STABLE DISTRIBUTED, REAL-TIME SCHEDULING OF FLEXIBLE MANUFACTURING / ASSEMBLY / DISASSEMBLY SYSTEMS *

J. Perkins and P.R. Kumar†
Dept. of Electrical and Computer Engineering, and
Coordinated Science Laboratory
University of Illinois
1101 West Springfiled Avenue
Urbana, IL 61801
USA

Abstract

We consider general flexible manufacturing/assembly/disassembly systems with the following features:

(i) there are several part types, each with given processing time requirements at a specified sequence of machines;

(ii) each part-type needs to be produced at a prespecified rate;

(iii) parts may incur variable transportation delays when moving from one machine to another;

(iv) set-up times are required whenever a machine changes from a production run of parts of one type to another;

(v) some part-types may also need assembly or disassembly; and

*Please address all correspondence to the second author at: Coordinated Science Laboratory, University of Illinois, 1101 W. Springfield Avenue, Urbana, Illinois 61801/USA.

†The research of the authors has been supported in part by the National Science Foundation under Grant No. ECS-88-02576, in part by the U. S. Army Research Office under Contract No. DAAG-29-85-K0094, and in part by the Joint Services Electronics Program under Contract No. N00014-85-C0149.

(vi) a proportion of parts of a part-type may upon exiting from a machine require separate routing (due to say poor quality).

We exhibit a class of scheduling policies implementable in real-time in a distributed way at the various machines, which ensure that the cumulative production of each part-type trails the desired production by no more than a constant. The buffers of all the machines are guaranteed to be bounded, and the system can thus operate with finite buffer capacities. We exhibit finite upper bounds on these buffer levels for the given distributed real-time scheduling policies. We also exhibit a lower bound on the average buffer levels for *any* scheduling policy.

1 Introduction

To illustrate the type of manufacturing systems treated in this paper, consider Figure 1. It consists of 9 machines labeled $M1, M2, \ldots, M9$. There are 4 part-types $P1$, $P2$, $P3$,

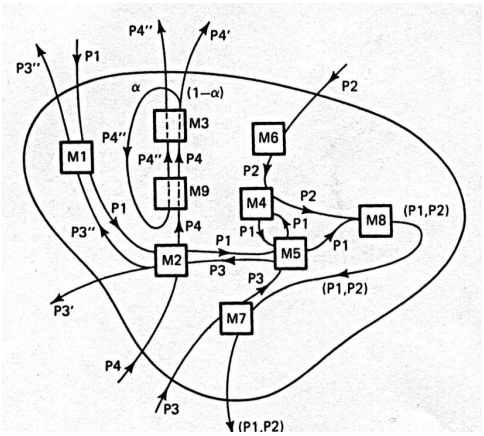

Figure 1: An example of a general system.

$P4$ flowing *into* the system. Parts of type $P1$ need to be processed at machine $M1$ for a *processing time* $\tau_{P1,M1}$, then at $M2$ for a time $\tau_{P1,M2}$, then at $M5$ for a time $\tau^{(1)}_{P1,M5}$, then at $M4$ for a time $\tau_{P1,M4}$, then at $M5$ *again*, this time for a time $\tau^{(2)}_{P1,M5}$, from which they proceed to $M8$, where each unit of part-type $P1$ has to be *assembled* with a part of type $P2$, which requires an assembly time of $\tau_{(P1,P2),M8}$. Similarly, parts of type $P2$ need work at $M6$ for a time $\tau_{P2,M6}$, then at $M4$ for a time $\tau_{P2,M4}$ before they are assembled with parts of type

$P1$ at $M8$. The assembled composite part $(P1, P2)$ then requires further processing at $M7$ for a time $\tau_{(P1,P2),M7}$. Parts of type $P3$ need work at machines $M7$ and $M5$ for times $\tau_{P3,M7}$ and $\tau_{P3,M5}$, respectively, from which they move to $M2$. At $M2$ parts of type $P3$ require a time $\tau_{P3,M2}$ to be *disassembled* into part-types $P3'$ and $P3''$. Parts of type $P3'$ leave the system, but parts of type $P3''$ need subsequent processing at $M1$ for a time $\tau_{P3'',M1}$. Parts of type $P4$ need processing at $M2$ for a time $\tau_{P4,M2}$, then at $M9$ for a time $\tau_{P4,M9}$ and then at $M3$ for a time $\tau_{P4,M3}$. However, only a *fixed proportion* $(1-\alpha)$ of parts of type $P4$, called $P4'$, then leave the system. The remaining proportion α of parts, called $P4''$, need rework, due to poor quality, and thus are rerouted back to $M9$ for further processing requiring a time $\tau_{P4'',M9}$, and then back to $M3$ for a further time $\tau_{P4'',M3}$,from which they then leave the system.

Let the cumulative demand for parts of the composite assembly $(P1, P2)$ at time t be $td_{(P1,P2)}$; thus the rate of desired production, called the *demand rate*, is $d_{(P1,P2)}$. Similarly, parts of type $P3'$ and $P3''$ have a *demand rate* $d_{P3'} = d_{P3''}$. Finally, parts of type 4 have a demand rate d_P4. This can be decomposed as $d_P4 = d_{P4'} + d_{P4''}$ where $d_{P4'} = (1-\alpha)d_P4$ is the desired rate for $P4'$, and $d_{P4''} = \alpha d_P4$, which is the desired rate for $P4''$.

We shall also suppose that parts incur a *transportation delay* when they move from one machine to another. This delay is *not* fixed, but may change with time depending on the existing load on the transportation network. We shall allow for this time variation by simply assuming that there is an *upper bound* $\sigma_{m\bar{m}}$ on the transportation delay from machine m to machine $\bar{m}$.

A critical feature of the manufacturing system is that *set-up times* are required whenever a machine changes the type of parts it is processing. Thus if machine m has been processing parts of type p, and it is desired to switch instead to the processing of parts of type q, then there is a set-up time δ_{pqm} required at machine m to handle parts of type q.

To summarize, the manufacturing systems we consider have the following features.

(i) There are several part-types $p = 1, 2, \ldots, P$, and several machines $m = 1, 2, \ldots, M$.

(ii) Each part type p needs to produced at a pre-specified demand rate d_p.

(iii) Parts of type p require a specified processing time at each machine in a given route, with $\tau_{p,m}^{(i)}$ being the processing time required on the i-th visit to machine m.

(iv) Parts may incur a variable transportation delay when moving from machine m to machine $\bar{m}$. This transportation delay is merely assumed to be bounded.

(v) Whenever machine m changes from the production of parts of type p to parts of type q, a set-up time δ_{pqm} is required.

(vi) Some parts may need assembly, thus giving rise to a composite part-type, or disassembly, thus splitting into several part-types.

(vii) A fixed proportion α_{pm} of parts of type p may require a separate routing after exiting from machine m.

The Main Results

In order for the system to be able to meet demand, it is clearly necessary that

$$\sum_{(p,i)} d_p \tau_{pm}^{(i)} < 1 \qquad \text{for every machine } m.$$

We will exhibit scheduling policies which guarantee that whenever this stability condition is satisfied, the manufacturing system is *stable* in the following sense.

(i) The cumulative production of every part-type p trails the cumulative demand by no more than a constant, i.e., if $y_p(t)$ is the total production of parts of type p in the time interval $[0, t]$, then

$$td_p \geq y_p(t) \geq td_p - M_p \qquad \text{for every } t > 0,$$

where M_p is a finite number.

Moreover, our policies have the following implementational properties.

(ii) They are implementable in a *distributed*, *real-time* fashion at the machines. By this we mean that the decision making regarding scheduling of parts to be worked on at machine m can be done by observing *only* the the buffers levels of the parts at machine m, and not the other machines. Moreover, these policies requires no computation at all, and so they can be implemented in real-time.

(iii) Our policies *decouple* the *transportation* problem from the *scheduling* problem. Since the scheduling policies for the machines guarantee stability as in (i) above, irrespective of transportation delays, the scheduling for the transportation system can be done independently, if so desired.

The *performance* of the scheduling policies is clearly measured by the magnitude of the buffer levels $\{x_{pm}(t)\}$.

(iv) We obtain upper bounds of the type,

$$\Sigma_p x_{pm}(t) \leq N_m \qquad \text{for every } t > 0.$$

This shows that our policies can be implemented with *finite buffers* of size N_m at each machine m, and guarantee that the buffers will never overflow. Note that by Little's Theorem, the *mean delay* is also bounded above by $N_m(\Sigma_p d_p)^{-1}$.

It is also of interest to know what the ultimate limits of performance are.

(v) We determine constants K_m, such that in the single machine case *every* stable policy satisfies,

$$\liminf_{t\to\infty} \frac{1}{t}\int_0^t \Sigma_p x_{pm}(s)ds \quad \geq \quad K_m.$$

Hence *no* stable policy exists which has average buffer levels less than K_m.

