

With the advent of semiconductor manufacturing, and more recently, thin film lines, however, this dichotomy needs to be expanded to consider a third class of manufacturing systems, which we shall label as *re-entrant lines*; see fig. 1. This type of system consists of one (or more) part types, each with its own fixed route through the plant. The part flow(s) along the route(s) may be high volume. The characteristic feature of each part flow route is that parts visit some machine (or set of machines) more than once. Thus parts at different stages of their life may be in contention for service at the same machine. This gives rise to problems of machine scheduling.

These re-entrant lines differ from flow shops since the part flow route is re-entrant, i.e., non-acyclic. As mentioned, this "re-entrant" feature gives rise to important scheduling problems. In addition, they differ from job shops since there may be only a single part flow route, which is thus of high volume. Thus, in contrast to job shop scheduling, which is usually regarded as a combinatorial problem, there is more structure to exploit when addressing scheduling problems. Depending on whether the re-entrant line is a full scale production facility or a research and development facility used for prototyping, and on whether the products manufac-

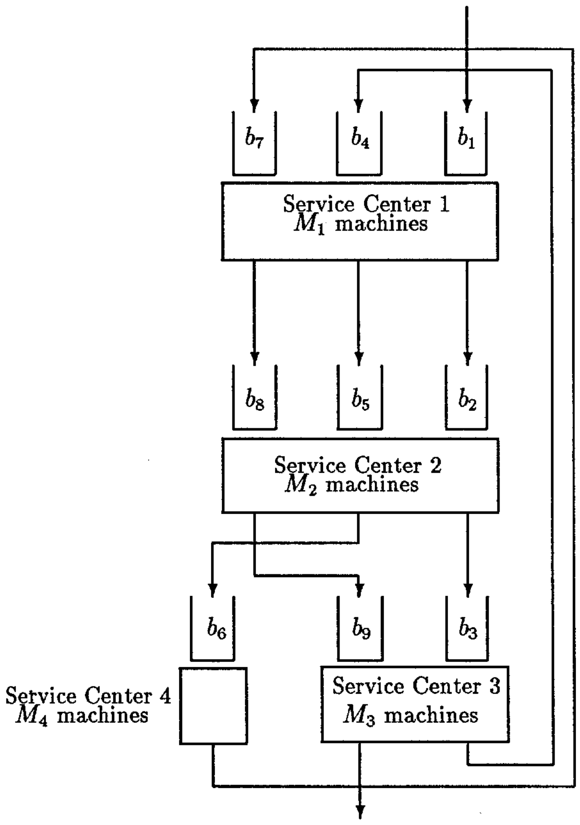

Fig. 1. A re-entrant line.

tured are memory or logic chips, the system may be classified as either a make-to-stock or make-to-order facility.

In the rest of the paper, we provide an account of some recent results in this area. In section 2, we describe a re-entrant line, and in sections 3 and 4 we provide some bounds on attainable mean delay in a stochastic framework, under the assumption that service and inter-arrival times are exponentially distributed. In section 5 we describe a scheduling policy for reducing delay. In sections 7–9 we examine the issue of stability of re-entrant lines, under a deterministic, bursty model of arrivals described in section 6. Section 10 provides some simulation results on the delay experienced by some scheduling policies. In section 11 we examine the problem of minimizing the variance of delay, suggest a scheduling policy, discuss its stability, and provide some simulation results. In section 12 we provide a stochastic control approach to the problem of production control under random machine failures. Finally, in sections 13 and 14 we turn to an examination of scheduling issues if set-up times are incurred whenever processing is switched from one buffer to another.

2. Description of a single re-entrant line

Consider a set $\{1, 2, \ldots, S\}$ of S "service centers" (see fig. 1). Each service center $\sigma \in \{1, 2, \ldots, S\}$ consists of M_σ identical machines. Parts of a certain type enter at a service center $\sigma(1) \in \{1, 2, \ldots, S\}$, where they are stored in buffer b_1. Then they visit service center $\sigma(2)$, where they are stored in buffer b_2, etc. Let buffer b_l at service center $\sigma(l)$ be the last buffer visited. The sequence $\sigma(1), \ldots, \sigma(l)$ is the *route* of the part. We shall allow for the possibility that $\sigma(i) = \sigma(j)$ for some stages $i \neq j$, and accordingly call this type of a system as a *re-entrant line*.

To describe the system more fully, we need to specify the nature of the arrival process(es), the service times at the service centers along the routes, and how parts are to be selected for service at the service centers. Finally, if machines are subject to failure, we need to specify the failure and repair times.

3. BCMP and Kelly networks

Let us assume that the machines do not fail, and

(i) the arrivals are a Poisson process of rate λ,
(ii) the service times at service center σ are all exponentially distributed with mean $1/\mu_\sigma$,
(iii) the scheduling discipline at each service center is first-come-first-serve (FCFS), i.e., a part which arrives first at a service center is served first.

Under these assumptions, the system is a BCMP or "Kelly" network (see [2,3]) and the steady-state distribution has a "product" form. Some properties of the steady-state distribution (taking $M_\sigma \equiv 1$ for simplicity) are,

$$\text{Prob (there are } n \text{ parts at service center } \sigma) = (1-\rho_\sigma)\rho_\sigma^n,$$

where

$$\rho_\sigma := \frac{\lambda V_\sigma}{\mu_\sigma},$$

and

$$V_\sigma := \text{number of visits to service center } \sigma.$$

Also,

$$E[\text{number of parts in buffer } b_j \text{ at service center } \sigma] = \frac{\lambda}{\mu_\sigma - \lambda V_\sigma}. \qquad (1)$$

We have assumed here that the arrival rate λ is within the capacity of the system, i.e., $\rho_\sigma < 1$ for all σ.

(For the more general case where there are many such routes, many machines at each service center, etc., and for more details about the "product-form" steady-state distribution, see [3].)

There are two crucial, restrictive features of BCMP and Kelly networks. First, in assumption (ii) above, all parts at a service center σ are required to have the *same* mean $1/\mu_\sigma$, regardless of the buffer they are in. Second, the scheduling discipline (FCFS above) is *not* allowed to depend on the buffer occupied by the part. Thus, priorities based on the "stage" of production of a part are disallowed.

To illustrate our ignorance when the mean service times are different on revisits, consider the system in fig. 2. When $\mu_1 \neq \mu_3$, even under an FCFS discipline, currently we do not know the steady-state distribution of the system, or even whether there is one. (We conjecture though that the system is stable whenever $\rho_\sigma < 1$ for all σ, and hence that there is in fact a steady-state distribution.) The state of the system consists of a binary valued *sequence* to describe the *order* of arrivals at Service Center 1, and an integer to denote the number at Service Center 2. The

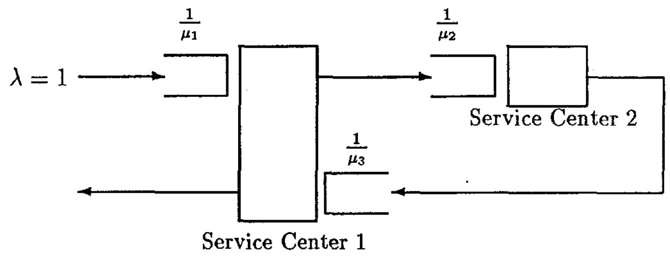

Fig. 2. A system with unequal service time distributions for the buffers at the same service station.

inclusion of the order of arrivals in the state is necessitated by the FCFS nature of the service discipline.

