Synthesis of robust control systems under resource constraints

Luigi Palopoli[2]* , Claudio Pinello[2], Alberto Sangiovanni Vincentelli[2], Laurent Elghaoui[2] and Antonio Bicchi[3]

[1] ReTiS Lab, Scuola Superiore S. Anna, Pisa, Italy, palopoli@sssup.it,
[2] Dept. of EECS, University of Berkeley California pinello@eecs.berkeley.edu, alberto@eecs.berkeley.edu, elghaoui@eecs.berkeley.edu,
[3] Centro Piaggio, Facoltà di Ingegneria, Università di Pisa, Italy bicchi@ing.unipi.it

Abstract. We address the problem of synthesizing real-time embedded controllers taking into account constraints deriving form the implementation platform. Specifically, we address the problem of controlling a set of independent systems by using a shared single CPU processor board. Assuming a time-triggered model of computation for control processes and a real-time preemptive scheduling policy, the problem of optimizing robustness of the controlled systems is formulated. Decision variables of the optimization problem are the processes' activation periods and the feedback gains. The analytical formulation of the problem enables an efficient numerical solution based on a Branch and Bound scheme. The resulting design is directly implementable without performance degradations due to scheduling and execution delays.

1 Introduction and related work

The production of quality embedded software from the specification of control algorithms is still a bottleneck in many industrial applications [1]. The most evident problem is the difficulty in establishing a clear flow of information among the activities of control designers and system engineers. The obvious results are design cycles that are longer than needed; implementations are often over-designed and serious difficulties arise in validating the overall design.

A possible solution is to mix algorithm design and implementation choice. However, this is hardly possible when considering the complexity of the designs that confront us in safety-critical applications such as transportation and industrial plant control. It seems that we are then in a quandary: on one side, we need more separation to cope with design correctness and time-to-market, on the other, we need to capture aspects of the design that couple all layers of abstraction.

* The research presented in this paper was developed while the author was hosted as a visiting scholar at the Department of EECS, University of California, Berkeley

To solve this problem, we advocate a design process that maintains clear separation of concerns but exposes to the algorithm expert the most critical parameters of the underlying hardware and software architectures. This goal can be achieved by identifying a set of abstraction levels that mark the progress from specification to implementation. Each abstraction layer has to be related to the next in the sequence by a "behavior containment" that should guarantee that whatever has been proven correct at one layer will stay correct in the next layer. The set of abstraction layers (also called platforms), the parameters that characterize the implementation layers, the constraints that are mapped down from the specification layers and of the tools that are used to map one layer into another constitute the so-called *platform-based design paradigms* [11].

In this paper, we apply this generic idea to the design of embedded controllers. In particular, we expose the scheduling policy parameters to the control layer so that an overall optimization problem can be formulated where control concerns such as stability and robustness are considered together with scheduling issues.

Other ideas have been proposed that can be cast in this overall framework. Generally speaking, a feedback controller is an agent exchanging a flow of information with the environment. From this standpoint, an implementation platform introduces limitations on the amount of information that can be circulated. An intriguing formulation of this concept can be found in a seminal paper by Roger Brockett [4]. Bandwidth and bit-rate constraints in communication links have been studied in [18] with respect to the stabilization of linear systems [18] and of state estimation [17, 13]. The application of LQG techniques to control synthesis with communication constraints is illustrated in [16]. A related area of research is the effect of quantization [19], where an optimization procedure is applied to the number of bits allocated to each signal and to other communication variables to minimize variance of the noise introduced by quantization. A different thread [3, 7, 6] considers quantization as an intrinsic feature of certain control systems.

Concurrency may be another source of complexity. Obvious examples are multi-agent applications, where a complex goal is attained by coordinating simpler behaviors [9]. Even a single agent may be specified as a set of periodically activated parallel processes. Very often parallelism in the application is not matched by parallelism in the implementation platform and a scheduling mechanism must then be introduced. In this context, one of the most interesting approaches is the use of parameters governing flows of information across shared resources (e.g, the processes' scheduling priorities) to optimize the control performance. Remarkable attempts to combine control synthesis and scheduling of communication channels are in [14, 8]. In both papers, an integrated formulation encompassing control and communication is proposed. The scheme that is considered for scheduling is Time Division Multiplexing. Authors provide results to design the control parameters once the schedule for the shared resource has been fixed. However, different scheduling choices can be only compared with exhaustive search, which is not necessarily a viable solution in many applications.

Two aspects are deemed relevant when dealing with a concurrent platform during control synthesis: 1) the timing behavior, 2) the constraints to the design parameters (e.g. sampling periods) deriving from limited resources. The two problems have strong dependencies. For example, in single processor scheduling, it is well known that preemptive algorithms[12, 5] ensure a better utilization of the processor resource. The price to be paid is the introduction of stochastic jitter in the emission of results which hinders the construction of realistic models for the delays.

