Optimal On-line (m,k)-firm Constraint Assignment for Real-time Control Tasks Based on Plant State Information

Flavia Felicioni, Ning Jia, Françoise Simonot-Lion, Ye-Qiong Song

Optimal On-line (*m,k*)-firm Constraint Assignment for Real-time Control Tasks Based on Plant State Information

Felicioni Flavia
FCEIA – UNR
Universidad Nacional de Rosario
Rosario - Argentina
flaviaf@fceia.unr.edu.ar

Jia Ning, Françoise Simonot-Lion, Song YeQiong
LORIA – INPL
Campus Scientifique – BP 239
54506 – Vandoeuvre lès Nancy France
{Ning.Jia, simonot,song}@loria.fr

Abstract

In this paper, we study the problem of scheduling a set of control tasks. We distinguish three different situations of states of controlled plants: not activated, steady state situation and transient situation. The infinite-horizon and finite-horizon cost functions are respectively used to represent the performance of each control task in last two situations. We propose a scheduling architecture in which, according to the plant state situation, the task handler switches between these two types of performance criterion to determine an on-line (m,k)-constraint based control task scheduling strategy, so that the overall control performance is maintained at a high level in each situation subject to the task schedulability. The approach is exemplified on a set of controllers for different plants.

1. Introduction

We consider N physical plants. One dedicated controller implemented as a real-time task controls each plant. Each instance of a task is responsible for carrying out the control law computation and has a deadline by which it is expected to complete its computation. We consider a centralized implementation of all the controllers. At any time, there are n activated plants (with $n \leq N$), i.e. n activated tasks. This raises the problem of the schedulability of these n tasks so that the stability is ensured for each controlled plant. In addition to this mandatory objective, we propose to define a scheduling approach that optimizes the control performance. The proposed approach is based on a task instance dropping strategy.

In [9], we have presented a scheduling approach for a set of control tasks. In this case, the infinite-horizon cost function was used to represent the control performance of each control task. This policy selectively rejects the execution of task instances, so that the overall control performance (sum of the cost function of each task) is maintained at high level and the schedulability is ensured.

In this paper, we focus on the scheduling decision based on the current states of controlled plants; i.e. control tasks are scheduled in a way that reflects the current control performance requirement. Concretely, we distinguish different situations according to the states of the n activated plants (*steady*: the plant output is in steady state; *transient:* the plant output is in transient state). The infinite-horizon and finite-horizon cost functions are respectively used in these situations for describing the control performance of each control task. Based on these cost functions, the proposed scheduling approach selectively rejects the execution of control tasks instances, so that the overall control performance is always maximized and the set of tasks are schedulable. Specifically, changes in plant references or disturbances affecting the plants, may induce the corresponding tasks to be executed more frequently than that in steady state.

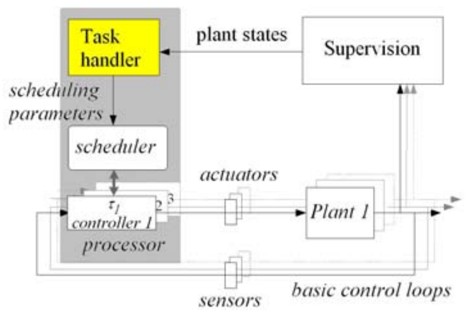


Figure 1. Overall system architecture.

The system architecture is shown in Figure 1. We suppose that a supervision function of all the controlled plants is implemented in a separate computer. The purpose of this function is to establish the new control objective and to notify the task handler that has to define the new scheduling parameters for this new objective. Specifically, the scheduling parameters are given for each task τ_i as a (m_i,k_i)-firm constraint [13][14]. A (m_i,k_i)-firm constraint means that the deadlines of at least m_i among any k_i consecutive instances of a control task τ_i must be met ($m_i , k_i \in N^+$, $m_i \leq k_i$). Since the discarded instances will not be executing the control law, this tends to degrade the control performance. However, if each controller is designed to accept a control performance degradation until k_i-m_i deadlines misses among k_i consecutive task instances (in fact, under a suited basic sampling period, control systems can tolerate misses of the control law updates to a certain extent), the controlled plant can then be conceived so that to offer the control performance levels between (k_i,k_i)-firm (ideal case) and (m_i,k_i)-firm (worst case) with as many intermediate levels as the possible values between k_i and m_i. This results in a controlled plant with graceful degradation of control

This work has been partially supported by ANR SSIA_NV_15.

performance.

As the discarded instances can be thought of as a sampling period variation, the controlled plant can be thought of as a concatenation of systems in time, and it can be modeled as a Discrete-Time Switched System (DTSS) [18]. For that reason, we propose an explicit analysis to ensure closed-loop stability for DTSS.

An alternative approach for achieving the scheduling objective of this paper is to adjust on-line the period of control tasks, as the approach proposed in [3][5][16]. However, this requires a convex cost function to represent the performance degradation, and that is not the general case. Furthermore, these approaches maintain the control performance optimality and control task schedulability by the regulation of the periods of control tasks. However, changing the period of a task may necessitate a change in the periods of dependant tasks as task periods are often carefully selected for an efficient exchange of information between relative tasks; in addition, the change in sampling period of a sub-system alters dynamics of the sub-system and leads to an unavoidable additional study for the approaches based on the regulation of the sampling periods.

The paper is organized as follows. The task model and the scheduling properties under (m,k)-firm constraint is contained in section 2. Section 3 shows how to choose the (m,k)-firm constraint for a control task, presents the finite and infinite-horizon cost function under (m,k)-firm constraint and analyzes closed-loop stability. A formal description and the solution of the scheduling problem are presented in section 4. Section 5 discusses a numerical example of the proposed scheduling approach. Finally, we summarize our work and show the perspectives.

2. Task model and scheduling properties

In this section, we provide the task model and a task instance classification strategy for fitting the (m,k)-firm constraint; some properties associated to the strategy required to assess the schedulability of control tasks and calculate the cost functions are given as well as a schedulability condition is introduced.

