Analysis of Preemptive Periodic Real Time Systems using the (max,plus) Algebra

With applications in robotics

François Baccelli[*], Bruno Gaujal[†] and Daniel Simon[‡]

Abstract

In this paper we present the model of a system of periodic real-time tasks with fixed priorities, preemption and synchronization, performed by a robot controller, using Marked Graphs.

Then, with the help of the (max,plus) algebra, we derive simple tests to check real time constraints on those tasks such as response times and the respect of deadlines. This method takes into account precedence and synchronisation constraints and is not limited to a particular scheduling policy.

keywords Periodic real-time systems, synchronisation, fixed priority preemption, marked graphs, (max,plus) algebra.

1 Introduction

Marked graphs can be used to model processes with highly synchronous behaviours, see [1, 2, 3] for example. Here, we focus on the study of several marked graphs which interact via certain preemption schemes.

Such systems appear in the modelling of sets of tasks performed by on-board processes in a robot controller. Some tasks have high priority and therefore must preempt the low priority tasks. All tasks, regardless of priority, have real time constraints to meet : each task is triggered by events (by a clock or by a precedent task), and is made runnable at each occurrence of these events. Each task must run to completion before the next activation period. The ORCCAD system presented in Section 2 is a good example of such a system.

This paper gives simple answers to a series of problems with an increasing degree of difficulty, all related to the quantitative behaviour of such systems :

- *(max,plus) representation.* We provide a (max,plus) representation of the system via a Petri net model using different time scales associated to the different priority levels (*contracted time* in Section 3 and *expanded time* in Section 5)
- *Periodicity.* A first structural problem that we address is the periodicity issue. Under rather general assumptions, we show that the whole system reaches a periodic regime after some transient behaviour.
- *Cycle time.* Another key practical problem is to compute the speed of the system once it has reached its periodic regime. When the contracted time model is valid, the answer to this question is quite simple and is given as the ratio of the cycle time without preemption by the busy period of the preemptive tasks (see section 3.2).
- *Response times.* A more precise performance measure is the response time of each task defined as the duration between the time it becomes runnable and the time of its completion. A (max,plus) representation of this quantity is given in Section 4.2 whereas a brief presentation of the (max,plus) algebra is given in appendix.
- *Real time constraints.* Finally, we give a simple test to check whether the system complies with its real time constraints, during the transient phase as well as during the periodic regime (Section 4.4).

The use of the (max,plus) framework to address a real time problem is rather new, especially when preemption is possible as it is the case here. Traditional techniques [4, 5] use worst case response times

[*]François Baccelli is with ENS, 45 rue d'Ulm, Paris, Francois.Baccelli@ens.fr

[†]Bruno Gaujal is with Loria, 615 rue du jardin Botanique, Villers lès Nancy, Bruno.Gaujal@loria.fr

[‡]Daniel Simon is with Inria Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot, Daniel.Simon@inrialpes.fr

to analyse fixed-priority real-time systems. The method proposed in this paper considers the interleaving between tasks to obtain less conservative estimates of system response times. Moreover the computational complexity of the method is lower than the computational complexity associated with a full search of the reachability graph of the system.

In the last part of the paper, we show that this approach can be partially generalised to an arbitrary set of Marked Graphs equipped with partial order relations between sets of transitions. In that case, we cannot always use the contracted time approach but rather expand the firing time of the transitions. The main result shows that the system reaches a periodic regime which does not depend on the initial conditions (see Section 5).

2 Modelling of a Real Time System : The ORCCAD Framework

This section is primarily a motivation section. We consider a specific framework for real time systems, ORCCAD, and show how to model its logical and timed behaviour by marked graphs. We believe that the mathematical models that we develop in this section and study later on are nevertheless generic.

ORCCAD is a software environment dedicated to the design, the verification and the implementation of robotic control systems. It also allows us to specify and validate robotic missions [6].

Periodic and multi-rate communicating tasks are executed under the control of an off-the-shelf real-time operating system : fixed priorities are assigned to the tasks, which can be preempted at any time by a higher priority task which becomes runnable. A preempted task resumes at the point it has been preempted when all tasks with higher priority become idle.

The structure of the periodic tasks, which are called module-tasks (MTs), is as follows : after initialisation, an infinite loop is executed where all input ports are first read, calculations are performed on these inputs and finally results are posted on all the output ports.

2.1 Synchronisation

The general idea is that a partial synchronisation of tasks allows for the specification of precedence constraints and thus improves the control performance by decreasing the computing latency [7]. Several kinds of one-to-one protocols can be used on input/output ports in order to synchronise more or less tightly the set of MTs :

- ASYN-ASYN : a communication of this type does not lead to synchronisation between the reading and writing tasks, thus it must be implemented using a fully asynchronous communication mechanism, e.g. [8].
- SYN-SYN : each communication of this type is a *rendez-vous*; the first task to reach the *rendez-vous* (either the writer or the reader) is blocked until the second one is ready for communication.
- ASYN-SYN : the writing task runs freely and posts messages on its output ports at each period; the reading task either reads the data if a new one is available or is blocked until the next data production.
- SYN-ASYN : symmetrical to the previous case : the reader runs freely, the writer is blocked until the next reading request except if a new one has been posted since the last reading.

2.2 Preemptions

In addition to synchronisations, MTs may interact through another mechanism, preemption.

A control system for robotics is generally made of several calculation paths : the direct control path computing control set-points from tracking errors is often quite simple and have a small computing duration [7]. Other tasks may be used to update some parameters of the non-linear robot model. These tasks are data-handling intensive, e.g. using trigonometric functions or matrix inversion. Their duration is usually longer than the period of the direct path. Thus they must be assigned with a low priority so that their execution is preempted by every execution of the direct path calculations.

The whole system is run over a limited number of CPUs. All the MTs using the same CPU are ordered according to their priorities. When a MT with high priority becomes runnable and starts its calculation cycle, the lower priority running MT is preempted and its context is stored. The activity of the runnable

MT with the immediately lower priority resumes at the point where it was stopped as soon as the higher priority MT has finished its cycle of calculations or is blocked waiting for a synchronisation event.

2.3 Modelling with Petri nets

We need modelling and analysis tools to automatically check for time inconsistencies in the network of synchronised MTs. Such problems have been addressed in the real-time community under very general assumptions, see for example [9]. Here, we will adopt a particular modelling tool, Petri nets which will provide a simple and efficient way to carry out inconsistency tests.

As shown in Figure 1, the sequential behaviour of the simplest periodic MT (reading an input port, performing a calculation, writing to an output port) may be modelled by a Petri net with three transitions. (Of course, when an MT have multiple input and output ports, we must be careful to associate a distinct transition with each input and each output port.) A fourth transition is required to activate the MT subject to the periodic awakening associated with a real time clock (RTC), also modelled by a Petri net. Since we are concerned with temporal analysis, we associate time intervals (also called "durations") with some of the Petri net's transitions. Such Petri nets are called *timed Petri nets.* We have chosen to associate the duration $[d]$ of the MT with the computation transition, and thereby assume that reading and writing are instantaneous events. A firing time $[\tau]$ is also assigned to the transition associated with the RTC (Transitions associated with non zero duration are drawn with thick lines). Since each place have just one input transition and one

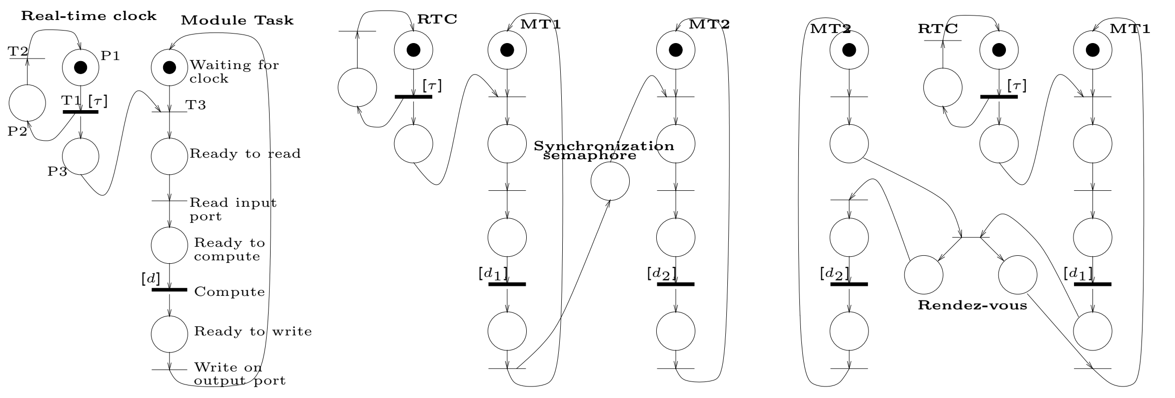

Figure 1: A Petri net model of : a periodic Module-Task, ASYN/SYN and SYN/SYN communications

output transition, the resulting Petri net is a so-called *marked graph* (or event graph). The synchronisation and communication mechanisms described in section 2.1 also have marked graph models as depicted in Figure 1.

Studying the temporal behaviour of the set of MTs is rather complex : classically this can be done through a more or less exhaustive exploration of the reachability graph of the Petri net, which can be costly in time and memory. Moreover, this Petri net does not model the priorities and the preemption due to the scheduler. Thus, this model must be refined accordingly, which will be the object of the present paper.

