Stable Scheduling Policies for Flexible Manufacturing Systems

Kevin Burgess and Kevin M. Passino

***Abstract*—In this brief note we provide a new analysis of the transient behavior of the clear-a-fraction policy of Perkins and Kumar. In addition, we show that a new "clear-average-oldest-buffer" policy and a "random part selection" policy (of which "first-come-first-served" is a special case) are stable. Finally, we introduce a stable and efficient "stream modifier" that can be used to obtain network level stability results.**

***Index Terms*—Discrete-event systems, manufacturing systems, scheduling, stability.**

I. Introduction

The flexible manufacturing systems (FMS's) considered here are of the type described in [1] where there are networks of machines, each of which has the capability to process N different part types $i \in P$, where $P = \{1, 2, \cdots, N\}$. Parts which arrive at a machine and are awaiting servicing are held in buffers, and the buffer levels are denoted by $x_i(k),\ i \in P$. Each machine can only process at some bounded rate one type of part at a time, and, in general, a machine incurs a "setup time" s_i (a bounded delay) when changing over to produce a new part type $i \in P$. The time during which a single part is being produced is called a "production run." We require the part flow to be composed of discrete parts because this is the case in most practical FMS's. In approaching the analysis and design of FMS's, the critical elements are the part production schedules, or *scheduling policies*, for the component machines. In [1], the authors study machines both in isolation and when interconnected in a nonacyclic fashion (i.e., when parts can revisit the same machine for processing) and analyze the stability of various scheduling policies (i.e., whether policies can keep the number of parts in the buffers bounded), including the clear-a-fraction (CAF) policy (which picks a buffer to process that has more than a fraction of the average number of parts in all the buffers). Related work is in [2].

In this paper we focus on isolated machines and specific network elements, not a network of machines (although our results can be useful for network level scheduling). In particular, in Section II, we present a stability analysis of the CAF policy of [1], which offers new insights into the transient behavior of the policy. In addition, we introduce the "time-based" clear-the-average-oldest-buffer (CAOB) policy and the random part selection (RPS) policy (of which the well known first-come first-serve (FCFS) policy is a special case) and perform stability analysis for these. The results for the FCFS, while conservative, do provide stability conditions for networks of FCFS machines (with setup times) when used with the regulator in [3] (certain FCFS networks have been shown to be unstable in [4]). In Section III, we introduce our stream modifier that is a generalization of the (σ, ρ) regulator of [5], which has been exploited in [3] and [6] to yield network level stability results. Our treatment of the stream modifier is important because we specify a realizable policy for the stream modifier which results in efficient stream modifier behavior (i.e., the policy maintains the smallest stream modifier buffer for all time).

Manuscript received April 26, 1995; revised February 13, 1996 and May 13, 1996. This work was supported in part by the National Science Foundation under Grant IRI-9210332.

The authors are with the Department of Electrical Engineering, The Ohio State University, Columbus, OH 43210-1272 USA (e-mail: passino@ee.eng.ohio-state.edu).

Publisher Item Identifier S 0018-9286(97)00499-6.

II. Machine Model and Stream Constraints

Let $W = \{0, 1, 2, \cdots\}$, and let $t_k,\ k \in W$ be the *real time* corresponding to discrete-time k. We will call the fixed length of real time between discrete-times k and $k+1$ one *period*. As is the case with any discrete-time model, the choice of period is important here. The single condition on the length of the period is that when in the midst of a production run, the machine must produce at least one part per period. Because such a period can be chosen for any realistic machine (by choosing the period sufficiently large), the condition is not restrictive. While this assumption simplifies the notational logistics of the entire analysis, only in the case of the RPS policy does the analysis rely on this assumption in a substantial manner. If parts arrive at the machine at real times other than the $t_k,\ k \in W$, then our analysis is valid at the real time points $t_k,\ k \in W$, if we consider all parts which arrive at or depart from the machine in any real time interval $[t_k, t_{k+1}),\ k \in W$, to have done so at real time t_k. Let $\mathbf{x}_k = [x_1(k), x_2(k), \cdots, x_N(k)]^t$ ("t" denotes transpose), where $k \in W$. Let Z be any set of times such that $Z \subset W$ and for all k_1, $k_2 \in Z$, if $k_1 \le k' \le k_2$, then $k' \in Z$, and there is some $i \in P$ such that $x_i(k') > 0$. For all of the analysis that follows, choose any such Z, and without loss of generality, assume that $\min Z = 0$. Let $A_i(k)$ be the (integer) number of parts of type $i \in P$ to arrive at the machine at time $k \in Z$, and let $D_i(k)$ be the (integer) number of parts of type $i \in P$ to depart from the machine at time $k \in Z$. At time $k \in Z$, the number of parts in buffer $i \in P$ is $x_i(k)$, and $x_i(k+1) = x_i(k) + A_i(k) - D_i(k)$. A part is considered to remain in its buffer until it exits the machine.

We define a function "ceil" such that ceil(y): $\Re^+ \to W$ and ceil$(y) = \min\{k \in W : k \ge y\}$, for all $y \in \Re^+$. We define a function "floor" such that floor: $\Re^+ \to W$ and floor$(y) = \max\{k \in W : k \le y\}$, for all $y \in \Re^+$. Let $F(k')$ be the number of production runs that have ended on or before time k'.

We will call any flow of parts into a machine an *input stream* and any flow of parts from a machine an *output stream*. We require that the input and output streams of the machine obey the following constraints.

1) For all $k_1, k_2 \in Z,\ k_1 \le k_2,\ i \in P$

$$0 \le \sum_{k=k_1}^{k_2} A_i(k) \le \text{ceil}\left(a_i^{\text{in}}(k_2 - k_1 + 1) + b_i^{\text{in}}\right) \tag{1}$$

(i.e., a_i^{in} is the maximum allowed rate, and b_i^{in} is the maximum allowed burstiness).