4. Bounds on attainable performance

Can we bound the performance attainable by optimal scheduling? Under assumptions (i) and (ii) of the preceding section, we can indeed obtain a lower bound on the mean "number of remaining stages", as follows.

Let us consider a *stable* scheduling policy, i.e., one for which there exists a steady-state distribution with a finite second moment for number of parts. Let $x_i(t)$ be the number of parts in buffer i at time t, including any in service, and define the *number of remaining stages* $R(t)$ as

$$R(t) = \sum_{i=1}^{l} (l - i + 1)x_i(t) .$$

We shall apply "uniformization" (see [4]) and suppose that an idle machine at service center σ is working on a fictitious customer whose service time is exponentially distributed with mean $1/\mu_\sigma$. Let $\{\tau_n\}$ be the sequence of times at which there is either an arrival, or a real or fictitious service completion, at some machine. We shall denote by $\mathcal{F}_{\tau_n}$ the σ-field generated by events up to time τ_n. Let us scale time so that $\lambda + \sum_{\sigma=1}^{S} \mu_\sigma = 1$.

Then

$$\begin{aligned} E[R^2(\tau_{n+1})|\mathcal{F}_{\tau_n}] =& \lambda[R(\tau_n) + l]^2 + \sum_{\sigma=1}^{S} \mu_\sigma \mathbf{1}(\sigma \text{ is busy at } \tau_n)[R(\tau_n) - 1]^2 \\ & + \sum_{\sigma=1}^{S} \mu_\sigma \mathbf{1}(\sigma \text{ is idle at } \tau_n) R^2(\tau_n) \\ \geqslant & R^2(\tau_n) + 2\lambda l R(\tau_n) + \lambda l^2 \quad (\text{using } \mathbf{1}(\sigma \text{ is busy at } \tau_n) \leqslant 1) \\ & - \sum_{\sigma=1}^{S} 2\mu_\sigma R(\tau_n) + \sum_{\sigma=1}^{S} \mu_\sigma \mathbf{1}(\sigma \text{ is busy at } \tau_n) . \end{aligned}$$

Taking expectations and telescoping,

$$\begin{aligned} \lim_{n\to\infty} E[R(\tau_n)] \geqslant & \frac{\lambda l^2 + \sum_{\sigma=1}^{S} \mu_\sigma \cdot (\lambda V_\sigma/\mu_\sigma)}{\sum_{\sigma=1}^{S} 2\mu_\sigma - 2\lambda l} \quad (\text{using } \rho_\sigma = E(\mathbf{1}(\sigma \text{ is busy at } \tau_n))) \\ = & \frac{\lambda l(l+1)}{2\sum_{\sigma=1}^{S} \mu_\sigma(1 - \rho_\sigma)} \quad (\text{using } l = \sum_{\sigma=1}^{S} V_\sigma) . \end{aligned} \qquad (2)$$

Thus we have obtained a bound on the attainable performance of *any* stable scheduling policy (with a finite second moment). This bound is due to Meyn [5]. (See also Meyn and Down [6], where such a "Lyapunov" function is used to establish the stability of "generalized" Jackson networks when the assumption of exponential distributions is relaxed.)

Let us compare this bound with the known performance of the FCFS discipline. Using (1), for the FCFS discipline we obtain,

$$\lim_{n\to\infty} E[R(\tau_n)] = \sum_{j=1}^{l} \frac{(l-i+1)\lambda}{\mu_\sigma - \lambda V_\sigma}\,.$$

If $\mu_\sigma \equiv \mu$, $V_\sigma \equiv V$ (and thus $\rho_\sigma \equiv \rho$, $SV = l$), we have

$$\lim_{n\to\infty} E[R(\tau_n)] = \frac{\rho S(l+1)}{2(1-\rho)}\,,$$

while the bound (2) is,

$$\lim_{n\to\infty} E[R(\tau_n)] \geqslant \frac{\rho(l+1)}{2(1-\rho)}\,. \tag{3}$$

Thus, we have

$$\frac{E[\text{number of remaining stages under FCFS}]}{E[\text{number of remaining stages under optimal policy}]} \leqslant S\,. \tag{4}$$

Since we do not really expect FCFS to be so bad when S is large, it appears that the bound (2) is rather poor when S is large.

Our real interest centers not on the mean "remaining number of stages of work", but on the *delay* experienced by a part, which is known as *cycle-time* in the semiconductor manufacturing parlance. Since (number of stages of remaining work) $\leqslant l$ (number of parts in system), we obtain from (1), (2) and Little's Theorem that

$$\frac{\text{Mean delay of FCFS}}{\text{Optimal mean delay}} \leqslant \frac{2Sl^2}{(l+1)}\,,$$

which is also rather gross when the number of stages or service centers is large.

An alternative bound, which is useful at light loading can be obtained as follows. Let us *dedicate* a server of rate μ_σ to *each* buffer. Then there is no conflict between buffers. Moreover, since all parts at a buffer are essentially identical, and since we have a tandem queueing system, any non-idling (also misleadingly called work conserving) policy, and in particular, FCFS, is optimal. Using the known result for tandem queues (a special case of Kelly networks), we obtain the bound,

$$\lim_{t\to\infty} E\left[\sum_{i=1}^{l} x_i(t)\right] \geqslant \sum_{\sigma=1}^{S}\sum_{i=1}^{V_\sigma} \frac{\lambda}{\mu_\sigma - \lambda}\,.$$

If $\mu_\sigma \equiv \mu$, $V_\sigma \equiv V$, we obtain

$$\lim_{t \to \infty} E\left[\sum_{i=1}^{l} x_i(t)\right] \geqslant \frac{l\rho}{V-\rho}. \tag{5}$$

Thus, combining bounds (5) and (4), we obtain

$$\frac{\text{Mean delay of FCFS}}{\text{Optimal mean delay}} \leqslant \min\left[\frac{2Sl^2}{(l+1)}, \frac{V-\rho}{V(1-\rho)}\right]$$
$$\leqslant \min\left[\frac{2Sl^2}{(l+1)}, \frac{1}{1-\rho}\right].$$

Clearly, more work is needed in the area of performance quantification!

5. Reducing the mean delay: The last buffer first serve policy

Let us consider two possible goals of scheduling a manufacturing system. First, one may want to reduce the *mean* cycle-time, i.e., the mean sojourn time or delay of a part in the system. Second, one may want to reduce the *variance* of the cycle-time, which is useful for reliably meeting due dates. In this section, we will focus on the goal of constructing a scheduling policy to reduce the mean cycle-time.

EXAMPLE 5.1

Consider the simple motivatory system of fig. 3. There is a single machine, and each part visits it l times in succession. Let us suppose that arrivals are a Poisson process of rate λ, and service times at buffer i are exponentially distributed with mean $1/\mu_i$. By Little's Theorem, (mean delay) $= 1/\lambda$(mean number in system).