The controller of each plant is realized by a real-time task and the n tasks are executed on a single processor. Each instance of each task is responsible for carrying out the control law computation; therefore each task τ_i is characterized by the following parameters:

- T_i, C_i, D_i: the time interval between two consecutive instances (period), the maximum execution time and the deadline of each instance; we consider $D_i=T_i$
- m_i, k_i: the (m,k)-firm constraint for τ_i with $m_i \le k_i$.

For meeting the (m,k)-firm constraint, a task instance classification strategy was proposed in [13][14], which classifies the instances of each task into mandatory and optional instances. Only the mandatory instances have to be executed while optional instances could be not executed at all in case of processor overload; for both categories, deadlines have to be met must be met.

We proposed in [7], [11] the equation (1) to determine this classification.

$$a = \left\lfloor \left\lceil a m_i / k_i \right\rceil k_i / m_i \right\rfloor \qquad (1)$$

For $a = 0,1,\ldots$, τ_i is classified as mandatory if the equation (1) is verified and as optional, otherwise.

Example 1. Let τ_i be under (3,5)-firm constraint, the condition (1) is verified only for $a=0+\alpha k$, $1+\alpha k$ and $3+\alpha k$ ($\alpha \in N$). Therefore the instances activated at 0,1,3,5, etc are mandatory; those activated 2,4,7 are optional.

The determination of the schedulability of a set of tasks under (m,k)-firm constraints has been proved in [13] to be NP-hard; however, using the task instance classification strategy (1), the schedulability of tasks can be explicitly proved; we demonstrated in [8] that for a single control loop, the control performance obtained with the task classification (1) is sub-optimal.

Equation (1) implies that the mandatory instances are periodically distributed; if the n^{th} instance is classified as mandatory instance, the $(n+k)^{th}$ instance is also classified as mandatory instance. Below, there are some properties of the instance classification strategy (1), which will be used to determine the schedulability of control tasks and to calculate the cost function.

Lemma 1 [7][14]. Under the task instance classification strategy (1), the index of the j^{th} mandatory instance for each task τ_i is $\left\lfloor j \cdot m_i / k_i \right\rfloor$

Lemma 2 [7][14]. Under task instance classification strategy (1), there are exactly $\left\lceil t m_i / (T_i k_i) \right\rceil$ mandatory instances for each task τ_i before time instant t.

Corollary 1. Under the task instance classification strategy (1), the number of optional instances between the j^{th} and $(j+1)^{th}$ mandatory instances for each task τ_i, is given by: $\left\lfloor (j+1)(m_i/k_i) \right\rfloor - \left\lfloor j(m_i/k_i) \right\rfloor - 1$

Proof. the proof follows from lemma 1 and 2.

In this work, the control tasks are scheduled using the fixed priority policy. The mandatory instances of all the tasks are assigned the rate-monotonic priorities: the mandatory instances of τ_i are assigned a higher priority than the mandatory instances of τ_j if $T_i<T_j$.

A task under (m,k)-firm constraint is said to be schedulable if its (m,k)-firm constraint is satisfied. For the case where instances of each task are classified using equation (1), the following theorem proposed in [9] can be used to determine the task schedulability:

Theorem 1 [9]. *Given a task set (τ_1, $\tau_2 \ldots \tau_n$) such that $T_1<T_2< \ldots <T_n$. Let:*

$$n_{ij} = \left\lceil m_j / k_j \cdot \left\lceil T_i / T_j \right\rceil \right\rceil$$

If $C_i + \sum_{j=1}^{i-1} n_{ij} C_j \le T_i$ *for all* $1 \le i \le n$, *then the* *(m_i, k_i)-firm constraint of each task τ_i is satisfied.*

The above schedulability condition is sufficient and necessary if the period of a task is multiple of the periods of all the lower priority tasks. In the other case, it degenerates to a sufficient condition.

3. Performance and Stability

In section 3.1, we establish an off-line method to choose T_i and k_i for each plant i. By using these values, we can design the controller thanks to a control design technique (state-feedback controller, LQ-optimal design). For each plant i, and for each m_i in the set [1.. k_i], we must calculate controller parameters as presented in section 3.2. Section 3.3 highlights a misconception introduced in [5] about closed-loop performance. Finally, a LMI formulation, which guarantees the stability of the control, is given in section 3.4.

3.1. Sampling time selection (T_i and k_i) satisfying the constraint of step response

In this section, we present how to choose both, the value of the sampling period T_i and the value of k_i for (m,k)-firm constraint of each i^{th} plant, so that the step response requirement is respected by using the control theory.

The sampling period selection is a compromise among many factors. Most authors coincide that it is the closed-loop bandwidth the main factor that provides the maximum bound for its value [1]. Its limitation is stated by the sampling theorem: ω_{CL} (closed-loop frequency) should be lower than π/T. Even more, to obtain a good closed-loop performance, based on the closed-loop temporal response, ω_{CL} should be inside the restricted interval proposed in [1], the "rule of thumb": $0.2<\omega_{CL}T<0.6$. As well, the open-loop plant frequency should respect the sampling theorem (section 3.3).

This rule considers the rise time as the step response parameter, which represents how much time the output of a second order under-damped system needs to go from 10% to 90% of its final value. Furthermore, the overshoot should be lower than an admissible threshold (typically 10%) and the rise time T_r will be lower than a given time T_{rM}.

In order to relate the continuous-time equivalent parameter with the sampling period (h), we propose to use N_r the number of sampling periods per rise time to be chosen between 2 and 10. Note that this rule characterizes the rise time only when it has, among the closed-loop poles, a conjugate-complex pole-pair that dominates the step response. In the case of closed-loop systems with real, stable and very different poles, a class of stiff systems where the rise time can be approximated by $Tr=3/a$ (95%) where a is the continuous-time equivalent slower pole mapped by the relationship $z=e^{sh}$, we propose to adopt the N_r limits as before (2 to 10).

