Scheduling of Embedded Controllers Under Timing Contracts

Mohammad Al Khatib, Antoine Girard, Thao Dang

Scheduling of Embedded Controllers Under Timing Contracts *

Mohammad Al Khatib
L2S, CNRS
CentraleSupélec
Université Paris-Sud
Université Paris-Saclay
F-91192 Gif-sur-Yvette
mohammad.alkhatib
@l2s.centralesupelec.fr

Antoine Girard
L2S, CNRS
CentraleSupélec
Université Paris-Sud
Université Paris-Saclay
F-91192 Gif-sur-Yvette
antoine.girard
@l2s.centralesupelec.fr

Thao Dang
Univ. Grenoble Alpes-CNRS
Verimag
F-38000 Grenoble
thao.dang@imag.fr

ABSTRACT

Timing contracts for embedded controller implementation specify the constraints on the time instants at which certain operations are performed such as sampling, actuation, computation, etc. Several previous works have focused on stability analysis of embedded control systems under such timing contracts. In this paper, we consider the scheduling of embedded controllers on a shared computational platform. Given a set of controllers, each of which is subject to a timing contract, we synthesize a dynamic scheduling policy, which guarantees that each timing contract is satisfied and that the shared computational resource is allocated to at most one embedded controller at any time. The approach is based on a timed game formulation whose solution provides a suitable scheduling policy. In the second part of the paper, we consider the problem of synthesizing a set of timing contracts that guarantee at the same time the schedulability and the stability of the embedded controllers.

Keywords

Embedded control; Sampled-data systems; Scheduling; Timed automata; Stability

1. INTRODUCTION

Cyber-physical systems (CPS) consisting of integration of computing devices with physical processes are to become ubiquitous in modern societies (autonomous vehicles, smart buildings, robots, etc.) and will practically impact the life of citizens in all their aspects (housing, transportation, health, industry, assistance to the elderly, etc.). Therefore, high-confidence tools for the analysis and design of CPS, being able to cope with the tight interactions between cyber and physical components are urgently needed.

Design of complex CPS can be tackled by decomposing the global design problem into smaller sub-problems. This approach can be formally implemented using *contract based design* [32, 8, 7]. For instance, for embedded controller implementation, [17] proposed the use of timing contracts, which specify the constraints on the time instants at which certain operations are performed such as sampling, actuation or computation. Under such contracts, the control engineers are responsible for designing a control law that is robust to all possible timing variation specified in the contract while the software engineers can focus on implementing the proposed control law so as to satisfy the timing contract. Several previous works have taken the point of view of control engineers, by developing techniques for robust stability analysis of embedded control systems under such timing contracts (see e.g. [13, 18, 5, 2]).

In the first part of this paper, we adopt the point of view of the software engineer who has to implement several controllers, each subject to a timing contract, on a shared computational platform. Given best and worst case execution times for each control task, we synthesize a dynamic scheduling policy, which guarantees that each timing contract is satisfied and that the shared computational resource is allocated to at most one controller at any time. Our approach is based on the use of timed game automata [30, 11] and we show that the scheduling problem can be formulated as a timed safety game, which can be solved by the tool UPPAAL-TIGA [6], and whose solution provides a suitable scheduling policy.

In the second part of the paper, we address the requirement engineering problem, which consists in synthesizing a set of timing contracts that guarantee at the same time the schedulability and the stability of the embedded controllers. We use a re-parametrization of contracts, which provides some monotonicity property to the problem and allows us to develop an effective synthesis method based on guided sampling of the timing contract parameter space.

The paper is organized as follows. The problem setting is given in Section 2, where we introduce the considered classes of systems and timing contracts and where we formulate the stability verification, schedulability verification and timing contract synthesis problems. Section 3 provides a solution

*This work was supported by the Agence Nationale de la Recherche (COMPACS project ANR-13-BS03-0004) and by the Labex DigiCosme, Université Paris-Saclay (CODECSYS project).

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

to the schedulability verification problem based on timed games. Then, the timing contract synthesis problem is addressed in Section 4. Section 5 shows an application of our approach using an illustrative example.

Notations. Let $\mathbb{R}$, $\mathbb{R}_0^+$, $\mathbb{R}^+$, $\mathbb{R}_0^-$, $\mathbb{R}^-$, $\mathbb{N}$, $\mathbb{N}^+$ denote the sets of reals, non-negative reals, positive reals, non-positive reals, negative reals, non-negative integers and positive integers, respectively. For $I \subseteq \mathbb{R}_0^+$, let $\mathbb{N}_I = \mathbb{N} \cap I$. Given a vector $p \in \mathbb{R}^n$, its i-th element is denoted by p_i. Given two vectors $p, p' \in \mathbb{R}^n$, the inequality $p \leq p'$ is interpreted conponentwise. For a set S, we denote the set of all subsets of S by 2^S.

2. PROBLEM FORMULATION

2.1 Sampled-data systems

In this work, we consider sampled-data systems that take into account the sequences of sampling and actuation instants $(t_k^s)_{k\in\mathbb{N}}$ and $(t_k^a)_{k\in\mathbb{N}}$:

$$\dot{x}(t) = Ax(t) + Bu(t), \quad \forall t \in \mathbb{R}_0^+ \tag{1}$$

$$u(t) = Kx(t_k^s), \qquad t_k^a < t \leq t_{k+1}^a \tag{2}$$

where $x(t) \in \mathbb{R}^n$ is the state of the system, $u(t) \in \mathbb{R}^m$ is the control input, the matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $K \in \mathbb{R}^{m\times n}$ and $k \in \mathbb{N}$. In addition, it is assumed that for all $t \in [0, t_0^a]$, $u(t) = 0$.

We assume that the sequence of sampling and actuation instants (t_k^s) and (t_k^a) satisfy a *timing contract* $\theta(\underline{\tau}, \overline{\tau}, \underline{h}, \overline{h})$ given by

$$\begin{array}{ll} 0 \leq t_0^s, & \\ t_k^s \leq t_k^a \leq t_{k+1}^s, & \forall k \in \mathbb{N} \\ \tau_k = t_k^a - t_k^s \in [\underline{\tau}, \overline{\tau}], & \forall k \in \mathbb{N} \\ h_k = t_{k+1}^s - t_k^s \in [\underline{h}, \overline{h}], & \forall k \in \mathbb{N} \end{array} \tag{3}$$

where $\underline{\tau} \in \mathbb{R}_0^+$, $\overline{\tau} \in \mathbb{R}_0^+$, $\underline{h} \in \mathbb{R}^+$ and $\overline{h} \in \mathbb{R}^+$ provide bounds on the sampling-to-actuation delays (which includes time for computation of the control law) and sampling periods provided that $t_k^s \leq t_k^a \leq t_{k+1}^s$ for all $k \in \mathbb{N}$. Note that we impose $\underline{h} \neq 0$ to prevent Zeno behavior. Moreover, these parameters must belong to the following set so that the time intervals given in (3) are always non-empty and it is always possible to choose $t_{k+1}^s \geq t_k^a$:

$$\mathcal{C} = \left\{(\underline{\tau}, \overline{\tau}, \underline{h}, \overline{h}) \in \mathbb{R}_0^+ \times \mathbb{R}_0^+ \times \mathbb{R}^+ \times \mathbb{R}^+ : \underline{\tau} \leq \overline{\tau} \leq \overline{h}, \underline{h} \leq \overline{h}\right\}.$$

2.2 Stability verification

We consider the notion of stability in the following sense:

Definition 1. The system $\mathcal{S} = (A, B, K)$ is *globally uniformly exponentially stable* (GUES) under timing contract $\theta(\underline{\tau}, \overline{\tau}, \underline{h}, \overline{h})$ if there exist $\lambda \in \mathbb{R}^+$ and $C \in \mathbb{R}^+$ such that, for all sequences $(t_k^s)_{k\in\mathbb{N}}$ and $(t_k^a)_{k\in\mathbb{N}}$ verifying (3), the solutions of (1-2) verify

$$\|x(t)\| \leq Ce^{-\lambda(t-t_0^s)} \|x(t_0^s)\|, \ \forall t \geq t_0^s.$$

We then define the stability verification problem as follows:

PROBLEM 1 (STABILITY VERIFICATION). *Given a system $\mathcal{S} = (A, B, K)$ and a timing contract $\theta(\underline{\tau}, \overline{\tau}, \underline{h}, \overline{h})$ verify that $\mathcal{S}$ is GUES under timing contract $\theta(\underline{\tau}, \overline{\tau}, \underline{h}, \overline{h})$.*

Several approaches are presented in the literature to solve instances of Problem 1. A non-exhaustive list is given in Table 1. From the modeling perspective, the problem can be tackled using difference inclusions, time-delay systems or hybrid systems. On the computational side, the approaches are based on semi-definite programming (Linear Matrix Inequalities (LMI) or Sum Of Squares (SOS) formulations), invariant sets or reachability analysis. Let us remark that approaches [13, 18, 5, 2] appear to be able to address all instances of Problem 1.