In this paper, a more ambitious goal is pursued: devising a numerically efficient procedure to optimize the control performance operating at the same time on all the available design parameters. As a first step towards a general approach to embedded controller design and for the sake of simplicity, we consider only the problem of controlling *a set of scalar linear systems using a shared computational resource*. The underlying architecture is assumed to be based on a single CPU board shared among concurrent processes via a *preemptive scheduling scheme*. The concurrent model of computation adopted to model the timing behavior of processes is the Time-Triggered approach, proposed by Kopetz and adopted in important hardware platforms [10]. More recently, the Giotto programming language [15] adopted the same paradigm.

The performance metric used in our optimization approach is related to the robustness of the overall system. This metric is of particular relevance in all contexts where tight cost constraints are not only placed on the computation resources but also on the quality of sensors and actuators. The variables of the optimization problem are activation periods of different processes and the gains of the controllers. This is a mixed integer program since the activation periods can only assume integer values. We have devised a branch-and-bound approach where the relaxation of the problem to a continuous optimization problem yields tight and numerically affordable estimations for the lower and upper bound functions.

The paper is organized as follows. In Section 2, we review briefly the Time-Triggered approach and the results on Real-Time preemptive Scheduling. In Section 3, we formulate the optimization problem. In Section 4, we describe the optimization algorithm and offer some computational results. In Section 5, we give conclusions and present our plan for extending this approach to a larger class of systems.

2 Background

2.1 The time-triggered MoC

The time triggered model of computation assumes a set of periodic tasks τ_i. Every p_1 time units process τ_i becomes active and reads its inputs; then it computes the output values. Outputs are released right at the beginning of the next activation period and their values are sustained between two subsequent writings (according to a ZoH model).

In the sequel the term *job* will indicate each periodic activation of a process. It is worth observing that the timing behavior resembles the one of a Moore sequential circuit, where the delay between inputs and outputs is equal to the activation period p. The fixed delay virtually nulls jitter in the release time of the outputs, greatly simplifying the control law design.

2.2 Real-time schedulability of periodic processes

A scheduling policy is said to guarantee *schedulability* of a set of processes if all real-time constraints are respected. A necessary condition for achieving schedulability on a single CPU is:

$$\sum_{i=1}^{m} \frac{e_i}{p_i} \le 1, \tag{1}$$

where e_i and p_i are integer numbers, counting the clock cycles required to compute process τ_i in the worst case and between successive activations of process τ_i, respectively. Moreover sufficient conditions can be derived for different scheduling algorithms. Popular examples are Rate Monotonic (RM) and Earliest Deadline First (EDF) for which a sufficient schedulability condition was derived in a seminal work by Liu and Layland [12]:

$$\sum_{i=1}^{m} \frac{e_i}{p_i} \le U, \tag{2}$$

where $U = 1$ for EDF and $U = m(2^{\frac{1}{m}} - 1)(> 0.69)$ for RM. Other scheduling algorithms, called *Pfair* [2], guarantee schedulability under condition 2 with $U = 1$ on a single processor architecture and scale well also to multiprocessor architectures, by setting U equal to the available number of processors. It is worth pointing out that in schedulability analysis e_i are the *worst-case* execution times for processes and that conservative choices on U allow to achieve robustness with respect to unmodeled delays or to reserve space for other processes, as required.

2.3 Platform definition

Given the choice of the time triggered MoC and of a class of real time schedulers, the implementation platform can be parameterized by:

$$U, e_1, \ldots, e_m, \tag{3}$$

to be used in (2). A m-tuple of activation periods $\mathbf{p} \in \mathbb{N}^m$ satisfying (2) is guaranteed to yield a design adhering to the MoC. It is worth noting that the set of parameters (3) may represent a family of implementation platforms, all meeting or exceeding those requirements. As a matter of fact information on the HW/SW architecture condensed in the proposed platform are: 1) it must ensure worst case execution time of e_i, 2) it has to be endowed with a preemptive RTOS, 3) the scheduling algorithm has to guarantee schedulability with total utilization U. As shown next, the proposed characterization for the platform leads to a compact analytical formulation of the control design problem.

3 Problem formulation

Consider the following collection of scalar systems $\mathcal{S}_i, i = 1, \dots, m$ each described by:

$$x_i(k+1) = a_i x_i(k) + b_i u_i(k). \tag{4}$$

Assume $b_i \neq 0 \ \forall i = 1, \dots, m$, i.e. all systems are completely controllable. Each system is controlled in feedback by a process τ_i. Such processes are implemented according to the time-triggered model of computation. The processing activity consists of deriving the x_i values from sensor data and computing the control values using a static feedback gain γ_i. The worst case execution time of each (*job* of) τ_i is denoted by e_i. Sensor processing activities may have very different execution times, depending on the type of sensors used on each sub-system[1], hence e_i may differ.