(vi) In addition, guided by our proof of this lower bound, we propose a policy in Section 4 which in several simulation tests for the single machine case has provided average buffer levels which are within 3.0% of the optimal.

Some Background and Some Philosophy

The work in this paper provides algorithms for fulfilling the functions of the intermediate layer of the hierarchical framework proposed by Gershwin [1]. In this suggested hierarchy, planning decisions regarding the capacity of a manufacturing system and its design are assigned to the topmost level, while detailed part-loading instructions are assigned to the lower-most level. Our work, incorporating set-up times, transportation delays and processing times of parts, can be regarded as an integrated package to deal with the intermediate level problem of real-time scheduling.

A key catalyst of our work has been the insight on how to deal with the issue of set-up times in manufacturing systems provided by Gershwin in [1], where he clearly formulates the issue by saying that, "set-up changes should not be performed too often because of the resulting reduction of capacity. They should not be performed too infrequently, because of the resulting increases of inventories and delays." The policies in this paper can be regarded as distributed feedback based policies to control the frequency of set-ups. Thus we eschew open-loop precomputation of production schedules, as has been traditional in the literature.

When employing feedback, the stability and dynamics of the resulting system are an important concern. Hence the philosophy of this paper is to introduce *feedback*, *stability*, and *dynamics* into the modeling, analysis and synthesis of manufacturing systems.

There has, of course, been much work in scheduling theory. A significant body of such work [2, 3, 4, 5, 6, 7, 8] deals essentially with *static* models. The typical problem addressed here is to schedule a *fixed* number of parts (or jobs), with known processing requirements, on a given set of machines, so as to minimize performance measures such as flowtime, makespan,

tardiness, lateness, etc. These issues are formulated as combinatorial optimization problems, including integer programming problems. Typically, however, the complexity of the solution procedure (usually branch and bound) grows exponentially with the number of parts and machines as well as the number of periods in the scheduling horizon. It is for this reason that for the dynamic real-time problems addressed here we eschew this computational approach. The interested reader will find good introductory accounts however in Baker [2], Conway, Maxwell and Miller [3] and French [4], while [6, 7, 8, 9, 10] are excellent surveys. An extension of this body of work deals with stochastic scheduling where the processing requirements are random. Excellent references are the works of Weiss [11], Weber, Walrand and Varaiya [12], and the references contained in them. Recently there has also been work on the issue of random machine failures, which can be regarded as the next higher level of decision making in the hierarchy. This can be found in Kimemia and Gershwin [13], Gershwin, Akella and Choong [14], Akella and Kumar [15], Bielecki and Kumar [16], and Sharifnia [17]. Some work on randomly changing demands can be found in Fleming, Sethi and Soner [18].

The remainder of this paper is organized as follows. In Section 2 we analyze the single machine scheduling problem in detail, and develop a class of stable, real-time scheduling policies, along with upper bounds on their performance. In Section 3 we develop a lower bound on the performance of the optimal scheduling policy, which we believe is fairly tight. In Section 4 we propose a particular policy which simulations have shown attains nearly optimal buffer levels. In Section 5 we show how distributed implementation of our single machine scheduling policies stabilizes all *acyclic* systems, and in Section 6 we consider some of the problems which arise when the manufacturing system is *not* acyclic. In Section 7 we describe a modification called "backoff" which stabilizes even non-acyclic systems, and in Section 8 we show how these policies are extended to handle assemblies, disassemblies and partial rerouting. Finally, Section 9 provides some concluding remarks.

2 SINGLE MACHINE SCHEDULING

Our approach to the theory will initially be modular. We will start with an elementary, idealized building block; later we will indicate how we proceed with a theory of interconnections, what difficulties arise, and our solutions to them.

Consider a single machine m in *isolation*, and suppose that there are several part-types labeled $1, \ldots, P$, which require production by machine m, with, say, prespecified demand rates $d_1, \ldots, d_P$. Let τ_p be the time required to produce one unit of part-type p and let us suppose that a set-up from parts of type p to parts of type q requires δ_{pq} units of time. For simplicity of notation in what follows we will assume that $\delta_{pq} \equiv \delta$ for all p, q.

Let us suppose that the machine receives an input of raw parts of type p at a constant rate d_p. Thus if $u_p(t) :=$ total input of parts of type p in $[0, t]$, then $u_p(t) = td_p$. Let us also define $y_p(t) :=$ total production of parts of type p in $[0, t]$, and $x_p(t) :=$ buffer level of parts of type p at time t. Clearly

$$x_p(t) \quad = \quad u_p(t) - y_p(t) = td_p - y_p(t), \tag{1}$$

as shown in Figure 2.

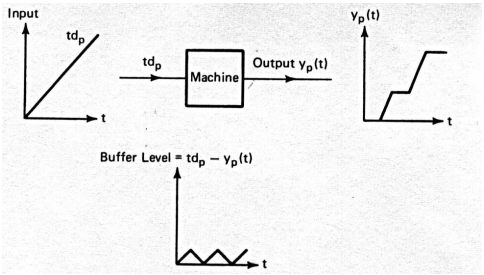

Figure 2: Input, Output and Buffer Levels at the Machine.

We will say that a scheduling policy is *stable* if,

$$\sup_t (td_p - y_p(t)) \quad \leq \quad M_p < +\infty \text{ for all part-types } p = 1, \ldots, P, \tag{2}$$

i.e., if the deviation of the actual amounts produced from the desired target is never off by more than a certain constant, say M_p. Note that by (1) this is equivalent to,

$$\sup_t x_p(t) \quad \leq \quad M_p < \infty \text{ for } p = 1, \ldots, P. \tag{3}$$

i.e., requiring that buffer levels at the machine are bounded.

Clearly, a *necessary* condition for stability of *any* policy is

$$\rho \quad := \quad \Sigma_p \tau_p d_p < 1.$$

We shall refer to ρ as the *machine load*, and

$$\rho_p \quad := \quad \tau_p d_p$$

as the *load on the machine due to part p*.

Note that ρ is the number of time units of work brought to the machine per time unit. Hence if the system is to meet demand, on the average the machine can only afford to spend a proportion $(1 - \rho)$ of the total time being *idle*. Since every set-up consumes δ units of idle time, it is clear that the average *frequency of set-ups* is bounded above by $\delta^{-1}(1 - \rho)$. If ρ is very close to 1, then $\delta^{-1}(1 - \rho)$ is very small, and set-ups are forced to be infrequent. This means that production runs have to be long. Hence the machine builds up huge buffer levels which are then cleared by long production runs, and so average buffer levels will be high. Thus one obtains performance curves for average buffer levels vs. machine load which are reminiscent of the average delay vs. load curves in queueing theory. Note however that our treatment is completely deterministic.

Consider now the following policy.

Definition: Clear-the-Largest-Buffer-Level (CLB) Policy. *Let $T_o = 0$. At time T_n, let the machine choose that part-type $\boldsymbol{p}^*(T_N)$ to produce which has the largest current buffer level, i.e.,*

$$x_{\boldsymbol{p}^*(T_n)}(T_n) \quad \leq \quad x_p(T_n) \text{ for all } p. \tag{4}$$

(If there is more than one such maximizer, any of the maximizers can be chosen.) Then, at time T_n, the machine sets up to begin the production of part-type $\boldsymbol{p}^*(T_n)$. This consumes a set-up time δ. Thus it is only at time $T_n + \delta$ that parts of type $\boldsymbol{p}^*(T_n)$ can actually be produced. This production run of part-type $\boldsymbol{p}^*(T_n)$ is then continued till the buffer is cleared, i.e., till a time T_{n+1} at which the buffer level hits 0. Thus T_{n+1} is given by

$$T_{n+1} \quad = \quad T_n + (1 - \rho_{\boldsymbol{p}^*(T_n)})^{-1}[\delta + \tau_{\boldsymbol{p}*(T_n)} x_{\boldsymbol{p}^*(T_n)}(T_n)]. \tag{5}$$

At time $T_n + 1$ this procedure is repeated and the machine sets up to produce that part-type $\boldsymbol{p}^*(T_{n+1})$, in the next production run, which has the largest buffer level at time $T_n + 1$, and so on. A graph of the resulting buffer levels is shown in Figure 3.

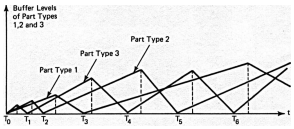

Figure 3: Buffer Levels vs. Time for CLB Policy.

In the above we have implicitly assumed that part flow is continuous as opposed to discrete. However, everything that follows in the sequel holds equally well for discrete flows as well as continuous flows.

The above policy is a special case of the following more general class of policies.

