Allocation Sequences of Two Processes Sharing a Resource

Bruno Gaujal, Mohsen Jafari, Melike Baykal-Gürsoy, and Gülgün Alpan

Abstract—**We study a Petri net model of a system composed of two processes sharing a resource. Conflicts may occur over the usage of the shared resource, thus making the system nondeterministic. Therefore, in the context of minimax algebra, it cannot be formulated as a linear system in order to compute its performance measures. However, if the sequence by which the resource is allocated to the two processes is known, we can transform the system into a decision-free net. For this system with an imposed constraint on the resource allocation frequencies, we show that the optimal allocation sequence is the most regular integer sequence satisfying that constraint. We also discuss the periodic behavior of this system under no constraints on the resource allocation frequencies.**

I. Introduction

We consider a system where two processes are sharing a common resource. Each process undergoes a number of activities (operations) in a cyclic manner. There are two types of activities, depending on whether or not the common resource is being utilized. For each process, the set of activities utilizing the common resource is followed by the set of activities not utilizing that resource. Each activity has a fixed duration or temporization.

The above system is typical in manufacturing applications. For instance, in a manufacturing work-cell, the two processes could correspond to two workstations (e.g., CNC machine tools) while the common resource corresponds to a robot in charge of material handling. The workstations may work independently or they may interact due to, for example, the flow of parts from one workstation to another. In this article, we shall use the term "parallel system" to refer to the cases where there is no interaction between the two workstations and "serial system" to refer to those where the two processes interact.

For some temporizations, it is possible not to have any conflict. In such a case and with no external constraints on the use of the resource, the system follows its natural frequency, which merely depends on the internal characteristics of the system (e.g., temporizations of the activities and the interaction between the two processes). Here, it is of interest to compute the number or the frequency of resource allocations for each process per period and the allocation sequence. If conflict was possible because of temporizations, then the sample path or the system frequency would also depend on how the conflicts are resolved. In such a case, it would also be of interest to determine the optimal conflict resolution scheme so that some performance measures, such as system period, are minimized.

It is also possible to impose an external control on the system frequency, so that the number of allocations of the resource to each process is fixed and prespecified. Here, given the resource allocation frequency, we are interested in the optimal resource allocation sequence so that some performance measures, such as the makespan to complete all the allocations, are minimized.

The above system can be modeled as a simple timed Petri net where the Petri net structure exhibits conflict. The term "simple" refers to the class of ordinary Petri nets, where a transition has at most one input place shared with other transitions (see [9] for more detail). Generally speaking, any ordinary timed Petri net containing a structural conflict can be transformed into a decision-free net under a specific transition firing schedule. A subclass of decision-free nets are marked graphs, which are ordinary Petri nets such that any place has one input transition and one output transition. No conflicts exist in a marked graph. Due to their deterministic behavior, the set of equations obtained by relating the earliest or latest firing times of transitions in a marked graph are linear in the context of minimax algebra [4], [3]. Thus, different performance measures can be obtained by solving these linear equations. There are also graph-theoretic based approaches [10], [7] for the performance analysis of timed marked graphs.

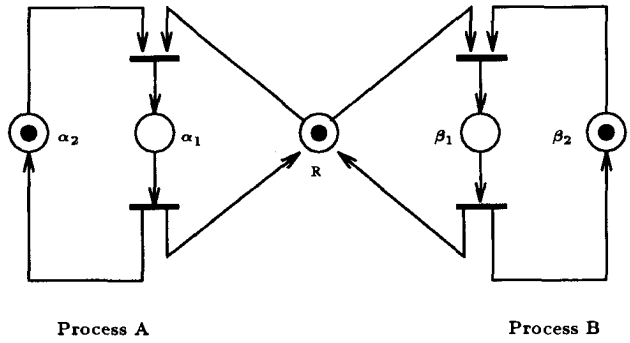

Fig. 1. Simple Petri net modeling the system.

Not knowing the transition firing schedule *a priori*, ordinary timed Petri nets (including the class of simple Petri nets we are considering) are nonlinear in the context of minimax algebra. There are no analytical techniques for solving these nonlinear systems. In this article, we analyze simple nonlinear Petri nets with some limited structure.

In Section II, we present some notations. In Section III, we analyze a simple timed Petri net where constraints are imposed on the transition firings. In this case, we attempt to compute the optimal resource allocation sequence, that is, the sequence by which the resource is utilized by the two processes with a given allocation frequency which gives the minimum makespan time. In Section IV, we discuss the periodicity and performance of a simple timed Petri net where there are no constraints imposed on the transition firings. In Sections V and VI, we consider extensions such as processes with inputs and the processes working in series, and show that they have equivalent systems that are already analyzed in the paper.

II. Preliminaries

We will be using the notations $\mathbb{N}$, denoting the set of positive integers, $\mathbb{R}$, denoting the set of reals, and lcm, denoting the least common multiple. Fig. 1 illustrates the system to be studied in this article. Places represent the activities and transitions the events taking place in the system. Associated with places are temporizations: α_1 and α_2 for process A, and β_1 and β_2 for process B. We denote by T the quadruplet $(\alpha_1, \alpha_2, \beta_1, \beta_2)$. The temporized system is denoted (S, T). As can be seen from Fig. 1, during α_1 (respectively, β_1) the common resource is being used by process A (respectively, B). During α_2 (respectively, β_2), process A (respectively, B) performs activities not involving the resource. Thus, the former activities have to be in mutual exclusion while the latter can be taking place in parallel. We note that the Petri net in this figure is a simple

Manuscript received April 13, 1993; revised November 1, 1994.

