Rate Monotonic Analysis: the Hyperbolic Bound

Enrico Bini, Giorgio C. Buttazzo, Giuseppe M. Buttazzo

Abstract— **In this paper we propose a novel schedulability analysis for verifying the feasibility of large periodic task sets under the rate monotonic algorithm, when the exact test cannot be applied on line due to prohibitively long execution times. The proposed test has the same complexity as the original Liu and Layland bound but it is less pessimistic, so allowing to accept task sets that would be rejected using the original approach. The performance of the proposed approach is evaluated with respect to the classical Liu and Layland method, and theoretical bounds are derived as a function of n (the number of tasks) and for the limit case of n tending to infinity. The analysis is also extended to include aperiodic servers and blocking times due to concurrency control protocols. Extensive simulations on synthetic tasks sets are presented to compare the effectiveness of the proposed test with respect to the Liu and Layland method and the exact response time analysis.**

Keywords— **Rate-Monotonic analysis, periodic scheduling, schedulability test.**

I. Introduction

DURING the last thirty years periodic task scheduling received much consideration in the real-time research community due to the large number of control applications using cyclical activities. Since a few years ago, the most critical control applications were developed using an off-line table-driven approach (timeline scheduling), according to which the time line is divided into slots of fixed length (*minor cycle*) and tasks are statically allocated in each slot based on their rates and execution requirements [1]. The schedule is then constructed up to the least common multiple of all the periods (called the *hyperperiod* or the *major cycle*) and stored in a table. At runtime, tasks are dispatched according to the table and synchronized by a timer at the beginning of each minor cycle. From one hand, timeline scheduling is straightforward to implement and does not introduce significant runtime overhead (since scheduling decisions are taken off-line). Moreover, tasks always execute in their preallocated slots, so the experienced jitter is very small.

On the other hand, timeline scheduling is fragile during overload situations, since a task exceeding its predicted execution time could generate (if not aborted) a domino effect on the subsequent tasks, causing their execution to exceed the minor cycle boundary (timeline break). In addition, timeline scheduling is not flexible enough for handling dynamic situations. In fact, a creation of a new task, or a little change in a task rate, might modify the values of the minor and major cycles, thus requiring a complete redesign of the scheduling table.

Scuola Superiore S. Anna in Pisa, Italy. Email: e.bini@sssup.it
University of Pavia, Italy. INFM Research Unit. Email: buttazzo@unipv.it
Department of Mathematics University of Pisa, Italy. Email: buttazzo@dm.unipi.it

Such problems can be solved by using a priority-based approach, according to which each task is assigned a priority (which can be fixed or dynamic) and the schedule is generated on line based on the current priority value. In 1973, Liu and Layland [2] analyzed the properties of two basic priority assignment rules: the Rate Monotonic (RM) algorithm (according to which priorities are inversely proportional to task periods) and the Earliest Deadline First (EDF) algorithm (according to which priorities are inversely proportional to absolute deadlines). Their major contribution was to derive two simple guarantee tests to verify the schedulability of a periodic task set under both algorithms.

Their results refer to the following task model. Each periodic task τ_i consists of an infinite sequence of jobs $\tau_{i,k}$ $(k = 1, 2, \ldots)$, where the first job $\tau_{i,1}$ is released at time $r_{i,1} = \Phi_i$ (the task phase) and the generic k^{th} job $\tau_{i,k}$ is released at time $r_{i,k} = \Phi_i + (k-1)\,T_i$, where T_i is the task period. Each job is characterized by a worst-case execution time C_i, a relative deadline D_i and an absolute deadline $d_{i,k} = r_{i,k} + D_i$. The ratio $U_i = C_i/T_i$ is called the *utilization factor* of task τ_i and represents the fraction of processor time used by that task. Finally, the value

$$U_p = \sum_{i=1}^{n} U_i$$

is called the *total processor utilization factor* and represents the fraction of processor time used by the periodic task set. Clearly, if $U_p > 1$ no feasible schedule exists for the task set.

The two schedulability conditions for RM and EDF are derived for a set Γ of n periodic tasks under the assumptions that all tasks start simultaneously at time $t = 0$ (that is, $\Phi_i = 0$ for all $i = 1, \ldots, n$), relative deadlines are equal to periods (that is, $d_{i,k} = k\,T_i$) and tasks are independent (that is, they do not have resource constraints, nor precedence relations). Under such assumptions, a set of n periodic tasks is schedulable by the RM algorithm if

$$\sum_{i=1}^{n} U_i \leq n\,(2^{1/n} - 1). \tag{1}$$

Throughout the paper, we will refer to the previous schedulability condition as the LL-test. We recall that

$$\lim_{n\to\infty} n\,(2^{1/n} - 1) = \ln 2 \simeq 0.69.$$

Under the EDF algorithm, a set of n periodic tasks is schedulable if and only if

$$\sum_{i=1}^{n} U_i \leq 1. \tag{2}$$

Since the work presented by Liu and Layland, a lot of work has been done on periodic tasks, to improve the schedulability bound of the RM algorithm or relax some restrictive assumption on the task set. In [3], Lehoczky, Sha, and Ding performed a statistical study and showed that for task sets with randomly generated parameters the LL-test is able to guarantee schedulability up to a processor utilization of about 88%. Exact schedulability tests for RM yielding to necessary and sufficient conditions have been independently derived in [4], [3], [5]. Using the Response Time Analysis (RTA) proposed in [5], a periodic task set is schedulable with the RM algorithm if and only if the worst-case response time of each task is less than or equal to its deadline. The worst-case response time R_i of a task can be computed using the following iterative formula:

$$\begin{cases} R_i^{(0)} = C_i \\ R_i^{(k)} = C_i + \sum_{j:D_j<D_i} \left\lceil \frac{R_i^{(k-1)}}{T_j} \right\rceil C_j. \end{cases} \quad (3)$$

where the worst-case response time of task τ_i is given by the smallest value of $R_i^{(k)}$ such that $R_i^{(k)} = R_i^{(k-1)}$. It is worth noting, however, that the complexity of the exact test is pseudo-polynomial, thus it is not suited to be used for on-line admission control in applications with large task sets.

The Rate Monotonic algorithm is probably the most used priority assignment in real-time applications, because it is very easy to implement on top of commercial kernels, which do not support explicit timing constraints on the task set. Indeed, a RM scheduler can be implemented just by assigning each task a fixed priority level inversely proportional to its period. On the other hand, implementing a dynamic scheme, like EDF, on top of a priority-based kernel would require to keep track of all absolute deadlines and perform a dynamic mapping between absolute deadlines and priorities. Such a mapping should be done every time a new task receives a priority falling between two adjacent priority levels. Such an additional implementation complexity and runtime overhead, due to dynamic priority management, often prevents EDF to be implemented on top of commercial real-time kernels, even though it would increase the total processor utilization.