2) For all $k_1, k_2 \in Z$ such that k_1 and k_2 lie in the same production run

$$\text{floor}(\delta_j(k_2 - k_1 + 1)) \le \sum_{k=k_1}^{k_2} D_j(k) \le \text{ceil}(d_j(k_2 - k_1 + 1)) \tag{2}$$

where $1 \le \delta_j \le d_j$ and j is the part type being produced. The floor and ceil functions in constraints 1) and 2) above are necessary because a_i^{in}, δ_i, and d_i for $i \in P$ may be noninteger

(e.g., if $a_1^{\text{in}} = 1.5$, we would like for three parts to be able to arrive at buffer 1 every two periods; however, if we remove the ceil function from constraint 1), only one part can arrive at buffer 1 each period).

Let $w_i = \frac{a_i^{\text{in}}}{\delta_i}$ and $w = \sum_{i \in P} w_i$, where w_i is the ratio of maximum input rate and minimum output rate of buffer $i \in P$. It is intuitive that to have any chance of maintaining bounded buffer levels, we must have $w < 1$ (this is often referred to as the *capacity condition*). In addition, for convenience let $u_i = \frac{b_i^{\text{in}}}{\delta_i}$ and $u = \sum_{i \in P} u_i$.

III. Stability Analysis of the CAF Policy

The CAF policy requires that once a production run is begun, it will be continued until the buffer being processed is empty, then it chooses part type $j \in \{i \in P : x_i(k) \geq \epsilon \sum_{m=1}^{N} x_m(k)\}$, where $\epsilon \in (0, \frac{1}{N}]$ to process. For notational simplicity let $\underline{\chi} := \min_i\{\chi_i\}$ and $\bar{\chi} := \max_i\{\chi_i\}$ for any variable χ_i which is defined for all $i \in P$.

Theorem 1: When the CAF part servicing policy is used to control the above machine (with $w < 1$), it has buffer levels that are bounded for all $k \in W$ by

$$\sum_{i \in P} x_i(k) \leq \bar{\delta}\left[\left(\sum_{i \in P} \frac{x_i(0)}{\delta_i} - \frac{\zeta}{1-\gamma}\right)\gamma^{F(k)} + \frac{\zeta}{1-\gamma} + \bar{s}w + u + \frac{N}{\underline{\delta}}\right] \tag{3}$$

where

$$\gamma = \max_i\left\{1 - \frac{\epsilon\underline{\delta}}{\delta_i}\left(\frac{1-w}{1-w_i}\right)\right\} \quad (0 < \gamma < 1) \tag{4}$$

$$\zeta = \max_i\left\{(u_i + s_i + 1)\left(\frac{w - w_i}{1 - w_i}\right) + (u - u_i) + \sum_{j \in P, j \neq i} \frac{1}{\delta_j}\right\}. \tag{5}$$

Proof: Choose $V(\mathbf{x}_k) = \sum_{i \in P} \frac{x_i(k)}{\delta_i}$. Because the following analysis is valid for any Z, the bounds obtained are valid for all $k \in W$ (the only k which are in the set W but not in any set Z are such that $x_i(k) = 0$ for all $i \in P$). We define the set of times $R = \{k_0, k_1, k_2, \cdots\} \subset Z$, $k_p < k_q$ if $p < q$, to include every time k' such that k' is the greatest time no longer than time k'' which immediately follows the end of any production run for some $k'' \in Z$. Notice that $k_0 = 0$. Let $j^*(k_p) \in P, k_p \in R$ denote the part type that is being setup for and processed by the machine between times k_p and k_{p+1}. We define $k_{p+1} - k_p \triangleq \Delta_p$. In order to bound Δ_p, we use an approach similar to that in [1] and after some manipulations obtain

$$\Delta_p \leq \frac{\frac{x_{j^*(k_p)}(k_p)}{\delta_{j^*(k_p)}} + \frac{b_{j^*(k_p)}^{\text{in}} + 1}{\delta_{j^*(k_p)}} + s_{j^*(k_p)} + 1}{1 - w_{j^*(k_p)}}. \tag{6}$$

We now bound $V(\mathbf{x}_{k_{p+1}})$ in terms of $V(\mathbf{x}_{k_p})$. Notice that $x_{j^*(k_p)}(k_{p+1}) = 0$ and $x_i(k_{p+1}) - x_i(k_p) \leq \text{ceil}(a_i^{\text{in}}\Delta_p + b_i^{\text{in}})$ for all $i \in P, i \neq j^*(k_p)$. From this it follows that

$$\sum_{i \in P} \frac{x_i(k_{p+1})}{\delta_i} \leq \sum_{i \in P} \frac{x_i(k_p)}{\delta_i} - \frac{x_{j^*(k_p)}(k_p)}{\delta_{j^*(k_p)}} + \sum_{i \in P, i \neq j^*(k_p)} \frac{a_i^{\text{in}}\Delta_p + b_i^{\text{in}} + 1}{\delta_i}. \tag{7}$$

After some manipulations, we see that

$$V(\mathbf{x}_{k_{p+1}}) \leq V(\mathbf{x}_{k_p}) - \frac{x_{j^*(k_p)}(k_p)}{\delta_{j^*(k_p)}}\left(\frac{1-w}{1-w_{j^*(k_p)}}\right) + \left(u_{j^*(k_p)} + s_{j^*(k_p)} + 1\right) \cdot \left(\frac{w - w_{j^*(k_p)}}{1 - w_{j^*(k_p)}}\right) + \left(u - u_{j^*(k_p)}\right) + \sum_{i \in P, i \neq j^*(k_p)} \frac{1}{\delta_i}. \tag{8}$$

From the definition of the CAF policy, we see that $x_{j^*(k_p)}(k_p) \geq \epsilon\underline{\delta}V(\mathbf{x}_{k_p})$. Up to this point, this proof is similar to the CAF stability proof in [1], except that in this proof, bounded input stream rate and burstiness constraints are allowed. If we define γ and ζ as in the statement of the theorem, we see from (8) that $V(\mathbf{x}_{k_{p+1}}) \leq V(\mathbf{x}_{k_p}) + \zeta$ for all $k_p \in R$ (notice that by definition, $0 < \frac{\epsilon\underline{\delta}}{\delta_i} < 1$ for all $i \in P$ and $0 < \gamma < 1$). This is simply a difference inequality which when solved yields

$$V(\mathbf{x}_{k_p}) \leq \left(V(\mathbf{x}_0) - \frac{\zeta}{1-\gamma}\right)\gamma^p + \frac{\zeta}{1-\gamma} \triangleq G(\mathbf{x}_0, p). \tag{9}$$