Definition: Clear-a-Fraction (CAF) Policies. *Fix $\epsilon > 0$. Let $T_0 = 0$. Consider a scheduling policy which at time T_n chooses any part-type $\boldsymbol{p}^*(T_n)$ to produce which satisfies,*

$$x_{p*(T_n)}(T_n) \quad \geq \quad \epsilon \Sigma_p x_p(T_n) \text{ for all } T_n, \tag{6}$$

This production is commenced at time $T_n + \delta$, and continued till the buffer level $x_{p*(T_n)}(t)$ hits 0 at the time $t = T_n + 1$ given by (5).

It is trivial to note that the CLB Policy is a special case of the above CAF type policy since the inequality (6) holds with the choice $\epsilon = P^{-1}$ by virtue of (4), where $P :=$ number of part-types. In fact a little thought shows that $\epsilon = (P-1)^{-1}$ is also valid.

Theorem 1 Performance of CAF Policies.

(i) All CAF policies are stable.

(ii) In particular,

$$\limsup_{t\to+\infty} \Sigma_p x_p(t) \;\leq\; \frac{\delta\rho}{\underline{\tau}} + \frac{\delta\overline{\tau}}{\epsilon\underline{\tau}(1-\rho)}\max_p\left(\frac{\rho-\rho_p}{\tau_p}\right),$$

where $\underline{\tau} := \min_p \tau_p$ *and* $\overline{\tau} := \max_p \tau_p$.

(iii) If the initial buffer levels satisfy, $\Sigma_p x_p(0) \leq \frac{\delta}{\epsilon(1-\rho)}\max_p\left(\frac{\rho-\rho_p}{\tau_p}\right)$, *then*

$$\Sigma_p x_p(t) \;\leq\; \frac{\delta\rho}{\underline{\tau}} + \frac{\delta\overline{\tau}}{\epsilon\underline{\tau}(1-\rho)}\max_p\left(\frac{\rho-\rho_p}{\tau_p}\right) \text{ for all } t \geq 0.$$

Hence CAF policies can be implemented with fixed, finite buffer capacities.

Proof: A key quantity to examine is the *net work at the machine* w(t) defined by,

$$w(t) \;:=\; \Sigma_p \tau_p x_p(t),$$

which is the total work contained in the buffers for the machine at time t. The quantity $w(t)$ plays a role similar to that of a Lyapunov function. Note that,

$$\begin{aligned}
w(T_{n+1}) &= \Sigma_p \tau_p x_p(T_{n+1}) = \sum_{p\neq \boldsymbol{p}^*(T_n)} \tau_p x_p(T_{n+1}) \qquad (\text{since } x_{p^*(T_n)}(T_n) = 0) \\
&= \sum_{p\neq \boldsymbol{p}^*(T_n)} \tau_p[x_p(T_n) + d_p(T_{n+1}-T_n)] \\
&= w(T_n) - \tau_{\boldsymbol{p}^*(T_n)} x_{\boldsymbol{p}^*(T_n)}(T_n) + (T_{n+1}-T_n)\sum_{p\neq \boldsymbol{p}^*(T_n)} \tau_p d_p \\
&= w(T_n) - \alpha(\boldsymbol{p}^*(T_n)) x_{\boldsymbol{p}^*(T_n)}(T_n) + \beta(\boldsymbol{p}^*(T_n)) \qquad (using (5)),
\end{aligned}$$

where $\alpha(p) := (1-\rho_p)^{-1}(1-\rho)\tau_p$ and $\beta(p) := (1-\rho_p)^{-1}(\rho-\rho_p)\delta$. Let $\overline{\tau} := \max_p \tau_p$, and note that

$$\begin{aligned}
\alpha(p*(T_n)) x_{p*(T_n)}(T_n) &\geq \alpha(p*(T_n))\epsilon\Sigma_p x_p(T_n), \qquad (\text{by } (6)), \\
&\geq \alpha(p*(T_n))\epsilon\overline{\tau}^{-1}\Sigma_p \tau_p x_p(T_n), \qquad (\text{since } \tau_p \leq \overline{\tau} \text{ for all } p), \quad (7) \\
&= \alpha(p*(T_n))\epsilon\overline{\tau}^{-1} w(T_n).
\end{aligned}$$

Hence,

$$w(T_n+1) \;\; <= \;\; [1-\epsilon\overline{\tau}^{-1}\alpha(p^*(T_n))]w(T_n)+\beta(p^*(T_n)). \tag{8}$$

Now let us define,

$$\gamma_n \;\; := \;\; w(T_n)-\frac{\overline{\tau}}{\epsilon}\max_p\frac{\beta(p)}{\alpha(p)}.$$

Then by (8) we obtain,

$$\begin{aligned}
\gamma_{n+1} &\leq \gamma_n\left[1-\frac{\epsilon}{\overline{\tau}}\alpha(p^*(T_n))\right]+\beta(p^*(T_n))\left[1-\frac{\alpha(p^*(T_n))}{\beta(p^*(T_n))}\max_p\frac{\beta(p)}{\alpha(p)}\right] \\
&\leq \gamma_n\left[1-\frac{\epsilon}{\overline{\tau}}\alpha(p^*(T_n))\right], \\
&\leq \gamma_n\left[1-\frac{\epsilon}{\overline{\tau}}\min_p\alpha(p)\right].
\end{aligned}$$

Since $0<\left[1-\frac{\epsilon}{\overline{\tau}}\min_p\alpha(p)\right]<1$, it follows that $\gamma_n\to 0$, and so

$$\text{lim-sup}_{n\to\infty}w(T_n) \;\; <= \;\; \frac{\overline{\tau}}{\epsilon}\max_p\frac{\beta(p)}{\alpha(p)}.$$

Now noting that $w(t)$ is piecewise-linear, and

$$w(t)\leq w(T_n+\delta) \;\; = \;\; w(T_n)+\rho\delta \text{ for } T_n\leq t\leq T_n+1, \tag{9}$$

we obtain,

$$\lim_{t\to\infty}w(t) \;\; <= \;\; \delta\rho+\frac{\overline{\tau}}{\epsilon}\max_p\frac{\beta(p)}{\alpha(p)}.$$

It then follows that

$$\Sigma_p x_p(t) \;\; \leq \;\; \frac{1}{\underline{\tau}}\Sigma_p\tau_p x_p(t)=\frac{1}{\underline{\tau}}w(t), \tag{10}$$

(where $\underline{\tau}:=\min_p\tau_p$) and the results (ii) and (i) follow.

For (iii), we simply note that since $\Sigma_p x_p(0)\leq\frac{1}{\epsilon}\max_p\frac{\beta(p)}{\alpha(p)}$, we have

$$w(T_0)=\Sigma_p\tau_p x_p(0) \;\; \leq \;\; \overline{\tau}\Sigma_p x_p(0)\leq\frac{\overline{\tau}}{\epsilon}\max_p\frac{\beta(p)}{\alpha(p)}.$$

Hence by iteratively applying (8) we get $\sup_n w(T_n) \leq \frac{\overline{\tau}}{\epsilon} \max_p \frac{\beta(p)}{\alpha(p)}$, from which the desired result follows through (9) and (10).

The performance bounds provided in the above Theorem are somewhat gross. It would be of interest to obtain sharper bounds.

Another interesting special case of CAF Policies is the following policy.

Definition: Clear-the-Largest-Work (CLW) Policy. *This policy chooses at time T_n that part $p^*(T_n)$ for which $\tau_p x_p(T_n)$ is largest, i.e.,*

$$\tau_{p^*(T_n)} x_{p^*(T_n)}(T_n) \quad \geq \quad \tau_p x_p(T_n) \text{ for all } p.$$

Theorem 1: Performance of CLW Policy. *For a CLW Policy,*

$$\textit{lim-sup}_{t \to +\infty} w(t) \quad \leq \quad \frac{\delta(P-1)}{(1-\rho)}(\rho - \min_p \tau_p d_p) + \delta\rho \leq \frac{\delta\rho(P-\rho)}{(1-\rho)}.$$

Proof: The proof is simply based on noting that (7) can be improved as follows.

$$\begin{aligned} \alpha(p^*(T_n)) x_{p^*(T_n)}(T_n) &= \frac{\alpha(p^*(T_n))}{\tau_{p^*(T_n)}} \tau_{p^*(T_n)} x_{p^*(T_n)}(T_n) \\ &\geq \frac{\alpha(p^*(T_n))}{(P-1)\tau_{p^*(T_n)}} \Sigma_p \tau_p x_p(T_n) \\ &= \frac{\alpha(p^*(T_n))}{(P-1)\tau_{p^*(T_n)}} w(T_n). \end{aligned}$$

The interested reader may note that the above policies are loosely related to the queueing systems with vacations treated in Bertsekas and Gallager [19].

Finally we point out that the above results can be generalized slightly to the case where the demands and inputs are "nearly linear" but not strictly linear, see Lemma 6 in Section 5.