B. Gaujal is with the Dimacs Center, Rutgers University, Piscataway, NJ 08855 USA and AT&T Bell Laboratories, NJ, USA.

M. Jafari, M. Baykal-Gürsoy, and G. Alpan are with the Department of Industrial Engineering, Rutgers University, Piscataway, NJ 08855 USA.

IEEE Log Number 9411928.

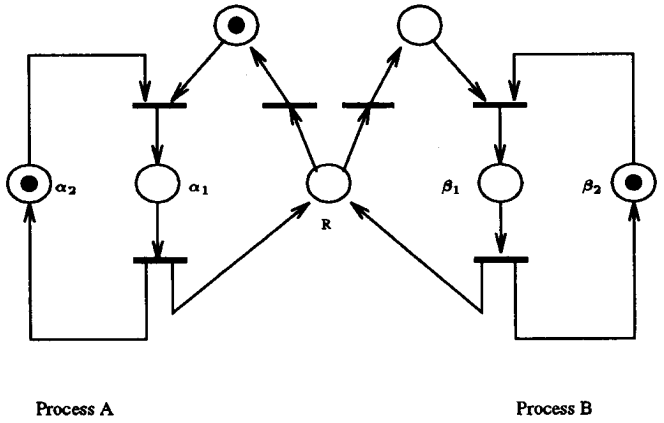

Fig. 2. Free-choice net modeling the system S. The resource is available if there is a token in place R. Once the allocation sequence is given, the resource is routed accordingly.

Petri net exhibiting structural conflict. According to this structure, the allocation of the resource to a given process is preconditioned by that process requiring the resource. In the case that there are constraints imposed on transition firings (i.e., constraints imposed on the allocation of the resource to the two processes), we can transform this Petri net into a free-choice net. This would give us control over allocating the resource to a given process independent of the state of the two processes. The reader notes that in free-choice nets, if two transitions are sharing a common input place, then it must be the only input place for either transition. See [8] for a comprehensive discussion on Petri nets and their variations and extensions.

III. A System with Imposed Frequency

In this section, we study the optimal resource allocation in the above system. The objective is to find the allocation sequence minimizing the idle time of the resource given the following constraint on the frequency of resource allocation: For every N allocations of the resource, process A must receive the resource p times and process B must receive it q times with $p + q = N$. The couple (p, q) will be called the *frequencies* of the system. The reader notes that we are still free to choose any resource allocation sequence satisfying this constraint. Therefore, the simple net system of Fig. 1 reduces to that of Fig. 2, which depicts a free-choice net. We further restrict ourselves to allocation sequences which are periodic with period N.

Under these conditions, minimizing the idle time of the resource is equivalent to minimizing the total time required to go through the N allocations. Indeed, if we denote by M the total time of running the system for $p + q = N$ allocations, the idle time of the resource is $M - p\alpha_1 - q\beta_1$.

We adopt the following notations. We consider the alphabet $\{A, B\}$ and the set $\mathcal{N}$ of the words formed from N letters of this alphabet. If X and Y are different letters in $\{A, B\}$, the word $\underbrace{XX\cdots X}_{i}\underbrace{Y\cdots Y}_{j}$ is denoted by X^iY^j.

Let $E(p, q)$ be the set of all the words in $\mathcal{N}$ formed with p A's and q B's. An element w in $E(p, q)$ corresponds to a period of a possible allocation sequence of the resource satisfying the given frequencies (p, q). The time that it takes for the system to execute the sequence w is called the *makespan* of the system (S, T, w) and is denoted by M_w.

A. Optimal Allocation Sequence

The problem can now be rewritten as: Find w^* in $E(p, q)$ such that M_{w^*} is minimum. In the following we assume without loss of generality that $q \geq p$.

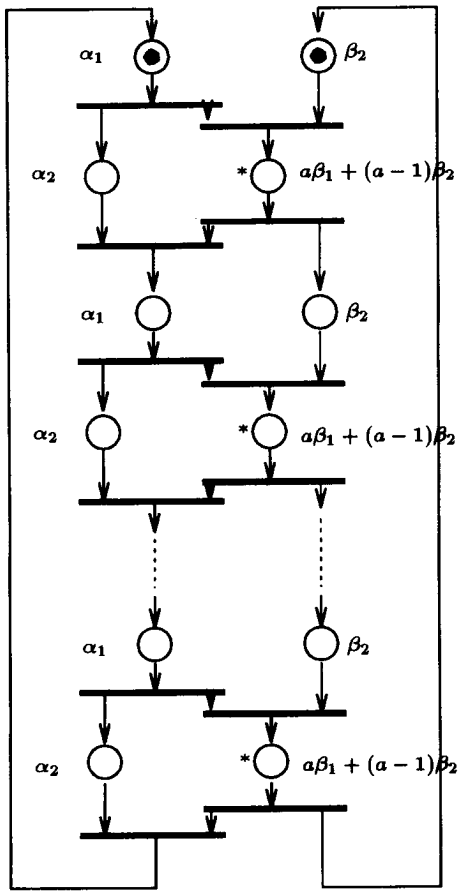

Fig. 3. Marked graph $MG(S, T, w)$. The places with weight $a\beta_1 + (a-1)\beta_2$ result from the aggregation of $(2a-1)$ places with holding times $\beta_1, \beta_2, \cdots, \beta_1$.