The control law applied to each system can be written as follows:

$$u_i(k) = \begin{cases} \gamma_i x_i(k - p_i) & k = hp_i, \\ u_i(k-1) & hp_i < k < (h+1)p_i, \end{cases} \tag{5}$$

where $h \in \mathbb{N}$. The resulting (time varying) closed loop dynamics is:

$$x_i(k+1) = a_i x_i(k) + b_i \gamma_i x_i(k - p_i - k \, mod \, p_i). \tag{6}$$

3.1 Closed loop stability analysis

In order to derive a time invariant representation, each $\mathcal{S}_i$ system is re-sampled taking one out of every p_i samples: $\hat{x}_i(h) = x_i(hp_i)$. The resulting dynamics can be written as: $\hat{x}_i(h+1) = a_i^{p_i} \hat{x}(h) + \sum_{j=0}^{p_i-1} a_i^j b_i \gamma_i \hat{x}_i(h-1)$. An equivalent form is given by:

$$\hat{\mathbf{x}}_i(h+1) = \begin{bmatrix} 0 & 1 \\ \beta_i \hat{\gamma}_i & a_i^{p_i} \end{bmatrix} \hat{\mathbf{x}}_i(h), \tag{7}$$

where $\beta_i = \sum_{j=0}^{p_i-1} a_i^j$, $\hat{\gamma}_i = b_i \gamma_i$, and $\hat{\mathbf{x}}_i(h) = [\hat{x}_i(h-1) \ \hat{x}_i(h)]^T$. The closed loop dynamics (6) of $\mathcal{S}_i$ can be related to the one of the considered subsequence thanks to the following:

Lemma 1. *System (6) is globally asymptotically stable if and only if system (7) is.*

System (7) is in standard companion form and its characteristic polynomial is given by

$$z^2 - a_i^{p_i} z - \beta_i \hat{\gamma}_i.$$

Applying Jury's criterion, system (7) is globally asymptotically stable if and only if, the following hold:

$$\beta_i \hat{\gamma}_i > -1 \tag{8}$$

$$\beta_i \hat{\gamma}_i < 1 + a_i^{p_i} \tag{9}$$

$$\beta_i \hat{\gamma}_i < 1 - a_i^{p_i}, \tag{10}$$

[1] In example, the position of a pendulum could be measured using a position encoder or processing the video feed from a camera.

where:

$$\beta_i = \begin{cases} p_i & a_i = 1 \\ \frac{1-a_i^{p_i}}{1-a_i} & a_i \neq 1. \end{cases}$$

The analysis of the system's stability turns out to be particularly easy for positive systems, i.e. $a_i > 0$. This assumption will be used throughout the rest of this paper and is valid whenever (4) is the result of sampling a scalar continuous LTI system. Hence, inequality (9) is implied by (10), and the following Proposition holds.

Proposition 1 *Consider system (6), then the following statements are true:*

1. *if $a_i = 1$ the closed loop system is asymptotically stable for all $\hat{\gamma}_i$ such that $-\frac{1}{p_i} < \hat{\gamma}_i < 0$;*
2. *if $a_i \neq 1$ the set of stabilizing solutions for $\hat{\gamma}_i$ is given by: $-\frac{1-a_i}{1-a_i^{p_i}} < \hat{\gamma}_i < (1-a_i)$. If the open loop system $\mathcal{S}_i$ is asymptotically stable ($0 < a_i < 1$) then this set is non empty (a trivial stabilizing solution is $\hat{\gamma}_i = 0$). If $\mathcal{S}_i$ is unstable ($a_i > 1$) then the set of stabilizing solutions for $\hat{\gamma}_i$ is non empty if and only if $p_i < \frac{\log 2}{\log a_i}$.*

3.2 Robustness metric

In order to characterize the robustness of the closed loop system we introduce the following definition:

Definition 1. *Consider the discrete time linear system*

$$\mathbf{x}(k+1) = A(\gamma)\mathbf{x}(k), \tag{11}$$

where $\mathbf{x} \in \mathcal{R}^n$, $\gamma \in \mathcal{R}^m$. Let $\Gamma \subseteq \mathcal{R}^m$ be such that the system is globally asymptotically stable if and only if $\gamma \in \Gamma$. Let $||.||$ denote some norm in $\mathcal{R}^m$. We define stability radius *μ the radius of the largest norm ball (induced by $||.||$) contained in Γ; we define* stability center *the center of the corresponding norm ball.*

Using feedback law (5) for each $\mathcal{S}_i$, the corresponding stability radius $\mu_{\mathcal{S}_i}$ is a measure of the largest perturbation of the gain γ_i that can be tolerated without jeopardizing stability of the closed loop system.

