Distributed Scheduling of Flexible Manufacturing Systems: Stability and Performance*

James R. Perkins†, Carlos Humes, Jr.‡, and P. R. Kumar§

Abstract

We consider a manufacturing system producing several part–types on several machines. Raw parts are input to the system. Each unit of a given part–type requires a predetermined processing time at each of several machines, in a given order. A set-up time is required whenever a machine switches from processing one part–type to another.

For a single machine system with constant demand rates, we present a class of Generalized Round Robin scheduling policies for which the buffer level trajectory of each part–type converges to a steady state level. Furthermore, for all small initial conditions, we show that these policies can be Pareto–efficient with respect to the buffer sizes required. Allowing the input streams to have some burstiness, we derive upper bounds on the buffer levels for small initial conditions.

For non–acyclic systems, we consider a class of policies which are stable for all inputs with bounded burstiness. We show how to employ system elements, called regulators, to stabilize systems. Using the bounds for the single machine case, we analyze the performance of regulated systems implementing Generalized Round Robin scheduling policies.

1 Introduction

We investigate the stability and performance of scheduling policies for manufacturing systems described by the following features:

*The research reported here has been partially supported by the National Science Foundation under Grant Nos. ECS-90-25007 and ECS-92-16487, the Joint Services Electronics Program under Contract No. N00014-90-J1270, Conselho Nacional de Pesquisas-CNPq(Brazil), and IBM-Brasil.

†Dept of Electrical and Computer Engg, and Coordinated Science Laboratory, University of Illinois, 1308 W. Main St., Urbana, IL 61801.

‡Depto. de Ciencia da Computacao, IME , Univ. of S. Paulo, Brazil.

§Dept of Electrical and Computer Engg, and Coordinated Science Laboratory, University of Illinois, 1308 W. Main St., Urbana, IL 61801.

(i) There are P part–types labeled $1, 2, \ldots, P$, and a set $\mathcal{M} = \{1, \ldots, M\}$ of M machines.

(ii) Parts of type p require processing at machines $\mu_{p,1}, \mu_{p,2}, \ldots, \mu_{p,n_p}$, in that order, where $\mu_{p,i} \in \mathcal{M}$.

(iii) Raw parts of type p arrive to the system at machine $\mu_{p,1}$. Denote by $u_p(t)$ the cumulative number of raw parts of type p which have entered the system in the time interval $[0, t]$.

(iv) At the i^{th} machine they visit, parts of type p enter a buffer labeled $b_{p,i}$, from which they are eventually processed at rate $\tau_{p,i}^{-1}$ by machine $\mu_{p,i}$.

(v) Machine m serves the buffers $B_m := \{b_{p,i} : \mu_{p,i} = m\}$. A set-up time $\delta_{p,p'}$ is incurred when machine m switches from processing parts in buffer $b \in B_m$ to processing parts in buffer $b' \in B_m$.

For such a system, let $x_{p,m}(t)$ denote the level of buffer $b_{p,m}$ at time t, and let $y_{p,m}(t)$ denote the cumulative output of part–type p from buffer $b_{p,m}$ in $[0, t]$. Thus,

$$x_{p,m}(t) = x_{p,m}(0) + y_{p,m-1}(t) - y_{p,m}(t), \tag{1}$$

with $y_{p,0}(t) := u_p(t)$,

A scheduling policy is said to be *stable for machine* m if for every initial condition $\{x_{p,i}(0)\}$, there exists a finite constant L_m, which is initial condition dependent, such that

$$x_{p,i}(t) \leq L_m < \infty \text{ for all } t \geq 0,$$

for all (p, i) such that $\mu_{p,i} = m$. Thus, a scheduling policy that is stable for machine m ensures that the levels of all of the buffers at the machine m are bounded. A machine will be said to be *stable* when it is operating under such a scheduling policy. For a multi–machine system, a scheduling policy is stable if it is stable for every machine m. Note from (1) that if a scheduling policy is stable, then the output of parts of each type from the system deviates from the input by no more than a constant.

In [1], the single machine system has been studied for the class of *linear deterministic* inputs, i.e., $u_p(t) = t\, d_p$. Also, a class of stable scheduling policies was presented, and for

every policy in this class, a lower bound on the average weighted sum of buffer levels was determined.

In order to study systems with more than one machine, it is useful to consider inputs which only satisfy,

$$\sup_t |u_p(t) - t\, d_p| \leq M_p < \infty. \tag{2}$$

Inputs of this type are included in the class of inputs with *bounded burstiness*, which are defined as those for which there exists a constant γ_p such that,

$$u_p(t) - u_p(s) \leq d_p\,(t-s) + \gamma_p \text{ for all } t \geq s \geq 0. \tag{3}$$

It should be noted that the class of bounded burstiness inputs, is one of the few classes for which bounds on important design parameters, such as buffer levels, can be determined. The use of bounded burstiness inputs has been introduced in Cruz [2, 3]. The results presented here will be derived for systems with linear deterministic and bounded burstiness inputs.

Let $\rho_{p,i} := d_p \tau_{p,i}$ be the load factor imposed by part–type p on the machine at the i-th stage of its route. An important role is played by the load factor ρ_m at machine m, defined by

$$\rho_m := \sum_{\{(p,i):\mu_{p,i}=m\}} \rho_{p,i}.$$

Note that ρ_m is the fraction of time which machine m must work in order to ensure stability. Thus, under linear deterministic or bounded burstiness inputs a necessary condition for stability of any scheduling policy is

$$\rho_m < 1, \text{ for } m = 1, \ldots, M. \tag{4}$$

Throughout the paper, we will assume that the condition (4) is satisfied.

Notice that, under linear deterministic inputs, if all of a machine's set-up times are zero, then once the machine empties its buffers it could keep them forever empty by infinitesimally dividing its attention among its buffers. This is done by dedicating a fraction $\rho_{p,i}$ of its capacity to buffer $b_{p,i}$. We shall therefore consider the non-trivial and interesting case where, at each machine, at least one set-up time is non-zero.

When set-up times are non-zero, it is not true that every non–idling policy (i.e., a policy where each machine works on some buffer at its maximal rate, as long as not all its buffers are empty) will lead to stability, even for a single machine. This is trivially seen by considering a system with three or more part–types. If set-ups are required between all pairs of part–types, then the policy which processes the part–type with the smallest buffer level can lead to instability.