Theorem 1: In the case where $q = ap$, where $a \in \mathbb{N}$, the sequence $w^* = (AB^a)^p$ is an optimal allocation sequence in the sense that for any temporizations $(\alpha_1, \alpha_2, \beta_1, \beta_2)$, $M_{w^*} \leq M_w, \forall w \in E(p, q)$.

This optimal sequence does not depend on the temporizations but only on the values of p and q. We also note that this sequence distributes the A's and the B's as evenly as possible. In the general case (q is not a multiple of p), the method used below cannot be applied. However, there is strong evidence suggesting that the most regular sequence should also be optimal. A related result is shown in [5]. The worst case has already been studied, and the worst sequence is known to be $w = A^pB^q$, which corresponds to a near complete sequentialization of the system (see [2]). Once the sequence w is given, the new system (S, T, w) can be modeled as a marked graph $(MG(S, T, w))$, i.e., a decision-free Petri net. If C is a circuit in the graph $MG(S, T, w)$, let $t(C)$ be the sum of the temporizations of the places belonging to the circuit. $t(C)$ is also called the temporization of the circuit. The makespan of the system is then given by the temporization of the slowest circuit in the net $MG(S, T, w)$ (see [1]), where the slowest circuit is the circuit with the maximum temporization. We say a circuit C in the $MG(S, w)$ is *critical* if there exists a temporization of the system $T = (\alpha_1, \alpha_2, \beta_1, \beta_2)$ such that C is the slowest circuit in $MG(S, T, w)$.

Proof: The proof directly follows from two lemmas which are given next. In Lemma 1, we show that $MG(S, T, w^*)$ (depicted in Fig. 3) has at most three critical circuits. In Lemma 2, we show that for any sequence $w \in E(p, q)$, we can find three circuits in $MG(S, T, w)$ which are slower than the critical circuits of $MG(S, T, w^*)$.

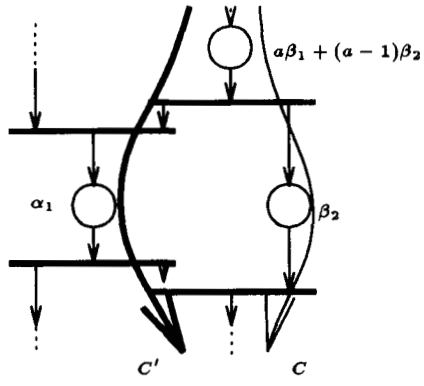

Fig. 4. If $\alpha_1 > \beta_2$, then the circuit C', which is equal to the circuit C except on this portion of $MG(S,T,w*)$ is slower than C, so that any circuit C containing a β_2-place is not the slowest circuit.

Lemma 1: The system $MG(S,T,w^*)$ has at most three critical circuits.

Proof: Case 1: $(\alpha_1 > \beta_2)$.

In this case, a circuit containing a β_2-place is not critical because this place can be replaced, as in Fig. 4, by the corresponding α_1-place to get a slower circuit. Therefore, slowest circuits necessarily contain all α_1-places.

Case 1.1: If $\alpha_2 > a\beta_1 + (a-1)\beta_2$, then the slowest circuit necessarily goes through all α_2-places, applying the same reasoning as previously. Thus the slowest circuit is C_1 going through α-places, and its temporization is $t(C_1) = p\alpha_1 + p\alpha_2$.

Case 1.2: If $\alpha_2 \leq a\beta_1 + (a-1)\beta_2$, then with a similar method, one can see that the slowest circuit is C_2, going through all α_1-places and through $[a\beta_1 + (a-1)\beta_2]$-places with temporization $t(C_2) = p\alpha_1 + q\beta_1 + (q-p)\beta_2$.

Case 2: $(\beta_2 \geq \alpha_1)$.

We consider two subcases:

Case 2.1: If $\alpha_2 \leq a\beta_1 + (a-1)\beta_2$, then the critical circuit goes through $[a\beta_1 + (a-1)\beta_2]$-places and through β_2-places. We call this slowest circuit C_3, and its temporization is $t(C_3) = q\beta_1 + q\beta_2$.

Case 2.2: If $\alpha_2 > a\beta_1 + (a-1)\beta_2$, suppose there exists a critical circuit C that goes through a β_2-place. (If not, $C = C_1$).

Let us now assume that circuit C goes through an α_1-place just before the β_2-place (if no β_2-place satisfies this condition, then $C = C_3$). If C is the slowest, then $\alpha_1 + \alpha_2 \leq a(\beta_1 + \beta_2)$. Now, the α_1-place itself must be preceded by an α_2-place, because otherwise replacing the α_1-place by a β_2-place would increase the temporization of the circuit. But now the inequality $\alpha_1 + \alpha_2 < a(\beta_1+\beta_2)$ makes this configuration impossible since this sequence of places can be advantageously replaced by the sequence of β-places. Thus this circuit can never be slowest circuit and there are only three critical circuits: C_1, C_2, C_3. □

Next, we prove that for any other allocation w, we can find circuits which are slower than the three critical circuits of the allocation w^*. Thus, the slowest circuit of any sequence w is certainly slower than the slowest circuit of the sequence w^*.

Lemma 2: For any sequence w, and for any critical circuit C of $MG(S,T,w^*)$, one can exhibit a circuit D in $MG(S,T,w)$ such that $t(D) \geq t(C)$.