Using a fixed priority scheme, like RM, however, does not prevent developing real-time applications with dynamic task activation. In fact, each task would still have a fixed priority level (assigned according to RM), but would be activated upon the arrival of an external event.

Consider, for example, a radar tracking application in an environment where there are a number of moving targets, having different but constant and known speed. In this case, using a RM priority assignment, the priority of a task is fixed, but proportional to the target speed. In this scenario, to optimize the available resources, a task is activated when a new target enters the monitored environment and killed when the target goes out of the visual field. In such a system, an on-line acceptance test needs to be executed at each task activation to guarantee that all active tasks will be scheduled within their given period. The failure of the acceptance test can be used to allocate additional computational resources (if available), to generate an alarm, if the system is not able to handle the overload, or to take a proper recovery action.

If the number of active tasks is high, the exact schedulability test provided by the response time analysis could require too much time to be executed on line, whereas the classical Liu and Layland test could fail on a feasible task set. The graph reported in Figure 1 shows the number of steps in the innermost loop of the RTA and LL tests as a function of the number of tasks (for the RTA test, the maximum and the average number of steps are reported because they are significantly different). In this simulation the periods are uniformly distributed in the interval $[T_{min}, T_{max}]$, where $T_{min} = 10$ and $T_{max} = 10000$ units of time, and the WCET, C_i, are uniformly distributed in $[0, T_i]$. For example, in an application with 50 tasks, where each step takes 10 microseconds to run, the RTA acceptance test would take 0.1 seconds, whereas the LL-test would run in 500 microseconds. It is worth noting that to accept a new task, when the current set has been already guaranteed, the LL-test can actually be performed in one step only, by summing the utilization of the incoming task to the total processor utilization previously computed. Conversely, in the same situation, the complexity of the RTA-test remains the same.

In the context described above, the acceptance test proposed in this paper, having the same O(n) complexity as the Liu and Layland test, can be efficiently executed on line to accept more tasks in the system and increase the exploitation of computational resources.

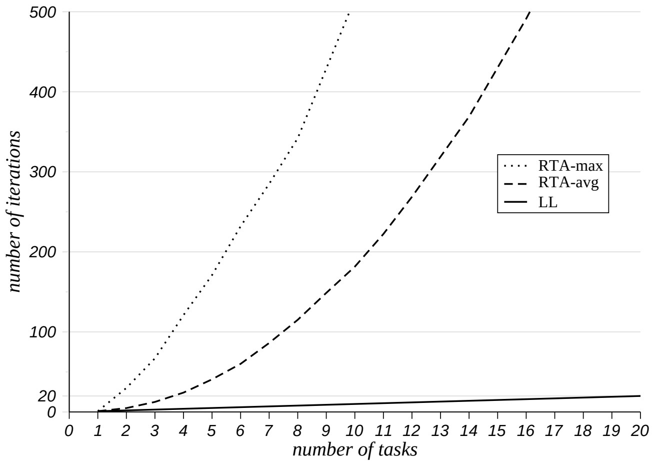


Fig. 1. Comparison between the RTA and the LL test in terms of complexity.

In [6], Sjödin and Hansson proposed an improved version of the RTA test which runs more efficiently in the average case, but has still pseudo-polynomial complexity in the worst-case. In [7], Sha, Rajkumar and Lehozcky extended the rate monotonic analysis in the presence of resource constraints, where access to resources is performed using concurrency control protocols, such as the Priority Inheritance

Protocol and the Priority Ceiling Protocol. In [5], Audsley et al. generalized the response time analysis including resource constraints.

A method similar to the one described here, has been independently developed by Oh and Song in [8] to verify the schedulability of periodic task sets in a multiprocessor environment. The authors, however, do not compare the effectiveness of the method against other classical approaches and the analysis is not extended to deal with resource constraints and aperiodic servers. The basic guarantee test has been also described in [9] in the context of uniprocessor scheduling, but without any performance characterization and comparison.

The present paper extends and integrates the results presented by the same authors in [10] and provides the following contributions:

- we present an efficient test for verifying the feasibility of large periodic task sets with fixed priorities in dynamic environments, when the response time analysis cannot be applied on line due to prohibitively long execution times. The proposed test has the same complexity as the original Liu and Layland bound but it is less pessimistic, so allowing to accept task sets that would be rejected using the original approach.
- we prove that the proposed test is thight, in the sense that it is the best possible bound that can be found given the same information on the task set (i.e., the individual task utilization factors).
- we analytically derive the asymptotic behavior of the bound, as the number of tasks tends to infinity.
- the performance of the test in terms of acceptance ratio is compared with respect to EDF, RTA and the LL-test.
- the schedulability test is extended to consider more realistic cases concerning task sets with shared resources and aperiodic servers.

The rest of the paper is organized as follows. Section II presents the hyperbolic feasibility bound for the RM algorithm, explaining its relation with the classical Liu and Layland approach in the utilization space. Section III evaluates the theoretical improvement of the proposed test with respect to the Liu and Layland bound as a function of n (the number of tasks) and computes its asymptotic value as n tends to infinity. Section V extends the schedulability test to take aperiodic servers and resource constraints into account. Section IV presents a number of simulation experiments performed on synthetic task sets aimed at comparing the proposed approach with other classical ones as a function of the number of tasks. Section VI presents two methods for improving the hyperbolic bound for task sets with particular characteristics. Finally, Section VII states our conclusions and future work.

II. The hyperbolic bound

The schedulability test we propose in this paper is derived from the same worst-case scenario identified by Liu and Layland in [2] for a set on n periodic tasks. However, instead of minimizing the processor utilization with respect to task periods, we manipulate the feasibility condition in order to find a tighter sufficient schedulability test as a function of the individual task utilizations. It is worth noting that the Liu and Layland proof has been shown to be incorrect by Devillers and Goossens in [11]; however, the bug in the proof does not affect our result since the worst-case scenario remains valid, as also shown by Jane Liu in [9].

The following theorem provides a sufficient condition for testing the schedulability of a task set under the RM algorithm.

