A NOTE ON "STABILITY OF CLEARING OPEN LOOP POLICIES IN MANUFACTURING SYSTEMS"

PAULO JOSÉ DA SILVA E SILVA†, MARCELO QUEIROZ†, AND CARLOS HUMES JR.‡

ABSTRACT. The stability of the Clearing Generalized Round-Robin scheduling policy for decentralized manufacturing systems, allowing for self-loops, was established in [2]. In fact, the existence of an unique limit cycle was shown, using subtle facts about eigenvalues of nonnegative matrices. This short-note presents not only a much simpler proof of stability and limit cycles but also tightens the convergence rate to the limit cycle.

AMS Classification: 90B35

1. Introduction and Notation

The relevance of Clearing Round-Robin policies and its natural extension Clearing Generalized Round-Robin (CGRR) has been established in [3], as those policies lead to Pareto-efficient solutions with respect to buffer lengths under certain relationships between the servicing cycle and the system date. In this paper the question of self-loops was not addressed.

The result about the stability of CGRR (in [2]) indicates a slow convergence to the limit cycle (in the case of several self-loops for the same product), besides being extremely involved and using strong lemmas (as [2, Lemma 3], suggested by Seidman).

The question of the convergence rate to the limit cycle is extremely important as this behavior is asymptotic and actual operation of manufacturing systems is associated to a finite horizon. Moreover, information about monotonicity of such convergence is also relevant, as an heuristic basis for storage cost minimization.

We shall study the system composed by one machine capable of processing P products, each of which demanding n_p tasks to be performed. The inputs for the system are assumed to be linear deterministic, with input rate d_p. This means that the total number of parts of product p entering the system up to time t is given by $d_p t$. Each product p runs through tasks $a_1^p, \ldots, a_{n_p}^p$, associated to which there are buffers $b_1^p, \ldots, b_{n_p}^p$ that hold the parts waiting processing. Because of the natural bijection between buffers and tasks, we allow ourselves to use these terms interchangeably.

Date: December 20th 2000.

Key words and phrases. Stability, Clearing Generalized Round-Robin, Limit Cycles, Manufacturing Systems.

†Supported by FAPESP, processo 96/09939-0 and processo 97/06227-2.

‡Partially supported by PRONEX, convênio 76.97.1008.00.

Each buffer b_i^p has a congestion level $\rho_i^p = d_p \times$ service time for a_i^p. We accept the capacity condition (CC),

$$\sum_{p,i} \rho_i^p < 1,$$

that is a natural and usual assumption when studying stability.

In order to clarify the notation presented above, we present an example of a simple manufacturing system. Consider the single machine system shown in Figure 1. This system has two products, $P = \{1, 2\}$. The first product has 3 tasks and the second only 1. We consider the inputs to be linear with rates $d_1 = 2$ parts/min and $d_2 = 4$ parts/min. The service times in minutes and the congestion rates for each task are:

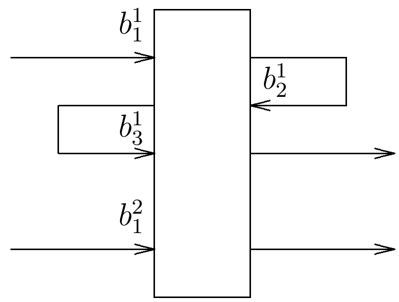

FIGURE 1. Single machine system with two products and two self-loops.

a_1^1	a_2^1	a_3^1	a_1^2
0.1	0.1	0.05	0.1

ρ_1^1	ρ_2^1	ρ_3^1	ρ_1^2
0.2	0.2	0.1	0.4

The clearing generalized Round-Robin (CGRR) policy demands that each task be given a time-slice long enough to empty the contents of its buffer, allowing for repeating buffers in a cycle. This policy is defined by a sequence $\sigma_1, \dots, \sigma_m$ of tasks, such that each task a_i^p appears at least once in the sequence. We assume there is a set-up cost δ_s to switch the machine from task σ_{s-1} to σ_s. We shall write $\rho_s = \rho_i^p$ when $\sigma_s = a_i^p$.