Let $\mathcal{S}$ denote the collection of systems $\mathcal{S}_i$ with $i = 1, \ldots, m$. A natural extension of Definition 1 for the collection is: $\mu_{\mathcal{S}} = \min_{i=1,\ldots,m} \mu_{\mathcal{S}_i}$. A direct consequence of Proposition 1 is the following

Corollary 1. *The stability radius of system (6) with respect to variations of $\hat{\gamma}_i$ and to the $||.||_\infty$ norm is given by*

$$\mu_{\mathcal{S}_i} = \begin{cases} \frac{1-a_i}{2(1-a_i^{p_i})}(2-a_i^{p_i}) & a_i \neq 1 \\ \frac{1}{2p_i} & a_i = 1 \end{cases} \tag{12}$$

Moreover, the stability center is given by:

$$\hat{\gamma}_i = \begin{cases} -a_i^{p_i} \frac{1-a_i}{2(1-a_i^{p_i})} & a_i \neq 1 \\ -\frac{1}{2p_i} & a_i = 1 \end{cases} \tag{13}$$

Remark 1. Definition 1 could be extended by considering a weigthing matrix $W \succ 0$ and by defining μ as the radius of the largest weighted norm ball. This extension could be useful in practical applications without significantly changing the results presented below. However, to keep the notation as simple as possible, we will still refer to the stability radius as defined above.

Remark 2. The stability radius $\mu_{\mathcal{S}_i}$ is a monotone decreasing function of p_i. Moreover, if the system $\mathcal{S}_i$ is open loop asymptotically stable (i.e. $0 < a_i < 1$) then it is obviously stabilized with any choice of the sampling period and a lower bound for the stability radius (obtained with $p_i \to \infty$) is given by $1 - a_i$.

3.3 Optimization problem

We are now in the condition to state the following problem:

Problem 1. Consider the collection $\mathcal{S}$ controlled by a set of time triggered periodic processes. Each τ_i, $i = 1, \ldots, m$ has a computation load e_i. Let U be the maximum utilization factor which guarantees schedulability for the assumed scheduling algorithm. Find the set of activation periods p_i and of feedback gains $\hat{\gamma}_i$ with $i = 1, \ldots, m$ such that: 1) the schedulability condition is respected, 2) the stability radius is maximized, 3) gains are at the stability center. e_i and p_i are integer numbers referred to the computer clock cycle, and $\hat{\gamma}_i \in \mathbb{R}$.

In essence the above problem consists of optimizing the system performance (stability radius) under real-time schedulability constraints.

Remark 3. Since in this paper we deal with scalar systems and the $|.|_\infty$ norm, the stability center and the stability radius can be computed in closed form from the activation periods p_i. Thanks to these assumptions, the mixed-integer Problem 1 can be reduced to the integer problem yielding the periods, the stability center being derived in a second step.

A related question is deciding whether for a given vector of computation requirements $e_1, e_2, \ldots, e_m$, a desired stability radius $\bar{\theta}$ can be achieved on a single CPU architecture. A solution to these problems will be presented next.

4 Optimization algorithm

Problem 1 can be formalized as:

$$\begin{cases} \max_{\mathbf{p}} \mu_{\mathcal{S}} \\ \sum_i \frac{e_i}{p_i} \le U \\ p_i < \frac{\log 2}{\log a_i}, \quad \text{if } a_i > 1 \\ \mu_{\mathcal{S}} \ge \bar{\mu} \\ p_i \in \mathbb{N} \backslash \{0\}, \end{cases} \tag{14}$$

where the decision variables $\mathbf{p} = [p_1,\ p_2, \ldots,\ p_m]^T$ are the activation periods of the processes τ_i, which are integer positive numbers, and $\bar{\mu}$ is the minimum stability radius acceptable for the problem. It is worth noting that if the problem encompasses open loop asymptotically stable systems, i.e. $a_i < 1$ for some i, then it is convenient to require, without loss of generality, $\bar{\mu} > 1 - a_i$. In fact if the optimal solution achieves an objective lower than $1 - a_i$, one could run control process τ_i at arbitrarily long periods p_i without effecting the cost function. Process τ_i would not actually improve robustness of the controlled system, the only real effect of including τ_i in the control problem would then be to make the admissible solution space infinite. Similarly systems with $a_i = 1$ may need a $\bar{\mu} > 0$ to ensure finiteness of the solutions space and to obtain a well-posed problem. For well posed problems the number of feasible solutions is finite. However, complete enumeration approaches may not be viable for large scale systems. A more effective approach is based on the use of a branch and bound scheme.

