Proceedings of the 2006 American Control Conference
Minneapolis, Minnesota, USA, June 14-16, 2006

Feedback control of 2-product server with setups and bounded buffers

J.A.W.M. van Eekelen, E. Lefeber and J.E. Rooda

Abstract— **A manufacturing machine processing two product types arriving at constant rate and setup times involved is considered in this study. An optimal process cycle is derived with respect to minimal weighted time averaged work in process (wip) level. In addition, a feedback law is proposed that steers the system to this optimal process cycle from arbitrary start point. The analysis has been done for both unbounded and bounded buffer capacity. Although the analysis is done for continuous models, the feedback law has been implemented successfully in a discrete event simulation.**

I. INTRODUCTION

In a lot of applications servers have to share capacity over competing resources. One could think of (Internet) communication, traffic lights, call-centers, or manufacturing systems. Switching between resources might take time. In this paper we consider a server in a manufacturing context, serving two job types, with nonzero setup times. We regard the scheduling problem in which a minimal (weighted) mean work-in-process (wip) level is looked for, assuming no setup costs. First, an optimal process cycle is determined. Then, a feedback policy is proposed that brings the trajectory to this optimal cycle. Next, the optimal cycle and feedback policy are determined for the situation with finite buffer capacity.

For deterministic systems, Savkin regards a similar type of problem [12], [13], where a switched server system serves n queues and stable limit cycle solutions are found with minimal cycle period. The control policy consists of fixed process periods for each job type before a setup to another type is performed. The schedule therefore is predetermined. The analysis is done for networks. The control policy does not account for uncertainties or perturbations. In this paper we determine a control action which depends on the state of the system.

For queueing systems, Boxma et al. [3], [4] provide some papers in which they propose a certain policy for getting a stable process cycle. In [11] systems are analyzed under a heavy traffic assumption. In most work, analysis and optimization is done within the given policy. Clearing policies (serve a queue until it is empty then switch to another queue) of threshold services (serve a queue until a value has been reached) are mostly considered in this area for both stochastic and deterministic environments, while ‘idling’ is not allowed. In this paper however, we do not start from a policy but we start with an objective and then propose a policy, in which we find trajectories with ‘idling’. Related work but then for stochastic systems has been done by Hofri and Ross [9]. However, they restrict themselves to equal maximum process rates for both product types. Moreover, they perform the analysis with infinite buffer capacities, whereas we consider finite buffer capacities.

Lan and Olsen [10] present a fluid model for a multi-product server which has to choose between competing queues (polling). They define a convex optimization problem which determines a, not necessarily achievable, lower bound on the waiting costs in a deterministic environment with both setup times and setup costs. They also claim that the polling table resulting from their optimization problem is rare to find and in most cases unachievable. Therefore, they propose heuristics to get close to the lower bound. In this paper however, we present a perfectly achievable optimal deterministic process cycle for 2 product types, even in an environment with limited buffer capacity (Lan and Olsen [10] assume infinite buffers).

A well known scheduling heuristic is based on the $c\mu$-rule (see e.g. [5]) where switching (without switchover times) takes place according to a $c\mu$ index where c is some cost rate and μ a process rate. Job types with highest indices have priority. In this paper, we derive an optimal process curve, in which a ‘slow mode’ may occur (also referred to as ‘idling’ [6] or ‘cruising’ [10]). In this mode, lots are processed at a lower rate, equal to their arrival rate until the other queue reaches some value. One could say that jobs of the former queue have higher priority than switching to the latter. If the ‘slow mode’ occurs, it takes place at the queue with the highest $c\lambda$ index, even if the $c\mu$ index of the other job type is higher, as is shown in an example in Section IV.

A different topic is studying optimal transient behavior: how to reach the steady state in an optimal way. Less attention is paid to deriving the actual optimal process cycle. The ‘slow mode’ had not been recognised by Connolly et al. in [7] and Boccadoro and Valigi only study symmetric systems in [2]. Although their work on transient behavior is interesting, this study focuses on the optimal steady state process cycle rather than optimal transient behavior.

The remainder of this paper is organized as follows. In Section II, we present the optimal steady state process cycle with respect to time averaged work in process (wip) level. Next we propose a feedback policy that brings a process trajectory to this optimal curve. The analysis is done based on continuous fluid models. In Section III we analyze the system with finite buffers. The optimal process cycle might change due to this limitation. The feedback policy is adjusted in a way that buffer capacities are never violated. In Section IV the feedback controller is successfully implemented in a discrete event example.

All authors are with the Department of Mechanical Engineering, Technische Universiteit Eindhoven, The Netherlands,
[j.a.w.m.v.eekelen,a.a.j.lefeber,j.e.rooda]@tue.nl

1-4244-0210-7/06/$20.00 ©2006 IEEE

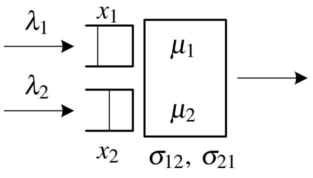

Fig. 1. Manufacturing machine.