Let t_s^r be the time-slice given to task $\sigma_s = a_i^p$ in cycle r, that is, t_s^r is the amount of time used to empty the contents of the buffer b_i^p in cycle r. Note that the time-slices are naturally ordered as follows:

$$t_1^0, \dots, t_m^0, t_1^1, \dots, t_m^1, t_1^2, \dots, t_m^2, \dots$$

Back to our example, let us consider that the system in Figure 1 operates under a CGRR policy with the following Round-Robin sequence of tasks and set-up times:

σ_1	σ_2	σ_3	σ_4	σ_5
b_1^2	b_1^1	b_3^1	b_2^1	b_1^1

δ_1	δ_2	δ_3	δ_4	δ_5
0.1	0.1	0.2	0.4	0.1

Numbering the time-slices according to the natural order just described, we have the following sequence of time-slices and related buffers:

t_1^0	t_2^0	t_3^0	t_4^0	t_5^0	t_1^1	t_2^1	t_3^1	t_4^1	t_5^1	t_1^2	t_2^2	t_3^2	t_4^2	t_5^2	...
b_1^2	b_1^1	b_3^1	b_2^1	b_1^1	b_1^2	b_1^1	b_3^1	b_2^1	b_1^1	b_1^2	b_1^1	b_3^1	b_2^1	b_1^1	...

The clearing property implies that we can express the duration of a time-slice, t_s^r, as a function of the time-slices in which the parts processed in t_s^r first entered the system. Let us call this set input(t_s^r). The clearing property also ensures that if $t_{l'}^{k'}$ and $t_{\hat{l}}^{\hat{k}}$ denote the first and last elements in input(t_s^r), then input(t_s^r) is composed by all the time-slices in-between these two. Then, $t_s^r = \delta_s + \rho_s(t_{l'}^{k'} + \ldots + t_{\hat{l}}^{\hat{k}})$.

When t_s^r is devoted to an *initial buffer*, i.e., one with external input, it is easy to see that the last and the first time-slices in input(t_s^r) are, respectively, t_s^r itself and the time-slice that follows the last clearing of that initial buffer that precedes t_s^r, i.e.,

$$\text{input}(t_s^r) = \{t_{l'}^{k'}, \ldots, t_s^r\}.$$

The duration of a time-slice t_s^r associated to an *internal buffer* b_i^p, i.e., one that receives its input from another task, depends on the number of parts it receives from the previous task in the p-production line. Because we are dealing with *Generalized* Round-Robin these parts may have entered the buffer b_i^p during various time-slices devoted to the previous task; these time-slices are called *predecessors* of t_s^r, in the sense that they correspond to direct supply of parts to be processed during t_s^r. Applying this idea recursively (by computing predecessors of predecessors and so on), we are able to compute the time-slices associated to the initial task of the p-production line which first processed the parts we are interested in. This will allow us to know eventually the time-slices in which these parts entered the system, i.e., input(t_s^r).

In order to carry such computation for time-slices devoted to internal buffers, we define

$$\text{pred}(t_s^r) = \left\{ t_l^k \mid \sigma_l = a_{i-1}^p \text{ and } \nexists t_j^i,\ s.t.\ \sigma_j = a_i^p \text{ and } t_l^k \le t_j^i < t_s^r \right\}.$$

Applying this definition to the cycle $t_s^2,\ s = 1, \ldots, 5$, of our example, we have:

	t_1^2	t_2^2	t_3^2	t_4^2	t_5^2
pred	$\{t_1^2\}$	$\{t_2^2\}$	$\{t_4^1\}$	$\{t_2^2, t_5^1\}$	$\{t_5^2\}$

To carry the recursion down to the level of the initial buffers, we define pred$^*(t_s^r)$ as the set of time-slices in which the parts were first processed. Clearly, for a time-slice t_s^r devoted to an initial buffer pred$^*(t_s^r) = \{t_s^r\}$. For internal buffers, we have

$$\text{pred}^*(t_s^r) = \bigcup_{t_l^k \in \text{pred}(t_s^r)} \text{pred}^*(t_l^k)$$

Using the above definitions and the fact that for an initial buffer input(t) is known, we can easily see that for internal buffers we have

$$\text{input}(t_s^r) = \bigcup_{t \in \text{pred}^*(t_s^r)} \text{input}(t) \tag{1}$$

In our example we have:

	t_1^2	t_2^2	t_3^2	t_4^2	t_5^2
pred*	$\{t_1^2\}$	$\{t_2^2\}$	$\{t_2^1, t_5^0\}$	$\{t_2^2, t_5^1\}$	$\{t_5^2\}$
input	$\{t_2^1, \ldots, t_1^2\}$	$\{t_1^2, t_2^2\}$	$\{t_3^0, \ldots, t_2^1\}$	$\{t_3^1, \ldots, t_2^2\}$	$\{t_3^2, \ldots, t_5^2\}$

An important property that simplifies the computation of input(t_s^r) is that this set is completely defined by the earliest time-slice $t_{l'}^{k'}$ and the latest time-slice $t_{\hat{l}}^{\hat{k}}$ it contains. This is true as every part entering the system in the time-slices between these two extremes will be processed by the first task of the p-production line during $t_{\hat{l}}^{\hat{k}}$ at the latest, and therefore will be processed thereafter during t_s^r. We have therefore input$(t_s^r) = \{t_{l'}^{k'}, \ldots, t_{\hat{l}}^{\hat{k}}\}$.