Proof: Any sequence w can be written as $w = AB^{s_1}AB^{s_2}\cdots AB^{s_p}$, where $s_1 + s_2 + \cdots + s_p = q$ and $s_i \geq 0, \forall i$. Let us rearrange the p-tuple $(s_1, \cdots, s_p)$ in the increasing order: $(t_1, \cdots, t_p) = (s_{[1]}, \cdots, s_{[p]})$ with $t_1 \leq \cdots \leq t_p$. Now, we can find a circuit D_1 (resp. D_3) in $MG(S,T,w)$, which is slower than C_1 (resp. C_3). Indeed, we can choose D_1 (resp. D_3) going through α_1-places and α_2-places (resp. β_1-places and β_2-places). Thus, $t(D_1) = t(C_1)$ and $t(D_3) = t(C_3)$.

For the remaining circuit, let us define b by $b\beta_1 + (b-1)\beta_2 \leq \alpha_2 < (b+1)\beta_1 + b\beta_2$. We note that b is well defined if $\alpha_2 \geq \beta_1$, else b is not defined. We consider two different cases: $(b \geq a)$ and $(b < a$ or $\alpha_2 < \beta_1)$

For $b \geq a$, $\alpha_2 \geq a\beta_1 + (a-1)\beta_2$, and therefore $t(C_2) \leq C_1 = t(D_1)$. This remark ends this case.

For $(b < a$ or $\alpha_2 < \beta_1)$, we consider the circuit D_2. This circuit is obtained by passing through the B^{s_j} if $s_j > b > a$ and passing through α_2 otherwise. Therefore its temporization is $t(D_2) = p\alpha_1 + i\alpha_2 + (t_p + \cdots + t_{i+1})(\beta_1 + \beta_2) - (p-i)\beta_2$, where i is defined by $t_i \leq b < t_{i+1}$ and $i = 0$ if b is not defined.

By the definition of b, $\alpha_2 \geq b\beta_1 + (b-1)\beta_2$. By the definition of i, $t_1 \leq b, \cdots, t_i \leq b$. Therefore,

$$\begin{aligned} \alpha_2 &\geq t_1\beta_1 + (t_1 - 1)\beta_2 \\ &\vdots \geq \vdots \\ \alpha_2 &\geq t_i\beta_1 + (t_i - 1)\beta_2. \end{aligned}$$

Eventually, we have $t(D_2) \geq p\alpha_1 + (t_1 + \cdots + t_i)\beta_1 + (t_1 + \cdots + t_i - p)\beta_2 = t(C_2)$. □

We have proved that once the constant a is given, no matter what the temporization T is, the optimal allocation is given by w^*. This ends the proof of the theorem. □

We note that w^* does not depend on p. Therefore, we can say that the optimal allocation, when we want to give the resource a times to process B while it is given only once to process A on the long run, is $w^* = (A\beta^a)^*$.

B. Computation of the Resource Allocation Frequencies

In this section we consider the problem of computing the optimal resource allocation frequencies: Given the temporization T of the system, and assuming that q is multiple of p, find the optimal frequencies p and q minimizing the idle time of the resource. Since when $q = ap$ there exist only three critical cycles, the idle time per allocation is therefore given by the function shown at the bottom of the page. We notice that h does not depend individually on p and q but depends only on the integral term a, and thus the above problem can be rewritten as minimization of the function $h(a)$, defined by:

$$h(a) = \max(h_1(a), h_2(a), h_3(a))$$

where $h_1(a) = \frac{\alpha_2 - a\beta_1}{a+1}$, $h_2(a) = \frac{(a-1)\beta_2}{a+1}$ and $h_3(a) = \frac{a\beta_2 - \alpha_1}{a+1}$. Note that $h_2(a) = h_3(a) + \frac{\alpha_1 - \beta_2}{a+1}$. This allows a simpler description of the function h (see equation at the bottom of the page).

If $\alpha_1 > \beta_2$ then $h(a) = \max(h_1(a), h_2(a))$, which achieves its minimum at the optimal point $a = \frac{\alpha_2+\beta_2}{\beta_1+\beta_2}$. This optimal value gives two possible integers: $a_1 = \lfloor \frac{\alpha_2+\beta_2}{\beta_1+\beta_2} \rfloor$ and $a_2 = \lceil \frac{\alpha_2+\beta_2}{\beta_1+\beta_2} \rceil$.

If $\alpha_1 \leq \beta_2$ then $h(a) = \max(h_1(a), h_3(a))$, and its minimum is at the point $a = \frac{\alpha_2+\alpha_1}{\beta_1+\beta_2}$. This gives two possible values for a: $a_1 = \lfloor \frac{\alpha_2+\alpha_1}{\beta_1+\beta_2} \rfloor$ and $a_2 = \lceil \frac{\alpha_2+\alpha_1}{\beta_1+\beta_2} \rceil$. In both cases, the determination

$$h(p,q) = \max\left(\frac{t(C_1) - p\alpha_1 - q\beta_1}{p+q}, \frac{t(C_2) - p\alpha_1 - q\beta_1}{p+q}, \frac{t(C_3) - p\alpha_1 - q\beta_1}{p+q}\right).$$

of the best constant a can be made by the analysis of the two networks with the two possibilities a_1 and a_2.