Thus, we have bounded $V(\mathbf{x}_k)$ for all $k \in R$. Consider now the set of times S_p such that if $k \in Z$ and $k \in (k_p, k_{p+1})$, then $k \in S_p$. In (9), we have found a bound $G(\mathbf{x}_0, p)$ for $V(\mathbf{x}_k)$, for $k = k_p$, and k_{p+1}. We now wish to bound $V(\mathbf{x}_k)$ for all $k \in S_p \cup \{k_p, k_{p+1}\}$. Clearly, the maximum of V over $S_p \cup \{k_p, k_{p+1}\}$ must occur at one of the following times: k_p, k_{p+1}, or at beginning of the production run that began before k_{p+1} which we denote by $b(k_{p+1})$. We can bound the increase in V that occurs between times k_p and $b(k_{p+1})$ as

$$V\left(\mathbf{x}_{b(k_{p+1})}\right) - V(\mathbf{x}_{k_{p+1}}) \leq \sum_{i \in P} \frac{a_i^{\text{in}}s_i + b_i^{\text{in}} + 1}{\delta_i} \leq \bar{s}w + u + \frac{N}{\underline{\delta}}. \tag{10}$$

Hence, for all $k \in S_p \cup \{k_p, k_{p+1}\}$, $V(\mathbf{x}_k) \leq G(\mathbf{x}_0, p) + \bar{s}w + u + \frac{N}{\underline{\delta}}$, and so for any $k \in Z$

$$V(\mathbf{x}_k) \leq G(\mathbf{x}_0, F(k)) + \bar{s}w + u + \frac{N}{\underline{\delta}} = \left(V(\mathbf{x}_0) - \frac{\zeta}{1-\gamma}\right)\gamma^{F(k)} + \frac{\zeta}{1-\gamma} + \bar{s}w + u + \frac{N}{\underline{\delta}}. \tag{11}$$

Therefore, a bound on the buffer levels for all $k \in Z$, and, hence, for all $k \in W$, is

$$\frac{1}{\bar{\delta}}\sum_{i \in P} x_i(k) \leq \sum_{i \in P} \frac{x_i(k)}{\delta_i} \leq \left(\sum_{i \in P} \frac{x_i(0)}{\delta_i} - \frac{\zeta}{1-\gamma}\right)\gamma^{F(k)} + \frac{\zeta}{1-\gamma} + \bar{s}w + u + \frac{N}{\underline{\delta}}$$

which completes the proof. □

Notice that from (3), a useful property of the machine buffer dynamics is apparent. If $\sum_{i \in P} \frac{x_i(0)}{\delta_i} > \frac{\zeta}{1-\gamma}$, then the right side of (3) must asymptotically *decrease* to $\bar{\delta}(\frac{\zeta}{1-\gamma} + \bar{s}w + u)$ as $k \to \infty$ so that the sum of the buffer levels will get no larger than the bound at time $k = 0$. If $\sum_{i \in P} \frac{x_i(0)}{\delta_i} < \frac{\zeta}{1-\gamma}$, then the right side of (3) must asymptotically *increase* to $\bar{\delta}(\frac{\zeta}{1-\gamma} + \bar{s}w + u)$ as $k \to \infty$. Hence, (3) helps to characterize both the transient and steady-state behavior of the machine. Notice that our proof is for a more general class of stream constraints for the parts arriving at and departing from the machine than in [1] and (3) provides a more detailed characterization of the transient behavior of the machine. It is also possible to define a generalized CAF (GCAF) in which only a fixed fraction of the

parts in a buffer at the beginning of a production run are cleared during that production run. Similar results hold for the GCAF policy. Finally, recall that the buffer bound established above is only valid at real times t_k. However, if the "inter-sample" input and output stream functions are known or can be bounded, then the above bound can be modified so that it is valid for all $t \in \Re^+$. If we know nothing of the "inter-sample" input and output stream functions, we can add d_i for all $i \in P$ to the buffer bounds so that the bounds are valid for all $t \in \Re^+$.

IV. Stability Analysis of the CAOB Policy

We now introduce what we call the "clear-the-average-oldest buffer" (CAOB) policy. For all $i \in P$ and for all $k \in W$, let $T_i(k)$ be the maximum number of periods that any part in buffer i has been waiting for service at time k and let $T(k) = [T_1(k), T_2(k), \cdots, T_N(k)]^{\mathrm{t}}$. Let the CAOB policy choose at time k a part $j \in \{i \in P : T_i(k) \geq \epsilon \sum_{m=1}^{N} T_m(k)\}$ to process where $\epsilon \in (0, \frac{1}{N}]$.

Theorem 2: If the CAOB part servicing policy is used to control the above machine and if $w_i < \frac{1}{N}$ for all $i \in P$, then it has buffer levels that are bounded for all $k \in W$ by

$$\sum_{i \in P} x_i(k) \leq \bar{a}^{\mathrm{in}} \left[\left(1 - \quad^{F(k-1)+1}\right) \frac{\zeta}{1-} + \bar{\eta} \right] + \sum_{i \in P} \left(x_i(0) + b_i^{\mathrm{in}} + 1 \right) \tag{12}$$

where

$$= \max_i \left\{ 1 - \epsilon \left(1 - \frac{(N-1) a_i^{\mathrm{in}}}{\delta_i (1 - w_i)} \right) \right\} \quad (0 < \gamma \ < 1) \tag{13}$$

$$\zeta = \max_i \left\{ \frac{N-1}{1-w_i} \left(\frac{2 b_i^{\mathrm{in}} + 1}{\delta_i} + s_i + 1 \right) \right\} \tag{14}$$

$$\eta_i = \max \left\{ k \in W : \mathrm{ceil}\left(a_i^{\mathrm{in}} (k+1) + b_i^{\mathrm{in}} \right) - \mathrm{floor}(\delta_i (k + 1 - s_i)) > 0 \right\}. \tag{15}$$