2.4 A Generic Marked Graph Model for Preemption Based Real Time Systems

We shall retain the following generic model from the above framework :

The model consists of a set of tasks $\mathcal{T}_i$, $i = 1, ..., N$. Each task $\mathcal{T}_i$ can be modelled by a connected marked graph $\mathcal{G}_i = (\mathcal{Q}_i, \mathcal{P}_i, M_i, \sigma_i)$,

- $\mathcal{Q}_i$ is the set of numbered transitions. We denote its size Q_i;
- $\mathcal{P}_i$ is the set of numbered places. We denote its size P_i;
- $\sigma_i = (\sigma_{i_1}, \cdots, \sigma_{i_Q})$ is the set of firing times, σ_{i_q} being the firing duration of transition q. We will assume that these numbers are all multiples of the smallest time unit that can be handled by the system. Therefore, they can be seen as integer numbers.
- $M_i(r, q)$ is the initial marking in the place between r and q, $r, q \in \mathcal{Q}_i$ (if this place does not exist, $M_i(r, q)$ is not defined).
- We also denote by $q^\bullet$ and ${}^\bullet q$ the output places and input places respectively, of transition q.

Moreover, the strongly connected components of $\mathcal{G}_i$ are partitioned into two sets :

- The set of initial components, denoted I_i. An initial component will be called a *clock*. In most practical cases, this clock is composed of a single recycled transition. However, nothing forbids to consider more elaborate clocks, and we will make no restrictive assumption concerning these initial components besides the fact that *they cannot be preempted* and that they have no computing activity. In other words, they always deliver ticks according to their timing specification and do not load the computing resources shared by the module-tasks.
- All the other components, denoted O_i. They are often simple cycles, for single task models but may be more complicated.

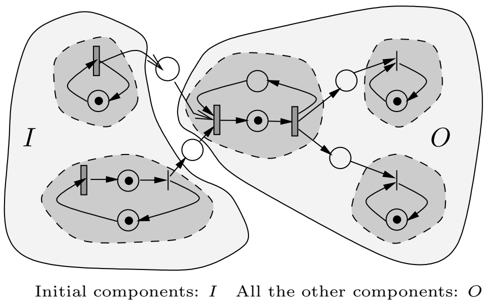

Figure 2: A graph with decomposition in its components I and O

The preemption between tasks is given under the form of an order relation between the graphs $\mathcal{G}_i$. If $\mathcal{G}_j \succ \mathcal{G}_i$, whenever a transition in O_j fires, every firing transition in O_i is suspended and resumes its firing as soon as all activities in O_j stop, e.g. after normal termination or waiting for some synchronisation event. Note that the clocks of $\mathcal{G}_i$ or $\mathcal{G}_j$ are not involved in the preemption process.

2.5 Examples

2.5.1 Case 1 : A Simple Case with no Preemption

This first example shows a group of two communicating tasks using a ASYN/SYN communication. However, there is no preemption in this case. This system is represented in Figure 3. Each task have its own clock that sets the period of each task (20 and 15 units of time respectively).

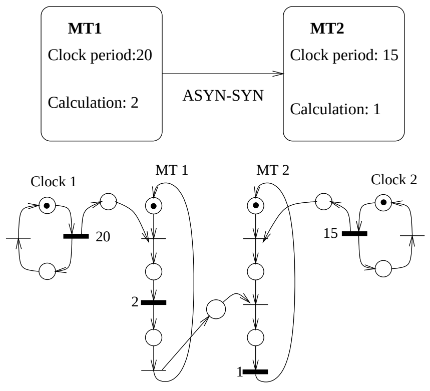

Figure 3: Two communicating tasks and their Marked Graph model

2.5.2 Case 2 : A case with preemption

Here, we show a realistic model of several tasks ($MT1$ to $MT7$) used for computed-torque robot control in the ORCCAD system. The priorities are set such that

$$\{MT1, MT2\} \succ \{MT3, MT4, MT5, MT6\} \succ \{MT7\}$$

Priorities has been set according to the relative urgency of the tasks. Here, MT1 and MT2 are observers checking for safety conditions, MT3–MT6 is the direct control path and the long duration MT7 computes an explicit model of the robot arm dynamics. Note that the analysis method we present is not limited to a particular scheduling policy, e.g. rate-monotonic or deadline-monotonic [5]. Thus the scheduling policy can be chosen to minimize a control performance index like a measure of the tracking error, e.g [10], while preserving the analysis capability. Figure 4 is the Marked Graph model of the system under a somewhat

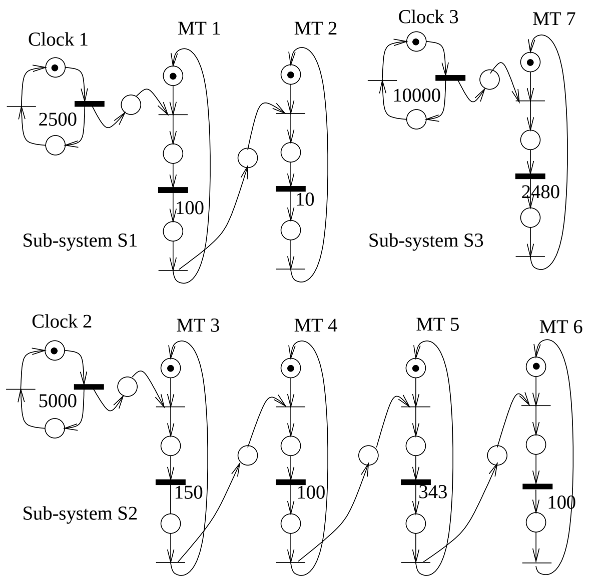

Figure 4: Marked graph model of a robot controller, involving preemption, with parallel execution

unrealistic assumption : as nothing prevents transitions of a timed Petri net to be active simultaneously, $MT1$ and $MT2$ can run in parallel, as well as tasks $MT3$ to $MT6$. Thus it is assumed that there is always a processing unit available for each runnable timed transition. If there is only one processing unit (which

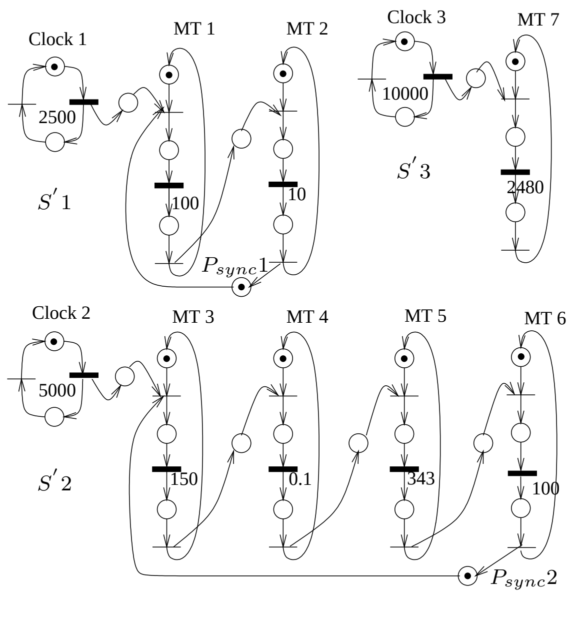

Figure 5: Marked graph model of a robot controller, involving preemption, with one processor

is often the case) then all tasks in $\{MT1, \ldots, MT7\}$ must run in mutual exclusion. The additional initially marked places $P_{sync}1$ and $P_{sync}2$ serialize the activities inside the cluster of tasks with equal priorities. Exclusion between the different clusters of tasks is handled by the preemption. The Marked Graph model of this system, given in Figure 5, is currently generated by the Orccad verifier from a block-diagram oriented GUI.

2.6 Problems to be addressed

The problem which this paper addresses is to check whether all tasks will meet their time constraints, that is if each task will be executed within the time slot that it is given by its clock. This general problem will be called Problem P_1 in the following text. In cases where there is no preemption this problem is rather classical and does not require the whole machinery developed below.

A less restrictive version of this problem is to check whether violation of the time constraints may only happen a finite number of time. This is called Problem P_2 in the following.

In the marked graph models, these problems can be formulated in the following way :

- *for Problem P_1 :* for all marked graphs $\mathcal{G}_i$, the marking in the places that connect initial components I_i to any component O_i, is bounded by one.
- *for Problem P_2 :* For all marked graphs $\mathcal{G}_i$, the marking in the places that connect initial components I_i to any component O_i, may get larger than one for a finite number of occurrences.

2.7 Real time scheduling analysis

Scheduling analysis over a set of recurrent real-time tasks have been studied thoroughly since [5]. Here , we give an overview of the techniques traditionally used to address real time scheduling analysis to put our approach in perspective.