IV. A System with No Control on the Resource

In this section, we consider the original system (two processes sharing a resource in mutual exclusion) without any frequency constraint on the resource allocation (see Fig. 1). In this system, the resource is given to the first process that is waiting for it. If the two processes become ready simultaneously to use the resource, we call this situation a conflict and different policies can be applied to solve it. A static policy would, for example, give a fixed priority level to each of the processes: The one with the higher priority acquires the resource. Dynamic policies may also be used. For example, the resource is given to the process which did not acquire the resource the last time it was allocated (alternative routing). Some priority policies can be imbedded in the Petri net structure. Here, we shall keep the Petri net structure independent of the conflict resolution policy (see Fig. 1). We will only assume that while executing the Petri net, B will be given a higher priority than A to use the resource in the case of a conflict.

First we show that this system is periodic and that once again, the allocation of the resource is the "most regular sequence." Consider the system without process B. Then, its behavior is periodic with period $\alpha_1 + \alpha_2$, the resource being used during time α_1 per period. The symmetric system (without process A) has a period $\beta_1 + \beta_2$, and the resource is used during β_1.

Let f and g be two periodic functions with respective periods $\alpha = \alpha_1 + \alpha_2$ and $\beta = \beta_1 + \beta_2$. On $[0, \alpha)$, f (respectively, g) is given by

$$f(x) = 1 \text{ if } 0 \leq x < \alpha_1$$
$$f(x) = 0 \text{ if } \alpha_1 \leq x < \alpha_2 + \alpha_1.$$

and

$$g(x) = 1 \text{ if } 0 \leq x < \beta_1$$
$$g(x) = 0 \text{if } \beta_1 \leq x < \beta_2 + \beta_1.$$

These two functions will represent the use of the resource by the two processes. $f(x) = 1$ means that process A holds the resource at time x. $f(x) = 0$ means that A does not hold the resource at time x and, respectively, with g and B (see Fig. 5). Now we define $f_{i_A}(x) = f(x - i_A)$ and $g_{i_B}(x) = g(x - i_B)$ to represent the two processes with their initial phases i_A and i_B.

If the two processes are superimposed, they behave as if they were alone as long as the intervals α_1 and β_1 do not overlap. If they overlap, then one of the processes has to wait for the resource. After a waiting point, two cases may happen. One option is that A has been waiting, in which case the system restarts with a phase ϕ_A: *A begins its interval α_1 and B its interval β_2*, and this phase corresponds to $i_A = 0$ and $i_B = \beta_2$ (see Fig. 6). In the other instance, B has been waiting for the resource, in which case the system restarts with a phase ϕ_B between the two processes: *B starts its interval β_1 and A starts its interval α_2*. This phase corresponds to $i_A = \alpha_2$ and $i_B = 0$ (see Fig. 7).

Theorem 2: The system is periodic following a transient period.

To prove the theorem, we first consider the case when $\frac{\alpha_1+\alpha_2}{\beta_1+\beta_2}$ is irrational.

The following lemma states that, for any initial phase, one of the two processes will eventually wait for the resource.

Lemma 3: If $\frac{\alpha_1+\alpha_2}{\beta_1+\beta_2}$ is irrational, there exists an $x \in \mathbb{R}$ satisfying $f_{i_A}(x) = g_{i_B}(x) = 1$.

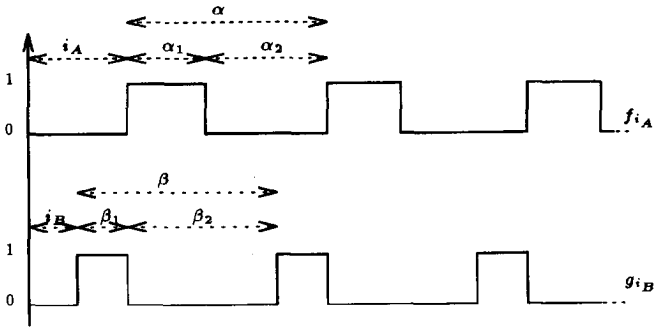

Fig. 5. The functions f_{i_A} and g_{i_B} are displayed on different axes. For simplicity, their right-continuity is not shown in the figure.

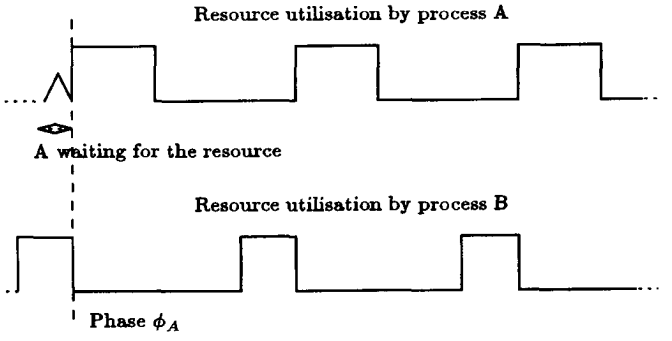

Fig. 6. A waiting point of process A always resets the phase between the two processes at ϕ_A.

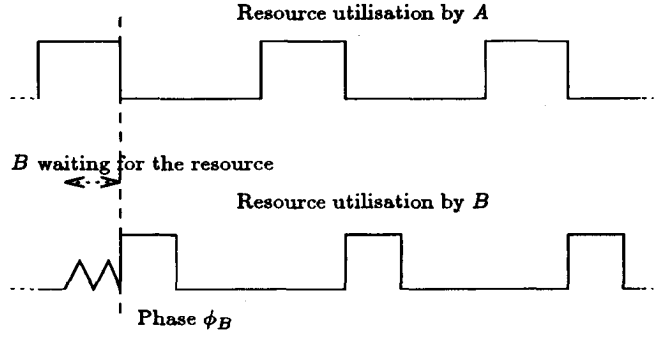

Fig. 7. A waiting point of process B alsays resets the phase between the two processes at ϕ_B.