Taking our cue from Little's Theorem, which prescribes that in order to minimize delay one must minimize the mean number of parts in the system, it is clear

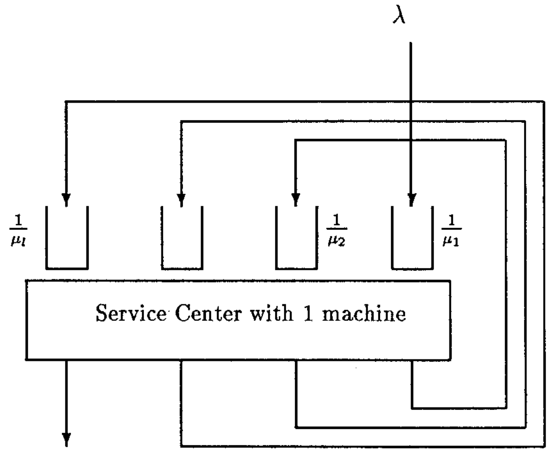

Fig. 3. System of example 5.1 where LBFS is optimal for mean delay.

that the optimal policy for the machine is to take up parts from buffer b_1 in the order of their arrival to b_1 (and hence in the order of their arrival to the *system*), and then continuously work on a part, through all its stages, until it exits the system. (This is provable using a sample path dominance argument.)

Under such a policy, the system is also equivalent to an $M/G/1$ system, where all the service times at the buffers are aggregated into a single service time which is a sum of exponentially distributed random variables, and operating under an FCFS discipline. Hence, by the Pollaczek–Khinchine formula, the mean delay in the system under this priority scheme is

$$\frac{2\rho - \rho^2 + \lambda^2[1/\mu_1^2 + \ldots + 1/\mu_l^2]}{2(1-\rho)},$$

where $\rho = \sum_{i=1}^{l} \lambda/\mu_i$.

The above policy can also be described as one where the part taken up next for processing by the machine is the one with the shortest expected remaining processing time, counting *all* its stages, the so called SERPT policy. Yet another description of the optimal policy is as follows. Order the buffers $b_l, b_{l-1}, \ldots, b_1$ in the *reverse* of the order that they are visited, and give priority to the buffers in this order. (Thus, a part in buffer b_3 should be taken up for processing only if buffers $b_l, \ldots, b_4$ are empty.) Moreover, parts are taken from the head of a buffer only. □

The optimality of the "shortest expected processing time" policy for minimizing the mean flow time of a set of fixed jobs under a non-preemptive discipline using a single machine or several parallel machines, has been established under an assumption of stochastic ordering of the service times; see [7,8]. Such optimal "time-sharing" results are also available for queueing networks; see Klimov [8,9] and Lai and Ying [10].

Taking our cue from such considerations, let us consider a policy for a re-entrant line which orders the buffers $b_l, \ldots, b_1$, and gives non-preemptive priority in that order. We shall refer to this as the *Last Buffer First Serve Policy* (LBFS). Since such a policy attempts to clear parts from the system, it is a good candidate for reducing the mean delay, particularly at high load factors. However, in evaluating any policy one must also consider whether it induces excessive forced machine starvations. In section 10 we provide the results of some simulations.

6. A deterministic model

We would like to analyze policies such as LBFS, and others. The available results cover only relatively simple systems, as in Simon [11]. Hence, in view of the difficulties faced by a probabilistic model of arrivals and services, let us adopt a deterministic model.

Let us suppose that in every time interval $[s, t]$,

$$\text{Number of arrivals in } [s,t] \leqslant \lambda(t-s)+\gamma \quad \text{for all } 0 \leqslant s \leqslant t < \infty . \tag{6}$$

We shall refer to λ as the "arrival rate" (though it is only an upper bound). The constant γ allows for some burstiness; see Cruz [12].

Regarding the service times, let us suppose that a part in buffer b_i requires τ_i units of service from one of the machines in service center $\sigma(i)$, and that there are M_σ identical machines at service center σ.

We shall also suppose that

$$\rho_\sigma := \sum_{\{i|\sigma(i)=\sigma\}} \frac{\lambda \tau_i}{M_\sigma} < 1 \quad \text{for all } \sigma \in \{1,2,\ldots,S\},$$

i.e., that the arrival rate λ is within the *capacity* of the system.

7. An unstable buffer priority policy

The last-buffer-first-serve (LBFS) policy is but one example of a scheduling policy based on ordering the buffers according to a priority ordering, and giving non-preemptive priority in that order (always choosing parts from the head of a buffer if there is more than one part in the buffer).

However, some buffer priority policies can be unstable, as shown in the following example drawn from Lu and Kumar [13].

EXAMPLE 7.1

Consider the system shown in fig. 4. Let us suppose that parts arrive periodically at rate $\lambda = 1$, at times 0, 1, 2, 3, ... to the system. Suppose that $\tau_1 = \tau_3 = 0$, and $\tau_2 = \tau_4 = 2/3$. Each service center consists of only one machine, i.e., $M_1 = M_2 = 1$. Even though τ_1 and τ_3 are zero, parts in buffers b_1 and b_2 require attention (of 0 seconds!) from the machines at service centers 1 and 2, before they can proceed to buffers b_2 and b_4 respectively. Consider the buffer priority policy employing the buffer priority ordering b_4, b_1, b_2, b_3.

Suppose that at time 0^-, there are x parts in buffer b_1, and 0 parts elsewhere in the system. At time 0^-, these x parts are transferred to buffer b_2. Service Center 2

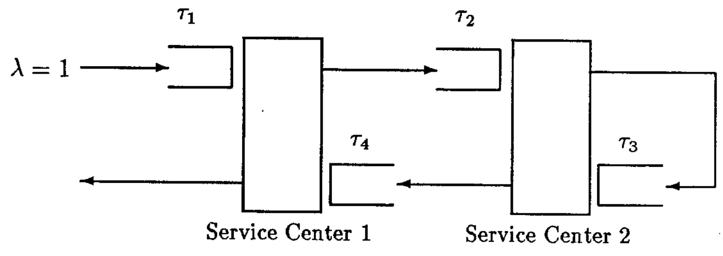

Fig. 4. System of example 7.1 which is unstable for a buffer priority policy.

commences a busy period which is completed at time T^-, satisfying $(T+x)2/3 = T$, i.e., $T = 2x$. At time T^-, the $3x$ parts in buffer b_3 are transferred to b_4. The machine at Service Center 2 then goes into a busy period lasting $2x$ time units, during which $2x$ parts have arrived into b_1. Hence the new state of the system at time $4x^-$ is $2x$ parts at buffer b_1, and 0 parts elsewhere.

Thus, in one cycle of operations described above, the number of parts in the system has doubled from x to $2x$. This process repeats itself, showing that the mean number of parts in the system, as well as mean delay, are unbounded. □

For an example of an unstable system with $\tau_i \neq 0$, see Tang and Shi [14].

8. Stability of the last-buffer-first-serve policy

It is a non-trivial fact that the LBFS policy is stable. The following contractive property of LBFS is proved in Lu and Kumar [13].

THEOREM 8.1: CONTRACTIVE PROPERTY OF LBFS

Let $w_\sigma := \sum_{\{i|b_i \text{ is served by service center } \sigma\}} \tau_i/M_\sigma$ be the work, in time units, brought in per incoming part to a machine at service center σ, and let $\bar{w} := \max_\sigma w_\sigma$, be the corresponding quantity for the "bottleneck" machine. Then, for every $\epsilon > 0$, there exists a constant $c(\epsilon)$ such that, if there are x parts in the system, however located, when a part arrives at the system, then its delay in the system satisfies,

$$\text{Delay} \leqslant c(\epsilon) + (\bar{w} + \epsilon)x \quad \text{for all } \epsilon > 0 .$$

The proof of this assertion can be found in [13]. The essential property on which it hinges is that there is only "unilateral temporal interference" by parts. To see this, suppose, for simplicity only, that the buffer priority discipline is pre-emptive. Consider a part arriving into buffer b_5 (say) at a service center σ which also serves buffers b_2, b_3, b_7 and b_{10}. Then by the pre-emptive priority, the part in b_5 is *not* subjected to delay by having to wait for parts in buffers b_2 and b_3. Its only delay is caused by parts already in b_7 and b_{10}, or parts which arrive into b_7 and b_{10} while it is waiting in b_5. Since all such parts must have arrived into the system *prior* to its own arrival into the system, we see that *delay is caused only by parts which arrived earlier*, not later. This is what we mean by *unilateral temporal interference*.