Worst case analysis Usually, real time systems are modelled by a set of recurrent tasks assigned to one or several processors, as this is the case since the seminal work of Liu and Layland [5] and further work [11]. Each task τ_i is characterized by a couple (C_i, T_i) where C_i is the execution time of each instance of task τ_i and T_i is the period of τ_i. In our context, a task is modelled by a transition of the Petri net and its execution time is the firing time of that transition. However, in our case, the pseudo-period of the task is not necessarily given *a priori* but can depend on other features of the system, e.g synchronisation with preceding tasks leading to periods inheritance. This is one important difference from the usual approach since one important test that must be done on our system is to check whether the periods are adequate (stability issues are addressed in Section 3.2). Another important difference lies in the data and temporal dependencies between tasks, which is in our case specified through synchronisations and also modelled with marked graphs (while conflicts on shared data access are avoided using an asynchronous multiple-buffers mechanism). Although dependence relations between tasks can be taken into account using the traditional approach (as in [12]), this has only been done for periodic tasks. To deal with sporadic tasks (which is our case here), some authors such as [13, 9] have replaced the data (C_i, T_i) by a *work arriving function.* However, precedences are difficult to take into account in this context. Yet another difficulty of addressing Problems P_1 and P_2 with classical approaches is the fact that neither problem can be simply written under the form of a deadline. They deal with the interleaving of events coming from several sources. This would require the computation of upper bounds (as is usually done) as well as lower bounds (this seems to be substantially more difficult).

Real time logic Real time logic has been introduced as an extension of temporal logic (see for example the survey [14]) and later extended, as for example in [15, 16]. This formalism is useful to formulate real time properties which are more complicated than deadlines and worst response times (such as Problems P_1, P_2, or even more subtle real time properties). The underlying model over which a real time logic formula can be written is a timed graph as defined in [17]. Here, that would be the state reachability graph of the Petri net enriched with timing information. From that point on, a real time formula can be checked using a formal derivation of the formula to its value (true or false). It can be shown (as in [15]) that most formulae are decidable (this is the case for P_2 and P_1). Although real time logic is a very powerful tool in terms of modelling power, it suffers from a high computational complexity. In our case, the state reachability graph can be exponential (and even infinite) in the number of transitions. Also, checking the value of a formula may be exponential in time and space.

In contrast, the (max,plus) approach that we present here stays at the net level (and not at the state level) and the time complexity for checking P_1 and P_2 is quadratic in the period of the system and cubic in the size of the net. Indeed, the (max,plus) formalism enables us to efficiently use the operators' properties (such as commutativity and associativity) to keep the complexity low, as in the case in Section 4.3. Moreover, our model of tasks is simple enough (but still useful for practical many applications) to stay with a linear model, thus allowing for efficient computing of numerical values, e.g. response time of tasks and length of the transient phase. This is now implemented in the ERS environment [18], a graphical interface dealing with Petri nets, (max,plus) systems and stochastic task graphs analysis.

It really seems that the Petri net model given here and the (max,plus) technique developed below are both well adapted to this case.

3 Modelling Under Contracted Time

In this section, we introduce a representation of the system where we contract time in the non-preempted transitions. This is a way to take into account the scheduler without modelling it by a Petri net, thus allowing one to still use marked graphs.

3.1 Description of the Model

We simply consider the case where $N = 2$ which is generic, as we will see in the following.

Let $\mathcal{G}_1 \succ \mathcal{G}_2$. In the following, $\mathcal{G}_1$ will only be seen through its activity process, $S_1(t)$ that is the process which is equal to 1 if a transition in O_1 is active and 0 otherwise. It is assumed to be periodic, with period T_1. As for $\mathcal{G}_2$, we will remove the index 2, since every variable considered in the following, will be for the marked graph $\mathcal{G}_2$.

We construct a copy of $\mathcal{G}(= \mathcal{G}_2)$, that we call $\mathcal{H}$ which will be considered without the preemption from $\mathcal{G}_1$.

In the following, for each Strongly Connected Component (SCC) $\mathcal{C}$, we will denote

- $\{X_q(n)\}$ the sequence of firing times of transition $q \in \mathcal{C}$, in the preempted system $\mathcal{G}$. The set of all these sequences will also be called the behaviour of the system;
- $\{V_q(n)\}$ the sequence associated with the same transition in $\mathcal{H}$.

By using the theory of timed marked graphs and Lemma A.1 in the appendix, we get for all SCC $\mathcal{C}$ in isolation, a *cycle time* $\lambda_\mathcal{C} \in \mathbb{R}_+$, a *cyclicity* $s_\mathcal{C} \in \mathbb{N}_+$ and a *transient period* $k_\mathcal{C} \in \mathbb{N}$, such that for all transitions $q \in \mathcal{C}$ and all $k \geqslant k_\mathcal{C}$,

$$V_q(k + s_\mathcal{C}) = V_q(k) + s_\mathcal{C}\lambda_\mathcal{C}. \tag{1}$$

Note that the activity process of this SCC is therefore periodic of period $s_\mathcal{C}\lambda_\mathcal{C}$. This period is an integer under the assumptions that we made on the firing durations.

3.2 Time Contraction

We denote by $S_1(t), t \in \mathbb{R}$ the activity process of $\mathcal{G}_1$, defined by $S_1(t) = 1$ if at time t a transition in $\mathcal{G}_1$ is active and $S_1(t) = 0$ otherwise. This function is assumed to be periodic of period T_1, where T_1 is an integer.

We define $\Gamma_1 \overset{\text{def}}{=} \int_0^{T_1} 1 - S_1(\tau)d\tau$ (under our assumptions, Γ_1 is also an integer number), and $F_1 : \mathbb{R} \to \mathbb{R}$ by

$$F_1(t) = \int_0^t 1 - S_1(\tau)d\tau.$$

Figure 6 gives a representation of S_1 and F_1. During each period T_1, $\mathcal{G}_1$ is not active for Γ_1 units of time. Also, F_1 is *pseudo-periodic* of period T_1 and increment Γ_1 : *i.e.* $F_1(t + T_1) = F_1(t) + \Gamma_1$. We construct F_1^{-1} as the unique left continuous function such that $F_1^{-1}(F_1(t)) = t$.

Let $X'_q(k) \overset{\text{def}}{=} F_1(X_q(k))$ for all $q \in \mathcal{Q}$. By definition of F_1^{-1}, we have, $X_q(n) = F_1^{-1}(X'_q(n))$. As shown by the following lemma, the sequences $\{X'_q(k)\}$, which give the firing times of transition q, after this time contraction, are also ultimately pseudo-periodic.

Lemma 3.1. *For all SCC $\mathcal{C}$ in isolation, we have :*
i- if $\mathcal{C} \subset I$, then for all transition $q \in \mathcal{C}$ and for all $n > k_\mathcal{C}$, $X'_q(n + s'_\mathcal{C}) = X'_q(n) + \lambda_\mathcal{C}\Gamma_1/T_1$ with $s'_\mathcal{C} = lcm(T_1, s_\mathcal{C}\lambda_\mathcal{C})/\lambda_\mathcal{C}$. ii- If $\mathcal{C} \subset O$, we have all transition $q \in \mathcal{C}$ and for all $n > k_\mathcal{C}$, $X'_q(n + s_\mathcal{C}) = X'_q(n) + s_\mathcal{C}\lambda_\mathcal{C}$.

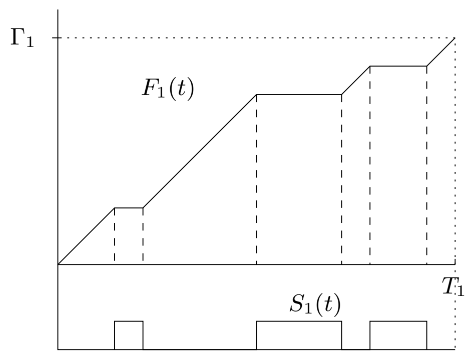

Figure 6: The contraction function F_1 associated with an activity S_1.

Proof. *i-* Since no transition in I is preempted, for all transitions $q \in \mathcal{C} \subset I$, $X_q(n) = V_q(n)$. Moreover, by equation (1), $V_q(n + s_{\mathcal{C}}) = V_q(n) + s_{\mathcal{C}}\lambda_{\mathcal{C}}$ for $n \geqslant k_{\mathcal{C}}$. Using the fact that $s'_{\mathcal{C}} = \alpha s_{\mathcal{C}} = \beta T_1/\lambda_{\mathcal{C}}$,

$$\begin{aligned} X'_q(n + s'_{\mathcal{C}}) &= F_1(V_q(n + s'_{\mathcal{C}})) \\ &= F_1(V_q(n + \alpha s_{\mathcal{C}})) \\ &= F_1(V_q(n) + \alpha s_{\mathcal{C}}\lambda_{\mathcal{C}}) \\ &= F_1(V_q(n)) + \beta\Gamma \\ &= X'_q(n) + s'_{\mathcal{C}}\lambda_{\mathcal{C}}\Gamma_1/T_1. \end{aligned}$$

ii- Recall that all transitions of SCC's belonging to O are simultaneously preempted by the activity of $\mathcal{G}_1$. Therefore, in contracted time, the behaviour of the every SCC in O considered in isolation is the same as in real time. □

The only difference in contracted time compared with the net in real-time is that the arrivals stemming from the SCCS of I have to be replaced by those obtained from the sequences $\{X'_q(n)\}$, $q \in I$ defined above. These arrivals are accelerated by the time change, but remain nevertheless pseudo-periodic as we just saw. However, Lemma 3.1 provides a period which is not necessarily the smallest period of the clock in contracted time.