Proof of the lemma: Without any loss of generality, we can assume that $i_A = 0$. We define the sequence

$$x_n = (i_B + n\beta) \bmod \alpha, \text{ where } \beta = \beta_1 + \beta_2.$$

If there exists an n such that $x_n < \alpha_1$, then $f_{i_A}(i_B + n\beta) = 1$ because $k\alpha \leq i_B + n\beta < k\alpha + \alpha_1$, where $\alpha = \alpha_1 + \alpha_2$. We also have $g_{i_B}(i_B + n\beta) = g_{i_B}(i_B) = 1$, so that the point $i_B + n\beta$ verifies the conditions given in Lemma 3.

Now, if α/β is irrational, the sequence x_n is dense in the interval $[0, \alpha]$ thanks to a Dirichlet argument like that in [6] (see the Appendix for a proof of this property). This proves the existence of an n verifying $x_n < \alpha_1$. This ends the proof of the lemma. □

Proof of the theorem: In the case where α/β is irrational, the initial phases of the two processes are arbitrary: i_A and i_B. However, according to Lemma 3 the two intervals α_1 and β_1 will eventually overlap, giving a waiting state. This results in a reset of the phases between the two processes to either phase ϕ_A or to phase ϕ_B. After this, if we apply Lemma 3 once more, with the new phase, the second

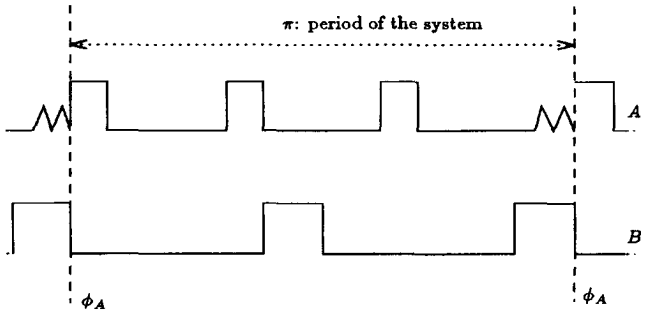

Fig. 8. First type of period: Only one process waits for the resource.

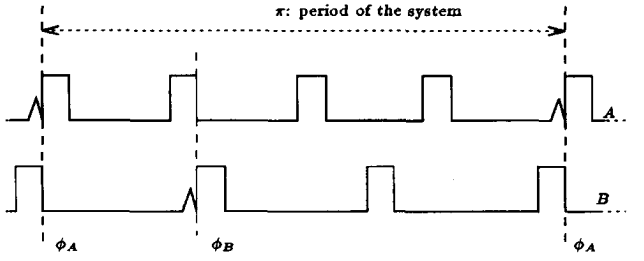

Fig. 9. Second type of period waiting states: Both processes wait for the resource during one period.

overlapping will reset the phases once more. The lemma is applied for the third time, and the third overlapping will reset the phases to a phase that has already been set so that the system is periodic.

In the case where α/β is rational, either the two requests of the resource will never overlap and the process is periodic with period $\text{lcm}(\alpha, \beta)$, or the two requests overlap and the phase is changed to phase ϕ_A or phase ϕ_B. We repeat this analysis twice more, and either we end with the same phase as the previous phase or with a periodic behavior with no waiting states; in any case, the system is periodic.□

This proof does not give the value of the period. The Dirichlet argument suggests that no closed formula of the period can be found in the general case. In the case where α/β is rational, an easy upper bound of the period can be given: $\text{lcm}(\alpha, \beta)$.

As stated previously, we are interested in finding the sequence of allocation to the two processes. As no closed formula of the period can be given, we cannot give a closed formula for the allocation sequence either. However, we can verify that once again the allocation sequence is the "most regular" sequence given the number of allocations of the resource to the two processes during one period.

Let us assume that during one period, the resource is given n times to A and k times to B. One can notice that n and k are the "natural" frequencies of the system (S, T) within $n + k$ allocations. If there are no waiting states, n and k are defined by $n\alpha = k\beta = \text{lcm}(\alpha, \beta)$.

In the case where the period contains waiting states, by the symmetry of the problem we can assume that the period begins with a phase ϕ_A and ends with phase ϕ_A. Now, only two situations are possible: The first waiting process during one period is A, in which case the period of the system is between these two waiting times (see Fig. 8), or the next waiting process is B and the third waiting process is A (see Fig. 9).

In the first case, the period of the system is a multiple of β: $\pi = k\beta$. As for process A, we define n by $n\alpha \geq \pi < (n+1)\alpha$. The allocation sequence is $w = A^{c_1}BA^{c_2}B \cdots A^{c_k}B$, where $c_i = \lceil i\frac{n}{k} \rceil - \lceil (i-1)\frac{n}{k} \rceil$. As k and n cannot be computed, we can also note that if $i < k$, $c_i = \lceil i\frac{\beta}{\alpha} \rceil - \lceil (i-1)\frac{\beta}{\alpha} \rceil$ and $c_k = \lceil (k-1)\frac{\beta}{\alpha} + \frac{\beta 2}{\alpha} \rceil - \lceil (k-1)\frac{\beta}{\alpha} \rceil$.