4.1 Branch and bound algorithm

In order to illustrate the proposed branch and bound algorithm it is useful to introduce some notation. $\mathcal{P} \subseteq \mathbb{N}^m$ will denote the set of period m-tuples $\mathbf{p}$ respecting the constraint inequalities; $\mu_l(\bar{\mathcal{P}})$, $\mu_u(\bar{\mathcal{P}})$ will denote respectively a lower bound and an upper bound of the objective function $\mu_{\mathcal{S}}(\mathbf{p})$ on the region $\bar{\mathcal{P}}$. $\mathcal{L}$ will denote a list of the disjoint subregions of the solution space which are currently candidate for the solution. The μ_l function is assumed to be derived from an admissible solution. The μ_u function is required to have the property that, when applied to a set containing a single element, it returns the value of the cost function computed for that element. The branch and bound algorithm can be formulated as follows.

Algorithm 1

```
Insert $\mathcal{P}$ into $\mathcal{L}$
set variable $\mu^*$ to $\mu_l(\mathcal{P})$
choose $\mathbf{p} \in \mathcal{P}$ s.t. $\mu_l(\mathcal{P}) = \mu_{\mathcal{S}}(\mathbf{p})$
set variable $\mathbf{p}^* = \mathbf{p}$
repeat
   choose a region $\mathcal{H}$ from $\mathcal{L}$ and remove it from $\mathcal{L}$
   partition $\mathcal{H}$ into non-empty disjoint subregions $\mathcal{H}_1, \dots, \mathcal{H}_h$
   for each $\mathcal{H}_i$
      if $\mu_u(\mathcal{H}_i) \leq \mu^*$
         discard $\mathcal{H}_i$
      else
         insert $\mathcal{H}_i$ into $\mathcal{L}$
         if $\mu_l(\mathcal{H}_i) > \mu^*$
            set $\mu^* = \mu_l(\mathcal{H}_i)$
            set $\mathbf{p}^* = \mathbf{p} \in \mathcal{H}_i$ s.t. $\mu_{\mathcal{S}}(\mathbf{p}) = \mu_l(\mathcal{H}_i)$
         endif
      endif
   end for each
until $\mathcal{L}$ is empty
```

For the problem under analysis the $\mathcal{H}_i$ subregions can be obtained by fixing some of the elements of the $\mathbf{p}$ vector. Let $\mathcal{I}(\mathcal{H}) \subseteq \{1, \dots, m\}$ be the index set of periods that are fixed to yield $\mathcal{H}$, then a finer partition $\mathcal{H}_1, \dots, \mathcal{H}_h$ can be obtained selecting one index z not in $\mathcal{I}(\mathcal{H})$ and fixing p_z to its h different admissible values[2]. The selection of the μ_u, μ_l functions is a particularly important issue. In fact, if the bounds are not tight, the algorithm does not "discard" a good number of regions resulting into a nearly exhaustive search. On the other hand accurate bounds may not be viable if overly expensive computation is required to derive them. In order to ensure correctness of the algorithm, the lower bound $\mu_l(\mathcal{H}_i)$ is assumed to be always constructed using some feasible solution $\mathbf{p}^* \in \mathcal{H}_i$ such that $\mu_{\mathcal{S}}(\mathbf{p}^*) = \mu_l(\mathcal{H}_i)$.

One possible upper bound $\mu_u(\mathcal{H}_i)$ can be derived using $\min_{j \in \mathcal{I}(\mathcal{H}_i)} \mu_{\mathcal{S}_j}(p_j)$, i.e. $\mu_u(\mathcal{H}_i) = \min\{\mu_u(\mathcal{H}), \min_{j \in \mathcal{I}(\mathcal{H}_i) \setminus \mathcal{I}(\mathcal{H})} \mu_{\mathcal{S}_j}(p_j)\}$. This upper bound may be quite optimistic and becomes tighter as the number of fixed periods increases. When all periods are fixed, i.e. $\mathcal{H}_i = \{\mathbf{p}\}$, it becomes $\mu_u(\mathcal{H}_i) = \mu_{\mathcal{S}}(\mathbf{p})$. Its computation cost is very low especially when periods are fixed incrementally. Different derivations for μ_u and for μ_l can be obtained considering the continuous relaxation of the problem, which is well posed for the positive systems ($a_i > 0$) considered in this paper.