We then get from the results following Lemma A.1 in the appendix that the sequences $\{\widehat{X}'_q(n)\}$, $q \subset \mathcal{C} \in O$ which give the firing time of the global system in contracted time become ultimately pseudo-periodic. More precisely, let us define the quantities $\lambda'_{\mathcal{C}}$ as follows : if $\mathcal{C} \subset I$, then $\lambda'_{\mathcal{C}} \overset{\text{def}}{=} \lambda_{\mathcal{C}}\Gamma_1/T_1$. If $\mathcal{C} \subset O$, then $\lambda'_{\mathcal{C}} \overset{\text{def}}{=} \lambda_{\mathcal{C}}$. These quantities are the inverse of the firing rate of the transitions in contracted time. There are two cases :

1. Either the system is stable (see Appendix A.3), which happens if and only if
$$\min_{\mathcal{C}\in I} \lambda'_{\mathcal{C}} \geqslant \max_{\mathcal{C}\in O} \lambda'_{\mathcal{C}}.$$
Let then $\mathcal{C}_0$ be a SCC such that $\min_{\mathcal{C}\in I} \lambda'_{\mathcal{C}} = \lambda'_{\mathcal{C}_0}$. In this case, all sequences $\{\widehat{X}'_q(n)\}$ couple in finite time with a pseudo-periodic regime such that
$$\widehat{X}'_q(n + s'_{\mathcal{C}_0}) = \widehat{X}'_q(n) + \lambda_{\mathcal{C}_0} s'_{\mathcal{C}_0}\Gamma_1/T_1,$$
where $s'_{\mathcal{C}_0}$is defined as in Lemma 3.1. In this case, the marking in all places is bounded, and both the marking and the activity processes are ultimately periodic functions of time.
2. Or the system is unstable, which happens if and only if
$$\min_{\mathcal{C}\in I} \lambda'_{\mathcal{C}} < \max_{\mathcal{C}\in O} \lambda'_{\mathcal{C}}.$$
In this case, some places have a marking which tends to ∞. All sequences $\{\widehat{X}'_q(n)\}$ couple in finite time with a pseudo-periodic regime which we will not study here because this case corresponds to an unproper behaviour of the system.

At this point, a few remarks are in order :

- The condition of stability of the whole system is stronger in the preempted case than in the non-preempted case since the stability region in this last case reads

$$\min_{\mathcal{C}\in I} \lambda_{\mathcal{C}} \leqslant \max_{\mathcal{C}\in O} \lambda_{\mathcal{C}}.$$

- The fact that the system is ultimately periodic in contracted time implies that it is also periodic in real time (as we see when applying back the function F_1^{-1} to the ultimately periodic sequences of interest).
- In the stable case, the activity process becomes ultimately periodic both in contracted and in real time, which allows one to proceed by induction when the number of levels N is larger that 2.

3.3 Analysis of a simple example

Consider the model of Figure 7, which is made of two SCC. The clock triggers the activity of the second SCC (O) and is not preempted. The second SCC is preempted by a system with an activity process with period $T_1 = 5$ and such that $F_1(T_1) = 3$. This preemption process is similar to the activity of $\mathcal{G}_1$ in Figure 7, for example. The firing time of the clock transition is $\sigma = 4$. The firing times of the two transition in O are 2 and 1 respectively.

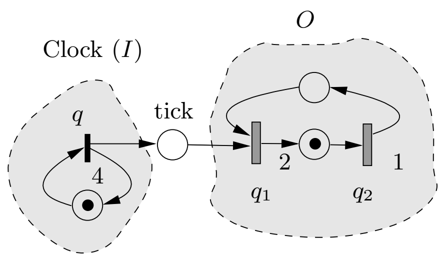

Figure 7: A clock and its synchronous circuit.

The stability condition can be tested here : first, note that with the data given in Figure 7, we have $T_1 = 5, \Gamma_1 = 3$. In contracted time, for the clock :
$\widehat{X}'_1(n+5) = \widehat{X}'_1(n) + 4 \times 3$
For the system O, we have : $\widehat{X}'_2(n+1) = (\widehat{X}'_2(n) \otimes 3) \oplus \widehat{X}'_1(n)$.
Therefore, although when there is no preemption the system is stable : $\lambda_1 = 4 > \lambda_2 = 3$, under preemption, we have $\lambda'_1 = 12/5 < \lambda'_2 = 3$ and the system becomes unstable. The marking of the place "tick" goes to infinity.
From the real-time computing point of view, this means that the task represented by O is restarted before completion, i.e. misses its deadline leading to a system failure.

4 Detailed Analysis of the Periodic Regime

In this section, we will consider further properties of the periodic regime of the system.
In our robot control applications, a real-time property to be met by the controller is : each task must be executed once between two ticks of its clock.
In the marked graph model, as mentioned in Section 2.6, this corresponds to verifying whether a token produced by the clock finds an empty place (the constraint is met) or not (the previous task has not been executed before the new clock tick, and the constraint is violated).

4.1 Formulation of the problems in the (max,plus) algebra

The following arguments will be done in contracted time. We focus on a system with one initial component $\mathcal{C}_0$ in I and one component $\mathcal{C}_2$ in O. $\mathcal{C}_0$ is connected to $\mathcal{C}_2$ through a place (called "tick" in Figure 7). The output transition of place "tick" (which belongs to $\mathcal{C}_2$) is numbered q_1.

Let us denote $u(n)$ the epoch of the nth arrival of a token in place tick, in contracted time. We have $u(n) = F_1(t_n)$, where t_n is the date of the n-th arrival in tick in real time. We also define $\tau(n) = u(n+1) - u(n)$.

Note that using Lemma 3.1, $\{u(n)\}$ is pseudo-periodic with period $T = lcm(T_1, s_{\mathcal{C}_0}\lambda_{\mathcal{C}_0})/\lambda_{\mathcal{C}_0}$ and $u(n+T) = u(n) + \sigma\Gamma$ with $\sigma = \lambda_{C_0}$ and $\Gamma = \Gamma_1 T/T_1$.

As for the process of $\mathcal{C}_2$, under contracted time, it is a marked graph identical to the original marked graph without preemption (in isolation). Again, this is a consequence of Lemma 3.1). Therefore, under contracted time, the whole system is an open (max,plus) system which can be represented under the form

$$X(n) = A \otimes X(n-1) \oplus B \otimes u(n). \tag{2}$$

We will denote by n_0, c, γ the coupling, cyclicity and maximal eigenvalue of matrix A, respectively (see Appendix A.1 and A.3 where (max,plus) notations and their relations with the dynamics of marked graphs are described).

Algebraic formulations of the problems

Lemma 4.1. *Problem P_1 can be written as*

$$X_1(n) - u(n+1) < 0, \quad \forall n \geqslant 1, \tag{3}$$

where $X_1(n)$ is the n-th firing time of transition q_1 and Problem P_2 can be written as

$$X_1(n) - u(n+1) < 0, \quad \forall n \geqslant n_0, \tag{4}$$

where n_0 is the time when the periodic regime is reached.

Proof. $X_1(n)$ is the instant when transition q_1 removes the nth token in place "tick" while $u(n+1)$ is the instant when the $n+1$th token is put in place "tick" by the clock. If $X_1(n) > u(n+1)$, then there are at least two tokens in place "tick" during the interval $[u(n+1), X_1(n)]$. □

In the following, we will assume that

$$X_1(1) = u(1). \tag{5}$$

This assumption is natural in our context : it means that the system is ready to start as soon as the clock emits its first signal.

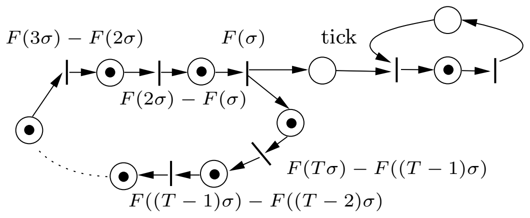
$F(3\sigma) - F(2\sigma)$
$F(\sigma)$
tick
$F(2\sigma) - F(\sigma)$
$F(T\sigma) - F((T-1)\sigma)$
$F((T-1)\sigma) - F((T-2)\sigma)$

Figure 8: Representation of the clock in contracted time

Remark In the case where the clock is a simple recycled transition (as in Figure 7) with firing duration σ, we have $u(n) = F_1(n\sigma)$, $n \geqslant 1$. We can also give an exact representation of the whole clock under contracted time, displayed in Figure 8. Under contracted time, the first token must arrive in place "tick" at time $F_1(\sigma)$; the second token at time $F_1(2\sigma)$, and so forth, with a period T. This allows one to derive response times in contracted times which can be converted in real time by applying F_1^{-1}.