Although we shall primarily be concerned with non-zero set-up times and inputs which have a continuous flow, we shall also obtain several results for discrete inputs and for zero set-up times.

2 Stable Policies And Acyclic Systems

Consider any scheduling policy for a single machine system, that stabilizes the single machine under bounded burstiness inputs. We will say that such a policy is *stable in isolation* (SI). Now consider a distributed policy for a multi–machine system, where every machine implements a stable in isolation scheduling policy. We will call the resulting overall policy a *distributed stable in isolation* (DSI) policy.

Definition: Clearing Policy

A scheduling policy is *clearing* if *all* of its production runs are clearing, i.e., if machine m is processing parts from buffer $b \in B_m$, then it will continue to process parts from buffer b until the buffer is empty (and some other buffer $b' \in B_m$ is non-empty).

A significant subclass of clearing policies, is the class of Clear–A–Fraction (CAF) policies introduced in [1].

Definition: Clear–A–Fraction (CAF) Policies

A CAF policy is a clearing policy with the property that for each machine m, there exists an $\epsilon_m > 0$, and an e_m such that, if the machine commences a set-up to buffer $b_{p,i} \in B_m$ at time t, then

$$x_{p,i}(t) \geq \epsilon_m \sum_{\{(q,k):\mu_{q,k}=m\}} x_{q,k}(t) - e_m. \tag{5}$$

The class of CAF policies, contained in the class of SI policies, includes many important subclasses of scheduling policies which can be implemented in real–time with very little computation. For example, clear the largest buffer, clear the buffer with the largest workload, and Round Robin, can all be shown to be CAF policies (see [1, 4]).

For a single machine system, the stability of CAF policies is derived from the fact that the system workload, $W(t) := \sum_p \tau_p x_p(t)$, plays a role similar to that of a Lyapunov function. When it is large, it can be shown to decrease.

By analyzing the behavior of $W(t)$, a bound $\bar{W}$ on $W(t)$ is provided in [1]. It has been refined by Lou, Sethi, and Sorger [5] for CFW (clear a fraction of the work) policies, which are equivalent to CAF policies. It should be noted that although upper bounds obtained for the workload can be transformed into bounds on the required buffer capacity, these bounds can be unnecessarily high.

Let us define the *connection graph of the system* as a directed graph, where the nodes are the machines, and the arc set A is,

$$A := \{(m, m') | m \neq m' \text{ and there exists } (p, l) \text{ such that } \mu_{p,l} = m, \mu_{p,l+1} = m'\}.$$

It can be verified easily that the ordering,

$$m \prec m' \iff \text{ there exists a path in the connection graph from } m \text{ to } m',$$

is a partial order for A. Moreover, $m \prec m' \Rightarrow m' \not\prec m$ if and only if the connection graph has no directed cycles. A system whose connection graph has no directed cycles will be called an *acyclic* system. Otherwise, it will be called a *non–acyclic* system.

The following theorem shows that DSI policies are stable for acyclic systems.

Theorem 1 *DSI policies are stable for acyclic systems with bounded burstiness inputs.*

Proof: The proof is by induction. Define $\mathcal{M}_i$ to be the set of minimal elements of $\mathcal{M} \backslash \mathcal{M}_{i-1}$ under the partial order "$\prec$" induced by the system structure, where $\mathcal{M}_0 := \phi$. Every machine $m \in \mathcal{M}_1$ is stable since all of its inputs come from outside of the system, and it is effectively in isolation. Hence the outputs from m are also of bounded burstiness.

Consider $m' \in \mathcal{M}_2$. All of its inputs come from outside the system or are the outputs of machines in $\mathcal{M}_1$, and hence of bounded burstiness. Thus, viewing the machine as though in isolation, and using the definition of SI policies, all machines $m' \in \mathcal{M}_2$ will be stable. Applying this argument recursively completes the proof. □

In [1] it has been shown that CAF policies are also stable when parts incur bounded transportation delays when moving from buffer to buffer. Such a property also holds for DSI policies, since the bounded transportation delays merely add to the burstiness in flows, while preserving their bounded burstiness property.

3 Instability In Non-Acyclic Systems

Although DSI policies stabilize acyclic systems, they do not, in general, stabilize non–acyclic systems. This fact was demonstrated in [6] for the systems shown in Figures 1 and 2. For these systems, all clearing policies lead to instability, even though the capacity constraints (4) are obeyed, when

$$\tau_1 < \tau_2 \text{ and } \tau_3 < \tau_4, \text{ but } d_1/\tau_1 + d_1/\tau_2 > 1,$$

and $d_1 = d_2$ for the system in Figure 2.

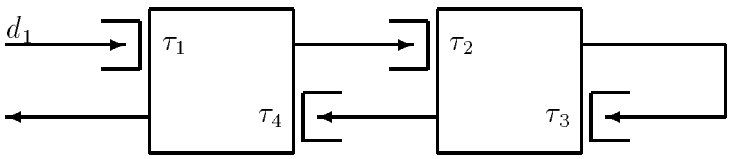

Figure 1: A two machine, one part–type non–acyclic system.

The instability is not only under continuous part flow but also if parts are discrete. For example, the system in Figure 1 is unstable for $d_1 = 1, \tau_1 = \tau_3 = 0$, and $\tau_2 = \tau_4 = 2/3$. For an example of an unstable system with $\tau_i \neq 0$, see Tang and Shi [7].

In the above systems, the fact that the service times are increasing along routes, i.e., $\tau_1 < \tau_2$ and $\tau_3 < \tau_4$, may lead to the false conjecture that under decreasing service times

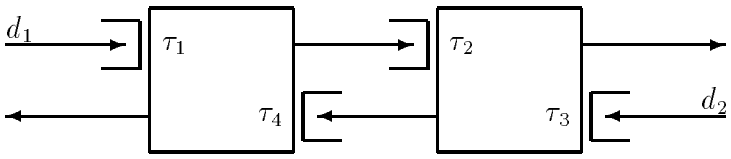

Figure 2: A two machine, two part–type non–acyclic system.

along the path of any part–type, CAF policies (or DSI policies) are stable. Recently, it has been shown that, under bounded burstiness inputs, this is not the case (see [8]). The example in [8] is a variation of the system in Figure 2, with each part–type requiring service three times, twice at the second machine which it visits. There, the CAF policy is unstable if $2\,d\,(\tau_2 + \tau_3) > 1$, where $d_1 = d_2 =: d$, $\tau_{1,j} = \tau_{2,j} =: \tau_j$, for $j = 1, 2, 3$, $\tau_1 > \tau_2 > \tau_3$, and $d\,(\tau_1 + \tau_2 + \tau_3) < 1$.