By changing the controller parameters at a change in sampling period, and maintaining the value of N_r within its bounds; the rise time remains approximately constant.

Then, this rule is useful to calculate approximately the minimum and maximum values for the sampling period. Specifically, we propose to assign to the sampling time T_i the maximum value of N_r, then $T_i = T_{irM}/10$. By using the minimum value N_r=2, k_i should be 5. Possible variations (4 or 6) will depend on the real value of N_r, evaluated as function of T_i. As we see in fig. 2, for a pendulum both, the time and the response times, remain approximately constant for the extreme N_r values.

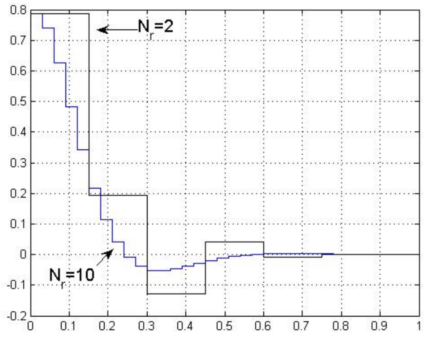


Figure 2: Minimum and maximum values of N_r.

3.2. Optimal LQ design under (m,k)-firm constraint

For each plant i and each k_i, we calculate the m_i controller parameters using a LQ design [5][14]. The LQ cost functions, with respect to task instance classification, are calculated for both, finite and infinite horizon strategy. These functions are used by the task handler (section 4.2) as the control performance index to calculate and assign the (m,k)-firm constraint to each control task.

Assume that each plant i is described by the differential equation: $dx_i = A_i x_i\,dt + B_i u_i\,dt + dv_{ic}$ (2) where x_i is the state vector containing state variables of the controlled plant, and u_i is the control signal. The process v_{ic} has mean value of zero and uncorrelated increments. The incremental covariance of v_{ic} is $R_{ic}dt$.

Let the control task realizing the controller be under a (m,k)-firm constraint, the period of control task to be T_i, and let its instances be classified using equation (1). For simplifying the cost function calculation, we suppose that the executions of all the optional instances are rejected (in practice, optional instances could be assigned the lowest priority and can be scheduled also, under an on-line schedulability test).

The control signal $u_{i,j}$ is computed by a mandatory instance, therefore the time that $u_{i,j}$ is held as constant can be calculated using *Corollary 1*. From that, the j^{th} control signal is held during $h_{ij} = T_i\left(\lfloor (j+1)(m_i/k_i)\rfloor - \lfloor j(m_i/k_i)\rfloor\right)$. During the time k_iT_i, the period h_{ij} assumes m_i values of jT_i.

The discrete time-varying model of the plant (2) is:

$$x_{i,j+1} = \Phi_{ij}x_{i,j} + \Gamma_{ij}\,u_{i,j} + v_{i,j} \quad j = 0,1,2,\ldots \qquad (3)$$

where $\Phi_{ij} = e^{Ah_{ij}} \qquad \Gamma_{ij} = \int_0^{h_{ij}} e^{As}dsB \qquad (4)$

$v_{i,j}$ is a discrete-time Gaussian white-noise process with zero mean value and the following property:

$$Ev_{i,j}v_{i,j}^T = R_{Vij} = \int_0^{h_{ij}} e^{A\tau}R_{ic}e^{A^T\tau}d\tau$$

Since the mandatory instances are distributed periodically in task instance sequence, we have $\Phi_{ij}=\Phi_{ij+mi}$, $\Gamma_{ij} = \Gamma_{ij+mi}$.

The sampled cost function that the controller aims to minimized is chosen as:

$$J_i = \frac{1}{m_i \frac{H_i}{k_i}} \left(\sum_{i=0}^{m_i \frac{H_i}{k_i T_i} - 1} \left(x_{i,j}^T Q_{ij}^{'} x_{i,j} + 2 x_{i,j}^T M_{ij} u_{i,j} + u_{i,j}^T R_{ij}^{'} u_{i,j} \right) + \right.$$

$$\left. + E\left(x^T_{m_i \frac{N_i}{k_i}} Q_{i0} x_{m_i \frac{N_i}{k_i}} \right) + \sum_{j=0}^{m_i \frac{H_i}{k_i T_i} - 1} \overline{J}_{ij} \right) \qquad (5)$$

$$Q_{ij}^{'} = \int_0^{h_{ij}} \Phi_i^T(t)\, Q \Phi_i(t) dt\ , \quad M_{ij} = \int_0^{h_{ij}} \Phi_i^T(t) Q \Gamma_i(t) dt$$

$$R_{ij}^{'} = \int_0^{h_{ji}} \left(\Gamma_i^T(t) Q \Gamma_i(t) + R_i \right) dt\ , \quad \overline{J}_{ij} = tr\left(Q \int_0^{T_{ij}} R_{ic}(\tau) d\tau \right)$$

and $\Phi_i(t) = e^{A_i t}$, $\Gamma_i(t) = \int_0^t e^{A_i s} ds \cdot B_i$; $m_i\, H_i / k_i T \in N^+$ and H_i is the i^{th} plant time horizon.