The last one, c_k, is slightly different because we have to take into account the waiting state of process A at the end of the period.

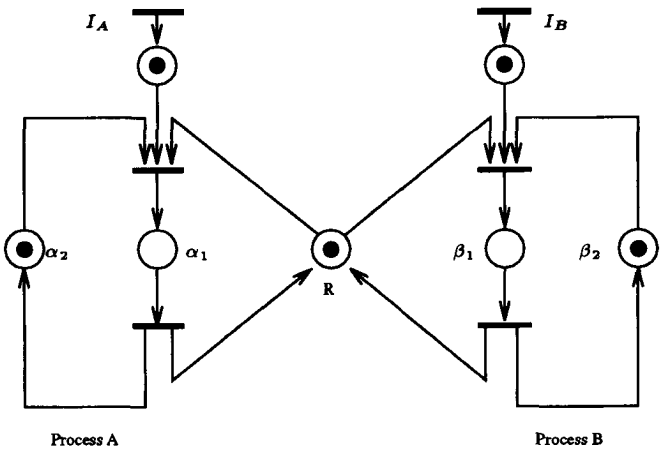

Fig. 10. A system with two inputs. The input transitions fire every I_A and I_B time units, respectively.

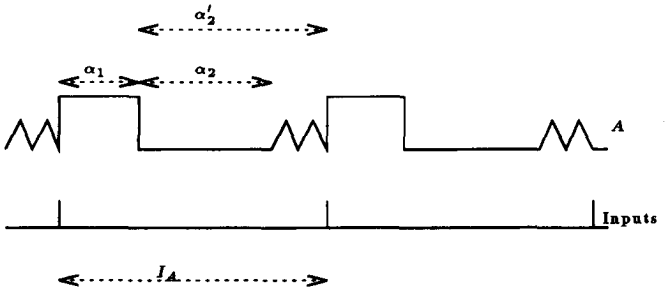

Fig. 11. If $I_A > \alpha_1 + \alpha_2$, the process A in isolation eventually behaves like a system with no inputs but with a longer period: I_A. The inputs are depicted on the lower line.

In the case where the period also contains a waiting time for process B, this waiting time is smaller than α_1 and therefore cannot alter the allocation sequence within one period, which remains the same as before. The only difference with the previous case is that now, k is also defined by an inequality: $k\beta < \pi < (k+1)\beta$.

The sequence c obtained is the most regular integer sequence (up to the last c_k, which is cut because of the waiting state). This sequence is optimal for various scheduling problems, as in [5].

V. Inputs

If we add periodic inputs to each of the processes (see Fig. 10), then we can show that this system is equivalent to a system without any input. Consider process A in isolation with a periodic input of period I_A. If $I_A > \alpha_1 + \alpha_2$, then the process will eventually wait for an input. After this point, the process becomes periodic with period I_A. This process can be seen as a process with no input but with temporization (α_1, α_2'), where $\alpha_2' = I_A - \alpha_1$. See Fig. 11.

If $I_A \leq \alpha_1 + \alpha_2$, the process may wait once for the input. Afterwards, the input token will always arrive before it is required. So, the input does not alter the behavior of process A and can be ignored (see Fig. 12). The same argument holds for process B with a periodic input every I_B time units. Therefore, this system with the temporizations $(\alpha_1, \alpha_2, \beta_1, \beta_2)$ and the inputs (I_A, I_B) is equivalent to a system without inputs with the temporizations $(\alpha_1, \max(\alpha_2, I_A - \alpha_1), \beta_1, \max(\beta_2, I_B - \beta_1))$.

VI. Two Processes in Series

In this case, process B has to wait for process A before getting the resource. An infinite buffer is placed between the processes (see

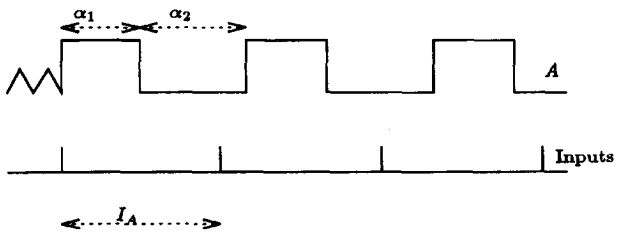

Fig. 12. If $I_A \leq \alpha_1 + \alpha_2$, the process A in isolation will eventually behave as if it had no inputs.

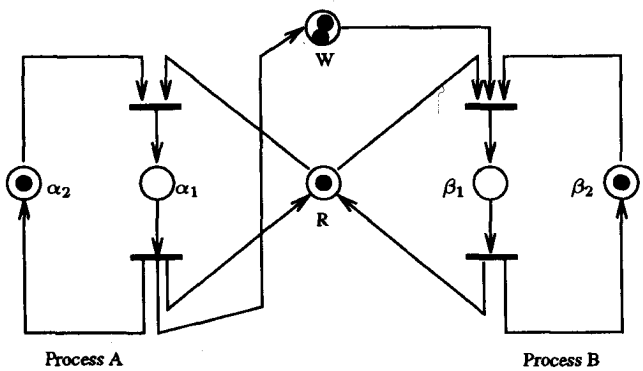

Fig. 13. A system where an infiniet buffer W is introduced from A to B.