Proof: Choose $V(T(k)) = \sum_{i \in P} T_i(k)$. Beginning in a similar way to the proof of Theorem 1 we know that (6) holds for CAOB. Notice that $T_{j^*(k_{p+1})}(k_{p+1}) = 0$, $T_i(k_{p+1}) - T_i(k_p) \leq \Delta_p$ for all $i \in P, i \neq j^*(k_p)$, and $x_i(k) \leq a_i^{\mathrm{in}} (T_i(k)) + b_i^{\mathrm{in}} + 1$ for all $i \in P, k \in Z$. From this and from the definition of the CAOB policy, we see that $\sum_{i \in P} T_i(k_{p+1}) \leq \sum_{i \in P} T_i(k_p) - T_{j^*(k_p)}(k_p) + (N-1)\Delta_p$. Hence

$$\begin{aligned} & V(T(k_{p+1})) \\ & \quad \leq V(T(k_p)) - \epsilon V(T(k_p)) \left[1 - \frac{(N-1) a_{j^*(k_p)}^{\mathrm{in}}}{\delta_{j^*(k_p)} (1 - w_{j^*(k_p)})} \right] \\ & \qquad + \frac{N-1}{1 - w_{j^*(k_p)}} \left[\frac{2 b_{j^*(k_p)}^{\mathrm{in}} + 1}{\delta_{j^*(k_p)}} + s_{j^*(k_p)} + 1 \right] \\ & \quad = V(T(k_p)) \left[1 - \epsilon \left(1 - \frac{(N-1) a_{j^*(k_p)}^{\mathrm{in}}}{\delta_{j^*(k_p)} (1 - w_{j^*(k_p)})} \right) \right] \\ & \qquad + \frac{N-1}{1 - w_{j^*(k_p)}} \left[\frac{2 b_{j^*(k_p)}^{\mathrm{in}} + 1}{\delta_{j^*(k_p)}} + s_{j^*(k_p)} + 1 \right]. \end{aligned} \tag{16}$$

If we define and ζ as in the statement of Theorem 2 we see from (16) that $V(T(k_{p+1})) \leq V(T(k_p)) + \zeta$, for all $k_p \in R$. Notice that by assumption, $0 < \gamma \ < 1$. Hence, we have $V(T(k_p)) \leq (1 - \ ^p) \frac{\zeta}{1-} \triangleq G(p)$ (note that $V(T(k_0)) = 0$) and so we have bounded $V(T(k))$ for all $k \in R$.

Consider now the set of times S_p such that if $k \in Z$ and $k \in (k_p, k_{p+1})$, then $k \in S_p$. We have found a bound $G(p)$ for $V(T(k))$, for $k = k_p$, and k_{p+1}. We now wish to bound $V(T(k))$ for all $k \in S_p \cup \{k_p, k_{p+1}\}$. To do this, we first must bound $T_{j^*(k_p)}(k') - T_{j^*(k_p)}(k_p)$ for all $k' \in S_p$. Consider the following expression:

$$\sum_{k = k_p - T_{j^*(k_p)}(k_p)}^{k_p - T_{j^*(k_p)}(k_p) + k'} \left[A_{j^*(k_p)}(k) - D_{j^*(k_p)}\left(k + T_{j^*(k_p)}(k_p)\right) \right] \tag{17}$$

for all $k_p + k' \in S_p$. Expression (17) is the sum of all parts which arrived at buffer $j^*(k_p)$ at any time $k \in W$, $k_p - T_{j^*(k_p)}(k_p) \leq k \leq k_p - T_{j^*(k_p)}(k_p) + k'$ minus the sum of all parts which leave buffer $j^*(k_p)$ at any time $k \in S_p$, $k_p \leq k \leq k_p + k'$. If expression (17) is positive for a given $k' \in W$, then more parts arrived at buffer $j^*(k_p)$ in the $k'+1$ periods beginning at time $k_p - T_{j^*(k_p)}(k_p)$ than have left buffer $j^*(k_p)$ in the $k'+1$ periods beginning at time k_p. Hence, there is at least one part which arrived at buffer $j^*(k_p)$ on or before time $k_p - T_{j^*(k_p)}(k_p) + k'$ which remains in the buffer at time $k_p + k' + 1$ so that $T_{j^*(k_p)}(k_p + k') > T_{j^*(k_p)}(k_p)$. By the same reasoning, if (17) is not positive, then $T_{j^*(k_p)}(k_p + k') \leq T_{j^*(k_p)}(k_p)$. It is clear that $T_{j^*(k_p)}(k_p + k') - T_{j^*(k_p)}(k_p) \leq k'$. We can bound expression (17) by specifying that parts in buffer $j^*(k_p)$ be serviced as slowly as possible so that

$$\sum_{k = k_p - T_{j^*(k_p)}(k_p)}^{k_p - T_{j^*(k_p)}(k_p) + k'} \left[A_{j^*(k_p)}(k) - D_{j^*(k_p)}\left(k + T_{j^*(k_p)}(k_p)\right) \right] \tag{18}$$

$$\leq \mathrm{ceil}\left(a_{j^*(k_p)}^{\mathrm{in}} (k' + 1) + b_{j^*(k_p)}^{\mathrm{in}} \right) - \mathrm{floor}\left(\delta_{j^*(k_p)} \left(k' + 1 - s_{j^*(k_p)} \right) \right) \tag{19}$$

for all $k_p + k' \in S_p$. For all $i \in P$, let η_i be defined as given in the statement of Theorem 2. Notice that because $a_i^{\mathrm{in}} \leq \frac{\delta_i}{N}$ for all $i \in P$ and because we are inherently assuming that $b_i^{\mathrm{in}} < \infty$ for all $i \in P$, $\eta_i < \infty$ for all $i \in P$. Then, for all $k_p + k' \in S_p$, $T_{j^*(k_p)}(k_p + k') - T_{j^*(k_p)}(k_p) \leq \eta_{j^*(k_p)}$. Hence, for all $k_p + k' \in S_p$, $V(T(k_p + k')) \leq G(p+1) + \eta_{j^*(k_p)}$. Notice in the above bound that $G(p+1)$ appears rather than $G(p)$. This is due to the fact that for all $i \in P$, $i \neq j^*(k_p)$, $T_i(k_{p+1}) > T_i(k_p + k')$, for all $k_p + k' \in S_p$. Hence

$$V(T(k)) \leq \left(1 - \quad^{F(k-1)+1}\right) \frac{\zeta}{1-} + \bar{\eta} \tag{20}$$

for all $k \in Z$, $k > 0$, and, hence, for all $k \in W$, $k > 0$. Because $x_i(k) \leq x_i(0) + a_i^{\mathrm{in}} T_i(k) + b_i^{\mathrm{in}}$ for all $k \in W$ and for all $i \in P$, it is clear that $\sum_{i \in P} x_i(k) \leq \sum_{i \in P} (x_i(0) + a_i^{\mathrm{in}} T_i(k) + b_i^{\mathrm{in}} + 1) \leq \bar{a}^{\mathrm{in}} V(T(k)) + \sum_{i \in P} (x_i(0) + b_i^{\mathrm{in}} + 1)$ which with (20) gives the final result for all $k \in W$, $k > 0$. □