The optimal control law that minimizes the cost function (5) is given by [1] as:

$$u_{i,j} = -L_{i,j}\, x_{i,j} \quad \mathrm{j} = 0,1,2,\ldots \qquad (6)$$

where

$$L_{i,j} = \left(\Gamma_{ij}^T S_{i,j+1} \Gamma_{ij} + R_{ij}^{'} \right)^{-1} \left(\Gamma_{ij}^T S_{i,j+1} \Phi_{i,j} + M_{ij}^T \right) \qquad (7)$$

and $S_{i,t}$ is obtained from the recurrent equation:

$$S_{i, m_i \frac{H_i}{kT_i}} = Q_{i0}$$

$$S_{i,j} = \Phi_{ij}^T S_{i,j+1} \Phi_{ij} + Q_{ij}^{'} \qquad (8)$$

$$- \left(\Gamma_{ij}^T S_{i,j+1} \Phi_{ij} + M_{ij}^T \right)^T \left(\Gamma_{ij}^T S_{i,j+1} \Gamma_{ij} + R_{ij}^{'} \right)^{-1} \left(\Gamma_{ij}^T S_{i,j+1} \Phi_{ij} + M_{ij}^T \right)$$

Because of the periodicity of Φ_{ij} and Γ_{ij}, the steady state solution of the Riccati equation (7) is periodic with period m_i [2], i.e., $S_{i,j} = S_{i,j+m_i}$. The controller gain matrix $L_{i,j}$ is designed using the steady-state solution of Riccati equation (7) and its solution is also periodic:

$$L_{i,j} = L_{i,j+mi}$$

Lemma 3. The minimal value of J_i under task instance classification strategy (1) is given as:

$$J_i(H_i, m_i, k_i) = \frac{1}{\left\lceil \frac{H_i m_i}{T_i k_i} \right\rceil T_i} \left(x_{i,0}^T S_{i,0} x_{i,0} + \sum_{j=0}^{\left\lceil \frac{H_i m_i}{T_i k_i} \right\rceil - 1} \left(tr S_{i,j+1} R_{Vi} + \overline{J}_{ij} \right) \right) \qquad (9)$$

where $\overline{J}_{i,j} = tr\left(Q \int_0^{h_{ij}} R_{ic}(\tau) d\tau \right)$

Proof. The minimal value of J_j given by optimal LQ controller (6) is derived in [1] as

$$J_i = \frac{1}{N_i T_i} \left(x_{i,0}^T S_{i,0} x_{i,0} + \sum_{j=0}^{N_i - 1} \left(tr S_{i,j+1} R_{Vij} + \overline{J}_{ij} \right) \right)$$

where N_i is the number of control law updates. As the control law is only updated by a mandatory instance, the number of control law updates is therefore equal to the number of mandatory instances. Lemma 2 shows that there are exactly $\lceil H_i m_i / (T_i k_i) \rceil$ mandatory instances before instant H_i, we therefore get $N_i = \lceil H_i m_i / (T_i k_i) \rceil$.

When time goes to infinity ($\lim H_i \to \infty$), the influence from the initial condition decreases and because $S_{i,j} = S_{i,j+m_i}$, (9) may be written as:

$$J_i(\infty, m_i, k_i) = \frac{1}{m_i T_i} \left(\sum_{j=0}^{m_i - 1} tr S_{i,j+1} R_{Vij} + \sum_{j=0}^{m_i - 1} \overline{J}_{ij} \right) \qquad (10)$$

According to [8], the instance sequence is chosen as uniformly as possible. For that, during each interval $k_i\, T_i$ the varying model (2) will update m_i times, and the period h_{ij} assumes a value in the set $\Psi_i = \{f_{i,0} T_i, f_{i,1} T_i, \ldots, f_{i,mi-1} T_i\}$ (j is replaced by $f_{i,p}$ obtained as in [8]). For example, if k_i/m_i is an integer, the optimal sequence in [8] gives $f_{i,0} = f_{i,1} = \ldots = f_{mi-1} = k_i/m_i$.

3.3. Closed-loop performance

From digital control theory, we know that if we reduce the sampling time of a plant, the open-loop performance will be closer to the continuous-time system one. So, a similar behavior for the closed-loop performance is expected (when the sampling period grows, the plant response becomes worse than the continuous time one, forcing performance degradation).

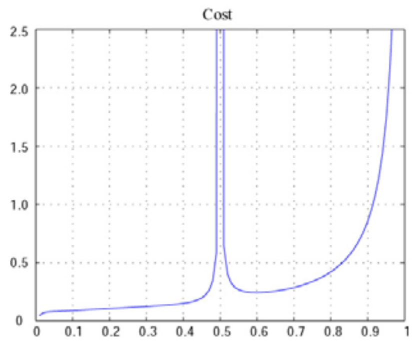


Figure 3: Cost vs. sampling time. Pendulum.

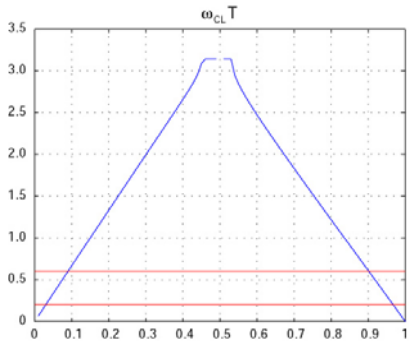


Figure 4: ω_{CL} T vs. sampling time. Rule of thumb.

But fig. 4 in [5] shows the closed-loop cost as function of the sampling time for a given plant (pendulum), where the controller parameters are obtained with an infinite time horizon LQ design. Surprisingly, in this figure, some high peaks appear for some specifics sampling periods. Then, the classical conception mentioned before is contradicted, i.e. faster sampling not necessarily increases control performance. To highlight this unexpected conclusion, we made the same analysis as in [5] (fig. 3), and we found that the first peak occurred when the sampling period is $T = \pi/\omega_A$, where ω_A is the open-loop frequency of the plant. The reason for this peak is that the sampling theorem is not respected for the open-loop plant (plant resonance), that is unacceptable. Even, as we show in fig. 4, one can satisfy the mentioned “rule of thumb” inside the section limited at top and at bottom by both red straight lines.

Summarizing, to select the maximum sampling period (k_iT_i) we add to the selection criteria presented in section 3.1, the verification of the sampling theorem for the *open-loop* plant. In this more restricted area, the classical digital control conception remains valid.