II. INFINITE BUFFERS ANALYSIS

We consider a manufacturing workstation as shown in Figure 1. Two different product types arrive at the workstation. The arrival stream is considered to be a continuous constant flow with rates $\lambda_1 > 0$ and $\lambda_2 > 0$ (in lots/hour) for products of type 1 and products of type 2 respectively. Products (lots) are stored in infinite buffers. The numbers of lots in the buffers are $x_1 \geq 0$ and $x_2 \geq 0$ respectively. The machine processes one product type at a time. The process rate for product type i is $\mu_i > 0$ (in lots/hour) with $i \in \{1,2\}$. Switching from product type 1 to 2 takes σ_{12} hours and σ_{21} hours in opposite direction. We define the utilization for product type i as

$$\rho_i = \frac{\lambda_i}{\mu_i} \text{ with } i \in \{1,2\}. \tag{1}$$

To be able to reach a steady state situation, the machine capacity must not be exceeded. This implies that

$$\sum_i \rho_i < 1. \tag{2}$$

This sum must be strictly less than 1, since setup times are involved. Objective is to reach a steady state cycle with minimum time averaged (weighted) wip level. The weighted time averaged wip level J is defined as:

$$J = \frac{1}{T}\int_0^T c_1 x_1(s) + c_2 x_2(s) \mathrm{d}s \tag{3}$$

where c_i is a weighting factor for queue length x_i with $i \in \{1,2\}$ and T denotes the period of the process cycle. A steady state point can never be reached because of the nonzero setup times involved.

Without loss of generality we assume that $c_1\lambda_1 \geq c_2\lambda_2$. Moreover, we define σ=σ_{12}+σ_{21}, the sum of the setup times.

Lemma 2.1: The steady state cycle which has the form depicted in Figure 2 minimizes the time averaged weighted work-in-process level.

In the left part of Figure 2, the queue lengths x_1 and x_2 are on the axes, the arrows indicate the direction of the cycle. During $\tau_1 > 0$, lots of type 1 are processed at rate μ_1. During $\tau_2 > 0$, lots of type 2 are processed at rate μ_2. During $\tau_3 \geq 0$, lots of type 1 are processed at rate λ_1. Setups from type 1 to 2 and vice versa have a duration of $\sigma_{12} > 0$ and $\sigma_{21} > 0$ hours respectively. On the right, the queue lengths are plotted against time for one period.

Proof: A first observation is that for an optimal steady state cycle, the machine either works at maximal rate μ_i or at the arrival rate λ_i in case $x_i = 0$, cf. [10, Proof of Lemma 2]. A second observation is that for an optimal steady state cycle both buffers will be emptied. Therefore, it only remains to be shown that an optimal steady state cycle does not contain a slow mode for each product type. This can be shown analytically along the lines of the proof of Lemma 2.2, but also made clear from the following reasoning. If the optimal steady state cycle would have a slow mode for each product type, we can reduce the duration of both slow modes. As a result both the mean wip-level of type 1 and type 2 reduces. Since we assumed to have no setup costs, this results in lower costs J.

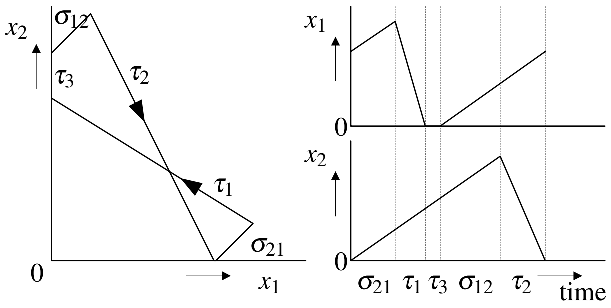

Fig. 2. Trajectory of steady state cycle.

An other way to prove this lemma is to use the fact that in [10, Theorem 1] a lower bound on the optimal costs J has been given by minimizing the time averaged wip level for each product type separately, ignoring the interplay between different product types. This lower bound might not be achievable, since product types do interact. However, the trajectory of Figure 2 minimizes the time averaged wip level for each product type separately and achieves the lower bound derived in [10]. Therefore, this trajectory minimizes the time averaged work-in-process level. ■

In the remainder of this paper, processing lots of type 1 or type 2 is abbreviated as ① and ② respectively. Performing a setup to type 1 or type 2 is abbreviated as ❶ and ❷ respectively. The system is always in 1 of the 4 modes {①,②,❶,❷}. Notice that when the system is in ①, it might be processing at rate μ_1 or λ_1. The latter mode is referred to as the 'slow mode'.

Lemma 2.2: For the minimal time averaged weighted wip level steady state cycle, we have a 'slow mode', i.e. $\tau_3 > 0$, iff $c_1\lambda_1(\rho_1 + \rho_2) + (c_2\lambda_2 - c_1\lambda_1)(1-\rho_2) < 0$.