2.3 Scheduling

We consider a collection of $N \in \mathbb{N}^+$ sampled-data systems $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$ of the form (1-2) where each system $\mathcal{S}_i = (A_i, B_i, K_i)$, is subject to a timing contract $\theta(\underline{\tau}^i, \overline{\tau}^i, \underline{h}^i, \overline{h}^i)$ of the form (3), with parameters $(\underline{\tau}^i, \overline{\tau}^i, \underline{h}^i, \overline{h}^i) \in \mathcal{C}$, $i \in \mathbb{N}_{[1,N]}$.

In addition, we assume that these systems share a single processor to compute the value of their control inputs given by (2). The time required to compute these inputs is assumed to belong to some known interval $[\underline{c}^i, \overline{c}^i]$, with $0 \leq \underline{c}^i \leq \overline{c}^i$, where $\underline{c}^i$ and $\overline{c}^i$ denote the best and worst case execution time to compute the input of systems $\mathcal{S}_i$, $i \in \mathbb{N}_{[1,N]}$.

The timing of events in the k-th control cycle of system $\mathcal{S}_i$ starts at instant $t_k^{s_i}$ when sampling occurs. Then, system $\mathcal{S}_i$ gains access to the computational resource at instant $t_k^{b_i}$, at which computation of the control input value begins. The computational resource is released at instant $t_k^{e_i}$, at which computation of the control input value ends. After that, actuation occurs at instant $t_k^{a_i}$. Formally, the sequences $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, $(t_k^{e_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfy the following constraints for all $i \in \mathbb{N}_{[1,N]}$:

$$\begin{array}{ll} 0 \leq t_0^{s_i} & \\ t_k^{s_i} \leq t_k^{b_i} \leq t_k^{e_i} \leq t_k^{a_i} \leq t_{k+1}^{s_i}, & \forall k \in \mathbb{N} \\ c_k^i = t_k^{e_i} - t_k^{b_i} \in [\underline{c}^i, \overline{c}^i], & \forall k \in \mathbb{N} \\ \tau_k^i = t_k^{a_i} - t_k^{s_i} \in [\underline{\tau}^i, \overline{\tau}^i], & \forall k \in \mathbb{N} \\ h_k^i = t_{k+1}^{s_i} - t_k^{s_i} \in [\underline{h}^i, \overline{h}^i], & \forall k \in \mathbb{N} \end{array} \tag{4}$$

In addition, a conflict arises if several systems request access to the computational resource at the same time. Let us define the following property, for $i \in \mathbb{N}_{[1,N]}$:

$$\mathsf{Com}(\mathcal{S}_i, t) \equiv \bigvee_{k\in\mathbb{N}} \left(t \in [t_k^{b_i}, t_k^{e_i})\right).$$

$\mathsf{Com}(\mathcal{S}_i, t)$ is true if and only if the computational resource is used by system $\mathcal{S}_i$ at time t. Then, in order to prevent conflicting accesses to the computational resource the following property must hold:

$$\begin{array}{c} \forall t \in \mathbb{R}_0^+, \forall (i,j) \in \mathbb{N}_{[1,N]}^2 \text{ with } i \neq j, \\ \mathsf{Com}(\mathcal{S}_i, t) \wedge \mathsf{Com}(\mathcal{S}_j, t) \equiv \mathsf{False}. \end{array} \tag{5}$$

REMARK 1. *It is straightforward to verify that for any sequences $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, $(t_k^{e_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfying (4-5), the sequences $(t_k^{s_i})_{k\in\mathbb{N}}$ and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfy the timing contract $\theta(\underline{\tau}^i, \overline{\tau}^i, \underline{h}^i, \overline{h}^i)$.*

We aim at synthesizing a dynamic scheduling policy, generating sequences of timing events satisfying (4-5). The scheduler has control over the sampling and actuation instants $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{a_i})_{k\in\mathbb{N}}$ and over the instants $(t_k^{b_i})_{k\in\mathbb{N}}$ when computation begins. However, the execution time $(c_k^i)_{k\in\mathbb{N}}$

Table 1: Methods that can solve instances of Problem 1 with description of the modeling, computational approaches, and list of restrictions.

	Models	Algorithm	Restrictions
[13]	difference inclusion	LMI	–
[18]		LMI	–
[24]		LMI	$\underline{\tau} = \overline{\tau} = 0$
[25]		LMI	$\underline{\tau} = \overline{\tau} = 0$
[33]		SOS	$\underline{\tau} = \overline{\tau} = 0$
[20]		Invariant sets	$\underline{\tau} = \overline{\tau} = 0$
[1]		Reachability analysis	$\underline{\tau} = \overline{\tau} = 0$
[2]		Reachability analysis	–
[28]	time-delay systems	LMI	$\underline{h} = 0$
[22]		LMI	$\underline{h} = \overline{h}$, $\underline{\tau} = 0$
[29]		LMI	$\underline{\tau} = \overline{\tau} = 0$
[21]		LMI	$\underline{h} = \underline{\tau} = \overline{\tau} = 0$
[5]	hybrid systems	SOS	–
[23]		LMI	$\underline{\tau} = 0$, $\underline{h} = 0$

and thus the instants when computation ends $(t_k^{e_i})_{k\in\mathbb{N}}$ is determined by the environment and are thus uncontrollable from the point of view of the scheduler. Then, we consider the following schedulability property:

Definition 2. The set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ is *schedulable* under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$, if for all sequences $(c_k^i)_{k\in\mathbb{N}}$ with $c_k^i \in [\underline{c}^i, \overline{c}^i]$, for all $k \in \mathbb{N}$ and $i \in \mathbb{N}_{[1,N]}$, there exist sequences $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, $(t_k^{e_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfying (4-5), for all $i \in \mathbb{N}_{[1,N]}$.

Then, we define the scheduling problem as follows:

Problem 2 (Schedulability verification). *Given a set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ and timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$, verify that $\mathcal{T}$ is schedulable under timing contracts Θ.*

In Section 3, we will provide a solution to the schedulability verification problem. In addition, our approach will allow us to synthesize a dynamic scheduling policy for generating the sequences $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, $(t_k^{e_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfying (4-5), for all $i \in \mathbb{N}_{[1,N]}$.

Related work. In real-time scheduling of multiple tasks on a single processor, the objective is to obtain a calculation model related to concurrent execution of a given number of tasks [12, 14, 10]. In case some of the tasks are control tasks, scheduling and controller co-design problems are studied for periodic [4, 31], event-triggered [34], and self-triggered [15, 26] real-time implementations of the controller. Unlike these approaches, we decouple the controller stability problem and the scheduling design, using timing contracts, in order to synthesize conflict-free schedules using timed game automata. In fact, scheduling with timed automata are examined in literature [16, 19, 26] where in [19] an application on a steel plant is studied.

The closest approach to our work is that depicted in [26] were the scheduling problem is reformulated in terms of timed game automata [11]. Although, therein the approach has the advantage of employing event-triggered controllers and eventually improves the resource utilization over the network, however it can only solve instances of our problem where $t_k^{s_i} = t_k^{b_i}$, $t_k^{e_i} = t_k^{a_i}$ and $\underline{\tau} = \overline{\tau} = \overline{c}^i = \underline{c}^i = c \in \mathbb{R}^+$ for all $k \in \mathbb{N}$ and $i \in \mathbb{N}_{[1,N]}$.

2.4 Timing contract synthesis

In Section 4, we will consider the problem of synthesizing a set of timing contracts that guarantee at the same time the stability of the systems and the schedulability of control tasks.