Theorem 1: Let $\Gamma = \{\tau_1, \ldots, \tau_n\}$ be a set of n periodic tasks, where each task τ_i is characterized by a processor utilization U_i. Then, Γ is schedulable with the RM algorithm if

$$\prod_{i=1}^{n}(U_i + 1) \leq 2. \tag{4}$$

Proof: Without loss of generality, we may assume that tasks are ordered by increasing periods, so that τ_1 is the task with the highest priority and τ_n is the task with the lowest priority. In [2], as well as in [11], it has been shown that the worst-case scenario for a set on n periodic tasks occurs when all the tasks start simultaneously (e.g., at time $t = 0$) and periods are such that

$$\forall i = 2, \ldots, n \quad T_1 < T_i < 2\,T_1.$$

Moreover, the total utilization factor is minimized when computation times have the following relations:

$$\left\{\begin{array}{lcl} C_1 & = & T_2 - T_1 \\ C_2 & = & T_3 - T_2 \\ \ldots & & \\ C_{n-1} & = & T_n - T_{n-1} \end{array}\right. \tag{5}$$

and the schedulability condition is given by:

$$\sum_{i=1}^{n} C_i \;\leq\; T_1. \tag{6}$$

Figure 2 illustrates an example of such a worst-case scenario for a sample set of 5 periodic tasks. From equations (5), the schedulability condition can also be written as

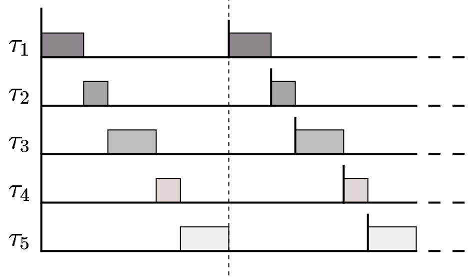


Fig. 2. Worst-case scenario for a set of 5 periodic tasks.

$$C_n \;\leq\; 2\,T_1 - T_n \tag{7}$$

which implies the schedulability of τ_n and all the other tasks in the set. Starting from such a worst-case scenario, the least upper bound U_{lub} of the processor utilization factor can be found by minimizing the expression of U_p with respect to periods. However, we show that the minimization process does not simplify the final result, but only reduces its applicability. In fact, an equally simple, but less stringent, result can be derived by manipulating equations (5) and (7), as described below. Defining:

$$R_i = \frac{T_{i+1}}{T_i} \quad \text{and} \quad U_i = \frac{C_i}{T_i}.$$

equations (5) can be written as follows:

$$\begin{cases} U_1 = R_1 - 1 \\ U_2 = R_2 - 1 \\ \ldots \\ U_{n-1} = R_{n-1} - 1. \end{cases} \tag{8}$$

Now we notice that:

$$\prod_{i=1}^{n-1} R_i = \frac{T_2}{T_1}\frac{T_3}{T_2}\cdots\frac{T_n}{T_{n-1}} = \frac{T_n}{T_1}.$$

If we divide both sides of the feasibility condition (7) by T_n, we get:

$$U_n \leq \frac{2T_1}{T_n} - 1.$$

Hence, the feasibility condition for a task set which fully utilizes the processor can be written as:

$$U_n + 1 \leq \frac{2}{\prod_{i=1}^{n-1} R_i}.$$

Since $R_i = U_i + 1$ for all $i = 1, \ldots, n-1$, we have

$$(U_n + 1)\prod_{i=1}^{n-1}(U_i + 1) \leq 2$$

and finally

$$\prod_{i=1}^{n}(U_i + 1) \leq 2,$$

which proves the theorem. ■

Notice that the Liu and Layland bounds expressed by equations (1) and (2) can easily be represented in the task utilization space, denoted as the U-space from now on. In such a space, a point $U = \{U_1, U_2, \ldots, U_n\}$ represents a periodic task set whose tasks have utilizations U_1, U_2, ..., and U_n, respectively. Notice, however, that different task sets with different period relations, but the same tasks' utilizations, are mapped on the same point.

In the U-space, the Liu and Layland bound for RM (LL bound) is represented by a n-dimensional plane which intersects each axis in $U_{\text{lub}}(n) = n\,(2^{1/n} - 1)$, whereas the EDF bound is represented by a n-dimensional plane which intersects each axis in 1. All points below the RM surface represent periodic task sets that are feasible with both RM and EDF, whereas the region above the EDF surface identifies those task sets whose total utilization is greater than one, and hence are not feasible with any algorithm. Finally, the region located between the two parallel planes of RM and EDF identifies those task sets which are schedulable by EDF but cannot be guaranteed to be schedulable by RM using condition (1).

In the U-space, the RM bound expressed by equation (4) is represented by a n-dimensional hyperbolic surface, tangent to the RM plane and having the same axis intersections as the EDF plane. For this reason, it will be referred to as the hyperbolic bound, or H-bound for short. Figure 3 illustrates such bounds for $n = 2$. Notice that the asymptots of the hyperbole are at $U_i = -1$. From the plots, we can clearly see that the feasibility region below the H-bound is larger than that below the LL-bound, and the gain is given by the dark gray area.

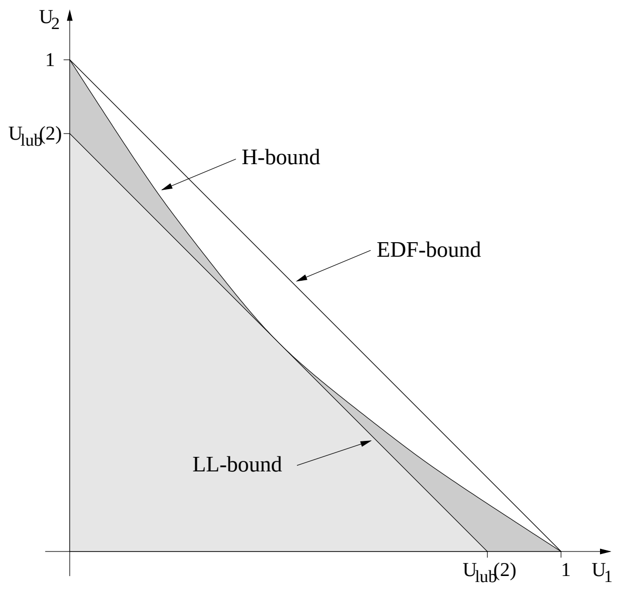


Fig. 3. Schedulability bounds for RM and EDF in the utilization space.

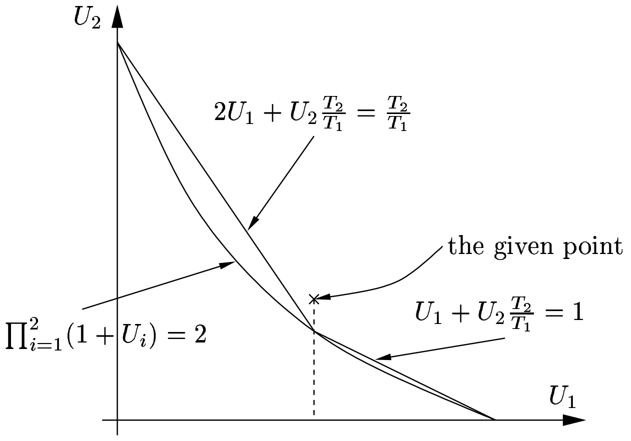
A quantitative evaluation of the gain (in terms of schedulability) achieved by the H-bound over the classical LL-bound as a function of the number of tasks is presented in Section III.