Note that the maximum number of cycles one must consider to find the input of all tasks in one cycle, denoted by L, has an upper bound given by the maximum number of tasks per product. In [2] only this upper bound was considered, leading to weaker results than the ones presented here.

2. The Main Theorem

The result presented below will be quite clear to those used to the analysis of the Gauss-Seidel method. To those not so familiar, we suggest referring to the appendix, where the ideas of the proof are illustrated in our example.

Theorem 1. *A machine operating under CGRR, receiving LD inputs such that CC is obeyed, is stable, with unique limit cycle, and the convergence to the limit is R-linear with ratio* $\rho^{\frac{1}{L}}$.

Proof. The key point of the proof is writing the equations defining the duration of a present time-slice, t_s^r, as a function of input$(t_s^r) = \{t_{l'}^{k'}, \ldots, t_{\hat{l}}^{\hat{k}}\}$, as

$$t_s^r = \delta_s + \rho_s(t_{l'}^{k'} + \cdots + t_{\hat{l}}^{\hat{k}}). \tag{2}$$

As the maximum number of cycles one must look back to find input is L, if we denote by $T^k = \left(t_1^{kL}, \ldots, t_m^{kL}, \ldots, t_1^{(k+1)L-1}, \ldots, t_m^{(k+1)L-1}\right)^t$ and by $\Delta = (\delta_1, \ldots, \delta_m, \ldots, \delta_1, \ldots, \delta_m)^t$, we may write (2) as

$$T^{k+1} = LT^{k+1} + UT^k + \Delta,$$

where L is lower-triangular and U is upper-triangular.

In order to analyze the stability and the existence of a limit cycle we shall follow the analysis of the existence (and uniqueness) of a fixed point, as done for the Gauss-Seidel method, by defining $P = L + U$, and studying the system

$$(I - P)T = \Delta. \tag{3}$$

We claim that $(I - P)$ is strictly (column-wise) diagonal dominant. Take the column j associated to t_l^k. Call the parts that entered the system during this time-slice the "red" parts. We claim that the red parts associated to a product p will go through each task of the p-production line exactly once. Because of the clearing property, at the end of each time-slice these parts will be in the same buffer. This means that the term t_l^k will appear at the right-hand side of (2) only once per task, in the equation defining the time-slice in which the red parts are being serviced.

If, for $t_s^r = t_l^k$, t_l^k appears in the right hand side of (2), due to (CC), we get the inequality

$$|(I-P)_{j,j}| = 1-\rho_l > \sum_{s=1, s\neq l}^{m} \rho_s = \sum_{i\neq j} |(I-P)_{i,j}|;$$

otherwise we get

$$|(I-P)_{j,j}| = 1 > \sum_{s=1}^{m} \rho_s = \sum_{i\neq j} |(I-P)_{i,j}|;$$

and so diagonal dominance is proven,

Since $(I-P)$ is strictly diagonal dominant, and the spectral ratio of $(I-P)$ is at most $\rho < 1$, we know that the Gauss-Seidel method for system (3) will converge with rate given by ρ to a vector with mL entries that we call T^*:

$$\| T^{k+1} - T^* \|_\infty \leq \rho \, \| T^k - T^* \|_\infty .$$

Moreover, we will show that T^* is of the form $(t^*, t^*, \dots, t^*)'$, where $t^* \in \mathbb{R}^m$. The rationale to prove this result corresponds to writing the Gauss-Seidel system for the variables $\widehat{T}^k = \left(t^{kL+1}, \dots, t^{(k+1)L-1}, t^{(k+1)L}\right)'$, where $t^{kL+1}, \dots, t^{(k+1)L} \in \mathbb{R}^m$, which is just a shift of T by a full cycle. This shift corresponds to applying Gauss-Seidel to (3) using a different starting point. Therefore the group of variables $t_1^{kL+1}, \dots, t_m^{kL+1}$ ought to converge to the same vector as $t_1^{kL}, \dots, t_m^{kL}$. A trivial inductive argument leads to the uniqueness of the limit cycle of the production system.