4.2 Problem P_2

- The first case is when the clock cycle time ($\sigma\Gamma/T$) is smaller than the maximal eigenvalue γ of matrix A. In this case, $X_1(n)$ is of the same order as $n\gamma$ when n goes to infinity and $u(n+1)$ is of order $n\sigma\Gamma/T$. Therefore, there exists some n_0 such that $u(n+1) > X_1(n)$ for all $n \geqslant n_0$. In this case none of the properties P_2 and P_1 are satisfied. - Now, consider $\sigma\Gamma/T > \gamma$ (**Assumption H1**)
(the case $\sigma\Gamma/T = \gamma$ can be treated by a similar method but requires some additional technicalities). We define the vector $Z(n) = X(n) - u(n+1)$. Since $\sigma\Gamma/T > \gamma$, $X(n)$ is also ultimately pseudo-periodic with period T and cycle time $\sigma\Gamma/T$ (see Lemma A.1). Therefore, the variable $Z(n)$ is ultimately periodic with period $T : \forall k \geqslant k_0, \quad \forall 0 \leqslant s \leqslant T-1$,

$$Z(kT+s) = Z((k-1)T+s). \tag{6}$$

Using the formula for $X(kT+s)$, we also have for all $k \geqslant 1$ and $0 \leqslant s \leqslant T-1$,

$$\begin{aligned}
&Z(kT+s) = X(kT+s) - u(kT+s+1)\\
&= \Big(A \otimes X(kT+s-1) \oplus B \otimes u(kT+s)\Big) - u(kT+s+1)\\
&= A \otimes D(-\tau(kT+s)) \otimes Z(kT+s-1) \oplus B \otimes (-\tau(kT+s))\\
&= A^{\otimes T} \otimes D(-\sigma\Gamma) \otimes Z((k-1)T+s)\\
&\bigoplus_{i=0}^{T-1} \Big(A^{\otimes i} \otimes D(-\tau(kT+s) - \cdots - \tau(kT+s-i)) \otimes B\Big),
\end{aligned}$$

where $D(x)$ is the diagonal matrix with x on the diagonal. All of this is true using the fact that the diagonal matrix $D(.)$ commutes with everything.

Let $M = A^T \otimes D(-\sigma\Gamma)$ and

$$C(s) \quad = \quad \bigoplus_{i=0}^{T-1} A^i \otimes D(-u(kT+s+1) + u(kT+s-i)) \otimes B.$$

Using Equation (6), we can rewrite the last relation

$$Z(kT+s) = M \otimes Z(kT+s) \oplus C(s), \quad \forall k \geqslant k_0. \tag{7}$$

Since the matrix M have a maximal eigenvalue which is negative by Assumption **H1**, then the matrix M^* exists (see Appendix A.2 for the definition and the condition of existence of matrix M^*) and the minimal solution of (7) is equal to

$$Z(kT+s) = M^* \otimes C(s), \quad \forall k \geqslant k_0. \tag{8}$$

Theorem 4.2. *Property P_2 is satisfied if and only if coordinate 1 in $M^* \otimes C(s)$ is non-positive for all $0 \leqslant s \leqslant T-1$.*

Computation Complexity The computation of $Z(kT+s)$ requires the calculation of $A^{\otimes 2}, \cdots, A^{\otimes T}$, which takes $O(T|Q|^3)$ units of time. The computation of M^* takes $O(|Q|^3)$ units of time and the computation of $C(s)$ for each s takes $O(T)$ units of time. The total complexity is $O(T|Q|^3 + T^2)$ Note however, that it is important to have a low complexity in T since T may be large.

4.3 Initial Phase Issues

It is often the case that the different tasks of the operating systems may have different initial phases each time the system is started anew.

Theorem 4.2 gives a test for Problem P_2 for a fixed given phase, however, we would like to derive a test to ensure that the time constraint is satisfied for all possible phases between the preemptive system and the preempted one. Such problems have been studied in [9] for example.

Since time is slotted, the total number of phases ϕ is finite, equal to T. A brute force formula to check problem P_2 under all possible phases is to check whether

$$\bigoplus_{\phi=0}^{T-1}\bigoplus_{s=0}^{T-1} M^* \otimes C_\phi(s),$$

have a non-positive first coordinate, with $C_\phi(s)$ defined as above with $\tau(kT+s)$ replaced by $\tau_\phi(n) \overset{\text{def}}{=} u_\phi(n+1) - u_\phi(n)$, where the variables $u_\phi(n) \overset{\text{def}}{=} F_1(t_n+\phi)$ are the clock ticks under phase ϕ.

However, the complexity of this formula is in $O(T^3)$, which could be prohibitive when T grows (this happens in particular when the preemption gets very complex, like in the case of the superposition of several preemptive tasks).

We now derive a better formula, by characterising the worst possible phase between the two systems.
We consider the first coordinate of $Z(kT+s)$, and the case where B is the vector $(0, -\infty, \cdots, -\infty)$. Then

$$\begin{aligned}
Z(kT+s)_1 &= \left(M^* \otimes \left(\bigoplus_{i=0}^{T-1} A^i \otimes D(-u_\phi(kT+s+1)\right.\right. \\
&\quad \left.\left. +u_\phi(kT+s-i)) \otimes B\right)\right)_1 \\
&= \bigoplus_{i=0}^{T-1} (M^*A^i)_{1,1} \otimes (-u_\phi(kT+s+1)) \\
&\quad \otimes u_\phi(kT+s-i).
\end{aligned}$$

Now, we maximise over all s and all ϕ,

$$\begin{aligned}
K &\overset{\text{def}}{=} \bigoplus_{\phi=0}^{T-1}\bigoplus_{s=0}^{T-1} Z(kT+s)_1 \\
&= \bigoplus_{i=0}^{T-1} (M^*A^i)_{1,1} \bigoplus_{\phi=0}^{T-1}\bigoplus_{s=0}^{T-1} (-u_\phi(kT+s+1)) \otimes u_\phi(kT+s-i) \\
&= \bigoplus_{i=0}^{T-1} (M^*A^i)_{1,1} \bigoplus_{\phi=0}^{T-1}\bigoplus_{s=0}^{T-1} (-u_\phi(s+i+1)) \otimes u_\phi(s)
\end{aligned}$$

We choose $s^* = s^*(i)$ such that $t_{i+1+s^*} - t_{s^*} \leqslant t_{i+1+s} - t_s$ for all $0 \leqslant s \leqslant T$. We choose ϕ^* such that $F_1(t_{i+1+s^*}+\phi^*) - F_1(t_{s^*}+\phi^*) \leqslant F_1(t_{i+1+s^*}+\phi) - F_1(t_{s^*}+\phi)$ for all $0 \leqslant i \leqslant T$ and for all $0 \leqslant \phi \leqslant T$.
Now, for all i, using the fact that F_1 is non-decreasing, we have

$$\begin{aligned}
-u_{\phi^*}(s^*+i+1) + u_{\phi^*}(s^*) &= -F_1(t_{i+1+s^*}+\phi^*) + F_1(t_{s^*}+\phi^*) \\
&\geqslant -F_1(t_{i+1+s}+\phi) + F_1(t_s+\phi) \\
&= -u_\phi(s+i+1) + u_\phi(s),
\end{aligned}$$

for all s and ϕ. Therefore, the worst case is reached with $s = s^*$ and $\phi = \phi^* = \phi^*(i)$. Checking Problem P_2 for all possible phases to done by checking whether

$$K = \bigoplus_{i=0}^{T-1} (M^*A^i)_{1,1} \otimes (-u_{\phi^*}(s^*+i+1)) \otimes u_{\phi^*}(kT+s^*)$$

is non-positive.

Complexity Note that the construction of the worst phases for the clock as well as for the preemption process must be done for all possible values of i. For a fixed i, this computation can be done independently of the rest. The computation of s^* takes $O(T)$ units of time, as well as the computation of ϕ^*.

Therefore, checking problem P_2 for all possible phases can be done in $O(T^2)$ units of time instead of $O(T^3)$ with the brute force computation.

4.4 Transient Issues : Problem P_1

In this section, we show how to check property P_1. For this we have to study the transient period.

We recall that A is ultimately pseudo-periodic with coupling time n_0 and period c. The process $\tau(n)$ is ultimately periodic with coupling time m_0 and period T. The vector $\mathbf{e}$ is the vector of size $|\mathcal{I}|$ with each component equal to 0.

We will derive an upper bound on the length of the transient period of the global system. Then, the verification of property (P_1) can be done simply by computing the behaviour of the system up to that bound and checking (P_1) at each step. To derive this upper bound, we use the variable $W(n) = X(n) - u(n)$ which satisfies an equation similar to Equation (7)

$$W(n+1) \quad = \quad A \otimes D(-\tau(n)) \otimes W(n) \oplus B, \quad \forall n \geqslant 1. \tag{9}$$

Note that $W(n) \geqslant 0$ for all n. We set $H(m,n) = A^{\otimes(n-m)} \otimes D(-\tau(n) - \cdots - \tau(m+1))$ if $m < n$ and $H(n,n) = Id$.