A. *Tightness of the H-bound*

In this section we show that the hyperbolic bound is tight, meaning that it is the best possible bound that can be found using the individual task utilization factors U_i as a task set knowledge. To show this property, we need to demonstrate that, for any set of utilization factors $\{U_1, U_2, \ldots, U_n\}$ which violates the hyperbolic bound, it is possible to find a task set $\Gamma = \{(T_1, C_1), (T_2, C_2), \ldots, (T_n, C_n)\}$ which is not schedulable.

The intuitive proof of the tightness for a set of two periodic tasks is illustrated in Figure 4.

Given an arbitrary set of U_i (represented by the point in the U-space) which does not satisfy condition (4), we can always find the task periods (which correspond to the linear constrains in Figure 4) such that the resulting task



U_2
$2U_1 + U_2\frac{T_2}{T_1} = \frac{T_2}{T_1}$
the given point
$U_1 + U_2\frac{T_2}{T_1} = 1$
$\prod_{i=1}^{2}(1+U_i) = 2$
U_1

Fig. 4. Tightness demonstration sketch (for $n = 2$).

set is not schedulable (that is, the point in the U-space misses the constraints).

More formally, the tightness of the hyperbolic bound is proved by the following theorem.

Theorem 2: For every set $\{U_1, U_2, \ldots, U_n\}$, $0 \leq U_i \leq 1$, such that

$$\prod_{i=1}^{n}(U_i + 1) > 2,$$

there exists a task set $\Gamma = \{\tau_1, \ldots, \tau_n\}$, whose tasks have utilizations $\{U_1, U_2, \ldots, U_n\}$, which cannot be scheduled by the RM algorithm.

Proof:

To build the non-schedulable task set we first choose T_1 arbitrarily and then calculate the other parameters as follows:

$$\begin{cases} C_i = U_i T_i & i = 1, \ldots, n \\ T_{i+1} = T_i + C_i & i = 1, \ldots, n-1. \end{cases} \tag{9}$$

We observe that the $2n-1$ equations expressed in (9) have $2n-1$ variables $(T_2, \ldots, T_n, C_1, \ldots, C_n)$, and in every equation there is only one unknown, so the system has a unique solution. Following the same approach used in the proof of Theorem 1, we notice that:

$$\prod_{i=1}^{n-1}(U_i + 1) = \frac{T_n}{T_1}.$$

Moreover, by adding the last $n-1$ equations in (9) we have:

$$T_n - T_1 = \sum_{i=1}^{n-1} C_i.$$

Hence, from the hypothesis, we can write:

$$\begin{aligned}
(U_n + 1)\prod_{i=1}^{n-1}(U_i + 1) &> 2 \\
(U_n + 1)\frac{T_n}{T_1} &> 2 \\
C_n + T_n &> 2T_1 \\
C_n + (T_n - T_1) &> T_1 \\
C_n + \sum_{i=1}^{n-1} C_i &> T_1 \\
\sum_{i=1}^{n} C_i &> T_1.
\end{aligned}$$

Hence the constructed task set is not schedulable in the worst-case scenario (see equation (6) and Figure 2). ■

III. Comparative evaluation

In this section we compute the gain (in terms of schedulability) achieved by the hyperbolic test over the classical Liu and Layland test as a function of the number of tasks. This is done by computing the volume in the U-space of the regions underlying the bounds and plotting their ratio as a function of n.

To evaluate the efficiency of Liu and Layland test, we will compute the measure of the region $L_n(A)$ defined as:

$$L_n(A) = \left\{ x \in \mathbb{R}^n : \quad x_i \geq 0, \quad \sum_{i=1}^{n} x_i \leq A \right\}. \tag{10}$$

In the following, $|E|$ will denote the n-dimensional measure of a subset E of $\mathbb{R}^n$.

Lemma 1: For every integer n and for all $A > 0$ we have

$$|L_n(A)| = \frac{A^n}{n!}. \tag{11}$$

Proof: We will proceed by induction on n. For $n = 1$, $L_n(A)$ reduces to the interval $[0, A]$, hence, its measure is clearly A.

Now, assume that equality (11) is true for $n-1$ and for all $A > 0$. In particular, we have

$$|L_{n-1}(A - x_n)| = \frac{(A - x_n)^{n-1}}{(n-1)!}.$$

Therefore

$$\begin{aligned}
|L_n(A)| &= \int_0^A \mathrm{d}x_n \int_{L_{n-1}(A-x_n)} \mathrm{d}x_1 \cdots \mathrm{d}x_{n-1} \\
&= \int_0^A \frac{(A - x_n)^{n-1}}{(n-1)!}\,\mathrm{d}x_n
\end{aligned}$$

and defining $y = (A - x_n)$ we can write:

$$|L_n(A)| = \int_0^A \frac{y^{n-1}}{(n-1)!}\,\mathrm{d}y = \frac{A^n}{n!}$$

as required. ■

To evaluate the effectiveness of the hyperbolic test, we will compute the measure of the region $H_n(A)$ defined as

$$H_n(A) = \left\{ x \in \mathbb{R}^n : \quad x_i \geq 0, \quad \prod_{i=1}^{n} (1 + x_i) \leq A \right\}. \quad (12)$$

Lemma 2: For every integer n and for all $A \geq 1$ we have

$$|H_n(A)| = (-1)^n \left[1 - A \sum_{k=0}^{n-1} \frac{(-\ln A)^k}{k!} \right]. \quad (13)$$

Proof: We will proceed by induction on n. For $n = 1$, $H_n(A)$ reduces to the interval $[0, A-1]$, whose measure is clearly $A - 1$.