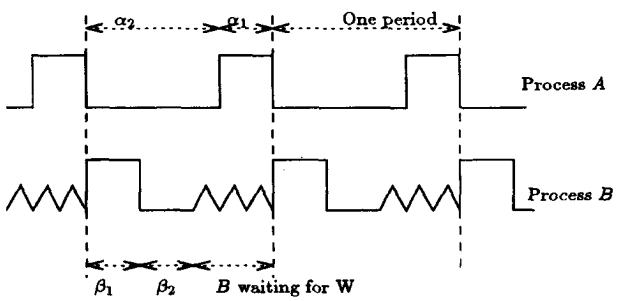

Fig. 14. If $\beta < \alpha$, the system in series eventually becomes periodic, with a period α.

Fig. 13). This case is equivalent to a system where the two processes work in parallel. We distinguish between two cases, as follows. If $\alpha \leq \beta$, then the system is truly equivalent to the parallel one. Indeed, if B has to wait once for the input from A, this situation will not happen again. After a possible first waiting time for a token in the buffer, whenever B wants the resource, the resource has been released by A and at least one token has been placed in the buffer. If $\beta < \alpha$, then after some time B will have to wait for the input from A. At that point, the system becomes periodic of period α, and the allocation sequence during one period is AB (see Fig. 14).

VII. Conclusion

In this article, we consider the problem of allocating a single resource to two processes in a repetitive manner. The system is studied with and without an imposed frequency. For the first case, if the required frequencies of each process is some known multiple of one another we show that the optimal resource allocation sequence is the most regular sequence. On the other hand, if the multiplication constant is not known the optimal one can be found that will minimize the idle time of the resource. For the second case, only under a known priority structure, we show that the system is periodic after a transient period and again the optimal sequence is the most regular one. We also consider the extensions of this problem to the case that includes inputs to each process and to the processes working in series. We show that these systems are equivalent to the systems discussed before.

Appendix

We will be using some of the notation and definitions given in the article.

We prove that if $\frac{\alpha}{\beta}$ is irrational, the sequence $x_n = (i_B + n\beta) \bmod(\alpha)$ is in $[0, \alpha)$ and there exists m such that $x_m < \alpha_1$. Choose n in $\mathbb{N}$ with $n > 1/\alpha_1$ and cut the interval $[0, \alpha)$ into n subintervals each one of size a/n. Now, consider the sequence $(x_0, x_1, \cdots, x_n)$. First, we remark that these $n+1$ numbers are all distinct. Indeed, if $x_k = x_j \bmod(\alpha)$ with $k > j$, this means that

$$i_B + k\beta = i_B + j\beta \bmod(\alpha)$$
$$(k-j)\beta = 0 \bmod(\alpha)$$

so that $\frac{\alpha}{\beta}$ is rational. This contradicts the assumption. Second, if $n+1$ different numbers are being distributed in n intervals, then at least two of them fall in the same interval. We get $k > j$ with $x_k - x_j = e \bmod(\alpha)$ and $|e| < 1/n$. $x_k - x_j = e \bmod(\alpha)$ implies $(k-j)\beta = e \bmod(\alpha)$, so $x_{k-j} = i_B + e \bmod(\alpha)$.

Finally, we can divide $\alpha_1 - i_B$ by e to get $\alpha_1 - i_B = ce + r \bmod(\alpha)$ with $c \in \mathbb{N}$ and $0 \leq r < |e|$.

$x_{c(k-j)} = i_B + ce = \alpha_1 - r \bmod(\alpha)$. As $0 \leq r < |e|$, we can conclude $x_{c(k-j)} < \alpha_1$. □

This proof can be easily adapted to prove that the sequence (x_n) is dense in $[0, \alpha)$. If we want to approach any number X in $[0, \alpha)$ within ϵ, choose n in $\mathbb{N}$ with $n > 1/\epsilon$ and replace $\alpha_1 - i_B$ by $X - i_B$ in the previous proof.

References

[1] F. Baccelli, G. Cohen, G. J. Olsder, and J. P. Quadrat, *Synchronization and Linearity*. New York: Springer-Verlag, 1992.
[2] C. Chaouiya, "Outils pour l'analyse de systemes synchronizes," Ph.D. dissertation, Univ. of Nice, INRIA, Sophia Antipolis, France, 1992.
[3] D. Dubois and K. Stecke, "Dynamic analysis of repetitive decision-free discrete event processes: Applications of production systems," Univ. of Michigan, Ann Arbor, Tech. Rep. 543, 1989.
[4] J. P. Quadrat, G. Cohen, P. Moller, and M. Viot, "Linear system theory of discrete-event systems," in *Proc. 23rd IEEE CDC*, Las Vegas, NV, vol. 1, pp. 539–544, 1984.
[5] B. Hajek, "Extremal splittings of point processes," *Math. Operation Research*, vol. 10, pp. 543–556, 1985.
[6] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*. Oxford: Oxford Univ. Press, 1983.
[7] H. P. Hillion and J. M. Proth, "Performance evaluation of job shop systems using timed event graphs," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 3–9, 1989.
[8] T. Murata, "Petri nets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, 1984.
[9] J. L. Peterson, "Petri nets," *ACM Comput. Surv.*, vol. 9, no. 3, 1977.
[10] C. V. Ramamoorthy and G. S. Ho, "Performance evaluation of asynchronous concurrent systems using Petri nets," *IEEE Trans. Software Eng.*, vol. SE-6, pp. 440–449, 1980.