Qualitatively, the instability results from the fact that the inputs for the buffers $b_{p,i}$ for $i \geq 2$, may have instantaneous growth rates temporarily exceeding machine capacity. This leads to extremely long production runs at one buffer, which blocks the other buffers, starving the other machines. Due to the cycle in the connection graph, there can be a "positive" feedback effect resulting in alternating periods of overflow and starvation, which grow without bound, thus resulting in instability.

Since CAF policies are not stable for some non–acyclic systems, it is of interest to determine necessary and sufficient conditions which will ensure that CAF policies will be stable.

Sufficient conditions for stability of non–acyclic systems have been determined in [6]. Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_K$ be the diconnected components of the connection graph, i.e., for all $(m, m') \in \mathcal{M}_k$ there is a directed path from m to m' and vice-versa. For a buffer $b_{p,i}$ with $\mu_{p,i} \in \mathcal{M}_k$, we shall define $d'_{p,i}$ as equal to d_p if the predecessor machine $\mu_{p,i-1}$ is not in $\mathcal{M}_k$, and equal to $\tau^{-1}_{p,i-1}$ if $\mu_{p,i-1}$ is not in $\mathcal{M}_k$. Let us say that a machine m is *conditionally burst stable* if,

$$\sum_{\{(p,i):\mu_{p,i}=m\}} d'_{p,i}\tau_{p,i} < 1.$$

Note that this is a stronger requirement than the capacity condition (4).

Theorem 2 *If every machine is conditionally burst stable, then the system is stable under any DSI policy.*

Proof: Let $\mathcal{M}_1$ be a minimal diconnected component, i.e., one for which there are no directed paths from any machine not in $\mathcal{M}_1$ to a machine in $\mathcal{M}_1$. Since every machine in $\mathcal{M}_1$ has a load factor of less than one, all machines in $\mathcal{M}_1$ have bounded buffers under any DSI policy. Since the machines in $\mathcal{M}_1$ are stable, they have outputs with bounded burstiness. Thus, the inputs are of bounded burstiness to the machines in the diconnected components which only have inputs from the minimal diconnected components or from outside the system. Stability follows by induction on the acyclic graph (with no directed cycles) defined by the partial order on the diconnected components. □

This conditional burst stability sufficient condition for stability has been enlarged in [9], but more work remains to be done.

4 Stabilizing Techniques

In the previous section we showed that, for some non–acyclic systems, DSI policies can lead to instability. Now we present stabilizing techniques to modify any DSI policy such that it is stable for *any* system.

First we note that the CAF Policy With Backoff of [1] offers part–types ordered time slices, with length determined by simulating the working of each machine as though it were in isolation. This leads to bounded production runs and thus stability. However, the stability is achieved at the price of possible enforced idleness and thus possibly unnecessarily large buffer levels.

Following the same idea of bounded production runs, [6] introduced Universally Stabilizing Supervisory Mechanisms (USSM). Under a USSM, when a buffer level exceeds a given threshold, the buffer joins a priority queue. All buffers in the priority queue are given a bounded time slice, which is proportional to the load factor they impose, and large enough to offset the idle–time lost in switching between part–types. If, even after processing, a buffer in the priority queue still has a level above the threshold, then it returns to the prior-

ity queue. This ensures stability without forced idleness, though of course some time is lost due to repeated switchings.

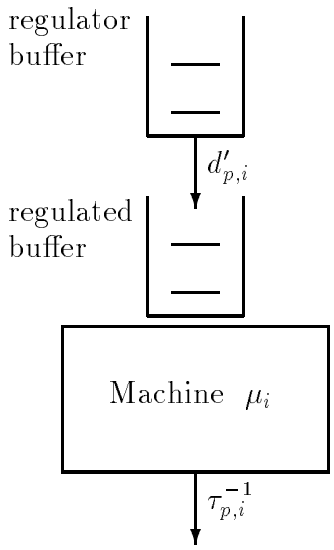

Figure 3: A regulator buffer constrains the input of parts available to machine μ_i.

We will consider a third approach, which implements system elements called *regulators*; see also Cruz [3]. Such regulators can be implemented as in Figure 3, by splitting the buffer into two (virtual) components, a *regulator* buffer and a *regulated* buffer. Only parts in the regulated buffer are available for processing by the machine. The flow of parts into the regulated buffer from the regulator buffer is constricted as follows:

(R.i) The cumulative input $u'_{p,i}(t)$ to the regulated buffer satisfies,

$$u'_{p,i}(t) - u'_{p,i}(s) \leq d'_{p,i}\,(t-s) + \gamma'_{p,i}.$$

(R.ii) If buffer $b_{p,j}$ precedes $b_{p,i}$, then $d_p \leq d'_{p,j} \leq d'_{p,i}$.

(R.iii) For every machine m, the rates $d'_{p,i}$ satisfy a capacity constraint, i.e.,

$$\rho'_m := \sum_{(p,i)\in B_m} d'_{p,i}\,\tau_{p,i} < 1.$$

Note that one choice of the regulator buffer speeds $d'_{p,i}$ which meets the above conditions is $d'_{p,i} \equiv d_p$. We note that the regulators can be implemented merely by using pointers to mark the dividing line between the regulator and regulated buffers.

Theorem 3 *If regulators satisfy $(R.i) - (R.iii)$, then the system will be stable for all DSI policies with bounded burstiness inputs.*

Proof: This proof is a generalization of that in Humes [9] to the case where the $\gamma_{p,i}$'s are not restricted to be 0. The inductive proof capitalizes on the ideas underlying the concept of conditional burst stability. □

It is interesting to observe that the sufficient condition of Theorem 2 corresponds to verifying that the system is automatically self-regulated. This is due to the fact that since the output of one machine feeds another machine in the same diconnected component, the maximum feeding speed is a speed for a regulator which would guarantee stability. For machines in different diconnected components, the stability of the previous component is equivalent to a non-smooth regulator $(d_p, \gamma_{p,i})$. In [9], it is also shown that CAF policies with backoff and USSM's can both be treated as particular cases of totally regulated systems, where every buffer is regulated.