Given the bounds on the parameters $0 \leq \tau_{min}^i \leq \tau_{max}^i$, $0 < h_{min}^i \leq h_{max}^i$, with $\tau_{min}^i \leq h_{min}^i$, $\tau_{max}^i \leq h_{max}^i$, let

$$\mathcal{D}_i = [\tau_{min}^i, \tau_{max}^i]^2 \times [h_{min}^i, h_{max}^i]^2, \quad i \in \mathbb{N}_{[1,N]}, \qquad (6)$$

with $N \in \mathbb{N}^+$, the timing contract synthesis problem is formalized as follows:

Problem 3 (Timing contract synthesis). *Given a collection of systems $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$, where $\mathcal{S}_i = (A_i, B_i, K_i)$ with $A_i \in \mathbb{R}^{n_i \times n_i}$, $B_i \in \mathbb{R}^{n_i \times m_i}$, and $K_i \in \mathbb{R}^{m_i \times n_i}$, $i \in \mathbb{N}_{[1,N]}$, a set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ with $0 \leq \underline{c}^i \leq \overline{c}^i$, $i \in \mathbb{N}_{[1,N]}$, and parameter sets $\mathcal{D}_i$, $i \in \mathbb{N}_{[1,N]}$, synthesize a set $\mathcal{P}^* \subseteq (\mathcal{C}^N) \cap (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N)$ such that for all $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N) \in \mathcal{P}^*$:*

1. *System $\mathcal{S}_i = (A_i, B_i, K_i)$ is GUES under timing contract $\theta(\underline{\tau}_i, \overline{\tau}_i, \underline{h}_i, \overline{h}_i)$, for all $i \in \mathbb{N}_{[1,N]}$.*
2. *The set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ is schedulable under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$.*

3. SCHEDULING

In this section, we provide a solution to Problem 2. Our approach is based on timed automata [3] and timed game automata [30], which we briefly introduce in the following.

3.1 Timed and timed game automata

Let C be a finite set of real-valued variables called clocks. We denote by $\mathcal{B}(C)$ the set of conjunctions of clock constraints of the form $c \sim \alpha$ where $\alpha \in \mathbb{R}_0^+$, $c \in C$ and

$\sim \in \{<, \leq, =, >, \geq\}$. We define a timed automaton (TA) and a timed game automaton (TGA) as in [11]:

Definition 3. A *timed automaton* is a sextuple (L, l_0, Act, C, E, I) where

- L is a finite set of locations;
- $l_0 \in L$ is the initial location;
- Act is a set of actions;
- C is a finite set of real-valued clocks;
- $E \subseteq L \times \mathcal{B}(C) \times Act \times 2^C \times L$ is the set of edges;
- $Inv : L \to \mathcal{B}(C)$ is a function that assigns invariants to locations.

Definition 4. A *timed game automaton* is a septuple $(L, l_0, Act_c, Act_u, C, E, I)$ such that $(L, l_0, Act_c \cup Act_u, C, E, I)$ is a timed automaton and $Act_c \cap Act_u = \emptyset$, where Act_c defines a set of controllable actions and Act_u defines a set of uncontrollable actions.

Formal semantics of TA and TGA are given in [11]. Informally, semantics of a TA is described by a transition system whose state consists of the current location and value of the clocks. Then, the execution of a TA can be described by two types of transitions defined as follows:

- time progress: the current location $l \in L$ is maintained and the value of the clocks grow at unitary rate; these transitions are enabled as long as the value of the clocks satisfies $Inv(l)$.
- discrete transition: an instantaneous transition from the current location $l \in L$ to a new location $l' \in L$ labelled by an action $a \in Act$ is triggered; these transitions are enabled if there is an edge $(l, G, a, C', l') \in E$, such that the value of the clocks satisfies G; in that case, the value of the clocks belonging to C' resets to zero.

The semantics of TGA is similar to that of TA with the specificity that discrete transitions labelled by a controllable action (i.e. $a \in Act_c$) are triggered by a controller, while discrete transitions labelled by an uncontrollable action (i.e. $a \in Act_u$) are triggered by the environment/opponent.

In the following, we consider *safety games* (see e.g. [11]) defined by a set of unsafe locations $L_u \subseteq L$. A solution to the safety game is given by a winning strategy for the controller such that under any behavior of the environment/opponent, the set of unsafe locations is avoided by all executions of the TGA.

3.2 Scheduling using TGA

In this section, we propose a solution to Problem 2 based on a reformulation using timed game automata.

Definition 5. Let $i \in \mathbb{N}_{[1,N]}$, the timed game automaton generated by control task $(\underline{c}^i, \overline{c}^i)$ and timing contract $\theta(\underline{\tau}^i, \overline{\tau}^i, \underline{h}^i, \overline{h}^i)$ is displayed in Figure 1 and is formally defined by $\mathsf{TGA}_i = (L^i, l_0^i, Act_c^i, Act_u^i, C^i, E^i, Inv^i)$ where

- $L^i = \{Init^i, Presam^i, Precomp^i, Comp^i, Preac^i\}$;
- $l_0^i = Init^i$;
- $Act_c^i = \{sample^i, begin^i, actuate^i\}$;
- $Act_u^i = \{end^i, in^i\}$;
- $C^i = \{c^i, k^i\}$;
- $E^i = \{(Init^i, c^i \geq 0, in^i, \{c^i\}, Presam^i),$
  $(Presam^i, c^i \geq \underline{h}^i, sample^i, \{c^i\}, Precomp^i),$
  $(Precomp^i, c^i \geq 0, begin^i, \{k^i\}, Comp^i),$
  $(Comp^i, k^i \geq \underline{c}^i, end^i, \emptyset, Preac^i),$
  $(Preac^i, c^i \geq \underline{\tau}^i, actuate^i, \emptyset, Presam^i)\}$;
- $Inv^i(Init^i) = \{c^i \geq 0\},$
  $Inv^i(Presam^i) = \{c^i \leq \overline{h}^i\},$
  $Inv^i(Precomp^i) = \{c^i \leq \overline{\tau}^i - \overline{c}^i\},$
  $Inv^i(Comp^i) = \{k^i \leq \overline{c}^i\},$
  $Inv^i(Preac^i) = \{c^i \leq \overline{\tau}^i\}.$

Let the sequences $(t_k^{s_i})$, $(t_k^{a_i})$, $(t_k^{b_i})$ and $(t_k^{e_i})$ be given by the instants of the discrete transitions labelled by actions $sample^i$, $actuate^i$, $begin^i$ and end^i, respectively. It is easy to see that these sequences satisfy the constraints given by (4). Conversely, one can check that all sequences satisfying (4) can be generated by executions of TGA_i.

Moreover, let us remark that the controllable actions are $sample^i$, $actuate^i$, $begin^i$ which means that the controller determines the instants when sampling and actuation occur and when computation begins. However, end^i is uncontrollable, which means that the execution time, and thus the instant at which computation ends is determined by the environment. This is consistent with the problem formulation in Section 2.

Finally, the computational resource is used by system $\mathcal{S}_i$ if the current location of TGA_i is $Comp^i$. To take into account the constraint given by (5), stating that two systems cannot access the computational resource at the same time, we need to define the composition of the timed game automata defined above:

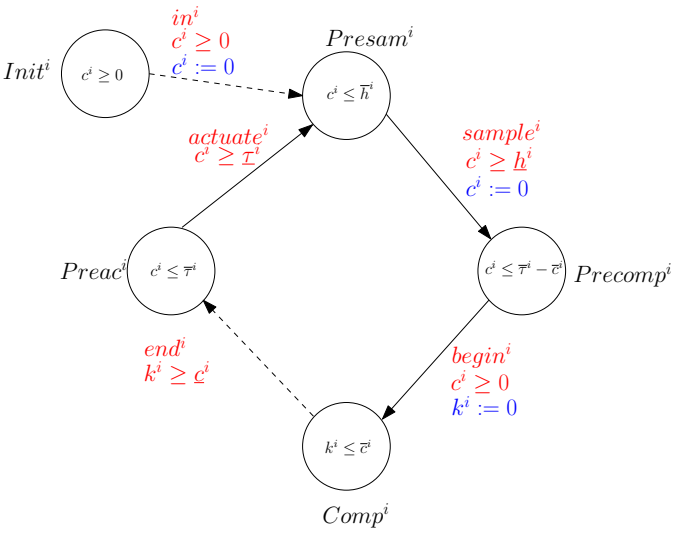

Figure 1: TGA^i where plain edges correspond to controllable actions while dashed edges correspond to uncontrollable actions.