Lemma 4.3. *For all $m \geqslant 0$, $H(m,n)$ have coupling time $t_0 = \max(n_0, m_0)$ and is pseudo-periodic in n with period $p = lcm(T,c)$, where c is the cyclicity of A.*

Proof. Indeed, for all $n \geqslant t_0$,

$$\begin{aligned} H(m,n+p) &= A^{\otimes(n+p-m)} \otimes D(-\tau(n+p) - \cdots - \tau(n+1)) \\ &\quad \otimes D(-\tau(n) - \cdots - \tau(m+1)) \\ &= (\gamma p/c - \sigma\Gamma p/T) \otimes A^{\otimes(n-m)} \otimes \\ &\quad D(-\tau(n) - \cdots - \tau(m+1)) \\ &= (\gamma p/c - \sigma\Gamma p/T) \otimes H(m,n). \end{aligned}$$

Note that this is true for all values of m. □

As for $W(n)$, we get for all $k > 1$,

$$\begin{aligned} W(t_0+kp) &= H(0,t_0+kp) \otimes W(1) \oplus \bigoplus_{i=1}^{t_0+kp} H(i,t_0+kp) \otimes B \\ &= k(\gamma p/c - \sigma\Gamma p/T) \otimes H(0,t_0) \otimes W(1) \\ &\quad \oplus \bigoplus_{i=1}^{t_0+kp} H(i,t_0+kp) \otimes B. \end{aligned}$$

Choosing

$$k \geqslant \beta_1 \overset{\text{def}}{=} \frac{\mathbf{e} \otimes H(0,t_0) \otimes W(1)}{-\gamma p/c + \sigma\Gamma p/T} + 1, \tag{10}$$

which is positive, we have

$$(k-j)(\gamma p/c - \sigma\Gamma p/T) \otimes \mathbf{e} \otimes H(0,t_0) \otimes W(1) \leqslant 0, \quad j = 0,1. \tag{11}$$

And choosing

$$k \geqslant \beta_2 \overset{\text{def}}{=} \frac{\mathbf{e} \otimes \bigoplus_{i=1}^{p+t_0} H(i, t_0 + p) \otimes B}{-\gamma p/c + \sigma \Gamma p/T} + 1, \tag{12}$$

we also have

$$H(i, t_0 + (k-j)p) \otimes B \leqslant 0, \quad \forall 1 \leqslant i \leqslant p + t_0, \quad j = 0, 1. \tag{13}$$

Therefore, for k larger than $\beta_1 \vee \beta_2$, we get

$$\begin{aligned}
W(t_0 + kp) &= \bigoplus_{i=1}^{t_0+kp} H(i, t_0 + kp) \otimes B & (14)\\
&= \bigoplus_{i=p+t_0+1}^{t_0+kp} H(i, t_0 + kp) \otimes B & (15)\\
&= \bigoplus_{i=t_0+1}^{t_0+(k-1)p} H(i, t_0 + (k-1)p) \otimes B & (16)\\
&= W(t_0 + (k-1)p), & (17)
\end{aligned}$$

where Equation (14) comes from Inequality (11), Equation (15) comes from Inequality (13), both with $j = 0$, and (16) from the fact that $H(i, t_0 + kp) = H(i - p, t_0 + (k-1)p)$, for all $i \geqslant p + t_o$. Equation (17), comes from the fact that using (10) and (12), Inequalities (11) and (13) are also valid for $j = 1$.

Finally, $W(t_0 + kp) = W(t_0 + (k-1)p)$ means that W has reached its periodic regime before step $t_0 + (k-1)p)$. Once W has reached its period, then this means that all the system also reached its periodic regime.

Theorem 4.4. *The periodic regime is reached after a transient period of length at most :*

$$t_0 + \frac{\mathbf{e} \otimes H(0, t_0) \otimes (W(1)) \oplus \bigoplus_{\alpha=1}^{p+t_0} H(\alpha, t_0 + p) \otimes B}{-\gamma/c + \sigma\Gamma/T}.$$

where $W(1)$ must be computed, using the initial conditions of the system.

Note that t_0 can be replaced by a larger bound (but easier to compute) :

$$t_0 + \frac{\mathbf{e} \otimes (I \oplus A)^{2t_0+p} \otimes (B \oplus W(1))}{-\gamma/c + \sigma\Gamma/T}.$$

4.5 Application to the Examples of § 2.5

Case 1 (Figure 3) can be solved by a simple application of Lemma A.1, because it does not contain preemption This is a simple marked graph. In that case neither P_1 nor P_2 are satisfied. Indeed, task MT1 is executed every 20 units of time. The ASYN/SYN communication between MT1 and MT2 imposes that task MT2 will eventually be executed every 20 units of time. However, it have a clock period requirement of 15 units of time, which will not be met from its second execution on. This is a case in which Assumption **H1** does not hold.

This very simple example has also been tested using symbolic verification over timed automata : this process required the Atp language to model the tasks, the TCTL temporal logic to express the temporal properties to be verified and the Kronos model-checker running for several minutes to check for the consistency of the tasks timings and synchronisations.

Case 2 with parallel processing (Figure 4) contains preemption

- The first thing to do is check whether the cycle times of all tasks are all well ordered so that the system is stable (Assumption **H1**).
  The sub-system (S1) made of clock 1, MT1 and MT2 is stable : $(2500 > 100$ and $2500 > 10)$.
  Now, we consider the second connected component, made of Clock 2, $MT3$, $MT4$, $MT5$ and $MT6$ (S2). This sub-system is preempted by the first component, which have an activity of period 2500, with $T = 2500$ and $\Gamma_2 = 2390$.
  Therefore, for the stability of the second component, we have to check that $5 \times \frac{2390}{2500} > 150$, $\quad 5 \times \frac{2390}{2500} > 100$, $\quad 5 \times \frac{2390}{2500} > 343$, which is true.
  The last component (S3) is preempted by both sub-systems S1 and S2. The whole preemption process have period $T = 5000$, and a total non busy time of $\Gamma_3 = 5000 - ((100+10) \times 2 + 150 + 100 + 343 + 100) = 4087$.
  The stability property becomes : $10000 \times \frac{4087}{5000} = 8174 > 2480$. We can conclude that the whole system is stable.
- The second test is to check whether property P_2 is satisfied. We will apply Theorem 4.2 for all sub-systems.
  Sub-system (S1) satisfies P_2 because its period is one and because it is stable.
  Sub-system (S2) is preempted by sub-system 1. However, it also have period 1 (in terms of number of firings). The input under contracted time is : $u(n) = u(n-1) + 2\Gamma_2$, with $2\Gamma_2 = 4780$. Its structure can be reduced to a scalar version of Equation (2): $x_{2_1}(n) = a_2 \otimes x_{2_1}(n-1) \oplus u(n)$, with $a_2 = 150$.
  In that case, we obtain
  $z_2(k+1) = x_{2_1}(k) - u(k+1) = (a_2 - 2\Gamma_2) \otimes z_2(k) \oplus -2\Gamma_2$,
  the maximal solution of which is
  $z_2(k+1) = (a_2 - 2\Gamma_2)^* \otimes -2\Gamma_2$,
  once the periodic regime of z_2 is reached (here, when $k > 1$).
  The numerical solution is $z_2(k+1) = -2\Gamma_2 = -4780$, which is negative.
  As for sub-system (S3), we get similarly, a periodicity equal to 1, and a solution $z_3(k) = (a_3 - 2\Gamma_3)^* \otimes -2\Gamma_3$ with $2\Gamma_3 = 8174$ and $a_3 = 2480$. The solution is $z_3(k) = -8174$. Therefore, Property P_2 is satisfied in the three sub-systems.
- Finally, one have to consider the transient regime of all marked graphs, to verify problem $P1$. Here, all systems have period 1 (in number of firings) as well as a transient regime of length 1. The periodic regime is reached immediately and property P_1 is verified without using Theorem 4.4.

Case 2b (Figure 5) contains preemption and multi-tasking on a single processor. The analysis is similar to the previous case

- cycle times : the sub-system (S'1) made of clock 1, MT1-MT2 is stable : $(2500 > (1000 + 10))$.
  Now, we consider the second connected component (S'2), made of Clock 2, MT3-...-MT6. This sub-system is preempted by the first component, which have an activity of period 2500, with $T = 2500$ and $\Gamma_2 = 2390$. Stability of (S'2) holds since $5000 \times \frac{2390}{2500} = 4780 > (150 + 100 + 343 + 100)$.
  The last component (S'3) is preempted by both sub-systems S'1 and S'2. The whole preemption process have period $T = 5000$, and a total non busy time of $\Gamma_3 = 5000 - ((100+10) \times 2 + 150 + 100 + 343 + 100) = 4087$.
  The stability property holds : $10000 \times \frac{4087}{5000} = 8174 > 2480$. The whole system is stable.
- Property P_2 : sub-system (S'1) satisfies P_2 as in the parallel case.
  Sub-system (S'2) is preempted by sub-system 1. However, it also have period one (in terms of number of firings). The input under contracted time is : $u(n) = u(n-1) + 2\Gamma_2$, with $2\Gamma_2 = 4780$. Its structure can be reduced to a scalar version of Equation (2): $z_2(k+1) = x_{2_1}(k) - u(k+1) = (a_2 - 2\Gamma_2) \otimes z_2(k) \oplus -2\Gamma_2$, with $a_2 = 693$. The solution is $z_2(k+1) = (a_2 - 2\Gamma_2)^* \otimes -2\Gamma_2 = -2\Gamma_2 = -4780 < 0$, once the periodic regime of z_2 is reached.
  As for sub-system (S'3), we get similarly, a periodicity equal to 1, and a solution $z_3(k) = (a_3 - 2\Gamma_3)^* \otimes -2\Gamma_3$, with $a_3 = 2480$. and $2\Gamma_3 = -8174$. The solution is $z_3(k) = -8174 < 0$.
- Transient regime : all systems have period 1 (in terms of number of firings) as well as a transient regime of length 1. The periodic regime is reached immediately and property P_1 is also satisfied immediately.

5 A More General Model

Some of the qualitative results which where established in the previous sections (periodicity) can actually be shown for more general models. However quantitative results do not extend easily to the most general models.

5.1 A Refined Preemption Scheme

The model presented in Section 2 can be generalised. To each marked graph $\mathcal{G}_i$, we associate $\mathcal{P}_i$ which is the set of preemptive transitions and $\mathcal{R}_i$ which is the set of preemptable transitions. Now, if $\mathcal{G}_j \succ \mathcal{G}_i$, then no transition in $\mathcal{R}_i$ may fire at the same time as a transition in $\mathcal{P}_j$. Note that the previous model corresponds to the case where $\mathcal{P}_j = O_j$ and $\mathcal{R}_i = O_i$.