The above idea of using regulators to stabilize systems is due to Humes [9]. There, however, only *smooth* regulators are considered, i.e., $\gamma'_{p,i} \equiv 0$ for all (p, i) pairs. A difficulty with restricting the regulators to be smooth is that this can result in an underutilization of the capacity of a machine. For example, if the physical buffer is not empty, but the regulated buffer is empty, then the machine can only process at the rate at which it receives input from the regulator buffer, which may be less than the maximum feasible processing rate of the machine. Smooth regulators are also analyzed in a different context by Cruz [2].

The use of non-smooth regulators (especially if $\gamma'_{p,i}$ is variable) helps to partially offset the problem of underutilization, and also allows for increased versatility in the control of the system.

Simulations indicate that the use of regulators can be very efficient in terms of the total number of parts in the system.

Under inputs corresponding to discrete part flow, some *buffer priority policies* that ensure stability are provided in [10]. Buffer priority policies are those for which all the buffers at a machine are ordered, and the machine provides service to the first nonempty buffer in the ordering. For an overview, the reader is referred to [11].

5 Generalized Round Robin Policies

Let us consider a single machine system. We will consider a class of policies, called Generalized Round Robin policies, which turn out to be a subclass of CAF policies.

Definition: Generalized Round Robin (GRR)

A GRR policy is defined by its cycle length which is an integer N ($\geq P$), and a function $i : \{1, 2, \ldots, N\} \to \{1, 2, \ldots, P\}$ such that

(i) Service is provided to part–types repetitively in the order $(i_1, \ldots, i_N)$, where $i_j := i(j)$. If the chosen buffer is not already empty, it is processed at its maximum service rate, until the buffer is cleared.[1]

(ii) The function $i(\cdot)$ is onto, i.e., every part–type is processed at least once in each cycle.

Thus, a GRR scheduling policy is an open-loop clearing policy which chooses part–types for production according to a predetermined, fixed, repetitive schedule. A special case of a GRR policy is the well known Round Robin (RR) policy, which has $N = P$ above.

Let us study the behavior of GRR policies for a single machine with linear deterministic inputs. It will be assumed that $\delta_{p,p'}$, the set-up time incurred when changing from processing parts of type p to parts of type p', is the same for all (p, p') pairs; we will denote it by $\delta > 0$. It should be noted that it is not difficult to extend our analysis to general set-up times, some of which, but not all, can be zero.

[1] As defined here, the GRR policy is a clearing policy Hence, if $i_{j+1} = i_j$, then the $(j+1)^{th}$ time slice will have null duration, and then it is not a CAF policy. However, there is no loss of generality in assuming that $i_{j+1} \neq i_j$. An extension of the GRR policy would keep the buffer level at 0 for some time at the end of each production run by continuing to work on the buffer at the reduced rate matching the input of parts (see, for example, Chase and Ramadge [12]). This results in a sometimes idling policy. The results presented here can be extended to that case.

Let $\{i_1, \ldots, i_N\}$ describe a GRR policy. Define

$$t_j^{k+1} := \text{ length of the } j^{th} \text{ production run in the } k^{th} \text{ cycle of the GRR policy.}$$

That is, t_j^{k+1} is the $(k\,N + j)^{th}$ overall production run. Let $\sum_{\{p,i\}:\mu_{p,i}=m} \tau_{p,i} x_{p,i}(t)$ be the "work" for machine m contained in its buffers. When part–type $i_j = p$ is being processed, work is cleared at rate $(1 - \rho_p)$ units per unit of time; while, when part–type p is not being processed, work for part–type p enters at rate ρ_p units per unit of time. Thus, if the j^{th} time slice of cycle k is the n^{th} production run of part–type $i_j = p$, then

$$\begin{aligned} t_j^{k+1} &= \frac{\rho_p}{1-\rho_p} \cdot (\text{time between end of } n^{th} \text{ and start of } (n+1)^{th} \text{ production runs of part–type } p) \\ &= \rho_p (t_j^{k+1} + \text{time between end of } n^{th} \text{ and start of } (n+1)^{th} \text{ production runs of part–type } p). \end{aligned}$$

The time which has elapsed since the end of the previous production run depends on whether the previous run was during cycle k or during cycle $k-1$. Thus, there are two cases.

Case I : Suppose $l < j$ and $i_l = p, i_{l+1} \neq p, \ldots, i_{j-1} \neq p, i_j = p$. Then

$$t_j^{k+1} = \rho_p [(j - l)\,\delta + \sum_{m=l+1}^{j} t_m^{k+1}]\,. \tag{6}$$

Case II : Suppose $l \geq j$ and $i_l = p, i_{l+1} \neq p, \ldots, i_N \neq p, i_1 \neq p, \ldots, i_{j-1} \neq p, i_j = p$. Then

$$t_j^{k+1} = \rho_p [(j + N - l)\,\delta + \sum_{m=l+1}^{N} t_m^{k} + \sum_{m=1}^{j} t_m^{k+1}]\,. \tag{7}$$

The recurrence equations (6) and (7) define $\{t_j^k\}$ for all $j = 1, \ldots, N$, and $k = 1, 2, \ldots$, given $\{t_1^1, t_2^1, \ldots, t_N^1\}$ or, equivalently, $\{x_1(0), \ldots, x_P(0)\}$. These equations can be represented in matrix form as

$$F\; t^{k+1} \;=\; G\; t^k + c, \tag{8}$$

where

$$\begin{aligned} t^k &= [t_1^k \; t_2^k \; \ldots \; t_N^k]^T, \\ c &= [c_1 \; c_2 \; \ldots \; c_N]^T, \text{ with } c_i > 0, \end{aligned}$$

and the matrices F and G are lower triangular and upper triangular, respectively. Specifically, if f_{ij} is the element of F in row i and column j, then

$$f_{ij} = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i < j, \end{cases}$$

and $f_{ij} \leq 0$ for $i > j$. Also, $g_{ij} = 0$ for $i \geq j$, and $g_{ij} \geq 0$ for $i < j$.