Utilizing theorem 8.1, one may establish the following stability and performance upper bound for the delay of an LBFS policy.

THEOREM 8.2

(i) Under the LBFS policy, the delays of all parts are bounded.

(ii) If $x(0)$ is the number of parts in the system at time 0, then the number of parts $x(t)$ at time t is bounded by,

$$x(t) \leqslant \max\left\{(1+\rho+\lambda\epsilon)x(0)+\lambda c(\epsilon)+\gamma, \frac{2(\lambda c(\epsilon)+\gamma)}{1-\rho-\lambda\epsilon}\right\}$$

for all $t \geqslant 0$, for every $\epsilon > 0$ with $1-\rho-\lambda\epsilon>0$, where $\rho := \max_\sigma \rho_\sigma$.

(iii) The asymptotic number of parts in the system is bounded by,

$$\limsup_{t\to\infty} x(t) \leqslant \frac{\lambda c(\epsilon)+\gamma}{1-\rho-\lambda\epsilon} \quad \text{for all } \epsilon>0 \text{ with } 1-\rho-\lambda\epsilon>0\,.$$

Proof

See [13]. □

The above theorem provides a bound on *transient* behavior in (ii), and a bound on asymptotic behavior in (iii). As one would expect, the asymptotic performance bound is independent of the initial number $x(0)$ in the system.

In contrast to the LBFS policy, parts moving under an FCFS scheduling policy are subject to *bilateral* temporal interference by parts which arrive both before as well as after it. We have been unable to establish whether FCFS is stable – a significant open question, and one which is well worth resolving. We conjecture that FCFS is stable.

9. The first-buffer-first-serve policy

Let us consider another buffer priority policy, the first-buffer-first-serve (FBFS) policy, which is the diametric opposite of the LBFS policy. It supplies non-preemptive service in the priority order $b_1, b_2, \ldots, b_l$, i.e., in the same order that buffers are visited.

The stability of FBFS is simple to establish; see [13].

THEOREM 9.1

The FBFS policy is stable.

Outline of proof

For ease of exposition, suppose that the service discipline is preemptive. Then parts in buffer b_1 have highest priority, and the other buffers in the system are transparent to it. Since b_1 is also the entry point for parts, and their arrivals are nearly periodic as in (6), it follows that the level of b_1 is bounded. Thus the departures from b_1 are also nearly periodic. These however constitute the arrivals into b_2. One can repeat this argument, until one arrives at a buffer which is subject to interference by buffers earlier in the route. At this point, one may regard the servers (machines) as having allocated a dedicated fraction of their times to the earlier buffers (since their arrivals as well as departures are nearly periodic). Thus the parts in the later buffers have their services met by the remaining capacity at the machines. In this way, by induction on the buffers, in the order $b_1, \ldots, b_l$, one can prove that every buffer level in the system is bounded. □

It is worth noting that under the FBFS policy also the parts suffer only from unilateral temporal interference. However, unlike LBFS, the interference is due to parts which arrived *later* rather than earlier.

For a treatment of systems consisting of many part-types, with different re-entrant routes, we refer the reader to [13], which provides several stable buffer priority policies.

10. Some simulation results

While theorem 8.2(iii) provides a bound on delay, it is conservative. To obtain a better idea of performance of LBFS, simulations have been conducted in [13] on the system shown in fig. 5.

Ninety random systems were constructed. To construct each system, the deterministic service times at each buffer were randomly generated, and then normalized so that $\rho_\sigma = 0.8$ for all σ in thirty of the systems, and equal to 0.9 and 0.999 respectively, in the other two sets of thirty systems each. Thus each system had deterministic processing times (which were randomly chosen). The arrivals in all cases were a Poisson process. For each system, one simulation run was performed to compare the four policies, LBFS, FBFS, FCFS, and another policy LS to be introduced in section 11. The mean delays of all 30 simulations runs (one for each system) as well as the mean rank (among the four contending policies) are shown in table 1. At high loading ($\rho = 0.999$) LBFS was best in all thirty systems, LS was always second, while FCFS and FBFS were always third and fourth respectively.

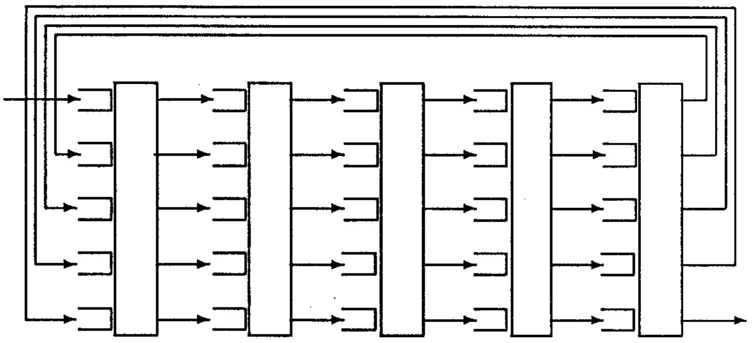

Fig. 5. Simulated system.

Table 1
Simulation results for mean delay and rank of mean delay on system of fig. 5.

ρ		FBFS[a]	FCFS	LBFS	LS
0.8	Average of 30 means	33.63	11.32	10.06	10.52
	Average rank of mean	4.00	2.80	1.40	1.80
0.9	Average of 30 means	164.04	31.54	22.42	26.56
	Average rank of mean	4.00	2.90	1.20	1.90
0.999	Average of 30 means	491.65	83.62	49.32	65.43
	Average rank of mean	4.00	3.00	1.00	2.00

[a] Due to an extremely large build-up of queue size, only 14 sets of values were recorded at the load factor 0.999.

This lends credibility to our intuition that LBFS is a good policy for reducing mean delay at high load factors, in two ways. First, it was always best among the contending policies, at $\rho = 0.999$. Second, the same intuition also suggests that FBFS should be a bad policy vis-a-vis mean delay, and it was indeed the worst in all 90 simulations.

11. Reducing the variance of the delay: The least slack policy

Reduction of the *variance* of the cycle-time is also often described as an important goal in semiconductor manufacturing. A low standard deviation of delay allows one to plan releases into the system to accurately meet due dates. Moreover, if a certain end product requires many separate parts of different types for its final assembly, then an accurate prediction of the delay allows a tight coordination of the production completion times of the various part-types, enabling on-time assembly.

Let us suppose that each part π entering into the system has a due date $\delta(\pi)$ stamped on it. Also suppose that to each buffer b_i we associate a number ζ_i. Let us define the *slack* $s(\pi)$ of a part π in buffer b_i as,

$$s(\pi) := \delta(\pi) - \zeta_i .$$

Consider the case where the number ζ_i is chosen to be equal to a (maybe empirical and rough) estimate of the *mean remaining delay till completion* for parts in buffer b_i. Then, clearly a part π' with $s(\pi') < s(\pi)$ can be regarded as having fallen further behind (or less ahead!) its schedule than π. Thus if the goal is to ensure that the lateness (defined as exit time $e(\pi)$ minus due date $\delta(\pi)$) of all parts is the same, then π' should be accorded higher priority over π. This suggests the following *Least Slack* (LS) policy; give highest priority to a part with the smallest slack $s(\pi)$. It is

Table 2
Simulation results for variance and rank of variance on system of fig. 5.