3.4. Closed-loop Stability

A change in the value of m_i, for a given k_i, produces a sampling period variation, and then we consider a Discrete Time Switched System (DTSS) description. To adapt control law parameters to this variation, we use the design presented in section 3.2. But, as it was shown in [15], controllers designed with optimal-LQ techniques, may suffer from instability under certain switching sequences, i.e. when the sampling period changes. Due to this undesirable result, [15] adopts a linear matrix inequalities (LMI) framework to design stable optimal controllers.

We will use a LMI framework to find a Common Quadratic Lyapunov Function (CQLF), then, asymptotic stability is guaranteed for any (*m,k*)-firm sequence proving the stability of the control. Firstly, we consider the set of m_i controller parameters to be calculated for each possible value of m_i $£_i=\{ L^0_{m_i}, L^1_{m_i}, \cdots, L^{m_i-1}_{m_i} \}$, by using equation (7), where $j=f_d$ (f_d depends on the delivery sequence [8]) and $d=0,1\ldots,m_i-1$, i.e. for the set Ψ_i. Secondly, we consider the set of open-loop discrete time models (2), $\Theta_i=\{ (\Phi_{i1},\Gamma_{i1}),(\Phi_{i2},\Gamma_{i2}),\cdots,(\Phi_{ik_i},\Gamma_{ik_i}) \}$ and evaluate the k_i periods, taking into account possible interruptions in a planned sequence at any time.

By using elements in both sets, we can establish a new set of $m_i{\cdot}k_i$ closed-loop models, (3) without noise, $A^i_{l,d} = \Phi_{il} + \Gamma_{il} L^d_{m_i}$, where l varies between 1 and k_i ($(\Phi_{il},\Gamma_{il}) \in \Theta_i$) and d between 0 and m_i-1 ($L^d_i \in £_i$).

In order to prove the stability of DTSS [18], we should find a CQLF for the set of matrices $A^i_{n,d}$, where $n = 1,..,k_i$, with $d = 0,..,m_{i-1}$. Then, we formulate a set of $m_i{\cdot}k_i$ inequalities: $\left(A^i_{n,d}\right)^T P A^i_{n,d} - P < 0, \forall n\, \forall d$, and for this set, we propose to use the LMI toolbox from Matlab in order to find the common matrix $P=P^T>0$.

4. Scheduling architecture and supervision

As we stated above, during the execution of the application, we distinguish 3 situations of states of controller plants: steady state, transient state and not activated plant. This information is provided by the supervision task. Section 4.1 describes how plant states are identified. A new system state, determined by the supervision task, requires the definition of new scheduling parameters by the task handler (section 4.2).

4.1. Plant State Detection

In this section, we specify how the supervision component identifies the situation of states of controlled plants. We consider the situation 1- for *non activated* plant and two situations for an activated one: 2- *Steady state* (or near) and 3- *Transient state*. Situation 1 is used when the plant does not exist for the overall system (plant controlled only during certain time interval, plant deactivated because its output is out of a given domain). Reaching or leaving situation 1 for a plant modifies the value of *n*.

The deadband approach presented in [12] is used to distinguished situation 2 and 3. Each controlled plant has a state, which asymptotically tracks the reference r, which is supervised by the supervision task. Let y_1 be this plant state. The following condition is set up:

$$\left| y_1(h_1 + n\,h_1) - y_1(n\,h_i) \right| < \min\left\{ \delta \left| y_1(n\,h_i) \right|, th \right\}$$

where *th* is a threshold to prevent false identifications due to noise. h_i is the detection period of the supervision component for the plant *i*. We proposed to select this time equal to the i^{th} sampling period. If this condition is verified, the plant is considered as in steady state, otherwise, it is in transient state. This plant state detection mechanism has as advantages that it depends on the actual evolution and it detects, in the same way, reference changes or/and non-modeled perturbations.

4.2. Task Handler

We implement a task handler which at each situation change, adjusts the (*m,k*)-firm constraint for each task by considering the current control performance indicator and the task schedulability objective. In this section, we formulate the scheduling problem and give the solution. After, the scheduling architecture is presented.

We suppose that the value k_i of each τ_i has been carefully chosen (section 3.1) and is constant during the execution of application. The value of m_i is chosen in [1 .. k_i] on line by the task handler. For each control task τ_i, each possible value of m_i is associated with two values g_{ij} and g_{ij}' corresponding to the control performance, respectively, in transient situation and in steady one. Suppose that a lower value of g_{ij} or g_{ij}' represents a better control performance, the aim of the task handler at a change in situation of plant states is to find, for each τ_i, a value $m_i \in$ [1.. k_i] so that the sum of g_{ij} or g_{ij}' (according to the situation which the plants fall into) for $j\in$ [1.. k_i] and $i\in$[1.. n] is minimized subject to the task schedulability. Then, condition in Theorem 1 is modified taking into account the mentioned performance indicators. This is formally formulated as the following optimization problem:

To determine the sequence $s_{i1}, s_{i2}, .., s_{il_i}$ for each task τ_i, $i=1,..,n$ that minimizes $\sum_{i=1}^{n}\sum_{j=1}^{l_i}\left(s_{ij} g_{ij} I + s_{ij} g'_{ij} F\right)$ (11)

with $s_{ij} \in \{0,1\}, \sum_{j=1}^{l_i} s_{ij} = 1,\ i = 1,..,n,\ j = 1,..,l_i, l_i = 1,..,k_i$

$I, F \in \{0,1\}, I + F = 1$

and such that

$$C_i + \sum_{j=1}^{i-1}\left\lceil \sum_{p=1}^{l_i} s_{jp} m_{jp} \Big/ k_j \left\lceil T_i/T_j \right\rceil \right\rceil C_j \le T_i,\ i=1,..,n,$$

where I and F are situation indicators: in transient state, I is 1 and F is 0, and in steady state, I is 0 and F is 1.