Lemma 1 *Let $\mathcal{F}^n$ be the class of all lower triangular $n \times n$ matrices F_n with all diagonal elements equal to 1, and all elements below the diagonal nonpositive. If $F_n \in \mathcal{F}^n$, then F_n^{-1} exists, and $F_n^{-1} \geq 0$, in the sense that all of its elements are nonnegative.*

Proof: The proof is by induction. It is clearly true for the singleton set $\mathcal{F}^1 = \{[1]\}$. Assume the lemma is true for $\mathcal{F}^{n-1}$. For any $F_n \in \mathcal{F}^n$,

$$F_n = \begin{bmatrix} F_{n-1} & 0 \\ f & 1 \end{bmatrix},$$

where $F_{n-1} \in \mathcal{F}^{n-1}$ and $f = [f_1 \; f_2 \; \ldots \; f_{n-1}]$, with $f_i \leq 0$. Using a standard matrix inversion lemma (see, for example, Kailath [13], page 656),

$$F_n^{-1} = \begin{bmatrix} F_{n-1}^{-1} & 0 \\ g & 1 \end{bmatrix},$$

where $g = -f F_{n-1}^{-1}$. Since $f_i \leq 0$, $-f F_{n-1}^{-1}$ has all nonnegative elements and $F_n^{-1} \geq 0$. □

It follows from Lemma 1 that F^{-1} exists and $F^{-1}G \geq 0$ (since $G \geq 0$). Thus, defining $A := F^{-1}G$ and $b := F^{-1}c$, (8) becomes

$$t^{k+1} = A\, t^k + b, \tag{9}$$

where $A \geq 0$ and $b > 0$.

If the sequence $\{t^k\}$ is convergent to $\bar{t}$, then the elements of $\bar{t}$ also satisfy (6) and (7) in steady state, i.e.,

$$\bar{t}_j = \rho_p[(j - l)\,\delta + \sum_{m=l+1}^{j} \bar{t}_m]\,. \tag{10}$$

for $l < j$ and $i_l = p, i_{l+1} \neq p, \ldots, i_{j-1} \neq p, i_j = p$, and

$$\bar{t}_j = \rho_p[(j + N - l)\,\delta + \sum_{m=l+1}^{N} \bar{t}_m + \sum_{m=1}^{j} \bar{t}_m], \tag{11}$$

for $l \geq j$ and $i_l = p, i_{l+1} \neq p, \ldots, i_N \neq p, i_1 \neq p, \ldots, i_{j-1} \neq p, i_j = p$. These equations can be represented in matrix form as

$$\bar{t} = \bar{A}\,\bar{t} + \bar{b}, \tag{12}$$

(i.e., from (10), the j^{th} row of $\bar{A}$ is $(0, \ldots, \rho_p, \rho_p, \ldots, \rho_p, 0, \ldots, 0)$, etc.).

Lemma 2 *In equation (12),*

(i) $\bar{A}$ has all nonnegative elements, and $\bar{b}$ has all positive elements.

(ii) The eigenvalues of $\bar{A}$ satisfy $\max_i |\lambda_i(\bar{A})| < 1$.

Proof: Note that each element of $\bar{A}$ is either 0 or $\rho_p > 0$ for some p. Also, since all the set-up times are positive, $\bar{b}$ has all positive elements. Define $\lambda_0 := \max_i |\lambda_i(\bar{A})|$, $\sigma_{min} :=$ the smallest column sum of $\bar{A}$, and $\sigma_{max} :=$ the largest column sum of $\bar{A}$. Then, from the theory of nonnegative matrices (see [14] page 194), it follows that λ_0 is real and nonnegative, and

$$\sigma_{min} \leq \lambda_0 \leq \sigma_{max}. \tag{13}$$

By inspection, *every* column sum of $\bar{A}$ equals

$$\sum_{p=1}^{P} \rho_p := \rho. \tag{14}$$

Since $\rho < 1$, it follows that $\lambda_0 \leq \rho < 1$. □

Lemma 3 *If each element of an $n \times n$ matrix A is nonnegative ($A \geq 0$) and each element of an n-vector b is strictly positive ($b > 0$), the following two conditions are equivalent:*

(i) $|\lambda_i(A)| < 1$ for all i;

(ii) There exists a vector unique $\bar{t} \geq 0$ satisfying $\bar{t} = A\,\bar{t} + b$,

where $\lambda_i(A)$ is the i^{th} eigenvalue of A.

Proof: These are standard results for nonnegative matrices (see, for example, Luenberger [14] Theorem 1, page 196). □

The following theorem proves that the sequence of production run times $\{t^k\}$ is convergent to $\bar{t}$.

Theorem 4 *Given any initial conditions, for a single machine operating under any GRR scheduling policy, the buffer level trajectory of each part–type will converge to a unique periodic trajectory, with steady state production run lengths given by the vector $\bar{t}$.*

Proof: Applying Lemmas 2 and 3 shows that there exists a unique vector $\bar{t} \geq 0$ which satisfies

$$\bar{t} = \bar{A}\,\bar{t} + \bar{b}.$$

Since (6) and (7) are equivalent to (9), $\bar{t}$ is also the unique solution to

$$\bar{t} = A\,\bar{t} + b.$$

Applying Lemma (3) again (using $(ii) \Rightarrow (i)$ now) yields $\max_i |\lambda_i(A)| < 1$. Therefore,

$$\lim_{k \to \infty} t^k = \bar{t} \geq 0,$$

where $\bar{t}$ is uniquely given by

$$\bar{t} = (I - \bar{A})^{-1}\,\bar{b}.$$

□

Thus, for any GRR policy, $\bar{t}$ can be explicitly computed by solving a set of N linear equations. The vector $\bar{t}$ describes the asymptotic behavior of the system.

The following interesting corollary to Theorem 4 allows a uniform upper bound on buffer levels, for all small initial conditions.

Corollary 1 *If, for any l, $t^l \leq \bar{t}$ componentwise, then, for all $k \geq l$,*

$$t^k \leq \bar{t}.$$

Proof: Define $r^l := \bar{t} - t^l$. Then, from (9),

$$t^{l+1} = A\,(\bar{t} - r^l) + b = \bar{t} - A\,r^l \leq \bar{t},$$

since A and r^l both have nonnegative elements. □

If the initial condition $x(0) = (x_1(0), \ldots, x_P(0))^T$ is such that $t^1 \leq \bar{t}$, let us say that we are in the *small initial condition* case.