Definition 6. The timed game automaton generated by the set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ and timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$ is given by $\mathsf{TGA} = (\overline{L}, \bar{l}_0, \overline{Act_c}, \overline{Act_u}, \overline{C}, \overline{E}, \overline{Inv})$ where

- $\overline{L} = L^1 \times \cdots \times L^N$, thus $l = (l^1, \ldots, l^N) \in L$ denotes the location of TGA;
- $\bar{l}_0 = (Init^1, \ldots, Init^N)$;
- $\overline{Act_c} = \bigcup_{i=1}^N Act_c^i$;
- $\overline{Act_u} = \bigcup_{i=1}^N Act_u^i$;
- $\overline{C} = \bigcup_{i=1}^N C^i$;
- $\overline{E} = \{(l_m, \lambda, act, C', l_n) \in \overline{L} \times \mathcal{B}(\overline{C}) \times (\overline{Act_c} \cup \overline{Act_u}) \times \overline{L} : \exists i \in \mathbb{N}_{[1,N]}, l_m^j = l_n^j \ \ \forall j \neq i \text{ and } (l_m^i, \lambda, act, C', l_n^i) \in E^i\}$;
- $Inv(l) = \bigwedge_{i=1}^N Inv^i(l^i)$, $i \in \mathbb{N}_{[1,N]}$.

TGA describes the parallel evolution of the $\mathsf{TGA}_1, \ldots, \mathsf{TGA}_N$. Then, the set of locations corresponding to conflicting accesses to the computational resources is $\overline{L}_u \subseteq \overline{L}$ defined by:

$$\begin{aligned}\overline{L}_u = \{l \in \overline{L} : \exists (i,j) \in \mathbb{N}^2_{[1,N]}, i \neq j, \\ (l^i = Comp^i) \wedge (l^j = Comp^j)\}.\end{aligned} \tag{7}$$

From the previous discussions, it follows that $\mathcal{T}$ is *schedulable* under timing contracts Θ if there is a winning strategy to the safety game defined by the timed game automaton TGA and the set of unsafe locations $\overline{L}_u$. From the practical point of view, the safety game can be solved using the tool UPPAAL-TIGA [6]. The tool synthesizes a winning strategy when it exists, which provides us with a dynamic scheduling policy for generating the sequences $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, $(t_k^{e_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$ satisfying (4-5), for all $i \in \mathbb{N}_{[1,N]}$.

4. TIMING CONTRACT SYNTHESIS

In this section, we propose a solution to Problem 3. Given a collection of systems $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$, a set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), \ldots, (\underline{c}^N, \overline{c}^N)\}$ with $0 \leq \underline{c}^i \leq \overline{c}^i$, $i \in \mathbb{N}_{[1,N]}$, and parameter sets $\mathcal{D}_1, \ldots, \mathcal{D}_N$, we use a previous result [2] and a monotonicity property to design an algorithm that synthesizes a set of timing contracts ensuring stability of each system $\mathcal{S}_i$ and schedulability of $\mathcal{T}$.

4.1 Guarantee on stability

For $i \in \mathbb{N}_{[1,N]}$, we consider each of the systems $\mathcal{S}_i = (A_i, B_i, K_i)$ and the set $\mathcal{D}_i$ given by (9), $i \in \mathbb{N}_{[1,N]}$. We use Algorithm 2 in [2] to synthesize a set $\mathcal{C}_i^* \subseteq \mathcal{C} \cap \mathcal{D}_i$ such that for all $(\underline{\tau}^i, \overline{\tau}^i, \underline{h}^i, \overline{h}^i) \in \mathcal{C}_i^*$, $\mathcal{S}_i$ is GUES under timing contract $\theta(\underline{\tau}_i, \overline{\tau}_i, \underline{h}_i, \overline{h}_i)$.

Then, defining the set $\mathcal{P}_{st} = \mathcal{C}_1^* \times \cdots \times \mathcal{C}_N^*$, it follows that for all $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N) \in \mathcal{P}_{st}$, system $\mathcal{S}_i = (A_i, B_i, K_i)$ is GUES under timing contract $\theta(\underline{\tau}_i, \overline{\tau}_i, \underline{h}_i, \overline{h}_i)$, for all $i \in \mathbb{N}_{[1,N]}$.

4.2 Guarantee on schedulability

In this section, given the set $\mathcal{P}_{st}$, defined in the previous section, we solve Problem 3 by synthesizing a set $\mathcal{P}^* \subseteq \mathcal{P}_{st}$ such that for all $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N) \in \mathcal{P}^*$, the set of control tasks $\mathcal{T}$ is schedulable under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$.

The timing contract parameters are given by the vector $p = (\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)$. We then define the following Boolean function for $p \in \mathcal{C}^N \cap (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N)$:

$\mathsf{Sched}(p) \equiv \mathcal{T}$ is *schedulable* under timing contracts Θ.

In order to solve Problem 3 we need to compute (a subset of) the set $\mathcal{P}_0$ defined by

$$\mathcal{P}_0 = \{p \in \mathcal{C}^N \cap (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N) : \mathsf{Sched}(p)\}.$$

Re-parametrization

We define a new parameter $p' \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N'$ with

$$\begin{aligned}\mathcal{D}_i' = \ & [\tau_{min}^i, \tau_{max}^i] \times [-\tau_{max}^i, -\tau_{min}^i] \times [h_{min}^i, h_{max}^i] \times \\ & [-h_{max}^i, -h_{min}^i], \ \ i \in \mathbb{N}_{[1,N]},\end{aligned} \tag{8}$$

such that $p' = (\beta_1^1, \beta_2^1, \beta_3^1, \beta_4^1, \ldots, \beta_1^N, \beta_2^N, \beta_3^N, \beta_4^N)$. We further define the map $f : (\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') \to (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N)$ such that $f(p') = p = (\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)$, where for all $i \in \mathbb{N}_{[1,N]}$

$$\underline{\tau}^i = \beta_1^i, \ \overline{\tau}^i = \min(-\beta_2^i, -\beta_4^i), \ \underline{h}^i = \beta_3^i, \ \overline{h}^i = -\beta_4^i.$$

We associate to the parameter p' a constraint set $(\mathcal{C}')^N$ where

$$\mathcal{C}' = \left\{\beta \in \mathbb{R}_0^+ \times \mathbb{R}_0^- \times \mathbb{R}^+ \times \mathbb{R}^- : \begin{array}{l}\beta_1 \leq \min(-\beta_2, -\beta_4) \\ \beta_3 \leq -\beta_4\end{array}\right\}.$$

Last we define the set $\mathcal{P}_o'$ by :

$$\mathcal{P}_o' = \left\{p' \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N' : ((p' \in (\mathcal{C}')^N) \wedge \mathsf{Sched}(f(p')))\right\}.$$

One can check that the following relation holds:

$$\mathcal{P}_o' = f(\mathcal{P}_o). \tag{9}$$

We can then show that $\mathcal{P}_o'$ satisfies the following monotonicity property:

PROPOSITION 1. *For all $p_1', p_2' \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N'$, the following implications hold:*

$$\big((p_2' \leq p_1') \wedge (p_1' \in \mathcal{P}_o')\big) \implies p_2' \in \mathcal{P}_o'.$$

$$\big((p_2' \leq p_1') \wedge (p_2' \notin \mathcal{P}_o')\big) \implies p_1' \notin \mathcal{P}_o'.$$

PROOF. Let $p_1' = (\beta_1^1, \beta_2^1, \beta_3^1, \beta_4^1, \ldots, \beta_1^N, \beta_2^N, \beta_3^N, \beta_4^N)$ and $p_2' = (\alpha_1^1, \alpha_2^1, \alpha_3^1, \alpha_4^1, \ldots, \alpha_1^N, \alpha_2^N, \alpha_3^N, \alpha_4^N)$. We assume $p_2' \leq p_1'$ and $p_1' \in \mathcal{P}_o'$. Then $p_1' \in (\mathcal{C}')^N$ which implies $\alpha_1^i \leq \beta_1^i \leq -\beta_2^i \leq \alpha_2^i$, $\alpha_1^i \leq \beta_1^i \leq -\beta_4^i \leq -\alpha_4^i$, and $\alpha_3^i \leq \beta_3^i \leq -\beta_4^i \leq -\alpha_4^i$ for all $i \in \mathbb{N}_{[1,N]}$. Thus $p_2' \in (\mathcal{C}')^N$. We also have $\mathsf{Sched}(f(p_1'))$. In this case, $p_1 = f(p_1') = (\underline{\tau}_1^1, \overline{\tau}_1^1, \underline{h}_1^1, \overline{h}_1^1, \ldots, \underline{\tau}_1^N, \overline{\tau}_1^N, \underline{h}_1^N, \overline{h}_1^N)$ and $p_2 = f(p_2') = (\underline{\tau}_2^1, \overline{\tau}_2^1, \underline{h}_2^1, \overline{h}_2^1, \ldots, \underline{\tau}_2^N, \overline{\tau}_2^N, \underline{h}_2^N, \overline{h}_2^N)$ satisfy $p_1 \in \mathcal{C}^N$, $p_2 \in \mathcal{C}^N$ and for all $i \in \mathbb{N}_{[1,N]}$

$$\underline{\tau}_2^i \leq \underline{\tau}_1^i, \ \overline{\tau}_2^i \geq \overline{\tau}_1^i, \ \underline{h}_2^i \leq \underline{h}_1^i, \ \overline{h}_2^i \geq \overline{h}_1^i. \tag{10}$$