The values g_{ij} and g_{ij}' are given in two way: one is using directly the cost (8) and (10), and the other way is to take the relative performance degradation percentage:

$$\frac{J(H,m_{ij},k_i)-J(H,k_i,k_i)}{J(H,k_i,k_i)} \text{ and } \frac{J(\infty,m_{ij},k_i)-J(\infty,k_i,k_i)}{J(\infty,k_i,k_i)} \quad (12)$$

Using the first control performance representation method, the optimization problem is to minimize the overall cost of the application. However, the sub-systems with lower costs may suffer from great control performance degradation due to a low value of m_i. That is, the task handler maintains the value of each m_i as great as possible for the sub-systems with greater costs by reducing the value of m_i for the sub-systems with lower costs. Using (12) as the control performance criteria avoids such a situation. The control performance degradation of each sub-system is treated equally. On the other hand, the overall cost of application may not be optimal. So, the choice of control performance representation should be identified according to the application requirements.

The time horizon H_i for the finite-horizon cost functions is an important design parameter, which directly affects the overall control performance, and need to be carefully chosen. Here, we choose H_i as the settling time (approx. three times the rise time). To calculate (12) in an off-line form, we considered the typical values of the reference values neglecting noise.

The optimization problem (11) has the similar form as that in [9] which was qualified as the multiple-choice multi-dimension knapsack problem (MMKP) [17], and was proved to be NP-hard problem. For solving the optimization problem, the heuristic algorithm based on a so-called computationally cheaper heuristic algorithm (HEU) proposed in [10] is used. It has been shown that the algorithm is efficient and suitable for an on-line use for real-time application.

At each change in the situation of at least one plant, the task handler receives the information about the current plants states. Based on this information (n current activated plants), and the values of v_{ij} that were evaluated off-line, it deduces the new (m,k)-firm constraint for each control task by solving the optimization problem (11). The control tasks are then scheduled according to these (m,k)-firm constraints.

5. Case study

In this section, we illustrate the scheduling approach presented above by studying the control of four plants. *Plant*$_1$ (resp. *Plant*$_2$, *Plant*$_3$, *Plant*$_4$) corresponds to a harmonic oscillator system, (resp. to a cart system, a pendulum and an inverted pendulum).

5.1. Plants and Controllers

Each plant is modeled by the differential equation (2):

Plant$_1$: $A=\begin{bmatrix}0 & 1\\ -18 & 0\end{bmatrix}$, $B=\begin{bmatrix}0\\ 516\end{bmatrix}$ and v_c has the incremental covariance: $R_{1c}=\begin{bmatrix}0.0025 & -0.005\\ -0.005 & 0.01\end{bmatrix}$;

Plant$_2$: $A=\begin{bmatrix}0 & 1\\ 0 & -12.6558\end{bmatrix}$, $B=\begin{bmatrix}0\\ 1.9243\end{bmatrix}$ and v_c has the incremental covariance $R_{1c}=\begin{bmatrix}0.005625 & -0.075\\ -0.075 & 1\end{bmatrix}$;

Plant$_3$: $A=\begin{bmatrix}0 & 1\\ -22.206 & -0.9424\end{bmatrix}$, $B=\begin{bmatrix}0\\ 0.48036\end{bmatrix}$ and v_c has the incremental covariance $R_{3c}=\begin{bmatrix}0 & 0\\ 0 & 22.2066\end{bmatrix}$;

Plant$_4$: $A=\begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & -14 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 28 & 0\end{bmatrix}$, $B=\begin{bmatrix}0\\ 2\\ 0\\ 2\end{bmatrix}$ and v_c has the incremental covariance $R_{1c}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0.0025 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$.

The controller of *Plant*$_i$ is denoted *Controller*$_i$.

Rise time specifications T_{rM} of each plant are respectively 0.2, 0.2, 0.3 and 0.5. Then, sampling periods are related to rise time specifications, i.e., 0.02s for *Plant*$_1$, 0.02s for *Plant*$_2$, 0.03s for *Plant*$_3$, and 0.05s for *Plant*$_4$.

The first state variable in vector x of each plant is the variable supervised by the supervision component, in other words, the controller tries to keep it asymptotically tracking the plant state reference. The step response target for the cart is an overdamped response, while for the others they are underdamped ones, being the damping coefficient upper than 0.6 (overshoot < 10%).

The controllers are designed using (8) for each m_i value. The design weights, which allow the satisfaction of the mentioned rise time and overshoot, are:

Controller$_1$: $Q_1=\begin{bmatrix}5 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 25\end{bmatrix}$, R_1=200

Controller$_2$: $Q_2=\begin{bmatrix}1.25 & 0\\ 0 & 0.0085\end{bmatrix}$, R_2=0.0001.

Controller$_3$: $Q_3=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$, R_3=0.00001.

Controller$_4$: $Q_4=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 2 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$, R_4=0.001.

Using the LMI control toolbox, the set of inequalities (for the overall set of discrete plants and controllers parameters), has a QCLF, guarantying stability.

To allow for fast changes between different (m,k)-firm constraints at an m adjustment, the controller parameters are calculated off-line and stored in a table.

Since the control tasks are assigned the rate-monotonic priority, the task with the largest period has the lowest priority; its execution has no influence on the other tasks. Therefore, no task instance classification will be applied to *Controller$_4$*, or in other words, it is executed under (*k,k*)-firm constraint.

Using the approach proposed in section 4, the value of k_i is set to respectively 6, 5, 5 and 4. The value of m for the plants may vary within [1.. k_i].

5.2. Performance specification and simulation setup

Table 1a gives, for *Controller$_1$* the optimal costs associated with different (*m,k*)-firm constraint. The column labeled "OC_IH/Degradation (%)" lists the optimal infinite-horizon costs and the relative performance degradation compared to (*k,k*)-firm constraint. The column "OC_FH / Degradation(%)" lists the optimal finite-horizon costs for typical initial plant states and the relative performance degradation compared to (*k,k*)-firm constraint. The time interval of the finite-horizon cost functions is set equal the settling time for each system (see section 4.2).