4.2 The continuous relaxation

The integrity constraint on the periods is dropped, i.e. $p_i \in [1, +\infty)$. In order to express the schedulability constraint in linear form it is convenient to make the change of variable $f_i = \frac{1}{p_i}$. The resulting problem is given by:

$$\begin{cases} \max_{\mathbf{f}} \mu_{\mathcal{S}} \\ \sum_i e_i f_i \leq U \\ f_i > \frac{\log a_i}{\log 2} \quad (if\ a_i > 1) \\ \mu_{\mathcal{S}} \geq \bar{\mu} \\ 0 < f_i \leq 1, \end{cases} \tag{15}$$

where $\mathbf{f}$ is the vector of decision variables f_i. Considering integer e_i and a uniprocessor platform ($U \leq 1$), constraint $f_i \leq 1$ is implied by $\sum_i e_i f_i \leq U$ and will be henceforth dropped. Moreover, the following assumption will implicitly be used: $\bar{\mu} \geq \max_j (1 - a_j), \forall a_j < 1$. Introduce the following function:

$$H(\mu) = \sum_i e_i \omega_i(\mu) - U, \tag{16}$$

where:

$$\omega_i(\mu) = \begin{cases} \frac{\log a_i}{\log 2 \frac{a_i - 1 + \mu}{a_i - 1 + 2\mu}} & a_i \neq 1 \\ 2\mu & a_i = 1. \end{cases} \tag{17}$$

The solution to Problem 15 can be found as shown in the following:

[2] This set is finite because it is a projection of some subset of $\mathcal{P}$ which is finite. The number h varies with z.

Proposition 2 *Problem 15 has a solution if and only if*

$$H(\bar{\mu}) \leq 0. \tag{18}$$

If condition 18 holds, then $H(\mu)$ has only one zero in the set $\mu \geq \bar{\mu}$ which is the optimal solution of Problem 15.

Proof. As a first remark observe that problem 15 can be written in the following form:

$$\begin{cases} \max_{\mathbf{f}} \mu \\ \mu_{S_i} \geq \mu \\ \sum_i e_i f_i \leq U \\ f_i > \frac{\log a_i}{\log 2} \quad (if\ a_i > 1) \\ \mu \geq \bar{\mu} \\ f_i > 0, \end{cases} \tag{19}$$

which, through simple computations, is equivalent to:

$$\begin{cases} \max_{\mathbf{f}} \mu \\ f_i \geq \omega_i(\mu) \\ \sum_i e_i f_i \leq U \\ \mu \geq \bar{\mu}. \end{cases} \tag{20}$$

Constraints $f_i > 0$ and $f_i > \frac{\log a_i}{\log 2}$ (for $a_i > 1$) have been removed since they are implied by $f_i \geq \omega_i(\mu)$. For a fixed μ, feasible solutions exist if and only if $H(\mu) \leq 0$. Since each ω_i is strictly increasing in $\mu \geq \bar{\mu}$ also $H(\mu)$ is and this consideration leads to the proof.

Remark 4. If $\mu^{(c)}$ denotes the optimal solution, then decision variables associated to the optimal solution are given by $f_i^{(c)} = \omega_i(\mu^{(c)})$.

Remark 5. Condition 18 is interesting in itself since it addresses the problem of whether a platform is powerful enough to provide a specified level of performance $\bar{\mu}$.

Finding the zero $H(\cdot)$ can be very costly for large scale systems. Hence, it is convenient to find a lower and an upper estimation for the solution so as to bound the search. To this end, observe that nonlinearity in function H is due to the terms relative to $a_i \neq 1$. Considering the case $a_i < 1$ and restricting to $\mu \geq 1 - a_i$, we can observe that $\omega_i(\mu) = \frac{\log a_i}{\log 2 \frac{a_i - 1 + \mu}{a_i - 1 + 2\mu}}$, is concave and strictly increasing. Hence it is very easy to show the following inequality:

$$\begin{aligned} &\overline{\omega_i}(\mu) \geq \omega_i \geq \underline{\omega_i}(\mu) \\ &\overline{\omega_i}(\mu) = (\tfrac{3}{2} + 2\mu \tfrac{1}{a_i - 1}) \log a_i \\ &\underline{\omega_i}(\mu) = (2 + 2\mu \tfrac{1}{a_i - 1}) \log a_i > 0. \end{aligned} \tag{21}$$

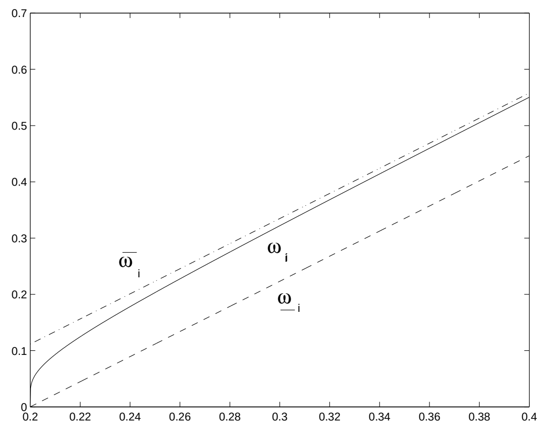

Fig. 1. Figure showing the upper and lower bound for ω_i for $a_i = 0.8$.