Now, assume equality (13) is true for $n - 1$ and for all $A \geq 1$. In particular, we have

$$\begin{aligned} &\left| H_{n-1}\left(\frac{A}{1+x_n}\right) \right| \\ &\quad = (-1)^{n-1} \left[1 - \frac{A}{1+x_n} \sum_{k=0}^{n-2} \frac{1}{k!} \left(-\ln \frac{A}{1+x_n} \right)^k \right]. \end{aligned}$$

Therefore

$$\begin{aligned} &|H_n(A)| \\ &\quad = \int_0^{A-1} \mathrm{d}x_n \int_{H_{n-1}(\frac{A}{1+x_n})} \mathrm{d}x_1 \cdots \mathrm{d}x_{n-1} \\ &\quad = \int_0^{A-1} (-1)^{n-1} \left[1 - \frac{A}{1+x_n} \sum_{k=0}^{n-2} \frac{1}{k!} \left(-\ln \frac{A}{1+x_n} \right)^k \right] \mathrm{d}x_n \end{aligned}$$

and defining $y = (1 + x_n)$ we have:

$$\begin{aligned} &|H_n(A)| \\ &= \int_1^A (-1)^{n-1} \left[1 - \frac{A}{y} \sum_{k=0}^{n-2} \frac{1}{k!} \left(-\ln \frac{A}{y} \right)^k \right] \mathrm{d}y \\ &= (-1)^{n-1} \left[y + A \sum_{k=0}^{n-2} \frac{(-1)^k}{(k+1)!} \left(\ln \frac{A}{y} \right)^{k+1} \right]_{y=1}^{y=A} \\ &= (-1)^{n-1} \left[A - 1 + A \sum_{k=0}^{n-2} \frac{(-\ln A)^{k+1}}{(k+1)!} \right] \\ &= (-1)^n \left[1 - A \sum_{k=0}^{n-1} \frac{(-\ln A)^k}{k!} \right] \end{aligned}$$

as required. ■

To evaluate the gain of the hyperbolic test with respect to the LL test, we have to compute the ratio

$$\rho_n = \frac{|H_n(2)|}{\left| L_n\left(n\left(2^{1/n} - 1\right)\right) \right|}. \quad (14)$$

In fact,

- according to equation (12), $|H_n(2)|$ represents the hypervolume in the U-space of the task sets found schedulable by the hyperbolic test;
- according to equation (10), $\left| L_n\left(n\left(2^{1/n} - 1\right)\right) \right|$ represents the hypervolume in the U-space of the task sets found schedulable by the LL test.

Proposition 1: As n tends to infinity, the asymptotic behavior of ρ_n is

$$\rho_n = \sqrt{2} + O(n^{-1}).$$

Proof: By observing that

$$\sqrt[n]{2} = \exp\left(\frac{\ln 2}{n}\right) = 1 + \frac{\ln 2}{n} + \frac{\ln^2 2}{2n^2} + O\left(n^{-3}\right)$$

we can write:

$$\begin{aligned} &\left| L_n\left(n\left(2^{1/n} - 1\right)\right) \right| \\ &\quad = \frac{\left[n\left(\sqrt[n]{2} - 1\right) \right]^n}{n!} \quad = \frac{\ln^n 2}{n!} \left[1 + \frac{\ln 2}{2n} + O\left(n^{-2}\right) \right]^n \\ &\quad = \frac{\ln^n 2}{n!} \exp\left[n \ln\left(1 + \frac{\ln 2}{2n} + O\left(n^{-2}\right) \right) \right] \\ &\quad = \frac{\ln^n 2}{n!} \exp\left(\frac{\ln 2}{2} + O\left(n^{-1}\right) \right) \\ &\quad = \sqrt{2} \frac{\ln^n 2}{n!} \left(1 + O\left(n^{-1}\right) \right). \end{aligned}$$

If we now define

$$S_k(x) = \sum_{j=0}^{k} \frac{x^j}{j!}$$

by Taylor expansion of e^x we obtain

$$e^x = S_k(x) + \frac{x^{k+1}}{(k+1)!} + e^\xi \frac{x^{k+2}}{(k+2)!}$$

where $\xi \in (x, 0)$. Therefore

$$\begin{aligned} &S_{n-1}(-\ln 2) \\ &\qquad = e^{-\ln 2} - \frac{(-\ln 2)^n}{n!} \left(1 - e^\xi \frac{\ln 2}{n+1} \right) \\ &\qquad = \frac{1}{2} - (-1)^n \frac{\ln^n 2}{n!} \left(1 - e^\xi \frac{\ln 2}{n+1} \right) \end{aligned}$$

where $\xi \in (-\ln 2, 0)$. Thus, we have:

$$\begin{aligned} |H_n(2)| &= (-1)^n \left[1 - 2\, S_{n-1}(-\ln 2) \right] \\ &= 2 \frac{\ln^n 2}{n!} \left(1 - e^\xi \frac{\ln 2}{n+1} \right) \\ &= 2 \frac{\ln^n 2}{n!} \left(1 + O\left(n^{-1}\right) \right) \end{aligned}$$

which gives

$$\rho_n = \sqrt{2} + O\left(n^{-1}\right) \quad (15)$$

as required. ■

IV. Simulation results

In this section, we present some simulation experiment we performed on synthetic task sets to evaluate the tightness of the H-bound (denoted by HB in the graphs), with respect to the Liu and Layland bound (LL in the graphs) and the exact test given by (3), resulting from the response time analysis (denoted by RTA in the graphs). Simulations have been conducted on randomly generated tasks sets, having desired total utilization.

In our experiment, we generated 10^6 task sets uniformly distributed in the region $L_n(1)$ of the U-space, as defined

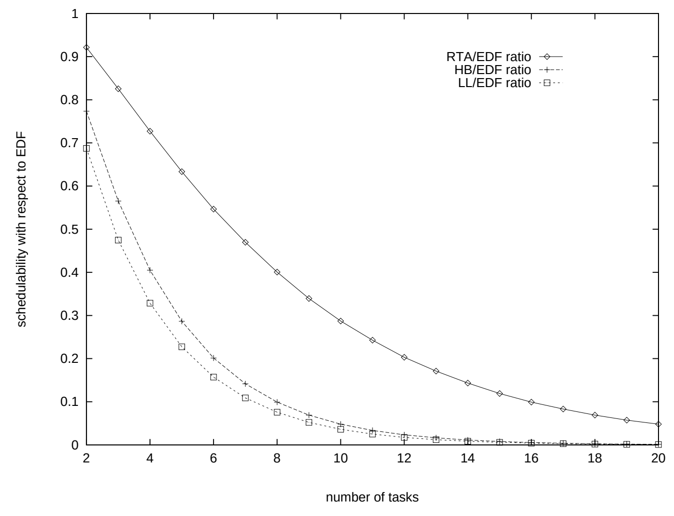


Fig. 5. Feasibility ratios with respect to EDF as a function of n.

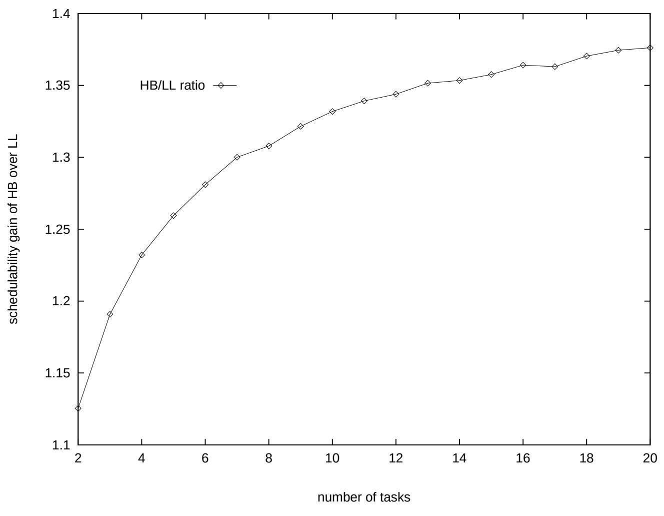


Fig. 7. Hyperbolic test versus LL-test as a function of n.