It is easy to check that if $\mathcal{T}$ is schedulable under timing contracts $\Theta_1 = \{\theta(\underline{\tau}_1^1, \overline{\tau}_1^1, \underline{h}_1^1, \overline{h}_1^1), \ldots, \theta(\underline{\tau}_1^N, \overline{\tau}_1^N, \underline{h}_1^N, \overline{h}_1^N)\}$ then $\mathcal{T}$ is schedulable under timing contracts $\Theta_2 =$

$\{\theta(\underline{\tau}_2^1, \overline{\tau}_2^1, \underline{h}_2^1, \overline{h}_2^1), \ldots, \theta(\underline{\tau}_2^N, \overline{\tau}_2^N, \underline{h}_2^N, \overline{h}_2^N)\}$ for all $p_2 \in \mathcal{C}$ satisfying (10). Thus, $\mathsf{Sched}(f(p_2'))$ holds and $p_2' \in \mathcal{P}_o'$.

This proves the first implication. For the second implication, it is sufficient to check that

$$\begin{aligned} & \big((p_2' \leq p_1') \wedge (p_1' \in \mathcal{P}_o')\big) \implies p_2' \in \mathcal{P}_o' \\ \equiv \quad & \neg(p_2' \leq p_1') \vee (p_1' \notin \mathcal{P}_o') \vee (p_2' \in \mathcal{P}_o') \\ \equiv \quad & \big((p_2' \leq p_1') \wedge (p_2' \notin \mathcal{P}_o')\big) \implies p_1' \notin \mathcal{P}_o'. \end{aligned}$$

□

Now using the Property 1 and the set $\mathcal{P}_{st}$ obtained in Section 4.1 we can sample the parameter space to solve Problem 3.

THEOREM 1. *Let* $\underline{p}^1, \ldots, \underline{p}^{M_1} \in \mathcal{P}_o'$, *and* $\overline{p}^1, \ldots, \overline{p}^{M_2} \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N' \setminus \mathcal{P}_o'$ *and let*

$$\underline{\mathcal{P}'} = \bigcup_{j=1}^{M_1} \{p' \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N' : \underline{p}^j \geq p'\},$$

$$\overline{\mathcal{P}'} = (\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') \setminus \bigcup_{j=1}^{M_2} \{p' \in \mathcal{D}_1' \times \cdots \times \mathcal{D}_N' : p' \geq \overline{p}^j\}.$$

Then, $\underline{\mathcal{P}'} \subseteq \mathcal{P}_o' \subseteq \overline{\mathcal{P}'}$. *Moreover,* $\mathcal{P}^* = f(\underline{\mathcal{P}'}) \cap \mathcal{P}_{st}$ *is a solution to Problem 3*

PROOF. $\underline{\mathcal{P}'} \subseteq \mathcal{P}_o' \subseteq \overline{\mathcal{P}'}$ is a direct consequence of Proposition 1. Then, it follows that $\mathcal{P}^*$ is a solution to Problem 3. □

4.3 Algorithm for timing contract synthesis

Theorem 1 shows that it is possible to compute under and over-approximations of the set $\mathcal{P}_o'$ by sampling the parameter space $\mathcal{D}_1' \times \cdots \times \mathcal{D}_N'$. In this section, given the set $\mathcal{P}_{st}$ from Section 4.1, we use this property to design a synthesis algorithm. Similar algorithms have been used in [27, 35] for computing an approximation of the Pareto front of a monotone multi-criteria optimization problem. Indeed, this latter problem can be tackled by computing an under and over-approximation of a set satisfying a monotonicity property similar to that of Proposition 1.

ALGORITHM 1. *Timing contract synthesis*

function *TC_Synth*($\mathcal{T}$,$\{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$,$\mathcal{P}_{st}$)
input: $\mathcal{T}$, $\mathcal{D}_i = [\tau_{min}^i, \tau_{max}^i]^2 \times [h_{min}^i, h_{max}^i]^2, i \in \mathbb{N}_{[1,N]}$, $\mathcal{P}_{st} \subseteq \mathcal{C}^N \cap (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N)$,
output: $\mathcal{P}^* \subseteq \mathcal{C}^N \cap (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N)$
parameter: $\varepsilon \in \mathbb{R}^+$
1: ***if*** $p_{max}' \in \mathcal{P}_o'$ ***then***
2: *return* $(\mathcal{D}_1 \times \cdots \times \mathcal{D}_N) \cap \mathcal{P}_{st}$;
3: ***else*** $\overline{\mathcal{P}'} := (\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') \setminus \{p_{max}'\}$;
4: ***end if***
5: ***if*** $p_{min}' \notin \mathcal{P}_o'$ ***then***
6: *return* $\emptyset$;
7: ***else*** $\underline{\mathcal{P}'} := \{p_{min}'\}$;
8: ***end if***
9: ***while*** $d(\underline{\mathcal{P}'}, \overline{\mathcal{P}'}) > \varepsilon$ ***do*** ▷ *main loop*
10: *Pick* $p' \in \overline{\mathcal{P}'} \setminus \underline{\mathcal{P}'}$; ▷ *select next sample*
11: ***if*** $p' \in \mathcal{P}_o'$ ***then***
12: $\underline{\mathcal{P}'} := \underline{\mathcal{P}'} \cup \{p_*' \in (\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') : p_* \leq p'\}$;
13: ***else*** $\overline{\mathcal{P}'} := \overline{\mathcal{P}'} \setminus \{p_*' \in (\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') : p' \leq p_*'\}$;
14: ***end if***
15: ***end while***
16: *return* $f(\underline{\mathcal{P}'}) \cap \mathcal{P}_{st}$;

Algorithm 1 computes an under-approximation $\underline{\mathcal{P}'}$ and an over-approximation $\overline{\mathcal{P}'}$ of the set $\mathcal{P}_o'$ by sampling iteratively the parameter space $\mathcal{D}_1' \times \cdots \times \mathcal{D}_N'$.

Lines 1 to 8 initialize these approximations by testing both the lower bound $p_{min}' = (\tau_{min}^1, -\tau_{max}^1, h_{min}^1, -h_{max}^1, \ldots, \tau_{min}^N, -\tau_{max}^N, h_{min}^N, -h_{max}^N)$ and the upper bound $p_{max}' = (\tau_{max}^1, -\tau_{min}^1, h_{max}^1, -h_{min}^1, \ldots, \tau_{max}^N, -\tau_{min}^N, h_{max}^N, -h_{min}^N)$ of the set $\mathcal{D}_1' \times \cdots \times \mathcal{D}_N'$. If $p_{max}' \in \mathcal{P}_o'$, then by Theorem 1, $f(\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') \cap \mathcal{P}_{st} = (\mathcal{D}_1 \times \cdots \times \mathcal{D}_N) \cap \mathcal{P}_{st}$ is a solution to Problem 3. Note that in that case, all timing-contract parameters, $(\underline{\tau}_1^1, \overline{\tau}_1^1, \underline{h}_1^1, \overline{h}_1^1, \ldots, \underline{\tau}_1^N, \overline{\tau}_1^N, \underline{h}_1^N, \overline{h}_1^N) \in \mathcal{D}_1 \times \cdots \times \mathcal{D}_N$ guarantee the schedulability of $\mathcal{T}$ under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$. If $p_{max} \notin \mathcal{P}_o'$, then $(\mathcal{D}_1' \times \cdots \times \mathcal{D}_N') \setminus \{p_{max}'\}$ is an over-approximation of $\mathcal{P}_o'$. Similarly, if $p_{min}' \notin \mathcal{P}_o'$, then by Theorem 1, $\mathcal{P}_o' = \emptyset$. Note that in that case, no timing-contracts can guarantee the schedulability of $\mathcal{T}$. If $p_{min}' \in \mathcal{P}_o'$, then $\{p_{min}'\}$ is an under-approximation of $\mathcal{P}_o'$.