In practice, the task handler can calculate the optimal finite-horizon costs on-line. They are listed here to illustrate how task handler chooses the (*m,k*)-firm constraints for each controller.

(*m,k*)-firm constraint	OC_IH/ Degradation(%)	OC_FH/Degradation(%)
(1,6)	0.0022639 / 18.05	0.2837 / 9.83
(2,6)	0.0020682 / 7.85	0.2645 / 2.4
(3,6)	0.0019941 / 3.99	0.26 / 0.66
(4,6)	0.0019682 / 2.64	0.2597 / 0.54
(5,6)	0.0019428 / 1.32	0.2587 / 0.15
(6,6)	0.0019175 / 0	0.2583 / 0

Table 1. Optimal costs associated with different (*m,k*)-firm constraints (controller 1)

The simulation model was created using Simulink and the TrueTime toolbox [4]. The execution time of each controller is approximately 9ms.

5.3. Simulation results

The simulation results obtained with the proposed approach are given in this section.

The model has been evaluated under the setup shown in Table 2. There, we identify the state (st) of each plant as follow: -1→Not activated, 0 → Steady-state and 1→ Transient-state. State changes (or transitions) are detected by the supervision component and, in the proposed simulation; they arrived at time « Time-Event ». Therefore, the task handler at the "Time-event", based on the overall system state, calculates the value of each m_i.

At the system starting (time 0^-) only Plant$_1$ and Plant$_3$ are activated (both in steady state situation). At time 0+, the task associated with Plant$_3$ is activated, and then the task handler is executed in order to admit the new task (Plant$_1$ -> state transition from -1 to 0). The schedulability condition at time 0^- {0.009<0.02, 0.018<0.02} allows the system to accept all the instances, i.e. $m_1 = k_1$ and $m_3 = k_3$. At time 0^+, with $m_i = k_i$, schedulability conditions are {0.009<0.02, 0.018<0.02, **0.063>0.05**}, i.e. the third condition is not verified, justifying the execution of the task handler and the changes in the values of each m_i.

At time 0.27, the transient state of Plant$_1$ is detected by the supervision component and the task handler is executed, then the values of all m_i are adjusted accordingly. Clearly m_1 is augmented with respect to its previous value to provide better transient performance of Plant$_1$. This choice of (*m,k*)-firm constraints is done verifying the condition proposed in section 4, whose goal is to reduce the overall performance degradation. Other choice makes either the tasks non-schedulable or a worse overall performance degradation.

At time 0.5, Plant$_2$ is activated, and then a new configuration is required to manage 4 Plants. In this case the schedulability condition {0.009<0.02, 0.018<0.02, **0.045> 0.03**, **0.081>0.05**}, i.e. if we do not reduce the values of m_i, the tasks will be non-schedulable.

The extinction of the Plant$_1$ transient, at time 0.62, produces a new set of m_i values. In the same way, the detection of the Plant$_2$ transient at 0.8 requires the adjustment of the m_i values. See Table 2.

At time 1.76, an inadmissible perturbation enters in the Plant$_3$, the output reaches π/2, and consequently this plant is deactivated reducing the number of tasks to 3. Then, the task handler can augment m_1 and m_2 values using condition in section 3. Both values cannot assume their maximum, k_1 and k_2, because the schedulability conditions for the 3 systems are {0.009<0.02, **0.027>0.02**, **0.045>0.03**}. Finally, at time 1.8 the detection of the Plant$_2$ transient forces to augment m_2.

Time event	0(-)		0 (+)		0.27		0.5		0.62	
	st	m_i	st	m_i	st	m_i	st	m_i	st	m_i
Plant$_1$	0	6	0	4	**1**	6	1	3	**0**	2
Plant$_2$	-1	0	-1	0	-1	0	**0**	1	0	1
Plant$_3$	0	5	0	5	0	2	0	2	0	5
Plant$_4$	-1	0	**0**	4	1	4	1	4	1	4
Time event	0.8		1.05		1.1		1.76		1.8	
	st	m_i	st	m_i	st	m_i	st	m_i	st	m_i
Plant$_1$	0	2	0	2	0	2	0	6	0	2
Plant$_2$	1	2	1	1	0	1	0	1	1	5
Plant$_3$	0	2	**1**	5	1	5	-1	0	-1	0
Plant$_4$	1	4	-1	4	0	4	1	4	1	4

Table 2. Simulation setup, m_i values vs Event.

Note that as the (*m,k*)-firm constraint of *Controller$_4$* is hold as (*k,k*) throughout the execution of application, the change in the state of the *Plant$_4$* has not effect on the decision of task handler.

Throughout the simulation, the rise-time requirements of plants are all met thanks to a judicious selection of k_i.

The minimum delay is 9ms, i.e. the computation time.

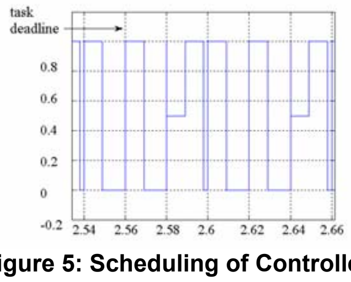
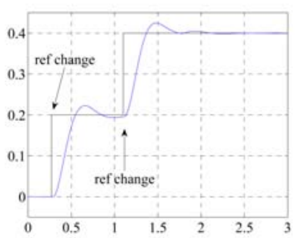
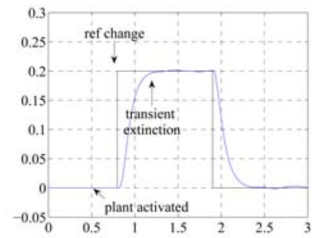


Figure 5: Scheduling of Controller2.