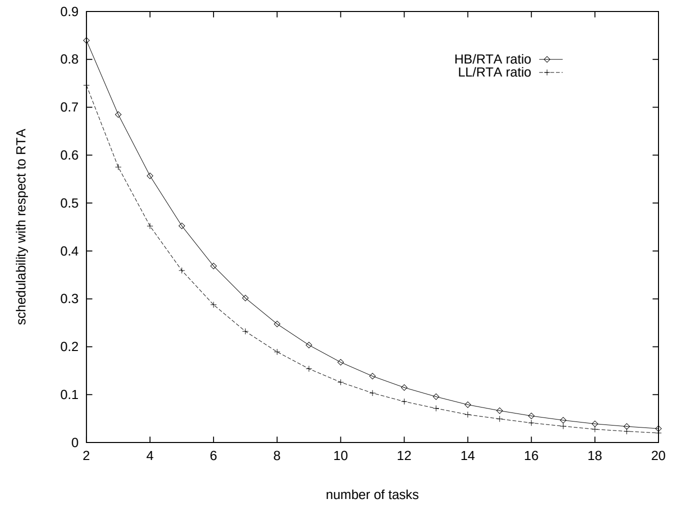


Fig. 6. Feasibility ratios with respect to RTA as a function of n.

by equation (10). This is the region where task sets are schedulable by EDF. For different values of n, we computed the number of task sets guaranteed by the LL test, by the hyperbolic test, and by the exact test, respectively. Figure 5 reports such ratios with respect to the total number of generated task sets (we recall that all task sets are generated to be feasible with EDF). Figure 6 compares the LL-test and the H-test with respect to the exact RTA test.

It is worth noting that, although all the ratios tends to zero as n tends to infinity, the schedulability gain achieved by the H-test over the LL-test increases as n gets larger. This can be clearly seen in Figure 7, which reports the ratio of the number of task sets guaranteed by the H-test and the number of task sets guaranteed by the LL-test, as a function of n. We observe that the ratio tends to $\sqrt{2}$, as predicted by the asymptotic analysis presented in Section III.

V. Extensions

In this section we extend the hyperbolic approach to take aperiodic servers and shared resources into account. First, we derive a schedulability condition for a Polling Server [12] scheduled by RM at the highest priority, and then generalize the analysis to a Deferrable Server. Finally, we show how to take blocking times into account.

Theorem 3: Let $\Gamma = \{\tau_1, \ldots, \tau_n\}$ be a set of n periodic tasks, where each task τ_i is characterized by a processor utilization U_i, and let S be a Polling Server with utilization $U_s = C_s/T_s$, such that $T_s \leq \min(T_1, \ldots, T_n)$ (that is, S is assigned the highest priority). Then, Γ is schedulable with the RM algorithm in the presence of server S if

$$\prod_{i=1}^{n}(U_i + 1) \leq \frac{2}{U_s + 1}. \tag{16}$$

Proof: Without loss of generality, assume that tasks are ordered by increasing periods, so that τ_1 is the task with the highest priority and τ_n is the task with the lowest priority.

Lehoczky, Sha, and Strosnider proved in [12] that the worst-case scenario for the task set occurs when all the tasks start simultaneously (e.g., at time $t = 0$) and

$$\begin{cases} T_s < T_i < 2T_s \quad \forall i = 1, \ldots, n \\ C_s \ = \ T_1 - T_s \\ C_1 \ = \ T_2 - T_1 \\ C_2 \ = \ T_3 - T_2 \\ \cdots \\ C_{n-1} \ = \ T_n - T_{n-1} \\ C_n \ = \ T_s - C_s - \sum_{i=1}^{n-1} C_i = 2\,T_s - T_n. \end{cases} \tag{17}$$

Hence, the feasibility condition for a task set which fully utilizes the processor and minimizes the total utilization factor can be written as

$$C_n \leq 2\,T_s - T_n$$

or (dividing both sides by T_n) as

$$(U_n + 1) \leq 2\,\frac{T_s}{T_n}. \tag{18}$$

Following the same approach used for proving Theorem 1, we define (for all $i < n$)

$$R_i = \frac{T_{i+1}}{T_i}$$

and notice that

$$\begin{aligned} \frac{T_1}{T_s} &= U_s + 1 \\ \frac{T_n}{T_1} &= \prod_{i=1}^{n-1} R_i = \prod_{i=1}^{n-1}(U_i + 1). \end{aligned}$$

Hence, equation (18) can be written as

$$(U_n + 1)\frac{T_n}{T_1} \leq 2\,\frac{T_s}{T_1}$$

which leads to

$$\prod_{i=1}^{n}(U_i + 1) \leq \frac{2}{U_s + 1}$$

as required. ■

For a Deferrable Server, the analysis performed in [12] by Lehoczky, Sha, and Strosnider can also be expressed in the hyperbolic form. By following the same reasoning presented in [12], in the presence of a high priority Deferrable Server, the constraint on C_n can be written as:

$$C_n \;\leq\; K T_1 - T_n.$$

where $K = \frac{U_s+2}{2\,U_s+1}$. Hence, the feasibility condition becomes

$$\prod_{i=1}^{n}(U_i + 1) \leq \frac{U_s + 2}{2\,U_s + 1}.$$

Analogous considerations can be done for the more precise analysis presented in [13], but they are omitted due to space limitations.

In the presence of resource constraints, blocking times due to mutual exclusion can be taken into account in the hyperbolic test by increasing tasks' execution times by a suitable blocking factor. Hence, the n schedulability conditions derived by Sha, Rajkumar, and Lehoczky in [7], can be expressed as follows:

$$\forall i = 1, \ldots, n \quad \prod_{k=1}^{i-1}(U_k + 1)\left(\frac{C_i + B_i}{T_i} + 1\right) \leq 2.$$

VI. Reducing pessimism

In this section we present two particular task set scenarios in which the pessimism of the hyperbolic bound can be reduced to improve schedulability:
1. the first scenario includes task sets with harmonic period chains, as also discussed by Kuo and Mok in [14];
2. the second scenario includes task sets where the condition $T_1 \leq T_i \leq 2T_1$ is relaxed.

A. Task sets with harmonic chains

Similarly to what Kuo and Mok presented in [14] for improving the Liu and Layland bound, in this section we extend the hyperbolic bound result, using the harmonic chains concept.

