Analysis of Preemptive Periodic Real Time Systems using the (max,plus) Algebra with Applications in Robotics

François Baccelli, Bruno Gaujal, Daniel Simon

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

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

With applications in robotics

François Baccelli , Bruno Gaujal , Daniel Simon

N°3778

October 15, 1999

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--3778--FR+ENG

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

With applications in robotics

François Baccelli* , Bruno Gaujal† , Daniel Simon‡

Thème 1 — Réseaux et systèmes
Projet TRIO

Rapport de recherche n ° 3778 — October 15, 1999 — 37 pages

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 controler, 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 synchronization constraints and is not limited to a particular scheduling policy.

Key-words: Periodic real time systems, synchronization, fixed priority preemption, marked graphs, (max,plus) algebra.

(Résumé : tsvp)

* ENS, 45 rue d'Ulm, Paris, Francois.Baccelli@ens.fr
† Loria, 615 rue du jardin Botanique, Villers lès Nancy, Bruno.Gaujal@loria.fr
‡ Inria Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot, Daniel.Simon@inrialpes.fr

Unité de recherche INRIA Lorraine
Technopôle de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY (France)
Téléphone : 03 83 59 30 30 - International : +33 3 3 83 59 30 30
Télécopie : 03 83 27 83 19 - International : +33 3 83 27 83 19
Antenne de Metz, technopôle de Metz 2000, 4 rue Marconi, 55070 METZ
Téléphone : 03 87 20 35 00 - Internatională: +33 3 87 20 35 00
Télécopie : 03 87 76 39 77 - International : +33 3 87 76 39 77

Analyse de systèmes temps-réels préemptifs par l'algèbre (max,plus) et applications à la robotique

Résumé : Dans cet article, nous présentons sous forme d'un Réseaux de Petri le modèle d' un système de tâches périodiques temps-réel, avec des priorités fixes, de la préemption et de la synchronisation qui sont exécutées par le controleur d'un robot. Ensuite, avec l'aide de l'algèbre (max,plus), nous établissons des tests simples pour vérifier des contraintes temps-réels sur ces tâches, comme le calcul des temps de réponse et le respect d'échéances. Cette méthode prend en compte les contraintes de précédence et de synchronisation et n'est pas limitée à une politique d' ordonnancement particulière.

Mots-clé : Systèmes temps-réel, synchronisation, préemption à priorité fixe, graphes d'événements , algèbre (max,plus)

1 Introduction

Marked graphs can be used to model processes with highly synchronous behaviors, see [1, 5, 3] for example. Here, we focus on the study of several marked graphs which interact via certain preemption schemes.

Such systems appear in the modeling of sets of tasks performed by on-board processes in a robot. Some tasks have high priority and therefore must preempt the low priority tasks. However, all tasks, those with high priority but also those with low priority have real time constraints to meet, which are of the following type. Each task is synchronized by a clock (or by a precedent task), and is made runnable at each tick of the clock. Each task must run to completion before the next synchronization tick. The ORCCAD system, detailed in Section 2 is a good example of such a systems.

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

- *(max,plus) representation.* This modeling problem can be approached under two different points of view: the *contracted time* approach is more suitable for the special case of the ORCCAD model and is presented in Section 3; the *expanded time* approach allows us to model more general problems but yields less precise answers. It is presented at the end of the paper 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 behavior.

- *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 period as well as during the periodic regime (Section 4.4).

In the last part of the paper, we show that this approach can be partially generalized 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 being the fact that the system reaches a periodic regime which does not depend on the initial conditions (see Section 5).

2 Modeling of a Real Time System: The ORCCAD Framework

This section is primarily a motivation section. We take the instance of a specific framework for real time systems, ORCCAD, and show how to model its logical and timed behavior 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 advanced robotics control systems. It also allows the specification and the validation of missions to be achieved by the system[1].

Periodic and multi-rate communicating tasks are executed under the control of a classical real-time operating system, using preemption based on fixed priorities assigned to tasks.

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

2.1 Synchronization

The general idea is that a partial synchronization of such tasks can improve the performances by decreasing the computing latency. However, using too many or incorrectly specified synchronizations may lead to deadlocks or temporal inconsistencies [8]. Several kinds of synchronizations can be used on ports in order to synchronize more or less tightly the set of MTs:

- ASYN-ASYN: a communication of this type does not lead to further synchronization.

[1] A freeware simplified prototype of the software and associated documentation are available through `http://www.inrialpes.fr/iramr/pub/Orccad/orccad-eng.html`.

- 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.
- ASYN-SYN: the writer runs freely and posts messages on its output ports at each period; the reader either reads the message if a new one is available or is blocked until the next message is posted.
- SYN-ASYN: symmetrical to the previous case: the reader runs freely, the writer is blocked until the next request except if a new one was posted since the last reading.

2.2 Preemptions

In addition to synchronizations, 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 has small computation duration. It can be activated with a fast period, thus improving the performance of the control law, e.g. reducing tracking errors or increasing robustness w.r.t. modeling errors [8]. 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. Such an example is given in section 2.6.

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 one MT with high priority becomes runnable and starts its calculation cycle, all MTs with lower priorities are preempted on the processor. The activity of the runnable MT with the immediately lower priority resumes where it was stopped as soon as the higher priority MT has finished its cycle of calculations.

2.3 Modeling with Petri nets

Design inconsistencies may arise in several ways. *Structural* deadlocks are due to the synchronization structure itself whatever are the numerical values of temporal attributes. In addition, preemption or badly chosen numerical values of temporal

attributes like tasks period and duration may lead to *temporal inconsistencies* such as non-periodic behavior of the system, even if it is free from structural deadlocks.

Therefore, we need modeling and analysis tools to automatically check for inconsistencies in the network of synchronized MTs. Such problems have been addressed in the real-time community under very general assumptions, see for example [7]. Here, we will adopt a particular modeling tool, Petri nets which will provide a simple and efficient way to carry out inconsistency tests.

As shown in Figure 1, the sequential behavior of the simplest periodic MT (reading an input port, performing a calculation, writing to an output port) may be modeled by a Petri net with three transitions. (Of course, when a MT has multiple input and output ports, we must be careful to associate a distinct transition with each read and each write.) A fourth transition is required to activate the MT subject to the periodic awakening associated with a real time clock (RTC), also modeled by a Petri net. As we shall be further concerned with temporal analysis, we may add some temporal properties to the model to obtain a *timed Petri net*, i.e. a Petri net where durations are associated with some transitions or places. We have chosen to associate the duration [d] of the MT with the calculation step transition, and thereby assume that reading and writing are instantaneous events, i.e. events of zero duration. A crossing time [τ] is also assigned to the transition associated with the RTC (Transitions associated with non zero duration are drawn with thick lines).

Since each place has just one input transition and one output transition, the resulting Petri net is a so-called *marked graph* (or event graph).

ASYN/SYN communication between two MTs is modeled as shown in Figure 2 on the left and SYN/SYN communication is modeled as shown in Figure 2 on the right. Note that the transition associated with the periodic awakening of MT2 is no longer present, since the temporal behavior of MT2 is bound to that of MT1. Once again, the combination of the two Petri nets is a marked graph.

Note that ASYN/ASYN communication does not add synchronization constraints, i.e. MTs communicating using this protocol have disconnected models.

Also, preemption is not shown directly in the Petri net model. It will be taken into account in the equations that describe the dynamics of the system.

Thanks to structural properties of marked graphs, checking for structural deadlocks can be easily done through the analysis of the initial marking of the p-invariants of the global Petri net [8]. Studying the temporal behavior of the set of MTs is more 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, such models usually do not take into account the effect of the real-time