Lines 9 to 14 describe the main loop of the timing contract synthesis algorithm. At any time of the execution, $\underline{\mathcal{P}'} \subseteq \mathcal{P}_o' \subseteq \overline{\mathcal{P}'}$ holds. We pick a sample $p' \in \overline{\mathcal{P}'} \setminus \underline{\mathcal{P}'}$ which is the unexplored parameter region lying in the over-approximation of $\mathcal{P}_o'$ but not in its under-approximation. If $p' \in \mathcal{P}_o'$ (or if $p' \notin \mathcal{P}_o'$), then we update the under-approximation $\underline{\mathcal{P}'}$ (or the over-approximation $\overline{\mathcal{P}'}$) according to Theorem 1. The algorithm stops when the Hausdorff distance between the $\underline{\mathcal{P}'}$ and $\overline{\mathcal{P}'}$ becomes smaller than ε. One crucial issue is that the choice of the sample $p' \in \overline{\mathcal{P}'} \setminus \underline{\mathcal{P}'}$, at line 10, is crucial for the efficiency of the algorithm. In our implementation of the algorithm, we use the selection criteria proposed in [27] which consists in choosing the sample that will produce the fastest decrease of the Hausdorff distance $d(\underline{\mathcal{P}'}, \overline{\mathcal{P}'})$. In [35] an alternative selection criteria based on multiscale grid exploration was proposed.

Finally, it is important to note that Algorithm 1 needs testing if the samples $p' \in \mathcal{P}_o'$, which require checking the condition $\mathsf{Sched}(f(p'))$. In our implementation, this is done using the method proposed in Section 3, which assures us that the set $f(\underline{\mathcal{P}'})$ tends to $\mathcal{P}_0$ as $\varepsilon \to 0$, where $\mathcal{P}_0$ is the set of all solutions $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \ldots, \underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)$ such that $\mathcal{T}$ is schedulable under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \ldots, \theta(\underline{\tau}^N, \overline{\tau}^N, \underline{h}^N, \overline{h}^N)\}$.

5. ILLUSTRATIVE EXAMPLE

In this section, we verify the stability and the schedulability of two systems sharing a common computational resource under given timing contracts. Then, we show an application of the timing contract synthesis algorithm.

We implemented the scheduling approach presented in Section 3 in UPPAAL-TIGA [6] and Algorithm 1 in Matlab. All reported experiments are realized on a desktop with i7 4790 processor of frequency 3.6 GHz and a 8 GB RAM.

We consider two systems $\mathcal{S}_1 = (A_1, B_1, K_1)$ and $\mathcal{S}_2 = (A_2, B_2, K_2)$, taken from [9], given by the following matrices:

$$A_1 = \begin{pmatrix} 0 & 1 \\ 0 & -0.1 \end{pmatrix}, \; B_1 = \begin{pmatrix} 0 \\ 0.1 \end{pmatrix}, \; K_1 = \begin{pmatrix} -3.75 & -11.5 \end{pmatrix}. \tag{11}$$

$$A_2 = \begin{pmatrix} 0 & 1 \\ -2 & 0.1 \end{pmatrix},\ B_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ K_2 = \begin{pmatrix} 1 & 0 \end{pmatrix}. \qquad (12)$$

Furthermore, we set the best and worst case execution times for each task as $\underline{c}^1 = 0.12$, $\overline{c}^1 = 0.35$, $\underline{c}^2 = 0.04$, and $\overline{c}^2 = 0.12$.

5.1 Stability verification

Using Algorithm 1 in [2] we could verify that systems $\mathcal{S}_1$ and $\mathcal{S}_2$ are GUES under timing contracts $\theta(0.1, 0.35, 0.3, 0.85)$ and $\theta(0.2, 0.6, 0.8, 1.15)$ respectively. The computation times required for stability verification are 1.96 seconds and 1.5 seconds, respectively.

5.2 Scheduling

Now, we consider the set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), (\underline{c}^2, \overline{c}^2)\}$ and the same timing contracts $\Theta = \{\theta(0.1, 0.35, 0.3, 0.85), \theta(0.2, 0.6, 0.8, 1.15)\}$ as in the previous section.

In order to solve the scheduling problem, we associate to $\mathcal{T}$ the timed game automaton TGA as given in Definition 6. Following the approach in Section 3, we solve the safety game on TGA to find a strategy (if it exists) for the triggering of controllable actions that occur at $(t_k^{s_i})_{k\in\mathbb{N}}$, $(t_k^{b_i})_{k\in\mathbb{N}}$, and $(t_k^{a_i})_{k\in\mathbb{N}}$, with $i \in \mathbb{N}_{[1,2]}$, guaranteeing that the set of bad states $\overline{L}_u$ of the system, given by (13), is never reached regardless of when uncontrollable actions occurring at $(t_k^{e_i})_{k\in\mathbb{N}}$, $i \in \mathbb{N}_{[1,2]}$, are exactly taken.

Using UPPAAL-TIGA, we successfully prove that $\mathcal{T}$ is schedulable under timing contracts Θ, and thus a scheduling policy was found. The computation time required to solve the game was 1.37 seconds.

Figure 2 shows the timing of events resulting from this scheduling policy. The first and second plots show that the timing contracts $\theta(0.1, 0.35, 0.3, 0.85)$ and $\theta(0.2, 0.6, 0.8, 1.15)$ are respected for both systems $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively. The third plot shows that only one of the two systems gains access to the shared processor at a time since it appear clearly that

$$\text{for all } t \in \mathbb{R}_0^+, \mathsf{Com}(\mathcal{S}_1, t) \wedge \mathsf{Com}(\mathcal{S}_2, t) \equiv \mathsf{False}.$$

One can notice that in the third and eighth control cycles of $\mathcal{S}_1$, the beginning of the computation has to be delayed until the computational resource is released by $\mathcal{S}_2$.

Using this scheduling policy, Figure 3 shows results of simulating $\mathcal{S}_1$ and $\mathcal{S}_2$, when they share a single processor to compute the value of their control inputs, for the initial states $x_0^1 = \left(\begin{smallmatrix} 2 \\ 3 \end{smallmatrix}\right)$ and $x_0^2 = \left(\begin{smallmatrix} 2 \\ 3 \end{smallmatrix}\right)$ with $t_0^{s1} = 0.4$ and $t_0^{s2} = 0.9$. As shown, trajectories of both systems converge to zero and therefore the scheduling policy in this case guarantees the exponential stability of each system.

5.3 Timing contract synthesis

We now consider the timing contract synthesis problem for systems $\mathcal{S}_1$ and $\mathcal{S}_2$ and the set of control tasks $\mathcal{T} = \{(\underline{c}^1, \overline{c}^1), (\underline{c}^2, \overline{c}^2)\}$. We fix $\underline{\tau}^1 = 0.1$, $\underline{h}^1 = 0.3$, $\underline{\tau}^2 = 0.2$, and $\underline{h}^2 = 0.8$ and consider the following bounds on parameters $\mathcal{D}_1 = [0.1, 0.1] \times [0.1, 0.76] \times [0.3, 0.3] \times [0.3, 1.72]$ and $\mathcal{D}_2 = [0.2, 0.2] \times [0.2, 1.16] \times [0.8, 0.8] \times [0.8, 2.02]$.

Using Algorithm 2 in [2], we synthesize the set $\mathcal{P}_{st} = \mathcal{C}_1^* \times \mathcal{C}_2^* \subseteq \mathcal{C}^2 \cap (\mathcal{D}_1 \times \mathcal{D}_2)$ such that for all $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \underline{\tau}^2, \overline{\tau}^2, \underline{h}^2, \overline{h}^2) \in \mathcal{P}_{st}$, system $\mathcal{S}_i = (A_i, B_i, K_i)$ is GUES under timing contract $\theta(\underline{\tau}_i, \overline{\tau}_i, \underline{h}_i, \overline{h}_i)$, for all $i \in \mathbb{N}_{[1,2]}$. The sets $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$, in the $(\overline{\tau}^1, \overline{h}^1)$ plane and $(\overline{\tau}^2, \overline{h}^2)$ plane respectively, are shown by Figure 4.