Let $P = \{T_1, T_2, \ldots, T_n\}$ be the set of periods of a set $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ of periodic tasks. A subset $R \subseteq P$ is said to be a *harmonic base* of Γ if there is a partition $\mathcal{P}$ of P into $|R|$ subsets such that:
1. each member of P is the smallest element in exactly one member of the partition $\mathcal{P}$;
2. if x and y are two elements in the same member of the partition $\mathcal{P}$, then either x divides y or y divides x.

We call each subset in the partition $\mathcal{P}$, a *harmonic chain.*

Kuo and Mok also suggested a polynomial algorithm to find the harmonic base given the task periods and they provided a new bound, better than the Liu and Layland one, taking into account such period properties. Their bound is given by:

$$\sum_{i=1}^{n} U_i \leq K\,(2^{1/K} - 1) \tag{19}$$

where $K \leq n$ is the number of harmonic chains.

In [14], they also proved that all the tasks belonging to the same member $\mathbf{S}$ of partition $\mathcal{P}$ can be merged together into a new task τ_k such that $U_k = \sum_{\tau_i \in \mathbf{S}} U_i$, *"without changing the schedulability of the whole task set"*. We refer to this modified task set as Γ^*.

Applying the hyperbolic bound to Γ^* leads to the next interesting theorem:

Theorem 4: Let $\Gamma = \{\tau_1, \ldots, \tau_n\}$ be a set of n periodic tasks, where each task τ_i is characterized by a processor utilization U_i. Let $\mathcal{P}$ be the partition in K subsets of the task indexes obtained by grouping the tasks into harmonic chains. Then, Γ is schedulable with the RM algorithm if

$$\prod_{\mathbf{S}\in\mathcal{P}}\left(1 + \sum_{i\in\mathbf{S}} U_i\right) \leq 2. \tag{20}$$

Proof: It directly follows from the equivalence, in terms of schedulability, between Γ and Γ^* (proved by Theorems 1 and 2 in [14]) and Theorem 1. ■

As can be guessed, condition (20) is a relaxation of (4) because we use the additional information on the harmonic properties of the tasks periods. This statement is formally proved in the next theorem.

Theorem 5: If a task set satisfies:

$$\prod_{i=1}^{n}(1 + U_i) \leq 2.$$

then it also satisfies:

$$\prod_{\mathbf{S}\in\mathcal{P}}\left(1 + \sum_{i\in\mathbf{S}} U_i\right) \leq 2.$$

where $\mathcal{P}$ is the partition in harmonic chains.

Proof: For the associative property of the product and for the properties of partitions, the hyperbolic bound can be written as:

$$\prod_{\mathbf{S}\in\mathcal{P}} \left[\prod_{i\in\mathbf{S}} (1+U_i)\right] \le 2.$$

Expanding the innermost product we have:

$$\begin{aligned}\prod_{i\in\mathbf{S}} (1+U_i) &\\ &= 1 + \sum_{i\in\mathbf{S}} U_i + \sum_{i\neq j} U_i\,U_j + \sum_{i\neq j\neq k} U_i\,U_j\,U_k + \cdots\\ &\ge 1 + \sum_{i\in\mathbf{S}} U_i.\end{aligned}$$

Hence,

$$\prod_{\mathbf{S}\in\mathcal{P}} \left(1+\sum_{i\in\mathbf{S}} U_i\right) \le \prod_{\mathbf{S}\in\mathcal{P}} \left[\prod_{i\in\mathbf{S}} (1+U_i)\right] = \prod_{i=1}^{n}(1+U_i) \le 2.$$

as required. ■

Figure 8 provides a graphical representation of the improvement for a simple case of three tasks, τ_1, τ_2, and τ_3, in which τ_1 and τ_2 have harmonic periods and, hence, can be grouped into a chain.

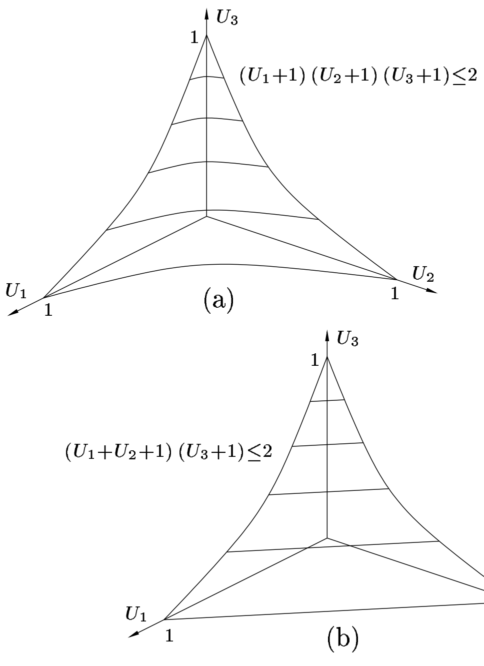

U_3
1
$(U_1{+}1)\,(U_2{+}1)\,(U_3{+}1)\le 2$
U_1
1
U_2
1
(a)
U_3
1
$(U_1{+}U_2{+}1)\,(U_3{+}1)\le 2$
U_1
1
U_2
1
(b)

Fig. 8. Schedulability regions of the plain H-bound (a) and its improvement with harmonic chains (b).

A precise evaluation of the gain introduced by the harmonic chain method would require computing the measure of

$$K_n(\mathcal{P}) = \left\{x \in \mathbb{R}^n : \quad x_i \ge 0, \quad \prod_{\mathbf{S}\in\mathcal{P}} \left(1+\sum_{i\in\mathbf{S}} x_i\right) \le 2\right\}$$

which is not trivial to compute.

B. *Relaxing* $T_1 \le T_i \le 2\,T_1$

By relaxing the worst-case condition for which $T_1 < T_i < 2\,T_1$, a better schedulability test can be found. In this section, we present the analysis for the simple case of two tasks. By defining

$$F = \left\lfloor \frac{T_2}{T_1} \right\rfloor,$$

the worst-case situation occurs when

$$\begin{cases} C_1 &= T_2 - F\,T_1 \\ C_2 &= T_2 - (F+1)\,C_1. \end{cases} \tag{21}$$

Hence, the schedulability condition can be written as

$$C_2 \le F\,T_2 \left[\frac{F+1}{T_2/T_1} - 1\right].$$

Now, observing that

$$\frac{T_2}{T_1} = U_1 + F,$$

the schedulability condition becomes

$$U_2 \le F \left[\frac{F+1}{U_1+F} - 1\right]$$

that is

$$\frac{U_2}{F} + 1 \le \frac{F+1}{U_1+F}$$

and finally

$$\prod_{i=1}^{2} \left(\frac{U_i}{F} + 1\right) \le \frac{1}{F} + 1. \tag{22}$$

Figure 9 plots equation (22) in the U-space for different values of the F parameter. In this case, the asymptots intersect each axis in $-F$, so the hyperbole tends to approach the EDF line as F gets larger.