ρ		FBFS[a]	FCFS	LBFS	LS
0.8	Average of 30 Std. deviations	14.16	4.19	3.44	3.01
	Average rank of Std. deviation	4.00	2.87	2.00	1.13
0.9	Average of 30 Std. deviations	18.29	7.64	5.52	4.76
	Average rank of Std. deviation	4.00	3.00	1.97	1.03
0.999	Average of 30 Std. deviations	15.22	4.04	4.41	3.35
	Average rank of Std. deviation	4.00	2.17	2.70	1.13

[a] Due to an extremely large build-up of queue size, only 14 sets of values were recorded at the load factor 0.999.

clear that such a policy will tend to reduce the variance of the lateness, since it tends to "equalize" the lateness of all parts.

If we instead choose $\delta(\pi)$ = *arrival time* of a part, rather than due date, then it is clear that the least-slack policy will tend to reduce the variance of *delay* in the system.

However, we need to resolve a "circular" feature of the mutual dependence between estimates $\{\zeta_i\}$ of delay and the policy LS. The policy LS is specified by the numbers $\{\zeta_i\}$, but the numbers $\{\zeta_i\}$ are estimates of the remaining delays which are determined by the policy. In [13] this issue is resolved as follows. First all $\{\zeta_i\}$ are set to zero, and the LS policy with $\zeta_i \equiv 0$ is implemented. A simulation of this LS policy is then run, and new estimates $\{\zeta_i\}$ of remaining delays are obtained. These are then used to refine the policy LS, and so on. This iterative procedure was used to obtain a policy LS whose empirically observed delays were close to the numbers $\{\zeta_i\}$ used in defining the policy itself. Such an iterative procedure is also used in Vepsalainen and Morton [15].

The simulation results, from [13], on the system of fig. 5 are reported in table 2. They indicate the effectiveness of LS in reducing the variances of lateness or delay. We should note that in our simulations above, the systems were balanced in that $\rho_\sigma \equiv \rho$ at all service centers. Also, no machine failures were considered.

The numbers ζ_i, which should more properly be called the "parameters" describing the LS policy, could also be chosen on some other basis. For example, choosing $\zeta_i \equiv 0$ results in $s(\pi) \equiv \delta(\pi)$, and so the LS policy then coincides with the well known Earliest Due Date Policy. Such "slack" based policies are often mentioned in the literature; see Baker [16], for example. In fact, if jobs arrive to the system in the order of their due dates, then it is worth noting that the LBFS policy is itself a special case of the LS policy which is obtained by setting all ζ_i to 0. This property can be used to generalize the proof of stability of LBFS to LS; see [13].

THEOREM 11.1
The least-slack policy is stable for every choice of parameters $\{\zeta_i\}$.

Noting, as above, that both the LBFS policy for reducing mean delay, and the LS policy for reducing standard derivation, are of the same form, with merely different choices for the ζ_i's, one may consider a "convex" combination of these two policies for reducing (Mean +3 Std. Devns) of delay. It is obtained by setting

$$\zeta_i = \alpha_i[\text{mean remaining delay from } b_i]$$

for some $0 < \alpha_i < 1$.

A more comprehensive set of simulations has been performed on models of semiconductor manufacturing plants in Lu and Kumar [17]. They indicate that low mean delay and standard deviation can be realized by "optimizing" over the parameters ζ_i; see the "Least Optimized Slack" policy in [17].

Recently, using a Brownian motion approximation of a network, Wein [18,19] has determined policies which attempt to reduce the mean and variance of delay.

12. Machine failures: A stochastic control approach

Machine failures (and maintenance, recalibration, or other down times) are a significant source of uncertainty in semiconductor manufacturing systems. We will now illustrate some structural features of an optimal solution suggested by a stochastic control formulation.

Let us adopt a "fluid" viewpoint, and suppose that when service center σ works at maximum rate on a buffer b_i, then μ_i units of parts *flow* out of buffer b_i and into buffer b_{i+1}, (or exit if $b_i = b_l$ is the last buffer) per unit time. Let $u_i(t)$ be the fraction of service center σ's service capacity which is provided to buffer b_i at time t. Clearly

$$u_i(t) \geqslant 0, \quad \text{and} \quad \sum_{\{i | b_i \text{ is served by } \sigma\}} u_i(t) \leqslant 1 \quad \text{for all } \sigma, \tag{7}$$

if service center σ is working, and

$$u_i(t) = 0, \quad \text{for all } i \text{ such that } b_i \text{ is served by } \sigma, \tag{8}$$

if service center σ has failed. (For simplicity let us suppose that service center σ consists of just one machine.)

Let $f_\sigma(t) = 0$ or 1 be a variable to denote whether service center σ has failed or is working at time t, and let $f(t) = (f_1(t), \dots, f_S(t))$ be the *failure status* of the entire system. From (7) and (8) it follows that for every system status vector f, there is a feasible set $U(f)$ of vectors $u = (u_1, \dots, u_l)^{\mathrm{T}}$. It should be noted from (7, 8) that each $U(f)$ is determined by *linear constraints*.

Thus one obtains the *constraint*,

$$u(t) \in U(f(t)) . \tag{9}$$

Let us suppose (for simplicity) that each $f_i(t)$ is an independent Markov chain with transition rate q_{i1} from 0 to 1, and q_{i0} from 1 to 0. Thus the constraint set $U(f(t))$ changes randomly with time as a Markov chain.

Now let us examine the part flows and the buffer levels. Let $x(t) = (x_2(t), \dots, x_l(t), x_{l+1}(t))$ be the levels of buffers $b_2, \dots, b_l, b_{l+1}$ at time t. We omit b_1 from consideration, regarding it as an infinite source of raw material instead. Thus, solving the stochastic control problem to be formulated shortly will also provide an optimal policy for releasing parts into the system, i.e., an optimal *release policy*. However, we do include an output buffer b_{l+1} to denote the *net inventory level* of finished parts. For this purpose, we suppose that there is a net demand rate λ for parts, which constitutes a constant *drain* on b_{l+1}, which is however sporadically replenished from b_l. We allow x_{l+1} to be positive or negative, the latter corresponding to a backlog, while

$$x_i(t) \geqslant 0 \quad \text{for } i = 2, \dots, l \tag{10}$$

for the remaining physical buffers. Thus we have the *state constraint* (10) in addition to the control constraint (9).

Clearly the evolution of the state of the buffer levels $x(t)$ is governed by a state equation of the form

$$\dot{x}(t) = Bu(t) - b\lambda \tag{11}$$

for an appropriate matrix B, and $b = [0, 0, \dots, 0, 1]^{\mathrm{T}}$. In fact the ith row of B is $[0, \dots, 0, \mu_{i-1}, -\mu_i, 0, \dots, 0]$.

To complete the formulation as a stochastic control problem, let us specify a *cost rate*

$$c(x(t)) = \sum_{i=2}^{l} c_i x_i(t) + c_{l+1}^{+} x_{l+1}^{+}(t) + c_{l+1}^{-} x_{l+1}^{-}(t) .$$

Thus c_i for $2 \leqslant i \leqslant l$ is the cost of maintaining one unit of work-in-process at b_i for one unit of time. The interpretation of c_{l+1}^{-} is as a *backlogging cost* for a finished part per unit of time.