5.2 Behaviour of the System

If $q \in \mathcal{G}_i$, we denote by $X_{i_q}(n)$ the epoch when transition q starts its n-th firing.

For every graph $\mathcal{G}_i$, we define the isolated version of $\mathcal{G}_i$, denoted $\mathcal{H}_i$, in which no transitions are ever preempted. More formally, $\mathcal{H}_i$ is a version of $\mathcal{G}_i$ with $\mathcal{R}_i = \emptyset$. The behaviour of $\mathcal{H}_i$ is denoted $U_i(n)$.

In the event graph $\mathcal{H}_i$ with constant firing times (σ_{i_q}), the variables $U_i(n) = (U_{i_1}(n), \cdots, U_{i_Q}(n))$ satisfy an evolution equation of the form (see [1]),

$$U_{i_q}(n) = \max_{r \in \bullet q} \big(U_{i_r}(n - M_{(r,q)}) + \sigma_{i_r}\big). \tag{18}$$

We also know from [1] that Equation (18) have the following property : there exists n_{i_0}, k_{i_q} and λ_{i_q} such that for $n \geqslant n_{i_0}$, $\quad U_{i_q}(n) = U_{i_q}(n - k_{i_q}) + \lambda_{i_q} k_{i_q}$.

Also note that if $\mathcal{G}_i \succ \mathcal{G}_j$, then the behaviour of $\mathcal{G}_i$ does not depend on the behaviour of $\mathcal{G}_j$. As a consequence, if $\mathcal{G}_i$ is not preempted by any other graph, its behaviour can be determined in isolation, that is $\forall q \in \mathcal{Q}_i, n \in \mathbb{N}, X_{i_q}(n) = U_{i_q}(n)$. In this case, we have the following property : there exists n_{i_0}, k_{i_q} and λ_{i_q} such that for $n \geqslant n_{i_0}$, $X_{i_q}(n) = X_{i_q}(n - k_{i_q}) + \lambda_{i_q} k_{i_q}$.

Definition 5.1. *The sequence of expanded firing times δ_{i_q} of a transition q in $\mathcal{G}_i$ are defined as follows :*
If $q \in \mathcal{R}_i$, then

$$\delta_{i_q}(x) = \inf\left\{u \ : \ \int_x^{x+u} \mathbf{1}_{\{S_j(t)=0, j \succ i\}} dt \geqslant \sigma_{i_q}\right\}. \tag{19}$$

If $q \notin \mathcal{R}_i$, then for all x, $\delta_{i_q}(x) = \sigma_{i_q}$.

This definition implies that $(\mathcal{G}_i, \sigma_i)$ have the same dynamical behaviour as $(\mathcal{H}_i, \delta_i)$ in the following sense :

$$X_{i_q}(n) = \max_{r \in \bullet q} X_{i_r}(n - M_i(r,q)) + \delta_{i_r}(X_{i_r}(n - M_i(r,q))). \tag{20}$$

This equation looks like a classical (max,plus) equation describing the dynamic of a marked graph, the only difference here being that the firing times depend on the current state.

5.3 Qualitative Analysis for $N = 2$

In the following, we will keep the assumption that time is slotted. The slot duration will be the time unit and all durations will be multiples of this unit.

We first study the case where $N = 2$, $\mathcal{G}_1 \succ \mathcal{G}_2$ and $\mathcal{P}_1$ and $\mathcal{R}_2$ are not empty.

Therefore, for n large enough and for all transitions $q \in \mathcal{Q}_1$, $X_{1_q}(n) = X_{1_q}(n - k_{1_q}) + \lambda_{1_q} k_{1_q}$.

Let $T \overset{\text{def}}{=} \text{lcm}_{q \in \mathcal{Q}_1}(\lambda_{1_q} k_{1_q})$. The activity $S_1(t)$ is a function which becomes eventually periodic with period T after a transient period of $n_1 \overset{\text{def}}{=} \max_q X_{1_q}(t_0)$ units of time.

Lemma 5.2. *The behaviour of transition q in $\mathcal{G}_2$ is either finite (q fires a finite number of times) or pseudo-periodic with period $p_q k_{2_q}$. In the latter case, there exists integers a and p_q and μ such that $\forall q \in \mathcal{G}_i$, $\forall n \geqslant a$,*

$$X_{2_q}(n + p_q k_{2_q}) = X_{2_q}(n) + \mu T.$$

Proof. If transition q fires an infinite number of times, then the preempting activity is not always equal to 1 in its period. This implies that the mapping $x \to \delta_{2_q}(x)$ is bounded.

Now, let $k_2 \overset{\text{def}}{=} \text{lcm}_{q \in \mathcal{Q}_2} k_{2_q}$. We consider the marking $M(nk_2T)$ in all places at time nk_2T, $n > \max(n_{2_0}, n_{1_0})$, as well as the residual firing time vector at time nk_2T, denoted $R(nk_2T)$. Since $\delta_{2_q}(x)$ is bounded and integer valued for integer x, and since all the involved quantities are integer numbers, then there exist integers $a > b > \max(n_{0_2}, n_{0_1})$ such that $M(ak_2T) = M(bk_2T)$ and $R(ak_2T) = R(bk_2T)$.

The whole process is such that the preemption as well as the initial condition are the same at times ak_2T and bk_2T. Therefore, the system evolves periodically. This implies that for each transition q $X_{2_q}(n+p_q k_{2_q}) = X_{2_q}(n) + \mu T$, for all $n > ak_2T$, where $p_q k_{2_q}$ is the number of firings of q between times ak_2T and bk_2T, and $\mu = bk_2 - ak_2$. □

The following technical lemma (which proof is straightforward) will be useful in the proof of Theorem 5.4, which is the main result of this section.

Lemma 5.3. *Let Φ be a component-wise non-decreasing function from $\mathbb{R}_+^Q \to \mathbb{R}_+^Q$, Then, if $\gamma \overset{\text{def}}{=} \lim_{n\to\infty} \Phi^{(n)}(x_0)/n$ exists and $\gamma > 0$, then, for all $x \geqslant x_0$, $\lim_{n\to\infty} \Phi^{(n)}(x)/n = \gamma$.*

Theorem 5.4. *There exists a constant γ_q such that $\lim_{n\to\infty} X_{2_q}(n)/n = \gamma_q$. Moreover, this constant does not depend on the initial conditions, $X_1(0)$ and $X_2(0)$.*

Proof. The sequence $X_{2_q}(n)$ is pseudo-periodic and non-decreasing. Therefore, the above limit γ exists and $\gamma = \frac{\mu T}{p_q k_{2_q}}$. Furthermore, the function δ_{2_r} given by the evolution equation (20) does not depend on n since the firing times of transitions in $\mathcal{H}_2$ are constant. We have the following evolution equation in this framework :

$$X_{2_q}(n) = \max_{r \in \bullet q} X_{2_r}(n - M_2(r,q)) + \delta_{2_r}(X_{2_r}(n - M_2(r,q))).$$

For all x, the function $x \to x + \delta_{2_r}(x)$ is non-decreasing in x, since the integral is taken over a non-negative function. Therefore, the function

$$\Phi \ : \ \mathbb{N}^Q \ \to \ \mathbb{N}^Q \quad \textit{where} \tag{21}$$

$$\Phi(X) \ = \ \left(\max_{r \in \bullet q} X_r + \delta_{2_r}(X_r) \right), \tag{22}$$

is component-wise non-decreasing. Also note that

$$X_{2_q}(n) = \Phi_q(X_{2_1}(n - M_2(1,q)), \cdots, X_{2_Q}(n - M_2(Q,q))).$$

By using lemma 5.3, we know that the limit of $X_{2_q}(n)/n$ does not depend on the initial value of $(X_1(0), X_2(0))$. □

Within the graph $\mathcal{G}_2$, we can distinguish the behaviour of the different strongly connected components.

Corollary 5.5. *If two transitions, q and r belong to the same strongly connected component in $\mathcal{G}_2$, then they have the same cycle time : $\gamma_q = \gamma_r$ and the same period : $p_q = p_r$.*

Proof. Suppose that $\gamma_q \neq \gamma_r$. Then we have $|X_{2_q}(n) - X_{2_r}(n)| \to \infty$. This means that the marking in a path between transitions q and r is unbounded, which is impossible in a strongly connected component. Moreover, if r and q are in the same component, then $k_{2_q} = k_{2_r}$. Therefore, the cycle period is equal for both of them, in which case we get : $p_q = p_r$. □

These results can be generalized to arbitrary N and priority assignment, as shown in [19]. Corollary 5.5 shows that it is useless to assign different priorities to transitions which belong to a same SCC, e.g. tasks with synchronisation relations running on the same processor. However an arbitrary priority assignment can be used to model more complex systems, e.g. for a set a tasks distributed on processors connected by a field-bus with prioritised messages.

6 Conclusion

In this paper we have explained the development of a new technique to analyse the quantitative temporal behaviour of a set of periodic tasks. We assume that the tasks are scheduled using preemption and fixed priorities, and that their deadline equals their period. The model also takes into account synchronisations between tasks enforcing precedence constraints. Under these assumptions, the set of tasks can be modelled by Timed Marked Graphs which have a linear model in the (max,plus) algebra.