The meaning of the linear bounding functions appear clearly in Figure 1, where the case $a_i = 0.8$ is considered. A similar result can be found for $a_i > 1$ and $\mu \geq 0$:

$$\begin{aligned} &\overline{\omega_i}(\mu) \geq \omega_i \geq \underline{\omega_i}(\mu) \\ &\overline{\omega_i}(\mu) = (\tfrac{3}{2} + 2\mu\tfrac{1}{a_i-1}) \log a_i \\ &\underline{\omega_i}(\mu) = (\tfrac{1}{\log 2} + 2\mu\tfrac{1}{a_i-1}) \log a_i > 0. \end{aligned} \tag{22}$$

The use of linear approximations allows for a very fast computation of an upper and a lower bound of the optimal value for μ_S in Problem 15, thus limiting the search for the zero of $H(\cdot)$ to a small region. It can be interesting to take a glance at the form of the approximate solution in a simple case. Suppose, for instance, that all systems are open loop unstable ($a_i > 1, \forall i$). Denote by $\overline{\mu}^{(c)}$ the linear upper bound for the optimum, by $\overline{f}_i^c$ the decision variables for which it is attained, and introduce the variables:

$$w_i = \frac{\log a_i}{a_i - 1} \; ; \; \bar{U} = U - \sum_i e_i \frac{\log a_i}{\log 2}.$$

It is very easy to find:

$$\overline{\mu}^{(c)} = \frac{\bar{U}}{2\sum_j e_j w_j} \; ; \; \overline{f}_i^{(c)} = \frac{\log a_i}{\log 2} + w_i \frac{\bar{U}}{2\sum_j e_j w_j}.$$

For each decision variable $\overline{f}_i^{(c)}$ we can distinguish a term $\frac{\log a_i}{\log 2}$ necessary to achieve stability, which is summed to a weighted fraction of the scaled utilization $\bar{U}$. In this computation both factors relative to the systems dynamics (w_i) and to the execution time of the processes (e_j) are accounted for.

	System 1	System 2	System 3
a_i	0.992	1.0512	1.005
b_i	1	1	1

Table 1. Dynamical parameters of the controlled systems

The analysis of the continuous relaxation of Problem 14 is useful for it provides a pair of very interesting upper bound functions μ_l for the branch and bound algorithm. A low cost choice is the the optimal value of the restricted problem obtained using the lower linear approximation $\underline{\omega_i}$ of functions ω_i (which correspond to looser constraints). A better upper bound is given by the exact resolution of the continuous relaxation. It requires solving the non-linear equation $H(\mu) = 0$. It is worth noting that fixing some of the decision variables f_i (as requested by the invocation $\mu_l(\mathcal{H}_i)$ in Algorithm 1) does not change substantially the way Problem 15 and its linear approximations are solved.

Lower bound $\mu_l(\mathcal{H}_i)$ are less easy to find. In this case the continuous relaxation provides good heuristics. A good lower bound can in most cases be obtained by solving Problem 15 (or its upper linear approximation) and and then by changing the non-integer periods to their ceiling (which is a conservative approximation with respect to the schedulability constraint).

4.3 Numerical Evaluation

We illustrate a possible design flow based on the results presented in the previous section on a simple example. We consider a set of three scalar systems, whose dynamics are described by the a_i, b_i parameters in table 1.

The results of the solution of the optimization problem related to different execution times for control processes are reported in table 2. The first three rows show different choices of computation times for control tasks. The second part of the table (from the third row) is devoted to the outcome of the Branch and Bound algorithm (optimal cost functions and periods for which it is attained). Finally the last rows show results of the continuous relaxation. In the first experiment, the task used to control the system $\mathcal{S}_1$ is assumed to be very demanding (100 time units for each execution).

This may be the result of a choice of a low quality sensor (requiring a heavy post-processing of the acquired data). As might be expected, the achieved stability radius is quite low (if compared with the other experiments). Moreover, the system requiring more "attention" (i.e. lower sampling period) is the most unstable one ($\mathcal{S}_2$). In the second experiment c_1 is lowered from 100 to 10 thus relieving the CPU of a remarkable workload. As a result, more aggressive policies on the choices of the activation periods may be selected improving the stability radius. In the third experiment the effects of increasing the computational load required by the task controlling system $\mathcal{S}_2$ are shown. It is possible to observe a remarkable performance degradation with respect to the second experiment. Interestingly enough, the optimization process tends to penalize systems which