The Hamilton–Jacobi–Bellman equation for the infinite horizon discounted cost function

$$\int_0^{+\infty} E[c(x(t))] \mathrm{e}^{-\alpha t} \mathrm{d}t , \quad \text{where } \alpha > 0 ,$$

is,

$$\begin{aligned}\min_{\substack{u\in U(f)\\ u_i=0 \text{ if } x_i=0}} u^{\mathrm{T}}B^{\mathrm{T}}\nabla_x V(x,f) &= \alpha V(x,f) + \sum_{\{\sigma|f_\sigma=1\}} q_{\sigma 0}[V(x,f) - V(x,f-e_\sigma)] \\ &\quad + \sum_{\{\sigma|f_\sigma=0\}} q_{\sigma 1}[V(x,f) - V(x,f+e_\sigma)] \\ &\quad - c(x) + \lambda^{\mathrm{T}} e_{l+1}^{\mathrm{T}} \nabla_x V(x,f)\,, \end{aligned} \tag{12}$$

where $e_n := (0,\ldots,0,1,0,\ldots,0,1)^{\mathrm{T}}$. The feature of interest here is the left hand side in (12). For fixed (x,f), it is a *linear programming problem*, and so an optimal solution is obtained at an extreme point of the constraint set. At each extreme point, either $u_i = 0$ or 1, and thus all service attention is dedicated to just one buffer.

However, as t increases, even if $f(t)$ does not change, $x(t)$ does change, and so the optimal $u(t)$ can switch from one extreme point to another. Thus, for each fixed f, we may divide $\mathbb{R}^l$ (actually restricted to $x_i \geqslant 0$ for $i = 2,\ldots,l$) into regions, where in each region one extreme point is optimal.

At the boundary between regions, more than one extreme point may be optimal, and so an entire edge or face may be optimal. This corresponds to $0 < u_i < 1$ for some i. If the vector fields $Bu - b\lambda$ of (11) in adjoining regions are outward directed at a boundary, then a "chattering" phenomenon takes place. This corresponds to choosing an optimal u_i at the boundary which is *not* an extreme point, and the trajectory of $x(t)$ then follows the boundary.

In this way, as successive boundaries are met, the trajectory of $x(t)$ may be successively restricted to lower dimensional surfaces, and there may even be a point $x = z^*$ at which $\dot{x}(t) = 0$. This corresponds to an *optimal buffer state* z^* where the system simply produces enough to match demand.

Kimemia and Gershwin [20], in a pioneering work, considered a system of the form (11) with $B = I$ and $b = (\lambda_1, \lambda_2, \ldots, \lambda_P)^{\mathrm{T}}$. This corresponds to a system producing a variety of part-types, but without taking into account the internal dynamics inside the system, i.e., neglecting the flows between buffers. They have determined suboptimal control laws.

A much simpler scalar system has been fully solved in Akella and Kumar [21] and Bielecki and Kumar [22]. This is a system consisting of just one machine producing a single part-type at rate $u, 0 \leqslant u \leqslant \mu$ when up, and at rate $u = 0$ when down; see fig. 6. The constant demand rate is λ. The machine state is described by a Markov chain with failure and repair rates q_0 and q_1. This is modeled as a system

$$\begin{aligned} \dot{x}(t) &= u(t) - \lambda \\ u(t) &\in [0,\mu] \quad \text{if } f(t) = 1\,, \\ &= 0 \qquad\ \ \text{if } f(t) = 0\,, \end{aligned}$$

where $f(t)$ is a Markov chain with state transition matrix

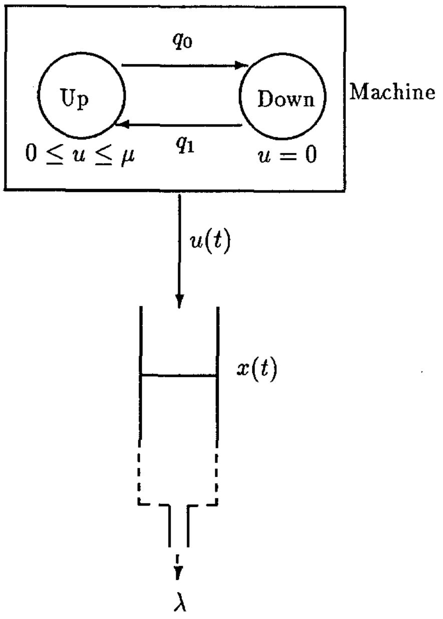

Fig. 6. A failure prone system.

$$\begin{bmatrix} -q_1 & q_1 \\ q_0 & -q_0 \end{bmatrix}.$$

Let us consider the average cost function,

$$\lim_{T\to\infty} \frac{1}{T} E \int_0^T [c^+ x^+(t) + c^- x^-(t)]\, \mathrm{d}t.$$

(The discounted cost version is treated in [21].)

The optimal control law is the following. There is an *optimal buffer level* z^*,

$$z^* = \max\left\{0, \frac{1}{\left(\frac{q_1}{\lambda} - \frac{q_0}{\mu - \lambda}\right)} \log\left[\frac{\lambda q_0 (c^+ + c^-)}{c^+(\lambda - \mu)(q_0 + q_1)}\right]\right\}, \tag{13}$$

and the production rate u should always be chosen to reach the inventory level z^* as quickly as possible, and maintain it there. Thus

$$\begin{aligned} u(t) &= 1 \quad \text{if } x(t) < z^* \text{ and } I(t) = 1, \\ &= \lambda \quad \text{if } x(t) = z^* \text{ and } I(t) = 1, \\ &= 0 \quad \text{otherwise}. \end{aligned}$$

The reader may wonder why the *optimal buffer level* z^* in (13) may be zero. This has its roots in three facts. First, the steady-state distribution of $x(t)$ has a mass at

z^*, and an exponentially decaying density for $x < z^*$. Second, changing z^* to some other z simply translates this steady-state distribution. Last, the cost function is of an absolute value type. Due to these facts, the optimal level z^* should be so chosen that the $(100c^+/(c^+ + c^-))$th percentile of the steady-state distribution (rather than the mean) is at the origin, see [22] for more details.

For a generalization to many machine states, see Sharifnia [23]. Lehoczky et al. [24] have carried out an asymptotic analysis of systems as above when the failure and repair rates are large. Also Mitra [25] has analyzed the steady-state distribution of a system with many "production" and "consumption" machines. Chen and Yao [26] have addressed the issue of accuracy of such fluid model approximations.

For flow shops, there has been much work over the past three decades on the problem of analyzing systems with machine failures. Recently, Lim et al. [27] have generalized the early work of Sevastyanov [28] and obtained asymptotic approximations of the throughput of such systems with finite buffers (also see the references in [27] for earlier work on this problem). They have also determined procedures to allocate the buffer sizes efficiently; see [29].

For more general systems, little appears to be known regarding buffer level control policies, and this is clearly an area where more work is needed.

Some heuristics for avoiding starvation at bottleneck machines in a semiconductor manufacturing plant, by maintaining healthy buffer levels of parts, are described in Lozinski and Glassey [30].

13. Set-up times

A more careful analysis of manufacturing systems should address the issue of set-up times, which was absent from our earlier discussion.