Notice that in bounding $V(T(k))$ in (20), that the bound increases from $\bar{\eta}$ to $\frac{\zeta}{1-} + \bar{\eta}$ as $k \to \infty$ so that we also characterize the transient properties of CAOB. A clear-the-oldest buffer (COB) policy is a special case of the CAOB policy; hence, the bounds above hold for this policy also. The COB policy is sometimes called the "first-come first-clear" (FCFC) policy since it will service the buffer which contains the part that arrived before all other parts in any of the other buffers. If parts tend to arrive at the machine such that a group of parts arriving at one buffer is followed by a group of parts arriving

at another buffer and so on (at a low enough frequency), then the CAOB policy will tend to behave like an FCFS policy.

V. Stability Analysis of the RPS Policy

We now introduce what we call the RPS policy. Under this policy, the machine is free to choose any nonempty buffer to service at any time just so long as it never sits idle (i.e., it is either setting up for or processing a part at every instant).

Theorem 3: If $a_i^{\text{in}} \leq \frac{1}{N(\bar{s}+1)}$ for all $i \in P$, and the RPS part servicing policy is used to control the above machine, then $\sum_{i=1}^{N} x_i(k) \leq b + N + 1 + \sum_{i=1}^{N} x_i(0)$, for all $k \in W$, where $b = \sum_{i=1}^{N} b_i^{\text{in}}$.

Proof: Let $A(k)$ be defined so that $A(k) = \sum_{i=1}^{N} A_i(k)$, for all $k \in W$. In the worst case in which the machine produces only a single part from any buffer before switching production to a different buffer, it is clear that it can take no longer than $\bar{s}+1$ periods to produce one part. If we let $D(k) = \max\{D_i(k) : i \in P\}$, we see that $\sum_{k=k_1}^{k_2} D(k) \geq \text{floor}(\frac{k_2-k_1+1}{\bar{s}+1})$, for all $k_1, k_2 \in Z$, $k_1 \leq k_2$. From this, the definition of $A(k)$, and the assumption on a_i^{in}, $i \in P$, in the statement of the theorem, it is apparent that

$$\sum_{k=0}^{k'} [A(k) - D(k)] \leq \sum_{i=1}^{N} \left(\frac{k'+1}{N(\bar{s}+1)} + b_i^{\text{in}} + 1 \right) - \text{floor}\left(\frac{k'+1}{\bar{s}+1} \right) \quad (21)$$

$$= \frac{k'+1}{\bar{s}+1} + b + N - \text{floor}\left(\frac{k'+1}{\bar{s}+1} \right) \quad (22)$$

$$\leq b + N + 1 \quad (23)$$

and that $\sum_{k=0}^{k'} [A(k) - D(k)] = \sum_{i=1}^{N} x_i(k'+1) - \sum_{i=1}^{N} x_i(0)$, for all $k', k'+1 \in Z$. Clearly, then, $\sum_{i=1}^{N} x_i(k'+1) \leq b + N + 1 + \sum_{i=1}^{N} x_i(0)$, for all $k', k'+1 \in W$. □

Notice that unlike the conditions on stability of the CAF policy, the condition for stability of the RPS policy does not depend on the processing speed of the machine (of course, we have required previously that the period length be chosen so that $1 \leq \delta_i \leq d_i$ for all $i \in P$). Rather, the condition simply limits the rates of the input streams of the machine. Intuitively, the RPS policy is stable because it is *persistent* in that if there are parts in any machine buffer, it will always be either processing parts or setting up to process parts. Under the conditions of Theorem 2, several commonly used policies are special cases of the RPS policy and hence are stable because they are persistent: 1) the first-come first-serve (FCFS) policy; (b) the priority policy (buffers are serviced in a fixed order, but empty buffers are skipped as in [7]), and fixed time policy (nonempty buffers are serviced for a fixed amount of time). Moreover, policies studied in [8], such as the "earliest due date" policy, are special cases of RPS. It is interesting to note that in [4] the author was able to show that FCFS is unstable for certain FMS topologies where there are no setup times. The key to obtaining stability here is that unlike in [4] we constrain the rates at which parts may be input to machines (so that if applied to a network of machines our results would require a stream modifier like in the next section to achieve stable operation).

The stability conditions for the RPS policy in Theorem 3 are somewhat dissatisfying because we cannot affect the input stream rate constraints by speeding up the machine, and this is contrary to our intuition. In light of this, we now reformulate the problem by altering the way that we look at part arrivals and departures. First of all, it is necessary to redefine the period for this analysis. We choose the new period and constants $a_i^{\text{in}'}$ so that for every $i \in P$ there are at least $a_i^{\text{in}'}$ periods for each part that arrives at buffer i that is not attributable to the input stream burstiness. Similarly, choose δ_i' so that when the machine is producing parts of type i it outputs parts no slower than one part every δ_i' periods. As before, let s_i be the number of periods needed to set up for production on buffer $i \in P$. Assume that for all $i \in P$, $a_i^{\text{in}'}$ and δ_i' are integers (this assumption is not limiting since we can choose the period to be as small as desired).

Theorem 4: If the RPS part servicing policy is used to control the above machine, and

$$\max_i\{\delta_i' + s_i\} \sum_{i \in P} \frac{1}{a_i^{\text{in}'}} \leq 1$$

then $\sum_{i=1}^{N} x_i(k) \leq b + N + 1 + \sum_{i=1}^{N} x_i(0)$, for all $k \in W$, where $b = \sum_{i=1}^{N} b_i^{\text{in}}$.