3 Lower Bound on Average Buffer Levels of Stable Scheduling Policies

How close to optimal are the buffer levels resulting from policies of the CLB, CLW or, more generally, of the CAF type? To answer this, we clearly need to know what the optimal buffer levels are, but the problem of determining the optimal scheduling policy which minimizes,

$$\frac{1}{T}\int_0^T w(s)ds \quad = \quad \text{Average work in system in the time interval } [0,T]$$

for large T, requires the solution of a very large mixed integer-linear programming problem, and a precise answer would require too much computation. The following Theorem gives a lower bound on the performance of *any* stable scheduling policy, which we conjecture is fairly tight.

Theorem 2: Lower Bound on Performance of any Stable Scheduling Policy. *Let $\{\gamma_p\}$ be any set of positive weighting factors. For any scheduling policy which is stable, i.e., one for which (3) holds, the average weighted buffer level in the system has the lower bound:*

$$\textit{lim-inf}_{t\to+\infty}\frac{1}{t}\int_0^t [\Sigma_p \gamma_p \tau_p x_p(s)]ds \quad \geq \quad \frac{\delta\left[\Sigma_p\sqrt{\gamma_p\rho_p(1-\rho_p)}\right]^2}{2(1-\rho)}.$$

Proof: Consider *any* stable scheduling policy. It gives rise to trajectories $\{x_p(t) : 0 \leq t < +\infty, p = 1,2,\ldots,P\}$ which satisfy,

$$\sup_t x_p(t) \quad \leq \quad c \text{ for every } p = 1,2,\ldots,P \tag{11}$$

for some number $c < +\infty$.

Now fix some time $T < +\infty$, and let n_p be the number of production runs for part-type p in the interval $[0,T]$, and let T_p be the total of these production run lengths. Note that

$$\Sigma_p T_p + \delta\left[(\Sigma_p n_p) - 1\right] \quad \leq \quad T, \tag{12}$$

since there are at least $(\Sigma_p n_p) - 1$ set-ups. Since $x_p(T) = x_p(0) + d_p T - T_p \tau_p^{-1}$ and since by (11), $x_p(T) - x_p(0) \leq c$, it follows that $d_p T - T_p \tau_p^{-1} \leq c$. Hence, $\Sigma_p \tau_p d_p T - \Sigma_p T_p \leq \Sigma_p c \tau_p$. Defining $\widehat{c} := \Sigma_p c \tau_p$, we thus have $\rho T - \Sigma_p T_p \leq \widehat{c}$. Hence, from (12) we obtain,

$$\begin{aligned} \delta \Sigma_p n_p &\leq \delta + T - \Sigma_p T_p \\ &\leq (1-\rho)T + \delta + \widehat{c}. \end{aligned} \tag{13}$$

Now fix attention on a part-type p. Let $\{\bar{x}_p(t) : 0 \leq t \leq T\}$ be a function which attains the *minimum* value of the cost function $\frac{1}{T}\Sigma_0^T \gamma_p \tau_p \bar{x}_p(t) dt$ subject to the constraints that:

(i) $0 \leq \bar{x}_p(t) \leq c$ for $0 \leq t \leq T$,

(ii) $\bar{x}_p(t)$ is continuous and piecewise linear, with the slopes of the linear segments being either d_p (for an idle period) or $(d_p - \tau_p^{-1})$ (for a production run).

(iii) the number of linear segments with slopes $(d_p - \tau_p^{-1})$ (i.e., the number of production runs) is less than or equal to n_p.

(iv) the total length of the linear segments with slopes $(d_p - \tau_p^{-1})$ is less than or equal to T_p.

(Note that a minimum exists because such functions can be characterized by the initial value $\bar{x}_p(0)$, a binary number indicating whether the period [0,T] starts with a production run or an idle period, and the lengths of the successive linear segments. Thus such functions are parameterized by a finite-dimensional vector lying in a compact set, and since the cost function is continuous with respect to this parameterization, there exists a minimizing function.)

Clearly the original scheduling policy gives rise to a function $\{x_p(t) : 0 \leq t \leq T\}$ which satisfies all the above conditions (and more, due to the fact that p and p' cannot simultaneously be on a production run at any given time), and since $\bar{x}_p(t)$ attains the minimum within this class, it follows that

$$\frac{1}{T}\int_0^T \Sigma_p \gamma_p \tau_p x_p(t) dt \geq \frac{1}{T}\int_0^T \Sigma_p \gamma_p \tau_p \bar{x}_p(t) dt. \tag{14}$$

Let $\bar{n}_p$ be the number of production runs of $\{\bar{x}_p(t) : 0 \leq t \leq T\}$ and let $\overline{T}_p$ be the total of all its production run lengths. Note that by (iii) $\bar{n}_p \leq n_p$.

We claim that every one of the $\bar{n}_p$ production runs of $\bar{x}_p(t)$ is *clearing*, i.e., at the end of each production run the value of $\bar{x}_p$ is 0. To see this we only need to consider the three cases shown in Figure 4. Suppose no production run is clearing, as in Figure 4a. Then, by

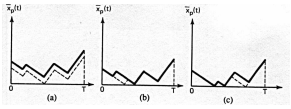

Figure 4:

translating the graph downwards to the dashed-line, we obtain a strictly lower cost, which is a contradiction. Hence there is at least one production run which is clearing. Now note that every production run *prior* to a clearing production run is itself clearing. To see this note that, as in Figure 4b, if this is not true then there is a perturbation of the schedule, indicated by the dashed line, which has lower cost, while still having the number of production runs bounded by $\bar{n}_p$, and preserving the total of production run lengths to be equal to $\overline{T}_p$. Finally, we note that every production run subsequent to a clearing production run is also clearing. Otherwise, as shown by the dashed line in Figure 4c, there is again a perturbation which has a strictly smaller cost, and which similarly meets the constraints.

Finally, we claim that the values $y_1, y_2, y_3, \ldots$ of $\bar{x}_p(t)$ at the beginning of every production run, except possibly for the first one, are all equal. Otherwise, as shown in Figure 5a the perturbation given by the dashed line has a strictly smaller cost. To verify this it is

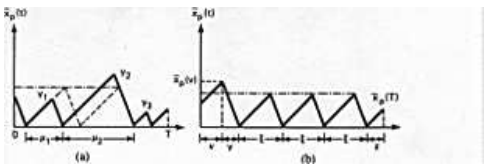

Figure 5:

just necessary to note that the areas of the triangles with bases μ_1 and μ_2 in Figure 5a

are $\frac{1}{2}\mu_1^2 d_p(1-\rho_p)$ and $\frac{1}{2}\mu_2^2 d_p(1-\rho_p)$, and the minimization of their total area, subject to the constraint that $\mu_1 + \mu_2$ =constant, is achieved when $\mu_1 = \mu_2$. Thus the schedule $\{\bar{x}_p(t) : 0 \le t \le T\}$ is of the form shown in Figure 5b, where v, y and z may be zero.

Due to the bound (11) on $\bar{x}_p(v)$ and $\bar{x}_p(T)$ it follows that $v \le c d_p^{-1} y \le c(\tau_p^{-1} - d_p)^{-1}$ and $z \le c d_p^{-1}$. Hence, $v + y + z \le \bar{c}$ where $\bar{c} := max\{3c d_p^{-1}, 3c(\tau_p^{-1} - d_p)^{-1}\}$ and so the base ξ of each of the other $(\bar{n}_p - 1)$ triangles in Figure 5b satisfies $\xi \ge \frac{T-\bar{c}}{\bar{n}_p - 1}$. Since the area of each triangle is $\frac{1}{2}\xi^2 d_p(1-\rho_p)$, as a simple calculation shows, we have

$$\int_0^T \bar{x}_p(t)dt \quad \ge \quad \frac{(T-\bar{c})^2 d_p(1-\rho_p)}{2(\bar{n}_p - 1)} \ge \frac{(T-\bar{c})^2 d_p(1-\rho_p)}{2n_p}. \tag{15}$$

Hence we obtain the lower bound,

$$\frac{1}{T}\int_0^T \Sigma_p \gamma_p \tau_p \bar{x}_p(t)dt \quad \ge \quad \Sigma_p \frac{\gamma_p (T-\bar{c})^2 \rho_p(1-\rho_p)}{2Tn_p}. \tag{16}$$

Now consider the optimization problem:

$$\min \Sigma_p \frac{\gamma_p \rho_p (1-\rho_p)(T-\bar{c})^2}{2n_p T} \tag{17}$$

subject to,

$$\delta \Sigma_p n_p \quad \le \quad (1-\rho)T + \delta + \widehat{c} \tag{18}$$

$$n_p \ge 0 \quad \text{for} \quad p = 1, 2, \ldots, P. \tag{19}$$

Since the constraints are a relaxation of (13) (due to the relaxation of the integrality condition on n_p), the minimal cost of this problem is a lower bound on $\frac{1}{T}\int_0^T \Sigma_p \gamma_p \tau_p x_p(t)dt$, the average cost of the original scheduling policy, over the interval $[0, T]$.