Proof: In steady state, the system reaches the same situation after completing one (periodic) cycle. So during processing at full rate, as many lots are processed as arrive during setups and processing of the other type. In other words:

$$\begin{aligned} \lambda_1(\sigma_{12} + \sigma_{21} + \tau_2) &= (\mu_1 - \lambda_1)\tau_1 \\ \lambda_2(\sigma_{12} + \sigma_{21} + \tau_1 + \tau_3) &= (\mu_2 - \lambda_2)\tau_2. \end{aligned} \tag{4}$$

We define: $\tau_3 = \alpha(\sigma_{12} + \sigma_{21})$ with $\alpha \geq 0$. Using (4) expressions can be derived for all τ's and setup times:

$$\begin{bmatrix}\sigma\\ \tau_1\\ \tau_2\\ \tau_3\end{bmatrix} = \frac{\sigma_{12}+\sigma_{21}}{1-\rho_1-\rho_2}\begin{bmatrix}1-\rho_1-\rho_2\\ \rho_1(1+\alpha\rho_2)\\ \rho_2(1+\alpha(1-\rho_1))\\ \alpha(1-\rho_1-\rho_2)\end{bmatrix} \tag{5}$$

We determine the mean wip (as defined in (3)) by computing the area of the right hand graphs of Figure 2 and divide it by the total period T:

$$\begin{aligned}\frac{1}{T}\int_0^T x_1(s)\mathrm{d}s &= \frac{\frac{1}{2}\cdot(\sigma+\tau_1+\tau_2)\cdot(\mu_1-\lambda_1)\tau_1}{\sigma+\tau_1+\tau_2+\tau_3}\\ \frac{1}{T}\int_0^T x_2(s)\mathrm{d}s &= \frac{\frac{1}{2}\cdot(\sigma+\tau_1+\tau_2+\tau_3)\cdot(\mu_2-\lambda_2)\tau_2}{\sigma+\tau_1+\tau_2+\tau_3}\end{aligned} \tag{6}$$

Using (1), (5) and (6) the time averaged weighted wip level (3) can be written as:

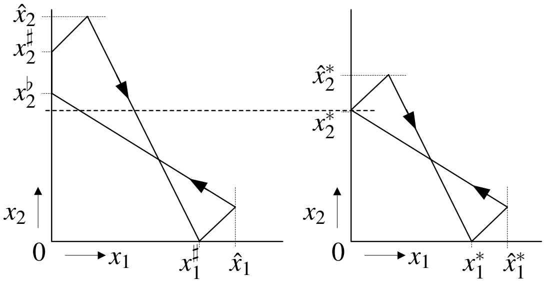

Fig. 3. Truncated bow tie (left) and bow tie (right).

$$J=\sigma\frac{c_1\left[1{+}\alpha\rho_2\right]^2\lambda_1(1-\rho_1)+c_2\left[1{+}\alpha(1-\rho_1)\right]^2\lambda_2(1-\rho_2)}{2(1-\rho_1-\rho_2)(1+\alpha(1-\rho_1))}. \tag{7}$$

Minimizing J with respect to α gives:

$$\begin{aligned}&\min_{\alpha}\frac{c_1\left[1+\alpha\rho_2\right]^2\lambda_1(1-\rho_1)+c_2\left[1+\alpha(1-\rho_1)\right]^2\lambda_2(1-\rho_2)}{1+\alpha(1-\rho_1)}\\ &\text{s.t. } \alpha\geq 0.\end{aligned} \tag{8}$$

From $\frac{\mathrm{d}J}{\mathrm{d}\alpha}=0$ we obtain:

$$\begin{aligned}&\left[c_1\lambda_1\rho_2^2(1-\rho_1)+c_2\lambda_2(1-\rho_1)^2(1-\rho_2)\right]\alpha^2+\ldots\\ &\quad 2\left[c_1\lambda_1\rho_2^2+c_2\lambda_2(1-\rho_1)(1-\rho_2)\right]\alpha+\ldots\\ &\quad\quad \left[c_1\lambda_1(\rho_1+\rho_2)+(c_2\lambda_2-c_1\lambda_1)(1-\rho_2)\right]=0.\end{aligned} \tag{9}$$

The coefficients in front of α^2 and α are both strictly positive, so this parabola has a positive real root iff $c_1\lambda_1(\rho_1+\rho_2)+(c_2\lambda_2-c_1\lambda_1)(1-\rho_2)<0$. Note that this is only possible if $c_1\lambda_1>c_2\lambda_2$. The value of α can be obtained by solving (9) taking into account that it has to be non-negative:

$$\alpha=\begin{cases}0 \text{ if } c_1\lambda_1(\rho_1+\rho_2)+(c_2\lambda_2-c_1\lambda_1)(1-\rho_2)\geq 0\\ \text{positive real root of (9) otherwise.}\end{cases} \tag{10}$$

■

Summarizing from Lemma 2.1 and Lemma 2.2:

Proposition 2.1: The optimal steady state periodic cycle for a machine processing two constantly arriving product flows with respect to minimal time averaged wip level has period 1 and the shape of Figure 2. The slow mode ($\tau_3>0$) only occurs when $c_1\lambda_1(\rho_1+\rho_2)+(c_2\lambda_2-c_1\lambda_1)(1-\rho_2)<0$.