Below we shall indicate some preliminary work in this area. Let us begin with the single machine system shown in fig. 7. There are P types of parts produced by a machine. Let us suppose that a delay δ is incurred whenever the production of parts

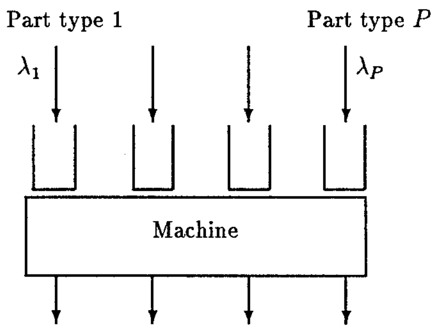

Fig. 7. A single machine with set-up times.

is switched from one buffer to another. We shall adopt a fluid model and suppose that the arrival rate of parts of type p is λ_p, and the service rate is μ_p. We assume that

$$\rho := \sum_{p=1}^{P} \frac{\lambda_p}{\mu_p} < 1 . \tag{14}$$

Let us denote $\rho_p := \lambda_p / \mu_p$.

Clearly, on the average there is only a fraction $(1 - \rho)$ of the total time that can be consumed by set-ups, and so on the average the time between set-ups should be no less than $\delta/(1 - \rho)$. Thus, set-ups cannot be too frequent. On the other hand, if set-ups are too infrequent, then the production runs for each part-type are large, leading to excessive inventories; see Gershwin [31]. Thus we are faced with a problem reminiscent of the dynamic "lot-sizing" problem, see Elmaghraby [32].

In an attempt to increase production run lengths, let us consider the class of "clearing" policies (also called "exhaustive service" policies). A *clearing policy* is one which continues to work on a buffer until it is emptied. In order to fully specify such policies, one only needs to specify how the next part-type to work on is to be chosen. Let $\{\tau_n\}$ be the sequence of times at which production runs end, and let p_n^* be the part-type whose production is commenced at time $(\tau_n + \delta)$, after the set-up. We shall say that a scheduling policy is a *Clear-A-Fraction* (CAF) policy if there exists an $\epsilon > 0$, such that

$$x_{p_n^*}(\tau_n) \geqslant \epsilon \sum_{p=1}^{P} x_p(\tau_n) \quad \text{for all } n ,$$

where $x_p(t)$ denotes the buffer level of part-type p at time t.

If a CAF policy is used, it is clear that whenever the total number of parts $\sum_{p=1}^{P} x_p(\tau_n)$ is large, then the next production run is guaranteed to be long, thus reducing the *work* at the machine. Utilizing this elementary fact, the following result is proved in Perkins and Kumar [33].

THEOREM: STABILITY OF CLEAR-A-FRACTION POLICIES

(i) Every CAF policy is stable, i.e.,

$$\sup_t \sum_{p=1}^{P} x_p(t) < +\infty .$$

(ii) In particular,

$$\limsup_{t \to \infty} \sum_{p=1}^{P} x_p(t) \leqslant \delta \rho \mu_{\max} + \frac{\delta \mu_{\max}}{(1 - \rho)\mu_{\min}} \max_p \mu_p(\rho - \rho_p) .$$

(iii) If $\sum_{p=1}^{P} x_p(0) \leqslant \delta/(\epsilon(1-\rho)) \max_p \mu_p(\rho - \rho_p)$, then the bound in the right hand side in (ii) holds for all $t \geqslant 0$.

Clearly it is of interest to minimize the mean buffer level while maintaining the same throughput rate. It is shown in [33] that for any scheduling policy which is non-idling (or which works at full rate whenever it is not being set up),

$$\lim_{T\to\infty} \frac{1}{T} \int_0^T \sum_{p=1}^{P} c_p x_p(t)\, \mathrm{d}t \geqslant \frac{\delta\left[\sum_{p=1}^{P} \sqrt{c_p \tau_p^{-1} \rho_p (1-\rho_p)}\right]^2}{2(1-\rho_p)} . \tag{15}$$

This provides a bound on attainable performance.

Moreover, what is of interest is that one can construct policies which come very close to the performance (15), in simulations. Specifically, let

$$a_\rho = \lambda_p \sqrt{c_p \tau_p^{-1} \rho_p (1-\rho_p)^{-1}}\,, \quad b_p := \delta \lambda_p^{-1} a_p\,,$$

(these constants are motivated by the analysis leading to the lower bound (15)), then the policy which chooses p_n^* to minimize $a_p x_p(\tau_n) + b_p$ (it is easy to see that it is nearly a CAF policy, and hence stable), yields performance which is very close to (15).

An improvement of the lower bound in (15), when idling or working at less than full rate is allowed, as well as an extension of the policy above, are provided in Chase and Ramadge [34].

14. Set-up times: Re-entrant lines

Since CAF policies yield attractive results for single machine scheduling, let us analyze them for their performance in systems of re-entrant lines. Specifically, let us suppose that each machine in a re-entrant line adopts a CAF policy in deciding which buffer to provide service to next. Such a policy has two attractive features. First, it can be implemented in a *distributed* fashion without any sharing of information or coordination of action between machines. Second, it is easy to implement, requiring little computation.

However, they can be unstable as the following example from Kumar and Seidman [35] shows; see also Chase [36].

EXAMPLE 14.1

Consider a system such as in fig. 4 with $\lambda = 1$, and $1/\mu_2 + 1/\mu_4 > 1$ even though the capacity condition (14) is satisfied. We allow $\delta = 0$ (no set-up times) or $\delta > 0$. Consider an initial state $x_1(0) = \xi > 0$, and $x_i(0) = 0$ for $i = 2, 3, 4$, and with service center 1 set-up for b_4, and service center 2 set-up for b_3 at time 0. Then it can be shown that for all large enough ξ there exists a time T at which a similar state reoccurs except that $x_1(T) = \lambda\xi + b$, where $\lambda := \mu_2/(\mu_2\mu_4 - \mu_4) > 1$, and $b > 0$. Thus a similar cycle of operations again repeats itself, leading to unbounded buffer levels. □

It is worth noting that, as in example 7.1, the instability is caused by the re-entrant nature of the line itself rather than the set-up times; in fact, instability can occur even with $\delta = 0$.

This instability is caused by the "positive feedback" effect induced by *two way material flow* between machines in a re-entrant line. In fact, as shown in [35], instability also occurs for systems such as in fig. 8 where the two way material flow is caused by *different part-types* rather than the same part-type; thus each separate route could still be acyclic. This suggests the validity of the theory of re-entrant lines to more general systems without re-entry, but with two way flow of jobs, as in communication networks.

It is easy to see that CAF policies are indeed stable for *acyclic* systems, where we require acyclicity of the graph when *all* part-types are considered (thus it is not enough for individual routes to be acyclic, as in fig. 8).

In fact [35] even determines *sufficient* conditions on non-acyclic systems, under which CAF policies are stable. This result has been generalized somewhat by Humes [37].

In general, however, some modifications have to be made to ensure stability for general non-acyclic systems. In [35] it is shown that if one implements a "distributed supervisor" at each service center which

(i) truncates all long enough production runs, and
(ii) maintains a FIFO list of all buffers with a large number of parts,

then such a supervisor can stabilize any non-acyclic system.

Another approach taken in [33] shows how to implement a CAF policy at each machine as though it were operating in a virtually isolated mode, so as to ensure stability.

An important open problem is to accurately quantify the performance of such policies.