Proof: Let $A(k) = \sum_{i \in P} A_i(k)$ and $D(k) = \sum_{i \in P} D_i(k)$ for all $k \in W$ so that

$$\sum_{k=k_1}^{k_2} A_i(k) \leq \text{floor}\left(\frac{k_2 - k_1 + 1}{a^{\text{in}'}{}_i} \right) + 1 + b_i^{\text{in}} \leq \frac{k_2 - k_1 + 1}{a^{\text{in}'}{}_i} + 1 + b_i^{\text{in}} \quad (24)$$

$$\sum_{k=k_1}^{k_2} A(k) \leq (k_2 - k_1 + 1) \sum_{i \in P} \frac{1}{a^{\text{in}'}{}_i} + N + \sum_{i \in P} b_i^{\text{in}} \quad (25)$$

and $\sum_{k=k_1}^{k_2} D(k) \geq \text{floor}\big(\frac{k_2-k_1+1}{\max_i\{\delta_i'+s_i\}}\big) \geq \frac{k_2-k_1+1}{\max_i\{\delta_i'+s_i\}} - 1$, for all $k_1, k_2 \in W$, $k_1 \leq k_2$. Notice that as in the previous analysis of the RPS policy, we have identified the maximum number of periods per part serviced (i.e., as long as there are parts in any machine buffer, the machine must output at least one part every $\max_i\{\delta_i' + s_i\}$ periods). Clearly, $\sum_{k=0}^{k'} [A(k) - D(k)] = \sum_{i \in P} (x_i(k'+1) - x_i(0)) \leq (k'+1)\big[\sum_{i \in P} \frac{1}{a^{\text{in}'}{}_i} - \frac{1}{\max_i\{\delta_i'+s_i\}}\big] + N + b + 1$, for all $k' \in W$. Now, by the assumption in the theorem we see that $\sum_{i \in P} (x_i(k+1) - x_i(0)) \leq N + b + 1$ or, equivalently $\sum_{i \in P} x_i(k+1) \leq b + N + 1 + \sum_{i \in P} x_i(0)$, for all $k \in W$. □

Notice that while the RPS stability condition in Theorem 4 is more flexible than the condition in Theorem 3 (in terms of our ability to design a machine that can achieve stability by speeding it up), the input stream rate constraints are still limited by the maximum machine setup time, regardless of how fast the machine is. This appears to be a fundamental property of RPS policies. Notice also that if $a^{\text{in}'}{}_i$ and δ_i' are considered to be inverse rate constraints, the stability condition of Theorem 4 can be thought of as reducing to the capacity condition as $s_i \to 0$ for all $i \in P$.

VI. Stream Modifier

The "stream modifier" is a network element which consists of a buffer and a part flow policy which selectively queues incoming parts in the buffer or passes them directly through to the output stream (we use the term "stream modifier" rather than "regulator" simply to emphasize that the two are different). In addition, the policy must decide when to release queued parts into the output stream. The purpose of the stream modifier is to alter the maximum rate and maximum burstiness of its input stream. Note that the stream modifier is an important element for FMS's since it can be used to modify the streams of parts between machines so that an *entire* FMS can be made stable [3].

At time $k \in W$, let the number of parts in the stream modifier buffer be $x(k)$, the number of parts arriving at the stream modifier

be $A(k)$, and the number of parts leaving the stream modifier be $D(k)$. In addition, let a^{in}, b^{in}, a^{out}, and b^{out} be real, nonnegative constants which are used to describe the input and output streams of the stream modifier. In order to clarify the following analysis, assume that a^{in} and a^{out} are integers. We specify the behavior of the input and output streams of the stream modifier with respect to constants a^{in}, b^{in}, a^{out}, and b^{out} as follows.

1) For all $k_1, k_2 \in W$, $k_1 \le k_2$, $\sum_{k=k_1}^{k_2} A(k) \le \text{ceil}(a^{\text{in}}(k_2 - k_1 + 1) + b^{\text{in}})$, $\sum_{k=k_1}^{k_2} D(k) \le \text{ceil}(a^{\text{out}}(k_2 - k_1 + 1) + b^{\text{out}})$.
2) For all $k' \in W$, $\sum_{k=0}^{k'}[A(k) - D(k)] \le \max\{(a^{\text{in}} - a^{\text{out}})(k'+1) + b^{\text{in}} - b^{\text{out}} + 1, -x(0)\}$.

Item 1) simply specifies the input stream constraint for the stream modifier and how we would like the output stream of the stream modifier to behave. Item 2) is included as a constraint on stream modifier behavior to ensure that its buffer is bounded. In fact, because $x(k+1) = x(k) + A(k) - D(k)$ and because $\sum_{k=0}^{k'}[A(k) - D(k)] = x(k'+1) - x(0)$, for all $k' \in W$, we see from item 2) that $x(k'+1) \le \max\{(x(0) + (a^{\text{in}} - a^{\text{out}})(k'+1) + b^{\text{in}} - b^{\text{out}} + 1), 0\}$, for all $k' \in W$. If we choose $a^{\text{out}} = a^{\text{in}}$, then for all $k \in W$, $x(k) \le x(0) + b^{\text{in}} - b^{\text{out}} + 1$, and the stream modifier buffer is bounded. Notice also that if $a^{\text{out}} = a^{\text{in}}$, then we can choose $b^{\text{out}} = 0$ so that the burstiness is completely removed from the output stream. In this case, if the input stream operates at its maximum rate, then the output stream can operate at its maximum rate and the extra bursts of parts on the input stream, that by definition will never total more than b^{in}, will be stored in the stream modifier buffer. Notice that if $a^{\text{out}} < a^{\text{in}}$, then either b^{out} is infinite or we cannot bound $x(k)$ for all $k \in W$. It is also clear that if a^{in} or b^{in} is infinite and a^{out} and b^{out} are finite, no bound exists for $x(k)$ for all $k \in W$.