Note that one can bound each n_p strictly away from 0, since we know a finite upper bound on the cost function. Applying the Kuhn-Tucker conditions we have

$$-\frac{\gamma_p (T-\bar{c})^2 \rho_p(1-\rho_p)}{2Tn_p^2} + \lambda\delta - \mu_p \quad = \quad 0 \text{ for } p = 1, 2, \ldots, P$$

where λ and μ_p are the Lagrange multipliers associated with (18) and (19). This is solved by,

$$n_p \;=\; k\sqrt{\gamma_p\rho_p(1-\rho_p)} \text{ for } p=1,2,\ldots,P, \tag{20}$$

where the constant k satisfies

$$k \;=\; \frac{(1-\rho)T+\delta+\widehat{c}}{\delta\Sigma_p\sqrt{\gamma_p\rho_p(1-\rho_p)}}. \tag{21}$$

and the Lagrange multipliers are $\lambda = \frac{(T-\bar{c})^2}{2k^2T\delta} > 0$, and $\mu_p = 0$ for $p = 1,2,\ldots,P$. Since the cost function (17) is convex in $(n_1, n_2, \ldots, n_P)$, and the constraints (18, 19) are linear inequalities, the Kuhn-Tucker conditions are also sufficient (see Varaiya[20]), and so (20, 21) is the optimal solution.

Since the resulting minimum cost is a lower bound for the average cost of our original schedule over $[0,T]$, see (14, 16), we obtain

$$\frac{1}{T}\int_0^T \Sigma_p\gamma_p\tau_p x_p(t)dt \;\geq\; \frac{(T-\bar{c})^2\delta\left[\Sigma_p\sqrt{\gamma_p\rho_p(1-\rho_p)}\right]^2}{2T[(1-\rho)T+\delta+\widehat{c}]}.$$

Taking the limit as $T\to+\infty$, we get the desired result.

4 A Suggested Policy that Performs Well in Simulations

In fact, the proof of the lower bound in the above Theorem 2 suggests a way to weight buffer levels of the parts to obtain a policy whose performance is very good. From (15) and (20) it follows that,

$$\text{Peak value of } \bar{x}_p \text{ prop } \frac{d_p(1-\rho_p)}{\sqrt{\gamma_p\rho_p(1-\rho_p)}} \;=\; d_p\sqrt{\gamma_p^{-1}\rho_p^{-1}(1-\rho_p)}.$$

Since this peak value occurs just *after* a set-up, this suggests a policy where we choose $p^*(T_n)$ as,

$$\begin{aligned} p^*(T_n) &= \arg\max_p \left\{ \frac{x_p(T_n) + \delta d_p}{\text{peak value of } \bar{x}_p} \right\} \\ &= \arg\max_p \left\{ \frac{x_p(T_n) + \delta d_p}{d_p \sqrt{\gamma_p^{-1} \rho_p^{-1} (1-\rho_p)}} \right\}. \end{aligned}$$

Note that this is a policy of the form, $p^*(T_n) = \arg\max_p \{a_p x_p(T_n) + b_p\}$. It is not a CAF policy due to the presence of the term $b_p = \frac{\delta}{\sqrt{\gamma_p^{-1} \rho_p^{-1}(1-\rho_p)}}$. However, it is easily checked by just a slight modification of the proof of Theorem 1 that the policies of the form above are also stable whenever $a_p > 0$, as is the case here.

This policy has been simulated on many examples with arbitrarily chosen values for δ, γ_p, τ_p, d_p and number of parts. To compare the performance of this policy with the lower bound, let us define

$$\eta = \frac{\text{Average buffer level realized in simulation}}{\text{Lower bound given in Theorem 2}}$$

Note that a value of $\eta = 1$ may not even be realizable.

The following four examples are representative of the results obtained. In all cases, $\gamma_p \equiv 1$, $\delta = 1$. (All systems were taken to be initially empty, and allowed to run for 10,000 production runs).

Test 1: 15 parts, $d_1 = d_2 = ., . = d_5 = 0.05$, and $d_6 = ., .., = d_15 = 0.01$, $\tau_p \equiv 1$. Note that $\rho = 0.35$.

Result: $\eta = 1.0021$.

Test 2: 10 parts, $d_1 = 0.24$, $d_2 = 0.16$, $d_3 = 0.12$, $d_4 = 0.09$, $d_5 = 0.08$, $d_6 = 0.06$, $d_7 = 0.04$, $d_8 = 0.03$, $d_9 = 0.02$, $d_{10} = 0.01$, $\tau_p \equiv 1$. Note that $\rho = 0.85$.

Result: $\eta = 1.0225$.

Test 3: 3 parts, $d_1 = 7$, $d_2 = 9$, $d_3 = 3$, $\tau_1 = 1/100$, $\tau_2 = 1/51$, $\tau_3 = 1/27$. Note that $\rho = 0.35758$.

Result: $\eta = 1.0264$.

Test 4: 3 parts, $d_1 = 18$, $d_2 = 3$, $d_3 = 1$, $\tau_1 = 1/35$, $\tau_2 = 1/7$, $\tau_3 = 1/20$. Note that $\rho = 0.99286$.

Result: $\eta = 1.0071$.

It is worth noting that for all these examples with more than three parts, the suggested policy has a performance no more than 3.0% above optimal. We feel that this policy is well worthy of consideration.

5 Acyclic Manufacturing Systems

Now we turn to the case where there are several machines. We first consider only *acyclic* manufacturing systems, such as the one shown in Figure 6. This models a situation such as a

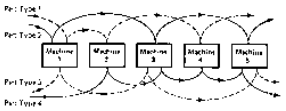

Figure 6: An Acyclic System

flow shop, where the machines can be ordered so that parts can only move from one machine to a machine higher in the ordering. We also suppose that there is a *transportation time* between any two machines which is *bounded* by σ, and a set-up time δ which for convenience does not depend on parts or machines.

As shown in Figure 6, let us suppose that raw parts of each type p are input at rate d_p into the system. For each machine m, let

$u_p m(t)$:= total input of part-type p into machine m in the interval $[0, t]$,

$$
\begin{aligned}
y_p m(t) &:= \text{ total output of part-type } p \text{ from machine } m \text{ in the interval } [0,t], \\
x_p m(t) &:= u_p m(t) - y_p m(t) = \text{ buffer level of part-type } p \text{ at machine } m \text{ at time } t \\
u_p(t) &:= td_p = \text{ total input of part-type } p \text{ into the } system\ r \text{ in } [0,t] \\
&\qquad \text{(i.e., into the } first \text{ machine at which part-type } p \text{ is processed)}
\end{aligned}
$$

and

$$
\begin{aligned}
y_p(t) &:= \text{ total output of part — type } p \text{ from the } systems \text{ in [0,t]} \\
&\qquad \text{(i.e., from the } last \text{ machine at which part-type } p \text{ is processed)}
\end{aligned}
$$

We shall say that the manufacturing system is *stable* if the condition (2) holds, i.e., if the production of every part-type deviates from the desired production by no more than a constant. The following Lemma shows that this is equivalent to the condition,

$$\sup_t x_{pm}(t) < +\infty \text{ for every } p, m, \tag{22}$$

i.e., that the buffer levels of parts are bounded at all machines.

Lemma 4: Equivalence of Stability and Bounded Buffer Levels. *The stability condition (2) is equivalent to (22), when the transportation delays are bounded by σ.*

Proof: Note that $\Sigma_m x_{pm}(t) \leq td_p - y_p(t)$ since some parts of a part-type may be in transit between machines at time t and thus unaccounted for in the buffer levels. Hence it is obvious that (2) implies (22). For the converse, note that the quantity $\bar{z}_{pm\bar{m}}(t)$ of part-type p in transit between machines m and $\bar{m}$ (if $\bar{m}$ is the next machine visited by part p) is bounded by $\bar{z}_{pm\bar{m}}(t) \leq y_p m(t) - y_p m(t-\sigma)$, since the transportation delay between machines m and $\bar{m}$ is bounded by σ. However, $y_{pm}(t) - y_{pm}(t-\sigma) \leq \sigma\tau_{pm}^{-1}$. Hence $td_p - y_p(t) \leq \sum_{\{m:\text{ part } p \text{ visits } m\}} \sigma\tau_{pm}^{-1} + \sum_m x_{pm}(t)$ and so (22) implies (2).