In addition to the optimal process cycle, we also consider the steady state trajectory that has minimal extreme values for x_1 and x_2. For this cycle, the slow mode does not occur (i.e. $\tau_3=0$). Both curves are shown in Figure 3. We refer to these curves as the truncated bow tie (left) and bow tie (right). Notice that even though the truncated bow tie always lies above the pure bow tie, it has a smaller mean wip level.

Both trajectories have some interesting points that we refer to in the remainder of this paper. We use hats ($\hat{}$) for denoting extreme values and stars (*) for points on the pure bow tie trajectory. Furthermore, coordinates with flat ($^\flat$) and sharp ($^\sharp$) symbols denote the points where the slow mode starts and ends respectively. On the x_1-axis, these coordinates are equal and denoted by $x_1^\sharp$ (cf. Figure 3). The coordinates of the interesting points of the trajectory are given by:

$$\begin{aligned}x_2^\flat &= \lambda_2(\sigma_{21}+\tau_1)\\ &= \lambda_2\sigma_{21}+\lambda_2\sigma\frac{\rho_1(1+\alpha\rho_2)}{1-\rho_1-\rho_2}\\ x_2^\sharp &= \lambda_2(\sigma_{21}+\tau_1+\tau_3)\\ &= \lambda_2\sigma_{21}+\lambda_2\sigma\frac{\alpha(1-\rho_1)(1-\rho_2)+\rho_1}{1-\rho_1-\rho_2}\\ \hat{x}_2 &= \lambda_2(\sigma_{21}+\tau_1+\tau_3+\sigma_{12})\\ &= \lambda_2\sigma\left(1+\frac{\alpha(1-\rho_1)(1-\rho_2)+\rho_1}{1-\rho_1-\rho_2}\right)\end{aligned} \tag{11}$$

with α as stated in (10). Note that if $\tau_3=0$, the pure bow tie trajectory is obtained: $x_2^\sharp=x_2^\flat=x_2^*$ and consequently $x_1^\sharp=x_1^*$. As can be seen in (11) and in Figure 3, $x_2^*\leq x_2^\flat\leq x_2^\sharp$. We use this property in the proof of the feedback control policy in Proposition 2.2. For the interesting points on the x_1-axis, the coordinates can be determined in a similar way.

Now that we characterized the optimal trajectory and the trajectory with minimal extreme values, we want to bring trajectories from arbitrary start points to the optimal trajectory.

Proposition 2.2: The following feedback control law brings the system of Figure 1 to the optimal periodic cycle with respect to minimal time averaged wip level. Dependent on the state of the system, the controller is in 1 of the 6 modes initially. This follows trivially from the controller mode descriptions.

- Mode 1: ① at μ_1 as long as $x_1>0$, then go to Mode 2.
- Mode 2: ① at λ_1 as long as $x_2<x_2^\sharp$, then go to Mode 3.
- Mode 3: perform ❷, after σ_{12} go to Mode 4.
- Mode 4: ② at μ_2 as long as $x_2>0$, then go to Mode 5.
- Mode 5: ② at λ_2 as long as $x_1<x_1^\sharp$, then go to Mode 6.
- Mode 6: perform ❶, after σ_{21} go to Mode 1.

Remark: Mode 2 and Mode 5 might have a duration of zero.

Proof: Assume the n^{th} start of ❷ is from coordinate $(0,x_2^n)$.

Now we wonder from which coordinate the $(n+1)^{\text{st}}$ start of ❷ takes place. Assuming a duration of zero in both Mode 2 and Mode 5 we obtain:

$$\begin{aligned}
(0, x_2^n) &\xrightarrow{❷} (\lambda_1 \sigma_{12}, x_2^n + \lambda_2 \sigma_{12}) \\
&\xrightarrow{②} \left(\lambda_1 \left(\sigma_{12} + \frac{x_2^n + \lambda_2 \sigma_{12}}{\mu_2 - \lambda_2} \right), 0 \right) \\
&\xrightarrow{❶} \left(\lambda_1 \left(\sigma + \frac{x_2^n + \lambda_2 \sigma_{12}}{\mu_2 - \lambda_2} \right), \lambda_2 \sigma_{21} \right) \qquad (12) \\
&\xrightarrow{①} \left(0, \lambda_2 \left(\sigma_{21} + \frac{\lambda_1 \left(\sigma + \frac{x_2^n + \lambda_2 \sigma_{12}}{\mu_2 - \lambda_2} \right)}{\mu_1 - \lambda_1} \right) \right)
\end{aligned}$$

which results in:

$$\begin{aligned}
x_2^{n+1} &= \lambda_2 \left(\sigma_{21} + \frac{\lambda_1 \left(\sigma + \frac{x_2^n + \lambda_2 \sigma_{12}}{\mu_2 - \lambda_2} \right)}{\mu_1 - \lambda_1} \right) \\
&= \frac{\rho_1 \rho_2}{(1-\rho_1)(1-\rho_2)} (x_2^n - x_2^*) + x_2^*.
\end{aligned} \qquad (13)$$