The result of convergence follows from

$$\begin{aligned} \| t^k - t^* \|_\infty &\leq \| T^{\lfloor \frac{k}{L} \rfloor} - T^* \|_\infty \\ &\leq \rho^{\lfloor \frac{k}{L} \rfloor} \, \| t^0_{\max} - t^* \|_\infty \leq \rho^{\frac{k}{L}} (\rho^{-1} \, \| t^0_{\max} - t^* \|_\infty), \end{aligned}$$

where $t^0_{\max} = \arg\max\{\| t^k - t^* \| \mid k = 0, \dots, L-1\}$, i.e., $\{t^k\}$ converges R-linearly to t^* with ratio $\rho^{\frac{1}{L}}$. □

3. Conclusion

In this paper, we presented a much simpler proof of Theorem 1 from [2]. Moreover, we improved the convergence rate estimate showing that it depends on the depth of the recursion used to compute input(·). Actually, if all the tasks of a product appear in the Round-Robin sequence in the same order of its production line, the convergence rate will be bounded by ρ, instead of by $\rho^{\frac{1}{\max \ \# \ \text{tasks/product}}}$, as shown in [2]. So, Round-Robin sequences of the above type tend to have faster convergence rates to the limit cycle than shuffled sequences, i.e., those in which the natural order of the production lines is not followed.

This relationship between the order of the tasks in the Round-Robin sequence and the convergence rate may have important practical implications in the actual operation of manufacturing systems, as mentioned in the Introduction. For instance, one might decide between two alternative CGRR schemes, given their respective limit cycles, storage costs and rates of convergence, considering the desirability of a slower or faster convergence to the limit cycle, heuristically trying to achieve a better storage allocation.

The analysis of how the changes in the Round-Robin sequence affect the limit cycle and the search for the minimal cost cycle according to a given criteria is open to further research. For the standard Round-Robin, a similar analysis is carried in [1], showing that optimal Round-Robin sequence is the one that minimizes the sum of the set-up costs.

References

[1] C. HUMES, L. BRANDAO and M.P. GARCIA, *On the optimal design of linear corridor policies,* In: Proceedings of the 33rd Allerton Conference on Communication, Control and Computing, 510-518, October 1995.

[2] A.F.P.C. HUMES and C. HUMES, *Stability of Clearing Open Loop Policies in Manufacturing Systems,* Computational and Applied Mathematics, 14(2):107-123, 1995.

[3] J.R. PERKINS, C. HUMES and P.R. KUMAR, *Distributed Control of Flexible Manufacturing Systems: Stability and Performance,* IEEE Transactions on Robotics and Automation, 37:132-141, 1994.

[4] J. STOER and R. BULIRSCH, *Introduction to Numerical Analysis, 2nd ed,* Springer-Verlag, New York, 1996.

Appendix A. The Limit Cycle for the Example

Let us use the ideas of the proof of Theorem 1 to the example presented in the Introduction. Recalling that

	t_1^2	t_2^2	t_3^2	t_4^2	t_5^2
input	$\{t_2^1,\dots,t_1^2\}$	$\{t_1^2,t_2^2\}$	$\{t_3^0,\dots,t_2^1\}$	$\{t_3^1,\dots,t_2^2\}$	$\{t_3^2,\dots,t_5^2\}$

we see that input$(t_3^2) = \{t_3^0,\dots,t_2^1\}$, therefore in order to write the system as in (3), we must consider three Round-Robin cycles, i.e., L = 3. Defining:

$$T_1^0 = t_1^0,\dots,T_5^0 = t_5^0, T_6^0 = t_1^1,\dots,T_{10}^0 = t_5^1, T_{11}^0 = t_1^2,\dots,T_{15}^0 = t_5^2, T_1^1 = t_1^3,\dots,$$