We now specify a practical policy for the stream modifier which we will show satisfies items 1) and 2) by releasing the maximum number of parts allowable without violating the second inequality at every time k in item 2). For every time $k \in W$, let $E_k = \{E_k^0, E_k^1, \cdots, E_k^k\}$, where $E_k^{k'}$ is the maximum allowable value of $D(k)$ such that

$$\sum_{l=k'}^{k} D(l) = \text{ceil}(a^{\text{out}}(k - k' + 1) + b^{\text{out}}). \tag{26}$$

Because the stream modifier cannot violate (26) for any k', $k' \le k$, let its policy choose

$$D(k) = \min\left(E_k \cup \{A(k) + x(k)\}\right) \tag{27}$$

(this policy is not implementable because as $k \to \infty$ $|E_k| \to \infty$; below we will show how to modify it so that it is an implementable policy). We now must ask whether our policy will satisfy items 1) and 2). Because our policy in (27) guarantees that (26) will not be violated for any $k', k \in W$, $k' \le k$, it is clear that it satisfies the second inequality in item 1) because (26) is more strict than the second inequality in item 1). Next consider item 2). Choose any time $k' \in W$. If $D(k') = \min\{E_{k'} \cup \{A(k') + x(k')\}\} = A(k') + x(k')$, then because $x(k'+1) = 0$ (the stream modifier buffer is cleared by the policy's choice of $D(k')$), it is clearly the case that $\sum_{k=0}^{k'}(A(k) - D(k)) = -x(0)$ so that item 2) holds. If, on the other hand, $D(k') = \min\{E_{k'} \cup \{A(k') + x(k')\}\} = \min(E_{k'})$, then from the definitions of E_k' in (26) and the stream modifier policy in (27), there is some $k_1 \in W$, $k_1 \le k'$, such that $\sum_{k=k_1}^{k'} D(k) = \text{ceil}(a^{\text{out}}(k' - k_1 + 1) + b^{\text{out}})$. Below we define a recursive procedure

TABLE I
E_k

$E_k^k = \text{ceil}\left(a^{\text{out}} + b^{\text{out}}\right)$
$E_k^{k-1} = \text{ceil}\left(2a^{\text{out}} + b^{\text{out}}\right) - D(k-1)$
$\vdots$
$E_k^0 = \text{ceil}\left(a^{\text{out}}(k+1) + b^{\text{out}}\right) - \sum_{n=0}^{k-1} D(n)$

TABLE II
E_{k+1}

$E_{k+1}^{k+1} = \text{ceil}\left(a^{\text{out}} + b^{\text{out}}\right)$
$E_{k+1}^{k} = \text{ceil}\left(2a^{\text{out}} + b^{\text{out}}\right) - D(k)$
$E_{k+1}^{k-1} = \text{ceil}\left(3a^{\text{out}} + b^{\text{out}}\right) - D(k-1) - D(k)$
$\vdots$
$E_{k+1}^{0} = \text{ceil}\left(a^{\text{out}}(k+2) + b^{\text{out}}\right) - \sum_{n=0}^{k} D(n)$

whose goal is to define a time $k_* \in W$ which we use later in the analysis. Upon initially entering the procedure, let $i = 0$.

1) If $i = 0$, let $n_i = k'$; otherwise, let $n_i = m_{i-1} - 1$.
2) If $D(n_i) = \min(E_{n_i})$, then find the smallest $q \in W$ such that

$$\sum_{k=q}^{n_i} D(k) = \text{ceil}(a^{\text{out}}(n_i - q + 1) + b^{\text{out}}) \tag{28}$$

and let $m_i = q$.
3) If $m_i = 0$, then let $k_* = 0$ and stop.
4) If $D(m_i - 1) < \min(E_{m_i - 1})$, then let $k_* = m_i - 1$ and stop; otherwise, let $i = i + 1$ and return to step 1.

Notice that the above procedure will always terminate. If $k_* = 0$, then the entire range of times $[0, k']$ is composed of adjacent subranges of times $[m_i, n_i]$, $i = 0, 1, 2, \cdots, Q$, where $n_0 = k'$, $m_Q = 0$, $n_i = m_{i-1} - 1$ for all $i = 1, 2, \cdots, Q$, and

$$\sum_{k=m_i}^{n_i} D(k) = \text{ceil}(a^{\text{out}}(n_i - m_i + 1) + b^{\text{out}}) \tag{29}$$

for all $i = 0, 1, 2, \cdots, Q$. Hence, we see that $\sum_{k=0}^{k'} D(k) = \sum_{i=0}^{Q} \text{ceil}(a^{\text{out}}(n_i - m_i + 1) + b^{\text{out}})$ and (because $\text{ceil}(a) + \text{ceil}(b) \ge \text{ceil}(a + b)$) that $\sum_{k=0}^{k'} D(k) \ge \text{ceil}(a^{\text{out}}(k'+1) + (Q+1)b^{\text{out}})$. Because we have established that our policy obeys item 1), we see that either Q or b^{out} must equal zero so that $\sum_{k=0}^{k'} D(k) = \text{ceil}(a^{\text{out}}(k'+1) + b^{\text{out}})$. Therefore, $\sum_{k=0}^{k'}(A(k) - D(k)) \le \text{ceil}(a^{\text{in}}(k'+1) + b^{\text{in}}) - \text{ceil}(a^{\text{out}}(k'+1) + b^{\text{out}}) \le (a^{\text{in}} - a^{\text{out}})(k'+1) + b^{\text{in}} - b^{\text{out}} + 1$, so that for $D(k') = \min(E_{k'})$ and $k_* = 0$, our policy satisfies item 2).

If $k_* > 0$, then the entire range of times $[k_* + 1, k']$ is composed of adjoining subranges of times $[m_i, n_i]$, $i = 0, 1, 2, \cdots, Q$, where $n_0 = k'$, $m_Q = k_* + 1$, $n_i = m_{i-1} - 1$ for all $i = 1, 2, \cdots, Q$, and (29) holds for all $i = 0, 1, 2, \cdots, Q$. Similar to before, it follows that $\sum_{k=k_*+1}^{k'} D(k) = \text{ceil}(a^{\text{out}}(k' - k_*) + b^{\text{out}})$. Because $\sum_{k=0}^{k_*}(A(k) - D(k)) \le 0$, we see that

$$\sum_{k=0}^{k'}(A(k) - D(k)) = \sum_{k=0}^{k_*}(A(k) - D(k)) + \sum_{k=k_*+1}^{k'}(A(k) - D(k)) \tag{30}$$

$$\le \sum_{k=k_*+1}^{k'}(A(k) - D(k)) \tag{31}$$

$$\leq \text{ceil}(a^{\text{in}}(k' - k_*) + b^{\text{in}}) - \text{ceil}(a^{\text{out}}(k' - k_*) + b^{\text{out}}) \quad (32)$$

$$\leq (a^{\text{in}} - a^{\text{out}})(k' - k_*) + b^{\text{in}} - b^{\text{out}} + 1 \quad (33)$$

$$\leq (a^{\text{in}} - a^{\text{out}})(k' + 1) + b^{\text{in}} - b^{\text{out}} + 1 \quad (34)$$

so that for $D(k') = \min(E_{k'})$ and $k_* > 0$, our policy satisfies item 2). Hence, items 1) and 2) are satisfied by our policy.