Corollary 2 *Under a GRR policy, for small initial conditions, the maximum buffer size required for each part–type is bounded above by*

$$\|x_p(\cdot)\|_\infty \leq \frac{1-\rho_p}{\tau_p} \max_{\{j\,:\,i_j=p\}} \bar{t}_j.$$

Proof: Due to the small initial conditions, the largest buffer levels are attained in steady state. In the steady state, each production run for part–type $i_j = p$ has a negative slope of $(\tau_p^{-1} - d_p)$ and length $\bar{t}_j$. The result then follows by maximizing over all runs for the part–type. □

When the inputs have bounded burstiness, as in (3), even though the trajectory is not asymptotically periodic, we can obtain a weaker version of the above corollary.

Corollary 3 *For small initial conditions,*

$$\|x_p(\cdot)\|_\infty \leq \frac{1-\rho_p}{\tau_p} \max_{\{j\,:\,i_j=p\}} \hat{t}_j,$$

where $\hat{t}$ is the solution of

$$\hat{t} = \bar{A}\,\hat{t} + (\bar{b} + \bar{\gamma}), \tag{15}$$

with $\bar{\gamma} = [\tau_{i_1}\gamma_{i_1}, \ldots, \tau_{i_N}\gamma_{i_N}]^T$.

Proof: Clearly, the length of a production run for part–type $i_j = p$ increases as the amount of material in buffer b_p increases. The worst case occurs when a burst amount γ_p is added to the input of parts of type p in each of its production runs. □

For linear deterministic inputs, Humes and Humes [15] have derived an upper bound on the length of all production runs (and not just in steady state), as a function of the initial conditions.

It should be noted that the asymptotically periodic behavior of GRR is dependent on having linear deterministic inputs and a system with only one machine. Clearly, if the input is aperiodic, then the output need not be periodic. Also, for multi–machine systems, the incommensurability of the processing times and input rates at the different machines may preclude periodic behavior.

6 Performance Bounds for Single Machine Systems

We now use the methods and results of the preceding section to obtain bounds for several performance criteria for single machine systems. Since there is only a single machine, we will omit the subscript m. We shall consider policies that only work on each part–type at its maximal possible rate τ_p^{-1}. In particular, if $x_p(t) = 0$ and some buffer $b_{p'}$ is not empty, then the machine will not continue to service buffer b_p. We should note that for the weighted average cost criterion, $\lim_{T\to\infty} 1/T \int_0^T \sum_{p=1}^P c_p x_p(t)dt$, Chase and Ramadge [12] have shown that under some conditions, it is optimal to work at less than the maximal rate. In particular they show that, for some lightly loaded systems, the cost can be reduced by working on a part at less than its maximum rate, by continuing to process it at a reduced rate matching the input even after its buffer level hits zero. However, we shall only concern ourselves with CAF policies which do not allow such behavior.

Under the assumptions of linear deterministic inputs and policies which operate at the maximum possible rate, we will first obtain a lower bound on the cost function,

$$\sum_{p=1}^{P} c_p \|x_p(\cdot)\|_\infty, \tag{16}$$

with all weights c_p strictly positive.

To find a lower bound $\underline{C}$ on (16), we shall relax the constraints, and obtain a set of independent problems. As in [1], we shall suppose that each part–type has received a copy

of the machine m for its private use, with the simple constraint that the copy for the p^{th} part–type can have at most n_p runs in the time interval $[0, T]$. This will be called the *one part–type, one machine relaxation.*

In [1], a lower bound, using this relaxation, is found for the average weighted work $\lim_{T\to\infty} 1/T \int_0^T \sum_p c_p x_p(t)dt$. Since the proof there attempts to make all production runs for a given part–type equal, it is not difficult to see that it also holds for the cost function (16) involving the maximum buffer levels. Thus, from [1], a lower bound on $\sum c_p \|x_p(\cdot)\|_\infty$ is

$$\underline{C} = \frac{\delta}{(1-\rho)} \left(\sum_{p=1}^{P} \sqrt{c_p\, d_p\, (1-\rho_p)} \right)^2 .$$

By applying Theorem 4, the asymptotic cost of any GRR policy can be easily determined. This is done for the Round Robin policy in the following theorem.

Theorem 5 : Asymptotic Performance of the Round Robin Policy

If the Round Robin Policy is implemented on a single machine with P part–types and linear deterministic inputs, then

(i) $\limsup_{t\to+\infty} x_p(t) = P\delta(1-\rho)^{-1} d_p(1-\rho_p)$ *for all* $p = 1, 2, \ldots, P$.

(ii) For small initial conditions (i.e., when all production runs are shorter than their steady state levels), we have $\sum_{p=1}^{P} c_p \|x_p(\cdot)\|_\infty = P\,\delta\,(1-\rho)^{-1} \sum_{p=1}^{P} c_p\, d_p\, (1-\rho_p) =: C_{RR}$.

Proof: The proof is a straightforward consequence of Theorem 4. See [4] for an alternative proof. □

Consider now the P-dimensional *vector* cost criterion $x_{\max}$ defined by

$$x_{\max} := (\|x_1(\cdot)\|_\infty, \|x_2(\cdot)\|_\infty, \ldots, \|x_P(\cdot)\|_\infty).$$

This is the vector of minimal buffer sizes needed to ensure that the buffers never overflow. We will consider Pareto–efficient[2] scheduling with respect to the vector cost criterion $x_{\max}$.

[2]Consider a vector cost criterion $J(u)$ of a decision variable $u \in U$. Then $u^* \in U$ is said to be Pareto–efficient if there is no $\bar{u} \in U$ with $J_i(\bar{u}) \le J_i(u^*)$ for all components i, with strict inequality for at least one i.

The following corollary shows that the Round Robin policy is Pareto–efficient with respect to the cost $x_{\max}$, by showing that it is optimal with respect to the cost function (16) for a particular choice of the weights c_p.

Corollary 4 *If $c_p = A[d_p(1-\rho_p)]^{-1}$ for $p = 1, 2, \ldots, P$ and any constant A, then $C_{RR} = \underline{C}$. Moreover, for small initial conditions, the lower bound $\underline{C}$ is achieved, and Round Robin is an optimal policy. Also, the Round Robin policy is a Pareto–efficient policy with respect to the P-dimensional criterion $x_{\max}$.*

Proof: Note that a policy is Pareto–efficient with respect to a P-dimensional criterion, if it is a minimizer of some linear combination, with strictly positive coefficients, of the P one-dimensional criteria. The corollary follows from the definitions of $\underline{C}$ and C_{RR}. □

As the following example demonstrates, more general GRR policies can be Pareto–efficient as well.