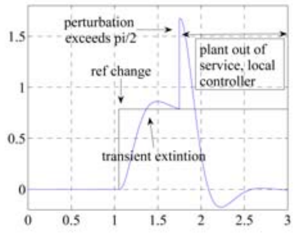
In fig. 5, we show the scheduling of Controller2. The three possible levels are 1–Running (task being executed by the processor), 0.5–Preempted (the execution is preempted by other task); and 0–Waiting (task waits for an activation). We could verify that each task deadline is always respected (it is guaranteed by the *event-triggered* execution of the task handler). Evolution of the plant outputs is given in fig.6.



a) Harmonic oscillator system



b) Cart system



c) Pendulum.

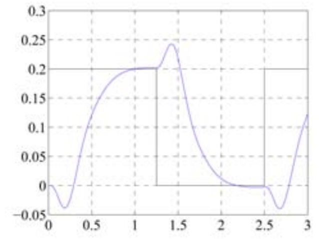

d) Inverted pendulum

Figure 6: Output evolution of Plants.

In order to analyze the performance degradation, we evaluate the LQ cost for each system during the simulation time. Considering that we have a dedicated CPU for each system, we calculated the nominal performance of each plant (Table 3) as the reference values. The values of *Performance overall system* were evaluated considering the simulation setup described before. These results depend on the simulation setup, and they are only exposed to show that using the proposed technique, the degradation of the performance should be maintained as small as possible in each situation subject to the task schedulability. Plant$_4$ suffered the lower cost degradation; due to the m_4 is always equal 4. Plant$_2$, suffered the maximum cost degradation, due to Plant$_2$ performance indicators, which generates the reduction of m_2 if the other plants require the use of CPU.

	Plant$_1$	Plant$_2$	Plant$_3$	Plant$_4$
Perf. Nominal	359	40.04	154.6	25.31
Perf. Overall System	395	52.5	199.4	26.347
Degradation	10%	31.1%	29.1 %	0.05 %

Table 3. Performance costs of *Plant$_i$*.

6. Conclusion

This paper has presented a scheduling approach based on the (*m,k*)-firm constraint model for scheduling a set of control tasks. Given the current states of the controlled plants, the proposed approach derives a (*m,k*)-firm constraint for each control task, and the control tasks are scheduled using these (*m,k*)-firm constraints so that the schedulability of control tasks is guaranteed and the overall control performance is maintained at a high level.

Compared with feedback scheduling approaches in the literature, the advantages of the proposed approach are: the approach does not depend on the type and property of the control performance (whatever the functions of the control performance are convex, the proposed approach can always keep the overall control performance at high level while guarantying the schedulability of control tasks); at a system configuration change, the *event-triggered* solution reacts immediately reducing the periods of the tasks.

7. References

[1] Åström, K. J. and B. Wittenmark, Computer-Controlled Systems, third edition. Prentice Hall. 1997.

[2] Bittanti, S. Bittanti, P. Colaneri, G. De Nicolao, The periodic Riccati equation, in The Riccati Equation, Springer-Verlag, Berlin, 1991.

[3] Cervin, A., Eker, J., Bernhardsson. B., and Årzén, K.-E., "Feedback feedforward scheduling of control tasks" *Real-Time System.*, vol. 23, no. 1-2, pp. 25--53, 2002.

[4] Cervin, A., Henriksson, D., Lincoln, B., Eker, J., d Årzén, K.-E., "How does control timing affect performance" *IEEE Control Systems Magazine*, 23:3, pp. 16-30, 2003

[5] Eker, J., Hagander, P., and Årzén, K.E., "A Feedback Scheduler for Real-Time Controller Tasks", *Control Engineering Practice*, vol. 12, no.8, p. 1369-1378, 2000.

[6] Henriksson, D., Cervin, A., Optimal On-line Sampling Period Assignment for Real-Time Control Tasks Based on Plant State Information, *ECC 2005*, Spain, Dec. 2005.

[7] Jia, N., Hyon, E., Song, Y.Q., "Ordonnancement sous contraintes (*m,k*)-firm et combinatoire des mots", RTS'2005, Paris, France, 2005.

[8] Jia. N., Song. Y.Q, Simonot-Lion. F., "Graceful Degradation of the Quality of Control through Data Drop Policy", *ECC'07*, Kos, Greece, July 2007.

[9] Jia. N., Song. Y.Q, Simonot-Lion. F., "Feedback scheduling based on (*m,k*)-firm constraint model for the handling of a set of real-time controllers", *RTNS'07*, France, March 2007.

[10]Khan, S., LI, K.F., Manning, E.G. and Akbar, M. "Solving the knapsack problem for adaptive multimedia systems", Studia Informatica Universalis, Vol. 2 (1), 157-178, 2002.

[11]Lothaire., M. "Algebraic Combinatorics on Words", *Cambridge University Press*, 2002.

[12] Otanez P., Moyne J., y Tilbury D., "Using Deadbands to Reduce communication in Networked Control Systems," *American Contr. Conf.* (2002).

[13] Quan, G., and Hu, X., "Enhanced Fixed-priority Scheduling with (*m,k*)-*firm* Guarantee ", *Proc. Of 21st IEEE Real-Time Systems Symposium*, 79-88, USA, 2000.

[14] Ramanathan, P., "Overload management in Real-Time control applications using (*m,k*)-firm guarantee", *IEEE Trans. on Parallel and Distr. Syst.*, 10(6), 549-559, 1999.

[15] Schinkel M., W.-H Chen, A. Rantzer. "Optimal control for systems with varying sampling rate". ACC 2002.

[16]Seto, D., Lehoczkyn, J. P., Sha, L., Shin, K. G., "On task schedulability in real-time control systems", *Proc. of 17th IEEE Real-Time Syst.*, 13-21, USA, 1996.

[17] Silvano Martello, Paolo Toth (1990). "Knapsack Problems: Algorithms and Computer Implementations", John Wiley & Sons. ISBN 0-471-92420-2, 1990.

[18] Theys J. "Joint Spectral Radius: theory and approximations", PhD Thesis. Univ. de Louvain. 2005.