In case it happens that either Mode 2 or Mode 5 has a strictly positive duration we end up on the optimal curve and get:

$$x_2^{n+1} = x_2^\sharp \qquad (14)$$

Combining (13) and (14) results in:

$$x_2^{n+1} = \max \left(\frac{\rho_1 \rho_2}{(1-\rho_1)(1-\rho_2)} (x_2^n - x_2^*) + x_2^*, \, x_2^\sharp \right) \qquad (15)$$

which has a solution

$$x_2^n = \max \left(\left[\frac{\rho_1 \rho_2}{(1-\rho_1)(1-\rho_2)} \right]^n (x_2^0 - x_2^*) + x_2^*, \, x_2^\sharp \right). \qquad (16)$$

Since

$$0 < \frac{\rho_1 \rho_2}{(1-\rho_1)(1-\rho_2)} = 1 - \frac{1 - \rho_1 - \rho_2}{(1-\rho_1)(1-\rho_2)} < 1 \qquad (17)$$

we get

$$\lim_{n \to \infty} x_2^n = \max(x_2^*, \, x_2^\sharp) = x_2^\sharp. \qquad (18)$$

Note that for $x_2^* < x_2^\sharp$ (i.e. $\tau_3 > 0$) as well as $x_2^0 < x_2^\sharp$, we obtain convergence in finite time. ■

Remark: Hofri and Ross [9] also use a similar double threshold policy: exhaustive processing and switch only whenever the other buffer length has reached some value.

In the next sections we consider finite buffers. The feedback control law needs to be adjusted and the optimal process cycle might be influenced.

III. FINITE BUFFERS ANALYSIS

In the previous section, the optimal process cycle with respect to wip level and a feedback have been proposed for the situation with infinite buffer capacity. In the remainder we extend the feedback to handle finite buffer capacities.

The maximum buffer capacities are denoted as $x_1^{\max}$ and $x_2^{\max}$. Notice that we need $x_1^{\max} \geq \hat{x}_1^*$ and $x_2^{\max} \geq \hat{x}_2^*$ since otherwise no periodic cycle exists which meets the buffer constraints. If the optimal process cycle (as determined in Section II) exceeds the buffer capacities $x_1^{\max}$ or $x_2^{\max}$, a new optimal process cycle must be constructed meeting the buffer bounds. Note that this can only happen if $\tau_3 > 0$, because if $\tau_3 = 0$, the optimal process cycle already equals the cycle with minimal extreme values.

So assume that $x_1^{\max} < \hat{x}_1$ or $x_2^{\max} < \hat{x}_2$. Figure 4 shows how the coordinates characterizing the (possibly) new optimal curve are determined. These coordinates (using bars ($\bar{\ }$) to indicate that we refer to the bounded buffer situation) are:

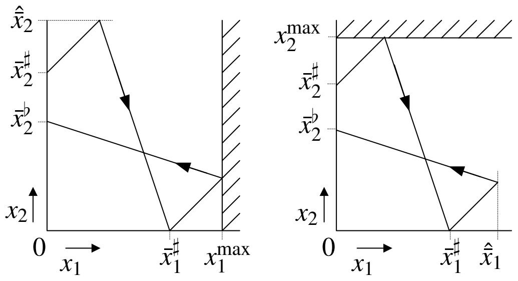

Fig. 4. Optimal curve subject to buffer constraints.

$$\begin{aligned}
\bar{x}_1^\sharp &= \min \left(x_1^{\max} - \lambda_1 \sigma_{21}, \, \lambda_1 \left(\sigma_{12} + \frac{x_2^{\max}}{\mu_2 - \lambda_2} \right), \, x_1^\sharp \right) \\
\hat{\bar{x}}_1 &= \min \left(x_1^{\max}, \, \lambda_1 \left(\sigma + \frac{x_2^{\max}}{\mu_2 - \lambda_2} \right), \, \hat{x}_1 \right) \\
\bar{x}_2^\flat &= \min \left(\lambda_2 \left(\sigma_{21} + \frac{x_1^{\max}}{\mu_1 - \lambda_1} \right), \right. \\
&\qquad \left. \lambda_2 \left(\sigma_{21} + \frac{\lambda_1}{\mu_1 - \lambda_1} \left(\sigma + \frac{x_2^{\max}}{\mu_2 - \lambda_2} \right) \right), \, x_2^\flat \right) \\
\bar{x}_2^\sharp &= \min \left(\frac{\mu_2 - \lambda_2}{\lambda_1} (x_1^{\max} - \lambda_1 \sigma) - \lambda_2 \sigma_{12}, \, x_2^{\max} - \lambda_2 \sigma_{12}, \, x_2^\sharp \right) \\
\hat{\bar{x}}_2 &= \min \left(\frac{\mu_2 - \lambda_2}{\lambda_1} (x_1^{\max} - \lambda_1 \sigma), \, x_2^{\max}, \, \hat{x}_2 \right).
\end{aligned} \qquad (19)$$

Note that if $\tau_3 = 0$, $\bar{x}_2^\flat = \bar{x}_2^\sharp = x_2^\flat = x_2^\sharp = x_2^*$, $\hat{\bar{x}}_2 = \hat{x}_2 = \hat{x}_2^*$, $\bar{x}_1^\sharp = x_1^\sharp = x_1^*$ and $\hat{\bar{x}}_1 = \hat{x}_1 = \hat{x}_1^*$.