We now show how to recursively calculate E_k for all $k \in W$. From (26), we can form Tables I and II. From the form of the expressions in Tables I and II, notice that $E_{k+1}^{k'} = E_k^{k'} + a^{\text{out}} - D(k)$ for all k', $k' \leq k$, so that $\min(E_{k+1} - \{E_{k+1}^{k+1}\}) = \min(E_k) + a^{\text{out}} - D(k)$. It is clear, then, that if we define the set

$$\bar{E}_k = \{\min(\bar{E}_{k-1}) + a^{\text{out}} - D(k-1)\} \cup \{\text{ceil}(a^{\text{out}} + b^{\text{out}})\} \quad (35)$$

for all $k \in W$, $k \neq 0$ and if we let $\bar{E}_0 = \{\text{ceil}(a^{\text{out}} + b^{\text{out}})\}$, then $\min(E_k) = \min(\bar{E}_k)$. Hence, we can define our policy for all $k \in W$ as $D(k) = \min\{\bar{E}_k \cup \{A(k) + x(k)\}\}$.

VII. Concluding Remarks

We have presented a new stability analysis of the CAF policy focusing on its transient behavior, shown that the CAOB and RPS policies are stable, and provided a stable implementation for a stream modifier. We have considered only the deterministic case throughout the paper. It would be interesting to study a stochastic version of the problem, for example for the case of failure-prone machines. Also, it is an important open question whether the policies proposed in this paper (and others) can help improve the performance of an FMS.

References

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

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

[3] C. Humes, Jr., "A regulator stabilization technique: Kumar–Seidman revisited," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 191–196, Jan. 1994.

[4] T. I. Seidman, "'First Come, First Served' Can Be Unstable!" *IEEE Trans. Automat. Contr.*, pp. 2166–2177, Oct. 1994.

[5] R. L. Cruz, "A calculus for network delay—Part I: Network elements in isolation," *IEEE Trans. Inform. Theory*, vol. 37, pp. 114–131, Jan. 1991.

[6] J. R. Perkins, C. J. Humes, Jr., and P. Kumar, "Distributed scheduling of flexible manufacturing systems: Stability and performance," *IEEE Trans. Robotics Automation*, vol. 10, pp. 133–141, Apr. 1994.

[7] K. M. Passino, K. Burgess, and A. N. Michel, "Lagrange stability and boundedness of discrete event systems," *J. Discrete Event Dynamic Syst.: Theory Appl.*, vol. 5, pp. 383–403, 1995.

[8] S. H. Lu and P. Kumar, "Distributed scheduling based on due dates and buffer priorities," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 1406–1416, Dec. 1991.

Comments on "A New Controller Design for a Flexible One Link Manipulator"

Susy Thomas and B. Bandyopadhyay

***Abstract*—In the above-mentioned paper[1] a variable structure sliding mode controller (VSSMC) design for the tip position control of a flexible one-link manipulator has been presented, where a switching line constructed from the tip position and its derivative was employed for the design. The claim was that if the slope of this line is chosen positive and the system variables are made to stay on this line, they will converge to zero exponentially, thus yielding a stable system in sliding mode (SM). The purpose of this comment is to show that the choice of a positive constant as the slope for this switching line will not guarantee the stability of the system in SM because, in view of the functional relationship of the tip position with the generalized coordinates of the system through the mode shape functions, what is presented as a switching line is in fact a switching hypersurface. Hence, the stability of the system in SM is guaranteed only if the motion on this hypersurface is asymptotically stable. A positive value for the slope of the switching line employed by Qian and Ma will not guarantee this stability because variations in the mode shape functions due to varying payload conditions or other disturbances at the tip will lead to a varying switching surface. These variations can be such that the resulting sliding motion becomes unstable. Also, the varying switching surface implies that the controller will fail to maintain sliding mode motion.**

***Index Terms*—Flexible manipulator, sliding mode, variable structure control.**

I. Introduction

From (12) of the above-mentioned paper,[1] the functional relationship between the tip position and the generalized coordinates considering only the first two vibratory modes is

$$y_{TP} = Lq_0 + \phi_1(L)q_1 + \phi_2(L)q_2 \quad (1)$$

where q_i denotes the generalized coordinates and $\phi_i(L)$ the mode shape functions of the flexible arm.

In the paper,[1] Qian and Ma define the tip position error x_1 as the difference between the current tip position y_{TP} and the set point x_{ST}. Without loss of generality it was assumed that

$$x_{ST} = 0. \quad (2)$$

Hence

$$x_1 = y_{TP} \quad (3)$$

and x_2 was defined as

$$x_2 = \dot{x}_1 = \dot{y}_{TP}. \quad (4)$$

The switching line was constructed as

$$S = x_2(t) + Cx_1(t) = 0. \quad (5)$$

Manuscript received April 4, 1995; revised November 15, 1995.

S. Thomas is with the Department of Electrical Engineering, IIT Bombay, Bombay 400076 India. She is on deputation from Calicut Regional Engineering College, Calicut, Kerala 673601, India.

B. Bandyopadhyay is with the Lehrstuhl für Elektrische Steuerung und Regelung, Ruhr-Universität Bochum, Bochum 44780 Germany.

Publisher Item Identifier S 0018-9286(97)01331-7.

[1] W. T. Qian and C. C. H. Ma, *IEEE Trans. Automat. Contr.*, vol. 37, pp. 132–137, 1992.