Then, we search for a set $\mathcal{P}^* \subseteq \mathcal{P}_{st}$ such that for all $(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1, \underline{\tau}^2, \overline{\tau}^2, \underline{h}^2, \overline{h}^2) \in \mathcal{P}^*$, the set of control tasks $\mathcal{T}$ is schedulable under timing contracts $\Theta = \{\theta(\underline{\tau}^1, \overline{\tau}^1, \underline{h}^1, \overline{h}^1), \theta(\underline{\tau}^2, \overline{\tau}^2, \underline{h}^2, \overline{h}^2)\}$. We set the parameter $\varepsilon = 0.04$, and apply Algorithm 1 to compute the set $\mathcal{P}^*$. The algorithm tested 944 parameter samples and the computation time was 43.4 minutes. A section of the sets $f(\underline{\mathcal{P}}')$ and $\mathcal{P}^*$ in the $(0.1, \overline{\tau}^1, 0.3, \overline{h}^1, 0.6, 0.6, 1.15, 1.15)$ domain is shown in Figure 5.

6. CONCLUSION

In this work, we proposed useful tools for contract-based design of embedded control systems under the form of a scheduling approach and an algorithm for timing contract synthesis. These tools can be used by control and software engineers to derive requirements that must be met by the real-time implementation of a control law. The validity of our approach has been shown on examples. As future work, it would be interesting to handle the problem of controller synthesis given a timing contract, and to co-synthesize the controller and the timing contracts guaranteeing the schedulability of the latter and the stability of each of the systems.

References

[1] M. Al Khatib, A. Girard, and T. Dang. Stability verification and timing contract synthesis for linear impulsive systems using reachability analysis. *Nonlinear Analysis: Hybrid Systems*, 2016. http://dx.doi.org/10.1016/j.nahs.2016.08.007.

[2] M. Al Khatib, A. Girard, and T. Dang. Verification and synthesis of timing contracts for embedded controllers. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pages 115–124. ACM, 2016.

[3] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.

[4] A. Aminifar, P. Tabuada, P. Eles, and Z. Peng. Self-triggered controllers and hard real-time guarantees. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 636–641. IEEE, 2016.

[5] N. W. Bauer, P. J. H. Maas, and W. P. M. H. Heemels. Stability analysis of networked control systems: A sum of squares approach. *Automatica*, 48(8):1514–1524, 2012.

[6] G. Behrmann, A. Cougnard, A. David, E. Fleury, K. G. Larsen, and D. Lime. UPPAAL-TIGA: Time for playing games! In *International Conference on Computer Aided Verification*, pages 121–125. Springer, 2007.

[7] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger, and K. Larsen. Contracts for systems design:

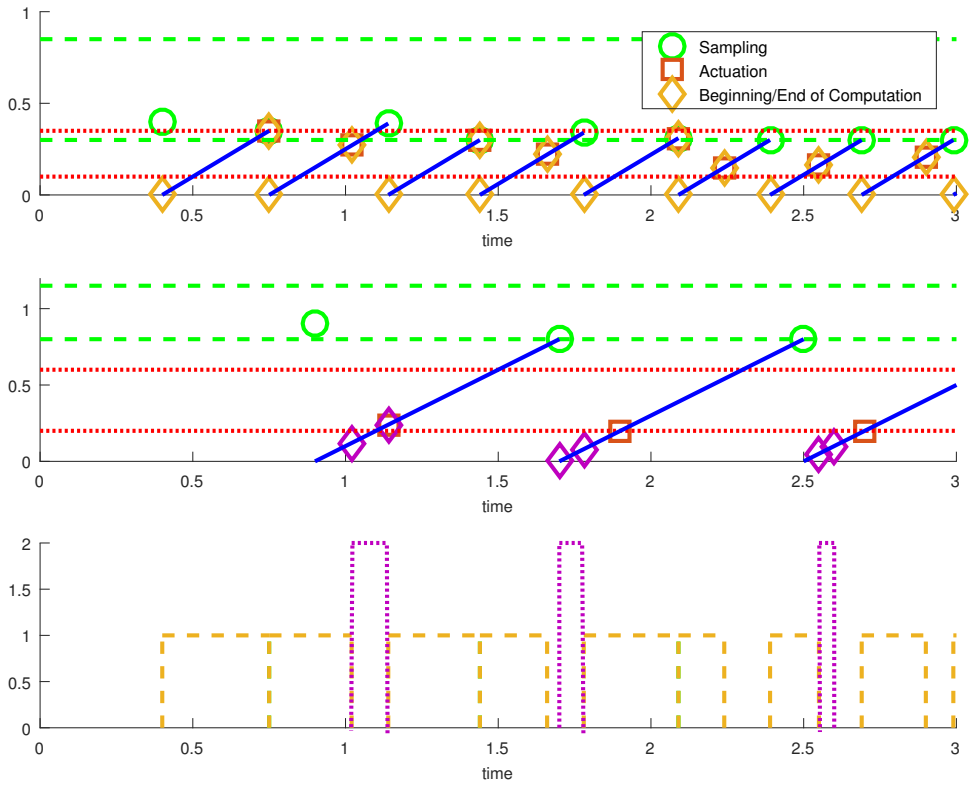

Figure 2: Timing of events (sampling, beginning/end of computation, and actuation) for systems $\mathcal{S}_1$ (first plot) and $\mathcal{S}_2$ (second plot) during the first 3 seconds; dotted lines represent constraints on actuation instants, while dashed lines represent constraints on sampling instants. In the third plot, the dotted line represents $\mathbf{Com}(\mathcal{S}_2, t)$ (less frequent) and the dashed line represents $\mathbf{Com}(\mathcal{S}_1, t)$ (more frequent).

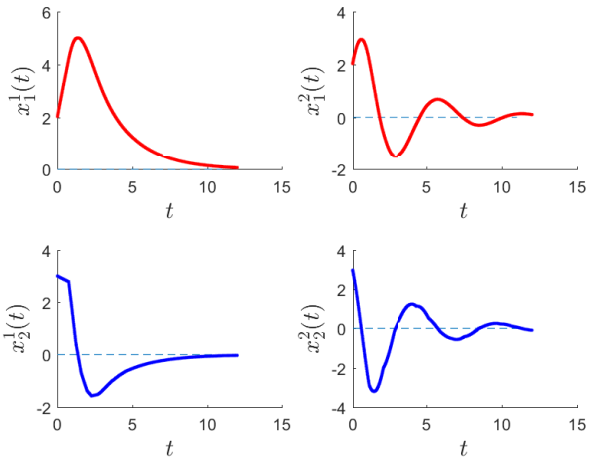

Figure 3: Trajectories for systems $\mathcal{S}_1$ (left) and $\mathcal{S}_2$ (right) using the synthesized scheduling policy.

methodology and application cases. Research Report RR-8760, Inria Rennes Bretagne Atlantique ; INRIA, July 2015.

[8] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger, and K. Larsen. Contracts for systems design: theory. Research Report RR-8759, Inria Rennes Bretagne Atlantique ; INRIA, July 2015.

[9] C. Briat. Convex conditions for robust stability analysis and stabilization of linear aperiodic impulsive and sampled-data systems under dwell-time constraints. *Automatica*, 49(11):3449–3457, 2013.

[10] G. Buttazzo. *Hard real-time computing systems: predictable scheduling algorithms and applications*, volume 24. Springer Science & Business Media, 2011.

[11] F. Cassez, A. David, E. Fleury, K. G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In *CONCUR 2005-Concurrency Theory*, pages 66–80. Springer-Verlag, 2005.

[12] A. Çela, M. B. Gaid, X.-G. Li, and S.-I. Niculescu. *Optimal design of distributed control and embedded systems*. Springer Science & Business Media, 2013.

[13] M. B. G. Cloosterman, L. Hetel, N. Van De Wouw, W. P. M. H. Heemels, J. Daafouz, and H. Nijmeijer.