Example 3

Suppose that a single machine processes three part–types, where

$$c_2 = c_3, d_2 = d_3, \tau_2 = \tau_3, \text{ and } c_1 d_1 (1-\rho_1) = 4 c_2 d_2 (1-\rho_2) =: 4Y^2.$$

Then

$$\underline{C} = \frac{\delta}{1-\rho} (2\,Y + Y + Y)^2 = 16\,Y^2 \frac{\delta}{1-\rho}$$

is a lower bound. For the GRR policy $(1, 2, 1, 3)$, we have

$$\begin{aligned} \|x_1\|_\infty &= 2\,d_1\,(1-\rho_1)\frac{\delta}{1-\rho}, \\ \|x_2\|_\infty &= \|x_3\|_\infty = 4\,d_2\,(1-\rho_2)\,\frac{\delta}{1-\rho}, \\ C_{1213} &= 16\,Y^2\,\frac{\delta}{1-\rho} = \underline{C}. \end{aligned}$$

This shows that the GRR policy $(1, 2, 1, 3)$ is Pareto–efficient if part–types 2 and 3 are identical. In fact, it is not difficult to show that the GRR policy $(1, 2, 1, 3, 1, 4, \ldots, 1, P)$ where part–types $2, 3, \ldots P$ are indistinguishable, is a Pareto–efficient policy. □

The one part–type, one machine relaxation approach also can be used to derive a lower bound for another cost criterion: Minimize $\max_p c_p \|x_p(\cdot)\|_\infty$.

Theorem 6 *For any stable policy implemented on a single machine with P part–types and linear deterministic input, a lower bound on*

$$\max_p c_p \|x_p(\cdot)\|_\infty$$

is given by

$$\underline{D} := \delta(1-\rho)^{-1} \sum_{p=1}^{P} c_p \, d_p \, (1-\rho_p).$$

Proof: The one part–type, one machine relaxation of [1] leads to

$$\underline{D} = \inf \left\{ \max_p \left[c_p d_p (1-\rho_p) \frac{1}{f_p} \right] \text{ such that } \sum_{p=1}^{P} f_p \leq \delta^{-1}(1-\rho) \right\}.$$

It is clear that the above infimum is the optimal value of the problem

$$\begin{cases} \min y \\ k_p \frac{1}{f_p} - y \leq 0, \qquad \text{where } k_p = c_p d_p (1-\rho_p) \\ \sum_p f_p \leq \delta^{-1}(1-\rho). \end{cases}$$

From the Kuhn-Tucker conditions the minimum is as claimed. □

Simple calculations demonstrate the optimality of the Round Robin policy for a P part–type system with respect to the cost criterion

$$\max_p c_p \|x_p(\cdot)\|_\infty,$$

when all part–types are identical in their requirements, i.e., $c_p \equiv c$, $d_p \equiv d$, and $\tau_p \equiv \tau$. Also, for a three part–type system, if $c_2 = c_3$, $d_2 = d_3$, $\tau_2 = \tau_3$, and $c_1 d_1 (1-\rho_1) = 2 c_2 d_2 (1-\rho_2)$, then the GRR policy $(1,2,1,3)$ is optimal as well.

7 Performance Bounds for Totally Regulated Systems

In the previous section we presented bounds for several buffer cost criteria for a single machine. We demonstrated that these bounds are tight, i.e., they are achievable for certain values of parameters. However, the results of the previous section do not extend to

multi–machine systems. In fact, the Round Robin policy may be unstable for non–acyclic systems, as the examples in [6] show. In addition, for general systems, the existing bounding techniques for maximum buffer levels, or maximum delays for policies which do not allow overtaking of parts (see Cruz [3]), do not yield tight bounds. The determination of tight bounds on performance becomes considerably more difficult for systems composed of several machines.

In order to address these issues, we now consider the implementation of DSI scheduling policies for *totally regulated systems*. These are systems where every buffer has a regulator. By Theorem 3, such systems are guaranteed to be stable.

One way to determine bounds on the buffer sizes, described in [3], considers the machines to be operating in isolation with bursty inputs, with burstiness dependent on the bounds on the output from the previous machine. Letting σ denote the vector of maximum delays, it is possible to determine an inequality of the form

$$\sigma \leq A\,\sigma + c.$$

If all elements of A are nonnegative and $\max_i \|\lambda_i(A)\| < 1$, then $(I - A)^{-1}$ exists and has no negative elements. Hence

$$\sigma \leq (I - A)^{-1}\, c = \bar{\sigma},$$

where $\bar{\sigma}$ is a componentwise bound for the maximum delays incurred. Unfortunately, because it is independent of the scheduling policy employed, this methodology often leads to large bounds, even for simple acyclic systems.

As an alternative, as in Cruz [2], we will consider regulators. Our regulators are as defined in Section 4. We will suppose that they are smooth, i.e., $\gamma'_{p,i} \equiv 0$, and that the system is totally regulated.

Suppose that each machine is operating under a DSI policy, and an upper bound on the workload at each machine is known. For example, for CAF policies, under small initial conditions, the maximum value of the total workload of the system is known. Similarly, for GRR policies, under small initial conditions, upper bounds for the individual buffers are

known. In cases where such bounds are known, bounds for the regulated buffer sizes can be determined, by noting that the input to each regulator buffer is constrained by the regulated buffer at the previous machine.

Example 4

Define

$R_{p,i}(t) :=$ the value of the regulator buffer for (p,i),

$\bar{r}_{p,i-1} :=$ an upper bound for the regulated buffer $(p,i-1)$, and

$\bar{t}_{p,i} = \frac{\tau_{p,i-1}}{1-d'_{p,i-1}\tau_{p,i-1}}\,\bar{r}_{p,i-1} :=$ time necessary to empty the regulated buffer $(p,i-1)$ when it is at its upper bound value and being fed at the maximal regulated rate.