Before we adjust the feedback control policy to handle buffer bounds properly, we first look closer to the x_1-x_2-plane. We identify some regions in this plane from which it is either possible or impossible to get on a steady state process cycle.

Lemma 3.1: Regardless of the feedback policy, the x_1-x_2 space can be divided into regions from which it is impossible to reach the steady state process cycle when in a certain mode, see Figure 5. The regions marked with ①† and ②† indicate that if the trajectory enters that region processing ① or ② respectively, the trajectory becomes infeasible (i.e. the buffer constraint will be violated). If the trajectory is on the bow tie shifted into the upper right corner of the x_1-x_2 space, the trajectory stays there.

Proof: It is easy to see that once the trajectory has crossed the dashed lines (Fig. 5), a setup will cause the buffer constraint to be violated. For the regions in or above the upper-right bow tie, the proof is included in the feedback control law proof. ■

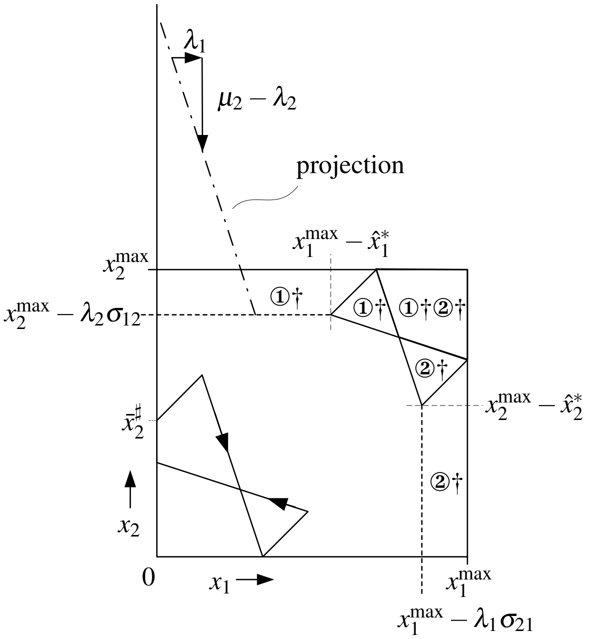
λ_1
$\mu_2 - \lambda_2$
projection
$x_1^{\max} - \hat{x}_1^*$
$x_2^{\max}$
$x_2^{\max} - \lambda_2\sigma_{12}$
①†
①†
①†②†
②†
$x_2^{\max} - \hat{x}_2^*$
$\bar{x}_2^\sharp$
②†
x_2
0
x_1
$x_1^{\max}$
$x_1^{\max} - \lambda_1\sigma_{21}$

Fig. 5. Feasible and infeasible regions for constrained problem.

Now that we have defined the optimal process cycle under buffer constraints and having identified infeasible regions, we can adjust the feedback in a way that it brings a trajectory from arbitrary start point to the (new) optimal process cycle.

Proposition 3.1: The feedback that stabilizes a trajectory to the optimal process cycle if started from a feasible start point (see Section III and Fig. 5), consists of 6 modes. Dependent on the state of the system, the controller is in one of these modes, which follows trivially from the mode description.

- Mode 1: ① at μ_1 as long as $x_1 > 0$ and $x_2 < x_2^{\max} - \lambda_2\sigma_{12}$, then go to Mode 2.
- Mode 2: ① at λ_1 as long as $x_2 < \bar{x}_2^\sharp$, then go to Mode 3.
- Mode 3: perform ❷, after σ_{12} go to Mode 4.
- Mode 4: ② at μ_2 as long as $x_2 > 0$ and $x_1 < x_1^{\max} - \lambda_1\sigma_{21}$, then go to Mode 5.
- Mode 5: ② at λ_2 as long as $x_1 < \bar{x}_1^\sharp$, then go to Mode 6.
- Mode 6: perform ❶, after σ_{21} go to Mode 1.

Proof: Similar to the proof of the case with unbounded buffers we are interested where the $(n+1)^{\text{st}}$ start of ❷ takes place given from which coordinate the n^{th} start of ❷ took place. With a point $(x_1,\, x_2^{\max} - \lambda_2\sigma_{12})$ we associate the point $(0,\, x_2)$ where x_2 is given by:

$$x_2 = \frac{\mu_2 - \lambda_2}{\lambda_1} x_1 + (x_2^{\max} - \lambda_2\sigma_{12}) \tag{20}$$

and vice versa. The choice of (20) has been made in such a way that it is the point in the unbounded x_1-x_2-plane from where ❷ and then ② results in the same trajectory in the feasible area (i.e. within the buffer bounds). See also Figure 5 for the projection. So given x_2^n we are interested in x_2^{n+1}. In case we do not suffer from the buffer constraints, we obtain (15) again. However, in case one or two buffer constraints become active, the resulting x_2^{n+1} will be larger, since switching earlier makes the system move along a line which is located higher. For the case one constraint becomes active, we introduce an auxiliary variable Z whose value depends on which constraint is active.