Unfortunately, generalizing equation (22) to the case of n tasks is not trivial, and will be investigated as a future work.

VII. Conclusions

In this paper we presented a hyperbolic schedulability bound for the Rate Monotonic algorithm and evaluated its effectiveness with respect to the classical Liu and Layland utilization bound and the necessary and sufficient condition computed through a response time analysis. The asymptotic behavior of the hyperbolic bound relative to the LL bound was also computed for n tending to infinity and found to be equal to $\sqrt{2}$. Since the hyperbolic test has an $O(n)$ complexity, it can be effectively used to perform on-line admission control in large periodic task sets under the RM algorithm, when the exact schedulability analysis cannot be applied for efficiency reasons.

We proved that the hyperbolic bound is the tightest among those using task utilizations as application parameters, and we showed how the proposed analysis can be improved by relaxing some pessimistic assumptions typically made on the task set, to take advantage of additional

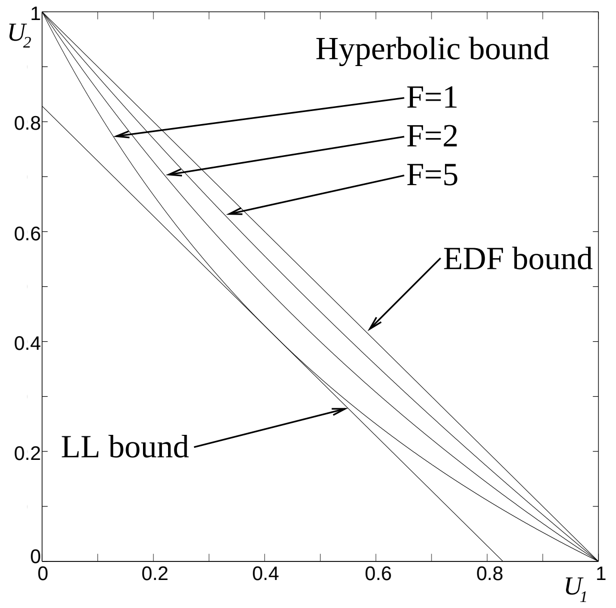


Fig. 9. Hyperbolic RM bound for different values of F.

knowledge on the application. In particular, we considered the case in which some tasks may have harmonic period relations and the case in which periods are not constrained to be in the worst-case configuration.

As a future work, we believe additional pessimistic assumptions can be relaxed, or special task set characteristics identified, to improve the schedulability of the RM algorithm within the same computational complexity. We are also investigating the possibility of deriving a tighter schedulability condition with polynomial complexity as a function of arbitrary period relations.

References

[1] J. Locke, "Designing real-time systems," Invited talk at IEEE International Conference of Real-Time Computing Systems and Applications, December 1997.

[2] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 40–61, 1973.

[3] J. P. Lehoczky, L. Sha, and Y. Ding, "The rate-monotonic scheduling algorithm: Exact characterization and average case behavior," in *Proceedings of the IEEE Real-Time Systems Symposium*, 1989, pp. 166–172.

[4] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.

[5] N. C. Audsley, A. Burns, K. Tindell, and A. Wellings, "Applying new scheduling theory to static priority preemptive scheduling," *Software Engineering Journal*, December 1993.

[6] M. Sjödin and H. Hansson, "Improved response-time analysis calculations," in *Proceedings of the* 19th *IEEE Real-Time Systems Symposium*, December 1998, pp. 399–409.

[7] L. Sha, R. Rajkumar, and J. P. Lehoczky, "Priority inheritance protocols: An approach to real-time synchronization," *IEEE Transactions on Computers*, vol. 39, no. 9, pp. 1175–1185, 1990.

[8] Y. Oh and S. H. Son, "Allocating fixed-priority periodic tasks on multiprocessor systems," *Real-Time Systems*, vol. 9, pp. 207–239, 1995.

[9] J. W. S. Liu, *Real-Time Systems*, Prentice Hall, 2000.

[10] E. Bini, G. C. Buttazzo, and G. M. Buttazzo, "A hyperbolic bound for the rate monotonic algorithm," in *IEEE Proc. of the* 13th *Euromicro Conf. on Real-Time Systems*, June 2001.

[11] R. Devillers and J. Goossens, "Liu and layland's schedulability test revisited," *Information Processing Letters*, vol. 73, no. 5, pp. 157–161, March 2000.

[12] J. P. Lehoczky, L. Sha, and J. K. Strosnider, "Enhanced aperiodic responsiveness in hard real-time environments," in *Proceedings of the IEEE Real-Time Systems Symposium*, 1987, pp. 261–270.

[13] J. K. Strosnider, J. P. Lehoczky, and L. Sha, "The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments," *IEEE Transactions on Computers*, January 1995.

[14] T.-W. Kuo and A. K. Mok, "Load adjustment in adaptive real-time systems," in *Proceedings of the IEEE Real-Time Systems Symposium*, December 1991.

Enrico Bini is a PhD student in computer engineering at Scuola Superiore S.Anna in Pisa, Italy. In 2000, he graduated in the same subject at University of Pisa and received in the same year the Licenza at Scuola Superiore S.Anna, where he won a national competition as a student. In 1999, through a EU student exchange program, he studied at the Delft University of Technology, in the Nederlands. His interests cover scheduling algorithms, real-time operating systems, linear programming and combinatorial optimization.

Giorgio C. Buttazzo graduated in electronic engineering from the University of Pisa in 1985, received the Master's degree in computer science from the University of Pennsylvania in 1987, and the PhD degree in computer engineering from the Scuola Superiore S.Anna of Pisa in 1991. He is an associate professor of computer engineering at the University of Pavia, Italy. His main research interests include real-time operating systems, dynamic scheduling algorithms, quality of service control, multimendia systems, advanced robotics applications, and neural networks. He is a member of the IEEE and the IEEE Computer Society.

Giuseppe M. Buttazzo graduated in Mathematics at the University of Pisa in 1976, and received in the same year the Diploma of Scuola Normale Superiore di Pisa, where he won a national competition as a student. He made the Ph. D. studies from 1976 until 1980 at Scuola Normale di Pisa, where he also got his first permanent position as a researcher in 1980. He obtained the full professorship in Mathematical Analysis at University of Ferrara in 1986, and moved in 1990 to University of Pisa, where he presently works at the Department of Mathematics. His main research interests include calculus of variations, partial differential equations, optimization problems, optimal control theory. On these subjects he wrote more than 120 scientific papers and several books.