Clearly, since it is unnecessary to introduce a regulator that is faster than its input, it can be assumed that

$$(\tau_{p,i-1}^{-1} - d'_{p,i}) \geq 0,$$

with equality holding for self-regulated buffers. Since the part flows are continuous, it is not too difficult to see that

$$R_{p,i}(t) \leq \begin{cases} R_{p,i}(0) + (\tau_{p,i-1}^{-1} - d'_{p,i})\,t & \text{for } 0 \leq t \leq \bar{t}_{p,i} \\ \max\{0\,,\ R_{p,i}(\bar{t}_{p,i}) - (d'_{p,i} - d'_{p,i-1})(t - \bar{t}_{p,i})\} & \text{for } t \geq \bar{t}_{p,i}. \end{cases}$$

Thus

$$\begin{aligned} \|R_{p,i}(\cdot)\|_\infty &\leq R_{p,i}(0) + \frac{1 - d'_{p,i}\,\tau_{p,i-1}}{1 - d'_{p,i-1}\,\tau_{p,i-1}}\bar{r}_{p,i-1} \\ &\leq R_{p,i}(0) + \bar{r}_{p,i-1}. \end{aligned}$$

□

As the above analysis indicates, if the bound for a regulated buffer is not tight, then the error will be propagated to the next regulator buffer. Thus, it is useful to determine good bounds for the regulated buffers.

Theorem 7 *If the system has small initial conditions at all of its regulated machines, then*

$$\|x_{p,i}(\cdot)\|_\infty \leq \bar{r}_{p,i} + \bar{r}_{p,i-1}, \tag{17}$$

with $\bar{r}_{p,0} := \gamma_p$.

Proof: The result follows as in Example 4, using the definition of small initial conditions.

□

Thus, for small initial conditions, an upper bound for the actual size of a buffer is given by the sum of the upper bounds in isolation for the size of its regulated buffer and the size of the regulated buffer at the previous machine. Therefore, for the overall system, one can obtain small bounds by minimizing the cost criterion

$$\sum_{(p,i)} [c_{p,i} + c_{p,i+1}] \, \|x_{p,i}(\cdot)\|_\infty .$$

or the vector cost criterion

$$\{\|x_{p,1}(\cdot)\|_\infty, \ldots, \|x_{p,n_p}(\cdot)\|_\infty\}.$$

These observations, together with the Pareto–efficiency of the Round Robin and other GRR policies, provide additional motivation for studying the behavior of GRR policies and the determination of tight $\|\cdot\|_\infty$ bounds.

8 Concluding Remarks

We have shown that for a single machine under linear deterministic inputs, GRR scheduling policies lead to periodic buffer level trajectories. It is an open question whether this result can be extended to acyclic systems.

We have also derived bounds for totally regulated systems operating under GRR scheduling policies. It is of much interest to simulate such systems and to compare the results with the analytical bounds, and bounds determined by other approaches.

Since regulators reduce the burstiness of the material flows, they cause each machine to more closely behave as if it were in isolation. Thus, understanding the behavior of GRR policies on a single machine is of interest. It may be possible to obtain optimal or near-optimal GRR policies for any set of machine parameters. Recently, Angsana and Passino [16] have conducted a comparative study, including the case of a fuzzy controller. More such studies are warranted.

The only uncertainty considered in this paper has been the burstiness allowed in the inputs. Another important open issue is the stability and performance of regulated systems and GRR policies when other forms of uncertainty are present, e.g., random arrival, service, and set-up times, and random machine failures and repairs.

References

[1] J. R. Perkins and P. R. Kumar, "Stable distributed real-time scheduling of flexible manufacturing/assembly/disassembly systems," *IEEE Trans. Automat. Control*, vol. AC-34, pp. 139–148, February 1989.

[2] R. L. Cruz, "A calculus for network delay, part I: Network elements in isolation," *IEEE Transactions on Information Theory*, vol. 37, pp. 114–131, January 1991.

[3] R. L. Cruz, "A calculus for network delay, part II: Network analysis," *IEEE Transactions on Information Theory*, vol. 37, pp. 132–141, January 1991.

[4] J. R. Perkins, "Stable, real-time scheduling of a single machine manufacturing systems," Master's thesis, University of Illinois, Urbana, IL, 1990. Also, CSL Rept. DC-121, UILU-ENG-90-2226, UIUC.

[5] S. Lou, S. Sethi, and G. Sorger, "Analysis of a class of real-time multiproduct lot scheduling policies," *IEEE Trans. Automat. Control*, vol. AC-36, pp. 243–248, February 1991.

[6] P. R. Kumar and T. I. Seidman, "Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems," *IEEE Trans. Automat. Control*, vol. AC-35, pp. 289–298, March 1990.

[7] Z. B. Tang and L. Y. Shi, "Note on 'Distributed scheduling based on due dates and buffer priorities' by S. H. Lu and P. R. Kumar," *IEEE Trans. Automat. Control*, vol. AC-37, pp. 1661–1662, October 1992.

[8] T. M. Homem de Mello and C. Humes, Jr., "Instability in manufacturing systems with decreasing service times," *Pesquisa Operacional*, vol. 11, December 1991.

[9] C. Humes, Jr., "A regulator stabilization technique: Kumar-Seidman revisited." To appear in *IEEE Transactions on Automatic Control*, 1993.

[10] S. H. Lu and P. R. Kumar, "Distributed scheduling based on due dates and buffer priorities," *IEEE Transactions on Automatic Control*, vol. 36, pp. 1406–1416, December 1991.

[11] P. R. Kumar, "Re–entrant lines," *Queueing Systems: Theory and Applications: Special Issue on Queueing Networks*, vol. 13, pp. 87–110, May 1993.

[12] C. J. Chase and P. J. Ramadge, "On real-time scheduling policies for flexible manufacturing systems," *IEEE Trans. Automat. Control*, vol. AC-37, pp. 491–496, April 1992.

[13] T. Kailath, *Linear Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1980.

[14] D. G. Luenberger, *Introduction to Dynamic Systems: Theory, Models, and Applications*. New York, NY: John Wiley and Sons, Inc., 1979.

[15] A. F. P. C. Humes and C. Humes, Jr., "Stability of clearing open-loop policies in manufacturing systems (revised version)." Tech. Rept. RT-MAC9305, Instituto de Matematica e Estatistica, U.S.P., Sao Paulo, 1993.

[16] A. Angsana and K. M. Passino, "Distributed intelligent control of flexible manufacturing systems." Technical report, 1993.