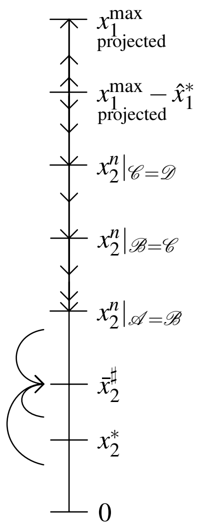
$x_1^{\max}$ projected
$x_1^{\max} - \hat{x}_1^*$ projected
$x_2^n|_{\mathscr{C}=\mathscr{D}}$
$x_2^n|_{\mathscr{B}=\mathscr{C}}$
$x_2^n|_{\mathscr{A}=\mathscr{B}}$
$\bar{x}_2^\sharp$
x_2^*
0

Fig. 6. Evolution of x_2^n at start of ❷.

$$Z = \min\left(\underbrace{\frac{\mu_2 - \lambda_2}{\lambda_1}(x_1^{\max} - \lambda_1\sigma) - \hat{x}_2^*}_{\text{is the smallest if } x_1^{\max} \text{ active}},\ \underbrace{\frac{\mu_1 - \lambda_1}{\lambda_2}(x_2^{\max} - \lambda_2\sigma) - \hat{x}_1^*}_{\text{is the smallest if } x_2^{\max} \text{ active}} \right) \tag{21}$$

We now distinguish 4 situations:

$\mathscr{A}$: no active constraints, iteration as in (14);
$\mathscr{B}$: no active constraints, iteration as in (12);
$\mathscr{C}$: one active buffer constraint during iteration;
$\mathscr{D}$: two active buffer constraints during iteration.

The endpoint of one iteration using the feedback law of Proposition 3.1 now becomes

$$x_2^{n+1} = \max\begin{pmatrix} \bar{x}_2^\sharp & (\mathscr{A}) \\ \frac{\rho_1\rho_2}{(1-\rho_1)(1-\rho_2)}(x_2^n - x_2^*) + x_2^* & (\mathscr{B}) \\ x_2^n - \frac{1-\rho_1-\rho_2}{(1-\rho_1)(1-\rho_2)} \cdot Z & (\mathscr{C}) \\ \frac{(1-\rho_1)(1-\rho_2)}{\rho_1\rho_2}(x_2^n - [x_1^{\max} - \hat{x}_1^*]) + [x_1^{\max} - \hat{x}_1^*] & (\mathscr{D}) \end{pmatrix} \tag{22}$$

where the calligraphic capital refers to 1 of the 4 situations. The evolution of an arbitrary point $(0,\, x_2^n)$ along the $x_1 = 0$ axis where ❷ starts can now be visualized, see Figure 6. The arrows indicate the direction in which x_2^n evolves. The distance between the arrows is a measure for the rate of the evolution. First consider being in region $\mathscr{D}$. Note that for $x_2^n > x_1^{\max} - \hat{x}_1^*$ (projected), i.e. we start to the right from the upper right bow tie (Figure 5), we have divergence, since $\frac{(1-\rho_1)(1-\rho_2)}{\rho_1\rho_2} > 1$, cf. (17). When starting on the upper right bow tie, we stay on it. This completes the proof of Lemma 3.1. For $x_2^n < x_1^{\max} - \hat{x}_1^*$ we have divergence from the upper right bow tie in the correct direction, i.e. towards the bottom left, and we will leave region $\mathscr{D}$ after a finite number of steps. If we are in $\mathscr{C}$ we also move in the correct direction with constant jumps and therefore after a finite number of steps leave region $\mathscr{C}$. So after a finite number of steps we will be in either region $\mathscr{A}$ or region $\mathscr{B}$. From the analysis in the previous section, convergence to $\bar{x}_2^\sharp$ follows. ■

IV. DISCRETE EVENT EXAMPLE

Although the analysis in this paper has been done for continuous (fluid) models, the feedback control law as proposed in Proposition 3.1 has been implemented in a discrete event manufacturing system. The system was modeled in the discrete event specification language χ, see [8], [1]. The settings that have been used are in Table I. With these settings, $\alpha = \frac{1}{4}, \bar{x}_1^\sharp = x_1^\sharp = 27, \hat{\bar{x}}_1 = \hat{x}_1 = 45, \bar{x}_2^\sharp = x_2^\sharp = 18, \bar{x}_2^\flat = x_2^\flat = 15$ and $\hat{\bar{x}}_2 = \hat{x}_2 = 24$. The $\bar{x}_2^\sharp$ value is implemented in the discrete event control policy. The results of the simulation are shown in figures 7 (buffer lengths over time) and 8 (trajectory). In this figure, the trajectory goes from light gray to black for better visual understanding. As can be seen, both buffer constraints become active (setup is performed before a buffer has been cleared). Convergence to the optimal cycle is reached.

TABLE I

DISCRETE EVENT SIMULATION SETTINGS.

λ_1:	9 lots/hr.	$x_1^{\max}$:	60 lots
λ_2:	3 lots/hr.	$x_2^{\max}$:	30 lots
μ_1:	24 lots/hr.	x_1^0:	36 lots
μ_2:	27 lots/hr.	x_2^0:	24 lots
σ_{12}:	2 hrs.	initial:	②
σ_{21}:	2 hrs.		