	Exp. 1	Exp. 2	Exp. 3
e_1	100	10	10
e_2	1	1	5
e_3	10	10	10
μ_S	0.0095	0.0226	0.0136
p_1	159	29	65
p_2	10	8	10
p_3	37	19	29
$\mu_S^{(c)}$	0.0096	0.0232	0.0138
$p_1^{(c)}$	155.6	29.07	65.65
$p_2^{(c)}$	10.93	8.43	10.02
$p_3^{(c)}$	37.6	18.6	28.66

Table 2. Results of the optimization

are closer to stability while keeping unvaried, as much as possible, the activation period of the process controlling $\mathcal{S}_2$. As a final remark, the continuous relaxation provides good estimation for both the objective function and the activation periods. For almost all cases periods corresponding to the optimal solution are lower or upper round up of the continuous relaxation. This is not true for in the first experiment for p_1, whose optimal value is 155.6 for the continuous relaxation and 159 for the discrete problem.

5 Conclusions and future work

In this paper, an integrated design procedure has been presented for determining optimal scheduling and control parameters for a set of controllers operating on scalar linear systems. The choice of a performance metric related to systems' robustness and of a preemptive scheduling model allows for a compact analytical formulation of the system design, amenable to an efficient numerical solution based on a branch and bound scheme. This is a first step towards more general results concerning mixed scheduling/control synthesis. The most natural continuation of this work is the extension of the approach to the control of multidimensional systems thus leading to potentially relevant industrial applications. Other important issues to investigate are alternative formulations for the platform constraints, with two distinct purposes: 1) alleviating, if possible, pessimism inherent to the choice of the time triggered approach, 2) extending the analysis to architectures with a higher degree of parallelism, such as multiprocessors and distributed architectures.

References

1. In Springer-Verlag, editor, *EMSOFT 2001: First Workshop on Embedded Software, Lecture Notes in Computer Science*, 2001.

2. S.K. Baruah, N.K. Cohen, C.G. Plaxton, and D.A. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 6, 1996.
3. A. Bicchi, A. Marigo, and B. Piccoli. Quantized control systems and discrete nonholonomy. *IEEE Trans. on Automatic Control*, 2001.
4. Roger Brockett. Minimum attention control, 1997.
5. G. Buttazzo. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications.* Kluwer Academic Publishers, Boston, 1997.
6. David F. Delchamps. Extracting state information from a quantized output record. *Systems and Control Letters*, 1989.
7. N. Elia and S. Mitter. Stabiliztion of linear systems with limited information. *IEEE Trans. on Automatic Control*, 2001.
8. Dimitris Hristu and Kristi Moransen. Limited communication control. *System and Control Letters*, 37(4):193–205, July 1999.
9. T. J. Koo, J. Liebman, C. Ma, and S. Sastry. Hierarchical approach for design of multi-vehicle multi-modal embedded software. In Springer-Verlag, editor, *EMSOFT 2001: First Workshop on Embedded Software, Lecture Notes in Computer Science*, 2001.
10. H. Kopetz and G. Grnsteidl. Ttp-a protocol for fault-tolerant real-time systems. *IEEE Computer*, January 1994.
11. K. Kuetzer, S. Malik, R. Newton, J.M. Rabaey, and A. Sangiovanni-Vincentelli. System-level design: Orthogonalization of concerns and platform-based design. *IEEE Transaction on coputer-aided design of integrated circuits and systems*, 21(27), 2000.
12. C.L. Liu and J.W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery*, 20(1), 1973.
13. G. N. Nair and R. J. Evans. State estimation under bit rate constraints. In *Proc. of the 37th IEEE Conference on Decision and Control*, Tampa, Florida, December 1998.
14. H. Rehbinder and M. Sanfridson. Scheduling of a limited communication channel for optimal control. In *Proc. of the 39th IEEE Conference on Decision and Control*, Sidney, Australia, December 2000.
15. C.M. Kirsch T. Henzinger, B. Horowitzm. Embedded control systems development with giotto. In *Proc. of ACM SIGPLAN 2001 Workshop on Languages, Compilers, and Tools for Embedded Systems (LCTES'2001)*, June 2001.
16. S. Tatikonda, A. Sahaim, and S. Mitter. Control of lqg systems under communication constraints. In *Proc. of the 37th IEEE Conference on Decision and Control*, Tampa, Florida, December 1998.
17. W.S. Wong and R. Brockett. Systems with finite bandwidth constraints - part i: State estimation problems. *IEEE Trans. on Automatic Control*, 42(9), 1997.
18. W.S. Wong and R. Brockett. Systems with finite bandwidth constraints - part ii: Stabilization with limited infromation feedback. *IEEE Trans. on Automatic Control*, 44(5), 1999.
19. L. Xiao, M. Johansson, H. Hindi, S. Boyd, and A. Goldsmith. Joint optimization of communication rates and linear systems. *submitted to IEEE Trans. on Automatic Control.*