Using this model we derived tests to check some temporal properties of the system such as periodicity, cycle time, response time and respect of deadlines, both for the transient regime and for the steady state regime. The method is quite general and is not limited to a particular scheduling policy like rate monotonic thus leaving freedom to choose the priority assignment according to, e.g., automatic control performance constraints.

The technique presented here is applied to the particular case of the ORCCAD environment which implements this particular model of tasks and synchronisation. In fact it is rather general and can be used in a more general framework of systems with preemption and fixed priority. However quantitative results about the system's activity can be computed only with a restrictive assumption on priority assignment, where tasks sharing a given synchronisation link must have the same priority. The analysis of more general real-time systems needs further investigation.

A Appendix : (max,plus) algebra and marked graphs

In this section we will list several properties of the so-called (max,plus) algebra. All these results can be found in [1], where they are presented in full details.

A.1 The (max,plus) semi-ring

$\mathbb{R}_{max}$ is the semi-ring $(\mathbb{R} \cup \{-\infty\}, \oplus, \otimes)$, where $\oplus$ stands for the max operation and $\otimes$ stands for the $+$ operator.

These operations are extended to vectorial operation in the canonical way, If A and B are matrices with appropriate dimensions on $\mathbb{R}_{max}$, then $C = A \oplus B$ is a matrix with $C_{ij} = A_{ij} \oplus B_{ij}$ and $D = A \otimes B$ is a matrix with $D_{ij} = \bigoplus_k A_{ik} \otimes B_{kj}$. For simplicity, $A \otimes A$ will be denoted $A^{\otimes 2}$ or A^2. When a is a scalar and A a matrix, in $\mathbb{R}_{max}$, then $a \otimes A$ is a matrix with each component equal to $a \otimes A_{ij}$.

A.2 Elements of spectral theory

Let A be an irreducible matrix over $\mathbb{R}_{max}$, then there exists a *coupling time* $n_0 \in \mathbb{N}$, a *cyclicity* $c \in \mathbb{N}$, a unique *eigenvalue* $\lambda \in \mathbb{R}$ and at least an *eigenvector* v such that

$$A \otimes v = \lambda \otimes v \; \forall n \geqslant n_0, A^{\otimes(n+c)} = \lambda^{\otimes c} \otimes A^{\otimes n}$$

If A is an irreducible matrix with a non-positive eigenvalue, then, the equation $X = A \otimes X + b$, where X is the unknown vector and b is a fixed vector, admits a unique finite solution $X = A^* \otimes B$, where A^* is the finite matrix :

$$A^* \overset{\text{def}}{=} \bigoplus_{i=0}^{\infty} A^{\otimes i} \tag{23}$$

A.3 Marked Graphs

To any marked graph $\mathcal{G}$ with an initial marking bounded by one, one can associate matrices, $A(k)$, $k \in \{0, 1\}$, of size $Q \times Q$, where the entry (i, j) in matrix $A(k)$ is σ_j, the delay or lag time of transition j, if there exists a place between transitions q_j and q_i with k initial tokens, and $-\infty$ otherwise.

Let $A(0)^* = \bigoplus_{i=0}^{\infty} A(0)^{\otimes i}$, and $A = A(0)^* \otimes A(1)$. Let $X_q(k)$ be the epoch when the k-th firing starts in transition q. Then if there are no input transitions, the Q-dimensional vectors satisfy the recurrence relation

$$X(n) = A \otimes X(n-1), \quad n \geqslant 1$$

If there is an input transition with arrival process u, where $u(n)$ gives the epoch of the nth release of a token by the input, then

$$X(n) = A \otimes X(n-1) \oplus B \otimes u(n),$$

where $B_i = 0$ if there is a place between the input and transition q and $-\infty$ otherwise.

By using the spectral theory with timed marked graphs, we get the following result.

Lemma A.1. *For all SCC $\mathcal{C}$ in isolation, there exists a cycle time $\lambda_{\mathcal{C}} \in \mathbb{R}_+$, a cyclicity $s_{\mathcal{C}} \in \mathbb{N}_+$ and a transient period $k_{\mathcal{C}} \in \mathbb{N}$, such that for all transitions $q \in \mathcal{C}$ and all $k \geqslant k_{\mathcal{C}}$,*

$$V_q(k + s_{\mathcal{C}}) = V_q(k) + s_{\mathcal{C}}\lambda_{\mathcal{C}}. \tag{24}$$

As for the whole system $\mathcal{G}$ (all SCC considered together), in the case with no input, we have the following result : we denote $\mathcal{C} \to \mathcal{C}'$ if the SCC $\mathcal{C}$ precedes the SCC $\mathcal{C}'$ for the topological ordering. If

$$\max\{\lambda_{\mathcal{C}} | \mathcal{C} \to \mathcal{C}'\} > \lambda_{\mathcal{C}'}, \tag{25}$$

then the SCC $\mathcal{C}'$ have the same cycle time as the preceding SCC achieving the maximum in Equation (25)*. If*

$$\max\{\lambda_{\mathcal{C}} | \mathcal{C} \to \mathcal{C}'\} < \lambda_{\mathcal{C}'}, \tag{26}$$

then the SCC $\mathcal{C}'$ (and hence the whole system) is said unstable (the marking in some places will grow to infinity).

A similar result holds for a marked graph with a pseudo periodic input u. In particular, if the inverse of the input rate is larger than or equal to the maximal cycle time of all SCCs in isolation, then the system is stable. Otherwise, it is unstable.

Acknowledgement

The authors would like to thank the associate editor and anonymous referees for their very thoughtful comments.

References

[1] F. Baccelli, G. Gohen, G. J. Olsder, and J-P. Quadrat, *Synchronization and Linearity*, Wiley, 1992.

[2] J. Mairesse, *Stabilité des systèmes à événements discrets stochastiques, approche algébrique*, Ph.D. thesis, École polytechnique, Palaiseau, France, 1995.

[3] B. Gaujal, *Parrallélisme et simulation des systèmes à événements discrets*, Ph.D. thesis, University of Nice, France, 1994, http://www.inria.fr/RRRT/TU-0288.html.

[4] K.W. Tindell, A. Burns, and A.J. Wellings, "An extensible approach for analyzing fixed priority hard real-time tasks," *Real Time Systems*, vol. 6, no. 2, pp. 133–152, 1994.

[5] Liu C.L. and Layland J.W., "Scheduling algorithms for multiprogramming in hard real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 40–61, Feb. 73.

[6] J.J. Borrelly, E. Coste-Manière, B. Espiau, K. Kapellos, R. Pissard, D. Simon, and N. Turro, "The ORCCAD architecture," *Int. Journal of Robotics Research*, vol. 18, no. 4, pp. 338–359, 1998, http://www.inrialpes.fr/iramr/pub/Orccad/orccad-eng.html.

[7] D. Simon, E. Castillo, and P. Freedman, "Design and analysis of synchronization for real-time closed-loop control in robotics," *IEEE Trans. on Control Systems Technology*, vol. 6, no. 4, pp. 445–461, july 1998.

[8] H.R. Simpson, "Multireader and multiwriter asynchronous communication mechanisms," *IEE Proceedings-Computer and Digital Techniques*, vol. 144, no. 4, pp. 241–244, 1997.

[9] J.M. Migge, *Real-Time Scheduling: a trajectory based model*, Ph.D. thesis, University of Nice, France, 1999, http://www-sop.inria.fr/mistral/personnel/Jorn.Migge/.

[10] K.-E. Årzén, A. Cervin, J. Eker, and L. Sha, "An introduction to control and scheduling co-design," in *39th IEEE Conference on Decision and Control*, Sydney, Australia, december 2000.

[11] N.C. Audsley, A. Burns, R.I Davis, K.W Tindell, and A.J. Wellings, "Fixed priority preemptive scheduling: An historical perspective," *Real-Time Systems*, vol. 8, pp. 173–198, 1995.

[12] J. Blazewicz, "Scheduling dependent tasks with different arrival time to meet deadlines," in *Modelling Performance Evaluation Computer Systems*, E. Gelenbe, Ed., pp. 57–65. 1976.

[13] C.S. Chang, *Deterministic and stochastic guarantees in communication networks*, Springer, 1998.

[14] R. Alur and T. Henzinger, "Real-time: The theory in practice," in *REX Workshop.* 1991, LNCS 600.

[15] A. Bouajjani, R. Echahed, and J. Sifakis, "On model checking for real time properties with durations," in *8th Symposium on Logic in Computer Science (LICS 93)*, 1993.

[16] J.S. Ostroff, "A visual toolset for the design of real-time discrete-event systems," *IEEE Trans. on Control Systems Technology*, vol. 5, no. 3, pp. 320–337, 1997.

[17] X. Nicollin, J. Sifakis, and S. Yovine, "From atp to timed graphs in hybrid systems," in *REX Workshop.* 1991, LNCS 600.

[18] "Ers: Étude de réseaux stochastiques," ERS is a free software, available at http://www.inria.fr/Information/logiciels-eng.html.

[19] F. Baccelli, B. Gaujal, and D. Simon, "Analysis of preemptive periodic real time systems using the (max,plus) algebra," Tech. Rep. 3778, INRIA, 1999, http://www.inria.fr/rrrt/rr-3778.html.