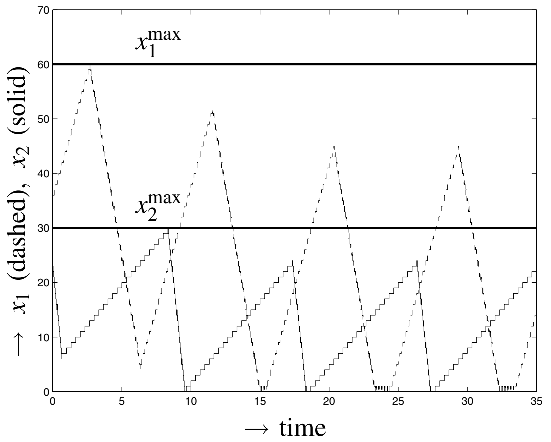

Fig. 7. Simulation results over time.

V. CONCLUSIONS AND FUTURE WORKS

In this paper we studied a switched server manufacturing system for 2 product types with setup times involved. First we determined an optimal process cycle with respect to minimal weighted time averaged work in process level for the case with infinite buffer capacity. For this optimal process cycle, we proposed a feedback control law that brings a trajectory to the optimal trajectory. Then, we considered finite buffers. The optimal process cycle might be adjusted to meet the buffer constraints. This (possibly) new optimal process cycle has been determined prior to the proposal of an adjusted feedback law. This new feedback law takes into account the buffer constraints. Although the complete analysis has been done with continuous (fluid-type) models, the controller has been implemented successfully in a discrete event simulation example.

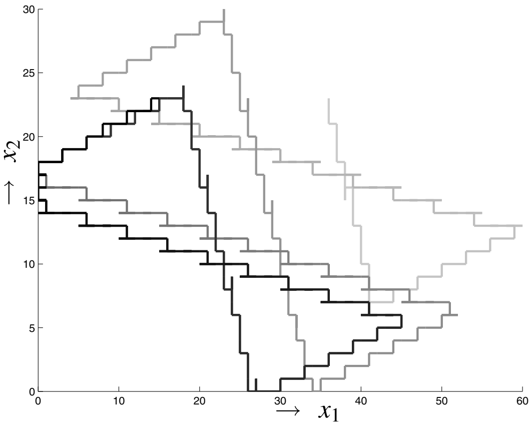

Fig. 8. Simulation results trajectory.

Future work includes extension of the analysis to more product types as well as to queueing networks. Another topic is to perform the analysis for stochastic inter-arrival times or process times. Both research topics should be applied to the case with both infinite and finite buffer capacities.

REFERENCES

[1] D.A. van Beek and J.E. Rooda. Languages and applications in hybrid modelling and simulation: Positioning of Chi. *Control Engineering Practice*, 8(1):81–91, January 2000.

[2] M. Boccadoro and P. Valigi. A modelling approach for the dynamic scheduling problem of manufacturing systems with non negligible setup times and finite buffers. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 5472–5477, 2003.

[3] O.J. Boxma and D.G. Down. Dynamic server assignment in a two-queue model. *European Journal of Operational Research*, 103(3):595–609, 1997.

[4] O.J. Boxma, H. Levy, and J.A. Westrate. Efficient visit frequencies for polling tables: minimization of waiting cost. *Queueing systems theory and applications*, 9(1–2):133–162, 1991.

[5] C. Buyukkoc, P. Varaiya, and J. Walrand. The $c\mu$-rule revisited. *Advances in applied probability*, 17:237–238, 1985.

[6] C. Chase and P.J. Ramadge. On real-time scheduling policies for flexible manufacturing systems. *IEEE transactions on automatic control*, 37(4):491–496, 1992.

[7] S. Connolly, Y. Dallery, and S.B. Gershwin. A real-time policy for performing setup changes in a manufacturing system. In *Proceedings of the 31st IEEE Conf. on Decision and Control*, pages 764–770, 1992.

[8] A.T. Hofkamp and J.E. Rooda. χ reference manual. Internal report Technische Univ. Eindhoven, http://se.wtb.tue.nl/documentation, 2002.

[9] M. Hofri and K.W. Ross. On the optimal control of two queues with server setup times and its analysis. *SIAM Journal on computing*, 16(2):399–420, 1987.

[10] W.L. Lan and T.L. Olsen. Multi-product systems with both setup times and costs: fluid bounds and schedules. *To appear in Operations Research*.

[11] D.M. Markowitz and L.M. Wein. Heavy traffic analysis of dynamic cyclic policies: a unified treatment of the single machine scheduling problem. *Operations Research*, 49(2):246–270, 2001.

[12] A.V. Savkin. Regularizability of complex switched server queueing networks modelled as hybrid dynamical systems. *Systems & Control letters*, 35:291–299, 1998.

[13] A.V. Savkin and A.S. Matveev. A switched server system of order n with all its trajectories converging to $(n-1)!$ limit cycles. *Automatica*, 37(2):303–306, 2001.