It is obvious that the necessary condition for stability is:

$$\sum_{\{p:\text{ part } p \text{ visits } m\}} \tau_{pm} d_p < 1 \qquad \text{for every } m. \tag{23}$$

We now show that the acyclic manufacturing system is stable when each machine implements a CAF Policy, which is as follows. Fix $\epsilon > 0$. Each machine m upon ending a production run (of possibly zero length, if it is starved of parts), chooses a part-type p^* for which $x_{p^*m} \geq \epsilon_m \Sigma_p x_{pm}$, and then, after a set-up, continues production of that part-type until its buffer level x_{p^*m} hits 0. Note that the completion time of the production run is not given by a clean formula such as (5) for T_{n+1} in the single-machine case, because both the input to the machine and the transportation delays are variable.

It should be noted the resulting scheduling policy is *distributed*. Each machine makes decisions on which part to produce based *only* on *local information* regarding the buffer levels of parts at its own buffers. Thus we shall call the resulting overall scheduling policy a *Distributed CAF Policy*. It is trivial to implement in *real-time*.

Theorem 5: Stability of Distributed CAF Policies for Acyclic Systems. *Distributed CAF Policies stabilize acyclic manufacturing systems when the condition (23) is satisfied.*

Proof: We will show by induction on the order of the machines that $\sup_t \Sigma_p \tau_{pm} x_{pm}(t) < +\infty$ for every m, which will prove stability by virtue of Lemma 4. Note that the inputs to machine 1 (see Figure 6) are linear functions td_p, and so by Theorem 1, $\sup_t x_{p1}(t) < +\infty$ for all part-types p that visit machine 1. Now suppose by induction that $\sup_t x_{pn}(t) < +\infty$ for all p and for $n = 1, 2, \ldots, m-1$. Then, as in Lemma 4, it follows that at machine m,

$$td_p \geq u_{pm}(t) \quad \geq \quad td_p - \gamma_{pm} \text{ for every part-type } p \text{ that visits } m, \tag{24}$$

where since the second term is an upper bound on the quantity of parts in transit between machines prior to m. The following Lemma completes the proof by showing that whenever a single machine m has inputs satisfying the "near linearity" condition (24), then $\sup_t x_{pm}(t) < +\infty$ for every p.

Lemma 6: Stability of CAF Policies Under Nearly Linear Inflows. *Consider a single machine m operating in isolation under a CAF Policy. Suppose that its input of part-type p satisfies $td_p + \gamma \geq u_{pm}(t) \geq td_p - \gamma$ for all p and $t \geq 0$, where $\gamma > 0$ is some constant. Then if the stability condition $\Sigma_p \tau_{pm} d_p < 1$ holds, the buffer levels of all parts are bounded.*

Proof: As in Theorem 1, let T_n be the time at which the n-th production run is completed, which may now however have zero length, and let $p^*(T_n)$ be the part-type chosen for production at T_n. Note that

$$\begin{aligned}(T_n + 1 - T_n - \delta)(\tau_{p^*(T_n)m})^{-1} &= x_{p^*(T_n)m}(T_n) + u_{p^*(T_n)m}(T_n + 1) - u_{p^*(T_n)m}(T_n) \\ &\leq x_{p^*(T_n)m}(T_n) + d_{p^*(T_n)}(T_n + 1 - T_n) + 2\gamma.\end{aligned}$$

Hence

$$(T_n + 1 - T_n) \leq \frac{\tau_{p^*(T_n)m} x_{p^*(T_n)m}(T_n) + 2\gamma\tau_{p^*(T_n)m} + \delta}{1 - \tau_{p^*(T_n)m} d_{p^*(T_n)}}.$$

Define $\rho_{pm} := \tau_{pm} d_p$ and $\rho_m := \Sigma_p \rho_{pm}$. Then if we let $w(T_n) := \Sigma_p \tau_{pm} x_{pm}(T_n)$, we have,

$$\begin{aligned}w(T_n + 1) &= \sum_{p \neq p^*(T_n)} \tau_{pm} x_{pm}(T_n + 1) \\ &= \sum_{p \neq p^*(T_n)} [\tau_{pm} x_{pm}(T_n) + \tau_{pm}(u_{pm}(T_n + 1) - u_{pm}(T_n))] \\ &\leq \left[\sum_{p \neq p^*(T_n)} \tau_{pm} x_{pm}(T_n)\right] + (\rho_m - \rho_{p^*(T_n)m})(T_n + 1 - T_n) + 2\gamma\Sigma_p\tau_{pm} \\ &= w(T_n) - \tau_{p^*(T_n)m} x_{p^*(T_n)m}(T_n) + (\rho_m - \rho_{p^*(T_n)m})(T_n + 1 - T_n) + 2\gamma\Sigma_p\tau_{pm} \\ &\leq w(T_n) - \tau_{p^*(T_n)m} x_{p^*(T_n)m}(T_n) \left[1 - \frac{(\rho_m - \rho_{p^*(T_n)m})}{(1 - \rho_{p^*(T_n)m})}\right] \\ &\qquad + \frac{(2\gamma\tau_{p^*(T_n)m} + \delta)(\rho_m - \rho_{p^*(T_n)m})}{(1 - \rho_{p^*(T_n)m})} + 2\gamma\Sigma_p\tau_{pm} \\ &\leq (1 - \epsilon_m) w(T_n) + \frac{(2\gamma\tau_{p^*(T_n)m} + \delta)(\rho_m - \rho_{p^*(T_n)m})}{(1 - \rho_{p^*(T_n)m})} + 2\gamma\Sigma_p\tau_{pm}\end{aligned}$$

where $\epsilon_m > 0$ is an appropriate constant due to the policy being of CAF type. Since the coefficient of $w(T_n)$ on the right hand side above is < 1, the difference equation has a bounded solution, and the proof concludes as in Theorem 1.

It is worth noting that this Lemma generalizes Theorem 1 to the case where inputs are not strictly linear functions.

6 Non-Acyclic Manufacturing Systems

Let us now consider the scheduling problem when the manufacturing system does *not* have the simplifying acyclic structure of the preceding section. Thus we consider manufacturing systems where there is no "direction" of flow which is uniform for all parts. Moreover, a part can revisit the same machine more than once. Clearly a necessary condition for stability is,

$$\rho_m \quad := \quad \Sigma_p d_p \left[\tau_{pm}^{(1)} + \tau_{pm}^{(2)} + \cdots + \tau_{pm}^{(n_p,m)} \right] < 1 \tag{25}$$

where $\tau_{pm}^{(i)}$ is the processing time required by a part of type p on its i-th visit to machine m, and $n_{p,m}$ is the total number of visits. Setting,

$x_{pm}^{(i)}(t) \quad = \quad$ quantity of parts of type p on their i-th visit to m which are awaiting processing at m,

the Distributed, Real-Time CAF Policies are then defined as those which operate at each machine m by choosing $(p^*(T_n), i*(T_n))$ at time T_n such that

$$x_{p^*(T_n)m}^{(i*(T_n))}(T_n) \quad \geq \quad \epsilon_m \sum_{p,i} x_{pm}^{(i)}(T_n)$$

and then processing parts of type $p^*(T_n)$, on their $i*(T_n)$-th visit, until their buffer level hits 0, where $\epsilon_m > 0$ is an initially chosen "fraction."

We have *not* been able to show the stability of such policies. Rather we have only been able to show the much weaker result that every part entering the manufacturing system eventually leaves it, as we prove in the following Theorem.

Theorem 7: Performance of CAF Policies for Non-Acyclic Systems. *If a Distributed CAF Policy is implemented at each machine, and condition (5) is satisfied, then*

(i) for every (p, m, i) there is a sequence of times $t_n \to +\infty$ such that $x_{pm}^{(i)}(t_n) = 0$, and

(ii) every part entering the system eventually leaves it.