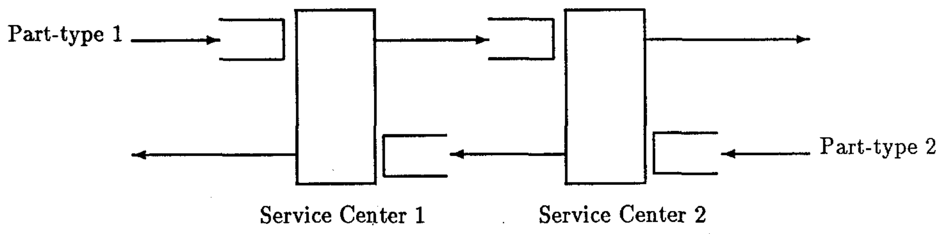

Fig. 8. A non-acyclic system, unstable under a distributed CAF policy, where the route of each part-type separately is acyclic.

15. Concluding remarks

We have made an attempt here to motivate several issues concerned with the scheduling of re-entrant lines. These types of manufacturing systems are of much current interest since they model semiconductor manufacturing plants, and more recently, thin film lines.

Aside from the re-entrant feature, some of the features of interest are random service and set-up times, as well as random machine failures and repairs.

Control is exercised over such systems by the release policy for introducing new parts into the system, and the scheduling policy for determining which part a machine should work on when it becomes idle. The goal of scheduling is to provide good performance relative to such measures as mean cycle-time and variance of cycle-time.

Due to the re-entrant form of the line, there is the possibility of instability. We have provided an account of some recent stability results in the field. Once stability is assured one wishes to improve the performance of the system, and we have provided results, simulations and conjectures concerning the behavior of various scheduling policies.

There are many open questions and we have suggested some of them. More work is definitely warranted to resolve these issues.

References

[1] S.C. Graves, A review of production scheduling, Math. Oper. Res. 29 (1981) 646–675.
[2] F. Baskett, K.M. Chandy, R.R. Muntz and F.G. Palacios, Open, closed and mixed networks of queues with different classes of customers, J. ACM 22 (1975) 248–260.
[3] F.P. Kelly, *Reversibility and Stochastic Networks* (Wiley, New York, 1979).
[4] S. Lippman, Applying a new device in the optimization of exponential queueing systems, Math. Oper. Res. 23 (1975) 687–710.
[5] S.P. Meyn, private communication (1991).
[6] S.P. Meyn and D. Down, Stability of generalized Jackson networks, submitted for publication (1991).
[7] R. Weber, P. Varaiya and J. Walrand, Scheduling jobs with stochastically ordered processing times on parallel machines to minimize expected flowtime, J. Appl. Prob. 23 (1986) 841–847.
[8] G.P. Klimov, Time-sharing service systems I, Theory Prob. Appl. 19 (1974) 532–551.
[9] G.P. Klimov, Time-sharing service systems II, Theory Prob. Appl. 23 (1978) 314–321.
[10] T.L. Lai and Z. Ying, Open bandit processes and optimal scheduling of queueing networks, Tech. Rep., Stanford University (1989).
[11] B. Simon, Priority queues with feedback, J. ACM 31 (1984) 134–149.
[12] R.L. Cruz, A calculus for network delay, part I: Network elements in isolation, IEEE Trans. Inf. Theory IT-37 (1991) 114–131.
[13] S.H. Lu and P.R. Kumar, Distributed scheduling based on due dates and buffer priorities, IEEE Trans. Auto. Control AC-36 (1991) 1406–1416.
[14] Z.B. Tang and L.Y. Shi, Note on distributed scheduling based on due dates and buffer priorities, by S. Lu and P.R. Kumar, Preprint (1991).

[15] A.P.J. Vepsalainen and T.E. Morton, Improving local priority rules with global lead-time estimates: A simulation study, J. Manuf. Oper. Manag. 1 (1988) 102–118.
[16] K.R. Baker, Sequencing rules and due-date assignments in a job shop, Manag. Sci. 30 (1984) 1093-1104.
[17] S.H. Lu and P.R. Kumar, Control policies for scheduling of semiconductor manufacturing plants, in: *Proc. 1992 American Control Conf.* (June 1992), to appear.
[18] L.M. Wein, Scheduling networks of queues: Heavy traffic analysis of a multistation network with controllable inputs, Preprint (June 1989).
[19] L.M. Wein, Reducing the variance of sojourn times in multi-class queueing systems, Preprint (November 1989).
[20] J. Kimemia and S.B. Gershwin, An algorithm for the computer control of a flexible manufacturing system, IIE Trans. 15 (1983) 353–362.
[21] R. Akella and P.R. Kumar, Optimal control of production rate in a failure prone manufacturing system, IEEE Trans. Auto. Control AC-31 (1986) 116–126.
[22] T. Bielecki and P.R. Kumar, Optimality of zero-inventory policies for unreliable manufacturing systems, Oper. Res. 36 (1988) 532–541.
[23] A. Sharifnia, Production control of a manufacturing system with multiple machine states, IEEE Trans. Auto. Control AC-33 (1988) 600–626.
[24] J. Lehoczky, S. Sethi, H.M. Soner and M. Taksar, An asymptotic analysis of hierarchical control of manufacturing systems under uncertainty, Math. Oper. Res. (1990), to appear.
[25] D. Mitra, Stochastic theory of a fluid model of producers and consumers coupled by a buffer, Adv. Appl. Prob. 20 (1988) 646–676.
[26] H. Chen and D.D. Yao, A fluid model for systems with random disruptions, Preprint (1990).
[27] J. Lim, S.M. Meerkov and F. Top, Homogeneous asymptotically reliable serial production lines: Theory and a case study, IEEE Trans. Auto. Control AC-35 (1990) 524–534.
[28] B.A. Sevastyanov, Influence of storage bin capacity on the standstill time of a production line, Theory Prob. Appl. 7 (1962) 429–438.
[29] J. Lim, S.M. Meerkov and F. Top, Analysis and synthesis of asynchronous asymptotically reliable, asymptotically controllable serial production lines, in: *Proc. 28th Conf. on Decision and Control*, Tampa, FL, vol. 1 (1989) pp. 60–64.
[30] C. Lozinski and C.R. Glassey, Bottleneck starvation indicators for shop floor control, IEEE Trans. Semicond. Manuf. SM-1 (1988) 147–153.
[31] S.B. Gershwin, Stochastic scheduling and setups in manufacturing systems, in: *Proc. 2nd ORSA/TIMS Conf. on Flexible Manufacturing Systems: Operations Research Models and Applications*, ed. K.E. Stecke (Elsevier, Amsterdam, 1986) pp. 431–442.
[32] S. Elmaghraby, The economic lot scheduling problem (ELSP): Review and extensions, Manag. Sci. 24 (1978) 587–598.
[33] J.R. Perkins and P.R. Kumar, Stable distributed real-time scheduling of flexible manufacturing/assembly/disassembly systems, IEEE Trans. Auto. Control AC-34 (1989) 139–148.
[34] C.J. Chase and P.J. Ramadge, On the real time control of flexible manufacturing systems, in: *Proc. 28th Conf. on Decision and Control*, Tampa, FL (1989) pp. 2026–2027.
[35] P.R. Kumar and T.I. Seidman, Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems, IEEE Trans. Auto. Control AC-35 (1990) 289–298.
[36] C. Chase, Manufacturing networks and stability, Preprint (1989).
[37] C. Humes Jr., A naive stabilization schema: Kumar–Seidman revisited, Preprint, Instituto de Matimatica et Estatistica-Universidade de S. Paulo, S. Paulo, Brazil (December 1990).