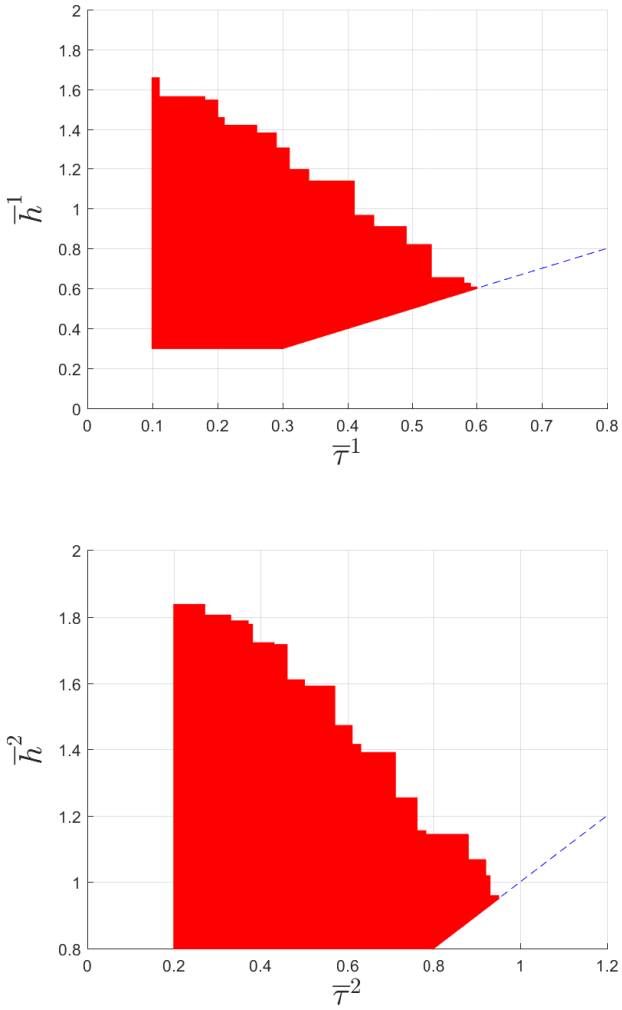

Figure 4: Timing contract parameters that guarantee stability for each system $\mathcal{S}_1$ and $\mathcal{S}_2$: $\mathcal{C}_1^*$ (top) and $\mathcal{C}_2^*$ (bottom).

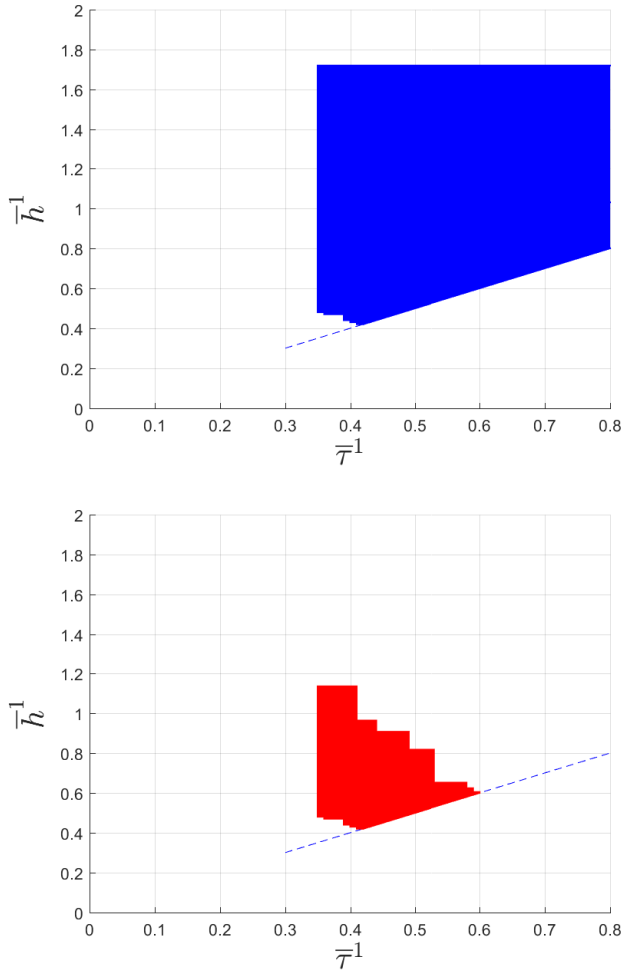

Figure 5: A 2D section of $f(\underline{\mathcal{P}}')$ (top) and $\mathcal{P}^*$ (bottom) in the $(0.1, \overline{\tau}^1, 0.3, \overline{h}^1, 0.6, 0.6, 1.15, 1.15)$ domain.

Controller synthesis for networked control systems. *Automatica*, 46(10):1584–1594, 2010.

[14] F. Cottet, J. Delacroix, C. Kaiser, and Z. Mammeri. Scheduling in real-time systems. *Scheduling in Real-Time Systems*, page 282, 2002.

[15] S.-L. Dai, H. Lin, and S. S. Ge. Scheduling-and-control codesign for a collection of networked control systems with uncertain delays. *IEEE Transactions on Control Systems Technology*, 18(1):66–78, 2010.

[16] A. David, J. Illum, K. G. Larsen, and A. Skou. Model-based framework for schedulability analysis using uppaal 4.1. *Model-based design for embedded systems*, 1(1):93–119, 2009.

[17] P. Derler, E. A. Lee, S. Tripakis, and M. Törngren. Cyber-physical system design contracts. In *ACM/IEEE International Conference on Cyber-Physical Systems*, pages 109–118, 2013.

[18] M. C. F. Donkers, W. P. M. H. Heemels, N. Van De Wouw, and L. Hetel. Stability analysis of networked control systems using a switched linear systems approach. *IEEE Transactions on Automatic Control*, 56(9):2101–2115, 2011.

[19] A. Fehnker. *Scheduling a steel plant with timed automata*. Computing Science Institute Nijmegen, Faculty of Mathematics and Informatics, Catholic University of Nijmegen, 1999.

[20] M. Fiacchini and I.-C. Morarescu. Set theory conditions for stability of linear impulsive systems. In *IEEE Conference on Decision and Control*, pages 1527–1532, 2014.

[21] H. Fujioka. Stability analysis of systems with aperiodic sample-and-hold devices. *Automatica*, 45(3):771–775, 2009.

[22] H. Gao, X. Meng, T. Chen, and J. Lam. Stabilization of networked control systems via dynamic output-feedback controllers. *SIAM Journal on Control and Optimization*, 48(5):3643–3658, 2010.

[23] W. P. M. H. Heemels, A. R. Teel, N. Van de Wouw, and D. Nešić. Networked control systems with communication constraints: Tradeoffs between transmission intervals, delays and performance. *IEEE Transactions on Automatic Control*, 55(8):1781–1796, 2010.

[24] L. Hetel, J. Daafouz, S. Tarbouriech, and C. Prieur. Stabilization of linear impulsive systems through a nearly-periodic reset. *Nonlinear Analysis: Hybrid Systems*, 7(1):4–15, 2013.

[25] L. Hetel, A. Kruszewski, W. Perruquetti, and J.-P. Richard. Discrete and intersample analysis of systems with aperiodic sampling. *IEEE Transactions on Automatic Control*, 56(7):1696–1701, 2011.

[26] A. S. Kolarijani, D. Adzkiya, and M. Mazo. Symbolic abstractions for the scheduling of event-triggered control systems. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 6153–6158. IEEE, 2015.

[27] J. Legriel, C. Le Guernic, S. Cotton, and O. Maler. Approximating the pareto front of multi-criteria optimization problems. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 69–83. Springer, 2010.

[28] K. Liu, E. Fridman, and L. Hetel. Networked control systems in the presence of scheduling protocols and communication delays. *SIAM Journal on Control and Optimization*, 53(4):1768–1788, 2015.

[29] K. Liu, V. Suplin, and E. Fridman. Stability of linear systems with general sawtooth delay. *IMA Journal of Mathematical Control and Information*, 27(4):419–436, 2010.

[30] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 229–242. Springer, 1995.

[31] P. Naghshtabrizi and J. P. Hespanha. Analysis of distributed control systems with shared communication and computation resources. In *2009 American Control Conference*, pages 3384–3389. IEEE, 2009.

[32] A. Sangiovanni-Vincentelli, W. Damm, and R. Passerone. Taming dr. frankenstein: Contract-based design for cyber-physical systems. *European journal of control*, 18(3):217–238, 2012.

[33] A. Seuret and M. Peet. Stability analysis of sampled-data systems using sum of squares. *IEEE Transactions on Automatic Control*, 58(6):1620–1625, 2013.

[34] P. Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic Control*, 52(9):1680–1685, 2007.

[35] P. Tendulkar. *Mapping and Scheduling on Multi-core Processors using SMT Solvers*. PhD thesis, Universite de Grenoble I-Joseph Fourier, 2014.