Proof: Consider a part-type p, and let m be the *first* machine at which it seeks processing. Suppose there is a time $T < +\infty$ such that machine m does not work after time T on any parts of type p on their first visit to machine m. Then the buffer level $x_{pm}^{(1)}(t) - x_{pm}^{(1)}(T)$ grows linearly with time t like $d_p(t - T)$ since raw material is input at a rate d_p. Hence we may as well assume that after time T (which may need to be increased) w have $x_{pm}^{(1)}(t) \geq \Gamma$ for $t \geq T$, where Γ is a large number that we shall specify later. Let $b_m(t)$ be the total work in time units of machine m's work, eventually destined for machine m, which is currently in the system at time t. Clearly this work grows due to raw material input by the term $(t - T) \sum_q, j d_q \tau_{qm}^{(j)}$ and depletes whenever machine m is not idle. Letting $B_m(T, t)$ denote that portion of the time interval [T,t] during which machine m is busy processing some part, we thus have the relation,

$$b_m(t) \;=\; b_m(T) + (t - T) \sum_{q,j} d_q \tau_{qm}^{(j)} - B_m(T, t) \text{ for } t \geq T. \tag{26}$$

Due to the nature of Clear-a-Fraction Policies, any part-type q on its j-th visit to machine m which is taken up for working at a time T_n larger than T must satisfy $x_{qm}(T_n) \leq \epsilon_m x_{pm}(T_n) \leq \Gamma \epsilon_m$. Hence the continuous production run of such parts of type q, for which a set-up is commenced at some time $T_n \geq T$, must last longer than $\Gamma \tau_{qm}^{(j)} \epsilon_m$. Hence if T_{n+1} is the time at which such a production run ends, then $B(T_n, T_{n+1}) \leq \Gamma \tau_{qm}^{(j)} \epsilon_m$, while the idle period, which is the time taken up for a set-up, is δ. Hence,

$$\frac{B(T_n, T_{n+1})}{T_{n+1} - T_n} \;\leq\; \frac{\Gamma \tau_q m^{(j)} \epsilon_m}{\Gamma \tau_{qm}^{(j)} \epsilon_m + \delta}.$$

Now suppose that Γ has been chosen so large that

$$\frac{\Gamma \tau_{qm}^{(j)} \epsilon_m}{\Gamma \tau_{qm}^{(j)} \epsilon_m + \delta} \;\leq\; \frac{1}{2} + \frac{1}{2} \sum_{(r,k)} d_r \tau_{rm}^{(k)}, \text{ for all } r, k,$$

which choice is feasible since the right hand side is < 1 due to (25), the stability condition. Hence,

$$B(T,t) \;\; \geq \;\; (t-T)\left(\frac{1}{2}+\frac{1}{2}\rho_m\right).$$

By (26) it then follows that $b_m(t) \leq b_m(T) - \frac{1}{2}(t-T)(1-\rho_m)$. Hence as $t \to \infty$, $b_m(t)$ converges to $-\infty$, while it can only be positive, providing a contradiction.

This shows that parts of type p on their visit to their first machine m do get an infinite number of production runs, and at the end of each run their buffer level $x_{pm}^{(1)}$ is reduced to zero. Now the result can be proved at the second and succeeding machines that part p visits by induction. Thus every part of type p eventually leaves the system.

However, it is not clear whether the buffer levels stay bounded, and it is an important open question whether

$$\sup_t x_{pm}^{(i)}(t) < +\infty \text{ for every } (p,m,i).$$

It is conceivable that there could be *dynamic instability*. By this we mean a scenario where a *long* production run of a certain part-type at a certain machine can temporarily *block* other part-types at this machine. This can then lead to *starvation* of parts at some downstream machines, leading to *short* production runs of the parts there. This results in much idle time being forced on these downstream machines due to frequent set-ups. However, when the long production run is completed, the huge buffer levels at the machine are cleared, dumping large quantities of parts on the downstream machines. These machines can then go into long production runs themselves, which can possibly cause a dangerous positive feedback effect since the system is not acyclic. We do not know whether such dynamic instability is possible under CAF policies.

However, in the next section, we will describe a slightly *modified* version of a Distributed CAF Policy which ensures stability.

7 Distributed CAF-Policies with Backoff

We now exhibit a class of policies which involve only a slight modification of Distributed CAF-Policies, but which can be proved to be stable.

Essentially, we will consider a policy which produces a part-type p only when it would have produced the same part-type if it had been in *isolation*, and its inputs had been strictly linear functions. Moreover, we show that this can be implemented in a feedback fashion by recursively updating a variable $z_p^{(i)}$ which keeps track of what the buffer level would have been if the machine had been in isolation.

Fix attention on machine m. We shall label parts of type p on their i-th visit to machine m as a part-type (p, i).

Let us now suppose that machine m is in *isolation*, i.e., we are back in the *single machine case* studied in Section 2. Thus the inflow of all parts of type (p, i) is a linear function td_p. Assume that a CAF-Policy is used at machine m, and let $\{T_n^{(p,i)}\}$ be the sequence of times at which the set-up for parts of type (p, i) is begun, $\{T_n^{(p,i)} + \delta\}$ the sequence of times at which the actual production of parts of type (p, i) is begun, and $\{\overline{T}_n^{(p,i)}\}$ the sequence of times at which the buffer levels of part-type (p, i) are cleared.

Now let us return to the case of interest where machine m is *not* in isolation, but is a part of the general manufacturing system. With $\{T_n^{(p,i)}, \overline{T}_n^{(p,i)}\}$ defined however from the *isolated* case, let us consider the following schedule:

> At machine m, during the time interval $[T_n^{(p,i)} + \delta, \overline{T}_n^{(p,i)}]$, produce parts of type (p, i) whenever possible, keeping it idle whenever it is starved of parts of type (p, i).

(Note that continuous production of parts of type (p, i) during the entire interval $[T_n^{(p,i)} + \delta, \overline{T}_n^{(p,i)}]$ may not be feasible if such parts have *not* been delivered to machine m from previous stages of production.)

As described above, this policy would seem to require the precomputation of the sequences $\{T_n^{(p,i)}\}$ and $\{\overline{T}_n^{(p,i)}\}$. However, a closed-loop real-time implementation of the same policy at each machine is as follows.

Time 0: Set $\zeta_0 := 0$ and $z_{pm}^{(i)}(\zeta_0) = 0$ for all (p,i).

Time ζ_n: Choose $(p^*(\zeta_n),\, i*(\zeta_n))$ such that

$$z_{p^*(\zeta_n)m}^{(i*(\zeta_n))}(\zeta_n) \;\; \geq \;\; \epsilon_m \sum_{(p,i)} z_{pm}^{(i)}(\zeta_n)$$

and compute,

$$\zeta_{n+1} := \zeta_n + \frac{z_{p^*(\zeta_n)m}^{(i*(\zeta_n))}(\zeta_n)\tau_{p^*(\zeta_n)m}^{(i*(\zeta_n))} + \delta}{1 - \tau_{p^*(\zeta_n)m}^{(i*(\zeta_n))} d_{p^*(\zeta_n)}} (\text{compare with (5)}).$$

At time ζ_n set up to produce $(p^*(\zeta_n), i*(\zeta_n))$ and produce it in the time interval $[\zeta_n + \delta, \zeta_n + 1]$ whenever possible, keeping the machine idle whenever production is not possible due to starvation of parts of type $(p^*(\zeta_n), i*(\zeta_n))$.

Update,

$$\begin{aligned} z_{pm}^{(i)}(\zeta_n + 1) \;\; &:= \;\; z_{pm}^{(i)}(\zeta_n) + (\zeta_n + 1 - \zeta_n) d_{pm} \text{ for } (p,i) \neq (p^*(\zeta_n), i*(\zeta_n)) \\ &:= \;\; 0 \text{ for } (p,i) = (p^*(\zeta_n), i*(\zeta_n)). \end{aligned}$$

Note that we now have a distributed, real-time scheduling algorithm, which we shall call the *Distributed, CAF-Policy With Backoff*. The following Theorem proves that it is stable.

Theorem 8: Stability of Distributed CAF-Policies With Backoff. *Distributed, CAF-Policies With Backoff are stable whenever the condition (25) is satisfied.*

Proof: Consider parts of type p flowing into their *first* machine m. Since the inflow is just the raw material, it is an "ideal" inflow, and so the outflow is also "ideal", by which

we mean that both coincide with the corresponding quantities in the *isolated* single machine case. Both are therefore bounded below by lines of slope d_p. But the outflow of such parts is just the inflow to the *second* machine, with possibly a bounded transportation delay, and so the actual inflow to the second machine is also bounded below by a line of slope d_p. Now the proof is completed by induction by showing that if the inflow of a part-type p to a machine m is bounded below by a line of slope d_p, as shown in Figure 7, then so is the outflow. To

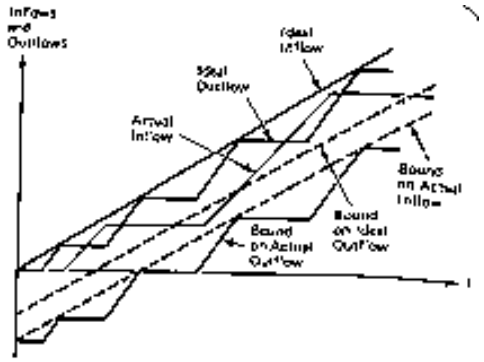

Figure 7: Inflows and Outflows at Machine m Under Distributed CAF Policy With Backoff.

see this consider the graph shown in Figure 7, which shows the ideal inflow, the ideal outflow and the actual inflow. Also shown is the line $(td_p - b)$ lower bounding the actual inflow. Consider now the curve obtained by translating downwards the ideal outflow graph by $-b$; call this curve C. We now claim that the actual outflow graph is bounded below by C. This is true because if the actual outflow graph ever hits C, then it will thereafter *coincide* with C for *all* future times, since the production periods of C exactly *coincide* with those of the ideal outflow graph, and moreover, since the graph C lies below the actual inflow graph, future starvations will never again occur. This proves that the actual outflow is bounded below by a line of slope d_p, thus proving the stability of the entire system.