Then, using the reasoning presented in the proof of Theorem 1, we write:

$$\begin{aligned}
T_1^0 &= \delta_1 + \rho_1^2(T_{12}^{-1} + T_{13}^{-1} + T_{14}^{-1} + T_{15}^{-1} + T_1^0)\\
T_2^0 &= \delta_2 + \rho_1^1(T_1^0 + T_2^0)\\
T_3^0 &= \delta_3 + \rho_3^1(T_8^{-1} + T_9^{-1} + T_{10}^{-1} + T_{11}^{-1} + T_{12}^{-1})\\
T_4^0 &= \delta_4 + \rho_2^1(T_{13}^{-1} + T_{14}^{-1} + T_{15}^{-1} + T_1^0 + T_2^0)\\
T_5^0 &= \delta_5 + \rho_1^1(T_3^0 + T_4^0 + T_5^0)\\
T_6^0 &= \delta_1 + \rho_1^2(T_2^0 + T_3^0 + T_4^0 + T_5^0 + T_6^0)\\
T_7^0 &= \delta_2 + \rho_1^1(T_6^0 + T_7^0)\\
T_8^0 &= \delta_3 + \rho_3^1(T_{13}^{-1} + T_{14}^{-1} + T_{15}^{-1} + T_1^0 + T_2^0)\\
T_9^0 &= \delta_4 + \rho_2^1(T_3^0 + T_4^0 + T_5^0 + T_6^0 + T_7^0)\\
T_{10}^0 &= \delta_5 + \rho_1^1(T_8^0 + T_9^0 + T_{10}^0)\\
T_{11}^0 &= \delta_1 + \rho_1^2(T_7^0 + T_8^0 + T_9^0 + T_{10}^0 + T_{11}^0)\\
T_{12}^0 &= \delta_2 + \rho_1^1(T_{11}^0 + T_{12}^0)\\
T_{13}^0 &= \delta_3 + \rho_3^1(T_3^0 + T_4^0 + T_5^0 + T_6^0 + T_7^0)\\
T_{14}^0 &= \delta_4 + \rho_2^1(T_8^0 + T_9^0 + T_{10}^0 + T_{11}^0 + T_{12}^0)\\
T_{15}^0 &= \delta_5 + \rho_1^1(T_{13}^0 + T_{14}^0 + T_{15}^0)
\end{aligned}$$

This system can be easily recognized as a Gauss-Seidel step where the matrix $(I-P)$ is presented in figure 2.

$$
\left(\begin{array}{ccccccccccccccc}
1-\rho_1^2 & & & & & & & & & & & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 \\
-\rho_1^1 & 1-\rho_1^1 & & & & & & & & & & & & & \\
 & & 1 & & & & & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & & & \\
-\rho_2^1 & -\rho_2^1 & & 1 & & & & & & & & & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 \\
 & & -\rho_1^1 & -\rho_1^1 & 1-\rho_1^1 & & & & & & & & & & \\
\hline
 & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 & 1-\rho_1^2 & & & & & & & & & \\
 & & & & & -\rho_1^1 & -\rho_1^1 & & & & & & & & \\
-\rho_3^1 & -\rho_3^1 & & & & & & 1 & & & & & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 \\
 & & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & & 1 & & & & & & \\
 & & & & & & & -\rho_1^1 & -\rho_1^1 & 1-\rho_1^1 & & & & & \\
\hline
 & & & & & & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 & -\rho_1^2 & 1-\rho_1^2 & & & & \\
 & & & & & & & & & & -\rho_1^1 & 1-\rho_1^1 & & & \\
 & & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & -\rho_3^1 & & & & & & 1 & & \\
 & & & & & & & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & -\rho_2^1 & & 1 & \\
 & & & & & & & & & & & & -\rho_1^1 & -\rho_1^1 & 1-\rho_1^1
\end{array}\right)
$$

FIGURE 2. $(I-P)$ matrix for the single machine system.

Note the structure of the matrix, where the two lower horizontal blocks consist of shifts of the first one by 5 and 10 columns, respectively.

Finally, the limit cycle can be obtained from the solution of the system

$$(I - P)T = \begin{bmatrix} \delta \\ \delta \\ \delta \end{bmatrix}.$$

This solution is

$$(3.7, 1.05, 1.1, 2.2, 0.95, 3.7, 1.05, 1.1, 2.2, 0.95, 3.7, 1.05, 1.1, 2.2, 0.95)^t,$$

and it is composed of three copies of the limit cycle, as was stated in the proof of Theorem 1.

(Paulo Silva) Dept. of Applied Mathematics
E-mail address: rsilva@ime.usp.br

(Marcelo Queiroz and Carlos Humes) Dept. of Computer Science
E-mail address: mqz@ime.usp.br, humes@ime.usp.br

University of São Paulo, Rua do Matão 1010, Cidade Universitária, CEP: 05508-900, São Paulo, SP, Brazil