It should be noted that upper bounds on buffer levels can be easily determined for *all* machines. These bounds will increase with the order of a machine in the path of a part. Moreover they are monotone increasing in the transportation delays. The calculations, being

straightforward but tedious, are omitted.

8 Assembly, Disassembly and Proportional Rerouting

We now come to the fully general case originally illustrated in Figure 1, where in addition to processing of parts at machines, there are also some part-types (such as $P1$ and $P2$ in Figure 1) which may need to be assembled (as at machine $M8$), and some part-types (such as $P3$) which may need to be disassembled (into part types $P3'$ and $P3''$ at machine $M2$), and some part-types (such as $P4$) of which a certain proportion (such as $P4''$, a proportion *alpha* of $P4$, after machine $M3$) may have to rerouted for reprocessing due to possible failure of quality inspections.

We shall exclusively consider Distributed CAF Policies With Backoff.

It is clear that disassembly and rerouting cause no additional problems beyond those considered in the previous section. In the case of disassembly, we just suppose that two part-types with different labels arise from each part disassembled. A point to note is that if part-type p is disassembled into two part-types p' and p'' at a certain machine, then we define $d_{p'} = d_{p''} = d_p$ for the demand rates of the part-types p' and p'', which are then used in the determination of their schedules at succeeding machines. These are then handled in the normal way at the succeeding machines to which they travel.

Similarly if a proportion of part-type p has a separate routing subsequent to a machine m, then we simply assume that we have two part types p' and p'' emerging from machine m, with demand rates $d_{p'} = \alpha d_p$ and $d_{p''} = (1 - \alpha)d_p$. These two part-types then retain their identities at subsequent machines and are scheduled for operation in the usual way.

Finally, let us consider assemblies. Suppose now that two part-types p' and p'' need to be assembled at machine m, and the demand rate for the composite part $(p', p'') =: p$ is d_p.

Then we set $d_{p'} = d_{p''} = d_p$ at machines prior to m, where they are handled in the usual way. However at machine m we treat them as constituting just *one* part-type p with buffer level defined by,

$$x_{pm}(t) \;\; := \;\; \min(x_{p'm}(t), x_{p''m}(t))$$

where $x_{p'm}$ and $x_{p''m}$ are their separate buffer levels. Then the Distributed CAF Policy With Backoff is defined with x_{pm} replacing both $x_{p'm}$ and $x_{p''m}$.

In order to see that the resulting scheduling policy is stable, we simply note that just as in the proof of Theorem 8, the inflows $u_{p'm}(t)$ and $u_{p''m}(t)$ can be shown to be bounded below by separate lines each of slope $d_{p'} = d_{p''}$. Hence $u_{pm}(t) := \min(u_{p'm}(t), u_{p''m}(t))$ is also so bounded, and the proof of Theorem 8 at machine m shows that the *outflow* of part-type p, $y_{pm}(t)$, is also bounded below by a line of slope d_p. Therefore, $x_{pm}(t) = u_{pm}(t) - y_{pm}(t)$ is bounded. Hence $\min(x_{p'm}(t), x_{p''m}(t))$ is bounded. Now noting that $x_{p'm}(t) - x_{p''m}(t) = u_{p'm}(t) - u_{p''m}(t)$ we see that $|x_{p'm}(t) - x_{p''m}(t)|$ is bounded since $td_p \geq u_{p'm}(t) \geq td_p - b'$ and $td_p \geq u_{p''m}(t) \geq td_p - b''$ for some constants b' and b''. Hence both $x_{p'm}(t)$ and $x_{p''m}(t)$ are bounded and so we have the following Theorem.

Theorem 9: Stability of General System Under Distributed CAF Policies With Backoff. *The general manufacturing/assembly/disassembly system with possible rerouting of proportions is stable when the Distributed CAF Policy With Backoff is employed, and the condition (25) is satisfied.*

9 Concluding Remarks

We have provided scheduling policies of the Clear-a-Fraction type which are implementable in a real-time distributed fashion at the machines of a general flexible manufacturing/assembly/disassembly system. These policies stabilize the entire system in that they produce all part-types at the desired rates while using only finite (and computable) buffer

sizes at the various machines. Our model allows for various complexities such as the differing processing times required by the different part-types at the various machines, differing routings of parts through the system, variable transportation delays and arbitrary set-up times. Moreover the scheduling policies achieve stability for *all* stabilizable systems, i.e., whenever the necessary condition for stability is met. We have also provided upper bounds on the buffer levels for our policies as well as lower bounds on the achievable average buffer levels for *any* policy.

There are however several open questions that need answering, and several directions in which future work can proceed. Foremost among these is that we do not know whether a Clear-a-Fraction Policy implemented at each machine (but without the "Backoff" modification introduced here) will stabilize non-acyclic systems.

On another note, our treatment has been happily deterministic, but it would be of interest to incorporate the vagaries of uncertainty such as random machine failures, random yields and random demands into our model.

It would also be of interest to examine notions such as "push" and "pull" for our class of models, to determine whether other and perhaps more efficient classes of policies can be designed.

In any case, we see our work as only a starting point for several iterations which will, we hope, lead ultimately to a tractable theory of dynamic scheduling for manufacturing systems.

References

[1] S. B. Gershwin, "A hierarchical framework for discrete-event scheduling," in *Manufacturing Systems* (P. Varaiya and A. B. Kurzhanski, eds.), no. 103 in Lecture Notes in Control and Information Sciences, Springer-Verlag, 1987. Presented at the IIASA Workshop on Discrete Event Systems: Models and Applications, Sopron, Hungary.

[2] K. Baker, *Introduction to Sequencing and Scheduling.* New York, NY: John Wiley and Sons, 1974.

[3] R. W. Conway, W. L. Maxwell, and L. W. Miller, *Theory of Scheduling.* Reading, MA: Addison-Wesley, 1967.

[4] S. French, *Sequencing and Scheduling.* New York, NY: John Wiley and Sons, 1982.

[5] E. G. Coffman, *Computer and Job-Shop Scheduling Theory.* New York, NY: John Wiley and Sons, 1976.

[6] S. S. Panwalker and W. Iskander, "A survey of scheduling rules," *Operations Research*, vol. 25, pp. 45–61, January-February 1977.

[7] S. C. Graves, "A review of production scheduling," *Mathematics of Operations Research*, vol. 29, pp. 646–675, July-August 1981.

[8] J. K. Lenstra and A. H. G. Rinnooy Kan, "Sequencing and scheduling." Amsterdam, The Netherlands. Preprint, 1984.

[9] S. Elmaghraby, "The economic lot scheduling problem (ELSP): Review and Extensions," *Management Science*, vol. 24, pp. 587–598, 1978.

[10] J. Blazewicz and et. al, "Scheduling under resource constraints-deterministic models," *Annals of Operations Research*, vol. 7, 1986. J.C. Baltzer.

[11] G. Weiss, "Multiserver stochastic scheduling." pp. 157–179, in *Deterministic and Stochastic Scheduling*, M.A.H. Dempster, Editor, D. Reidel Publishing Co., 1982.

[12] R. Weber, P. Varaiya, and J. Walrand, "Scheduling jobs with stochastically ordered processing times on parallel machines to minimize expected flowtime," *Journal of Applied Probability*, vol. 23, pp. 841–847, 1986.

[13] J. Kimemia and S. B. Gershwin, "An algorithm for the computer control of a flexible manufacturing system," *IIE Transactions*, vol. 15, pp. 353–362, December 1983.

[14] S. B. Gershwin, R. Akella, and Y. F. Choong, "Short-term production scheduling of an automated manufacturing facility," *IBM Journal of Research and Development*, vol. 29, pp. 392–400, July 1985.

[15] R. Akella and P. R. Kumar, "Optimal control of production rate in a failure prone manufacturing system," *IEEE Transactions on Automatic Control*, vol. AC-31, pp. 116–126, February 1986.

[16] T. Bielecki and P. R. Kumar, "Optimality of zero-inventory policies for unreliable manufacturing systems," *Operations Research*, vol. 36, pp. 532–541, July-August 1988.

[17] A. Sharifnia, "Production control of a manufacturing system with multiple machine states," *IEEE Transactions on Automatic Control*, vol. AC-33, pp. 600–626, July 1988.

[18] W. Fleming, S. Sethi, and M. Soner, "An optimal stochastic production planning problem with randomly fluctuating demand," *SIAM Journal on Control and Optimization*, vol. 25, pp. 1494–1502, November 1987.

[19] D. Bertsekas and R. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice-Hall, 1987.

[20] P. P. Varaiya, *Notes on Optimization.* New York, Van Nostrand, 1972.