Proc. of the 13th Euromicro Conference on Real-Time Systems (ECRTS 2001), Delft, The Netherlands, June 13-15, 2001.

A Hyperbolic Bound for the Rate Monotonic Algorithm

Enrico Bini
Scuola Superiore S. Anna
Pisa, Italy
e.bini@sssup.it

Giorgio Buttazzo
University of Pavia, Italy
INFM Research Unit
buttazzo@unipv.it

Giuseppe Buttazzo
Department of Mathematics
University of Pisa, Italy
buttazzo@dm.unipi.it

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.

1 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 [8]. 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.

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 [7] 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 kth 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, $d_{i,k} = kT_i$) 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

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

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

$$\sum_{i=1}^{n} U_i \leq 1. \qquad (2)$$

Although EDF has a better schedulability bound than RM, RM is easier to implement on top of commercial kernels, since task rates can directly be mapped into a small set of fixed priorities. RM is especially preferred in small embedded systems, where managing deadlines would require more memory space and a more sophisticated timer handling routine. For such reasons, a lot of work has been done to improve the schedulability bound of the RM algorithm or relax some restrictive assumption on the task set. In [6], 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, 6, 1]. Using the method proposed in [1], 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:

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

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 [10], Sha, Rajkumar and Lehoczky 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 [1], Audsley et al. generalized the response time analysis including resource constraints, and in [2], Burns, Davis, and Punnekats extended it to take fault-tolerant constraints into account.

In this paper we present an efficient test for verifying the feasibility of large periodic task sets under the RM algorithm. The proposed method is particularly useful for achieving on-line admission control in small embedded systems, when EDF cannot be implemented for efficiency reasons and the response time analysis cannot be applied on line due to prohibitively long execution times. The analysis has the same complexity as the original LL-test, but it is less pessimistic, so allowing to accept task sets that would be rejected using the original approach. A similar approach has been independently developed by Oh and Song in [9] to verify the schedulability of 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 rest of the paper is organized as follows. Section 2 presents the hyperbolic feasibility bound for the RM algorithm, explaining its relation with the classical Liu and Layland approach in the utilization space. Section 3 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 4 extends the schedulability test to take aperiodic servers and resource constraints into account. Section 5 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 6 states our conclusions and future work.

2 The hyperbolic bound

The schedulability test we propose in this paper is derived from the same the worst-case scenario identified by Liu and Layland in [7] 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.

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. Liu and Layland [7] have 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

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

Moreover, among the tasks that fully utilize the processor, the total utilization factor is minimized when computation times have the following relations:

$$\begin{cases} C_1 \;=\; T_2 - T_1 \\ C_2 \;=\; T_3 - T_2 \\ \cdots \\ C_{n-1} \;=\; T_n - T_{n-1} \\ C_n \;=\; T_1 - \sum_{i=1}^{n-1} C_i \;=\; 2T_1 - T_n. \end{cases} \tag{5}$$

Starting from such a worst-case scenario, the least upper bound 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) 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:

$$\begin{cases} U_1 = R_1 - 1 \\ U_2 = R_2 - 1 \\ \cdots \\ U_{n-1} = R_{n-1} - 1 \\ U_n = 2T_1/T_n - 1. \end{cases} \tag{6}$$

Now we notice that:

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

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

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

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. $\square$

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, $\ldots$, 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 $U_{lub}(n) = n(2^{1/n} - 1)$, 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 1 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 grey area.

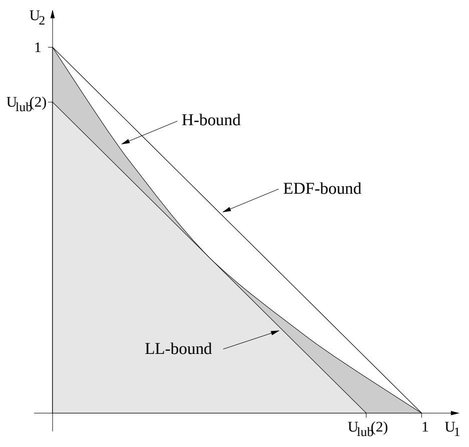

Figure 1. Schedulability bounds for RM and EDF in the utilization space.

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 3.

2.1 Reducing pessimism

By relaxing the worst-case condition for which $T_1 < T_i < 2T_1$, 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

$$\begin{cases} C_1 &= T_2 - FT_1 \\ C_2 &= T_2 - (F+1)C_1. \end{cases} \tag{7}$$

Hence, the schedulability condition can be written as

$$C_2 \leq FT_2 \left[\frac{F+1}{T_2/T_1} - 1 \right].$$

Now, observing that

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

the schedulability condition becomes

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

that is

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

and finally

$$\prod_{i=1}^{2} \left(\frac{U_i}{F} + 1 \right) \leq \frac{1}{F} + 1. \tag{8}$$

Figure 2 plots equation 8 in the U-space for different values of the F parameter. As we can see, the asymptots intersect each axis in $-F$, so the hyperbole tends to approach the EDF line as F gets larger.

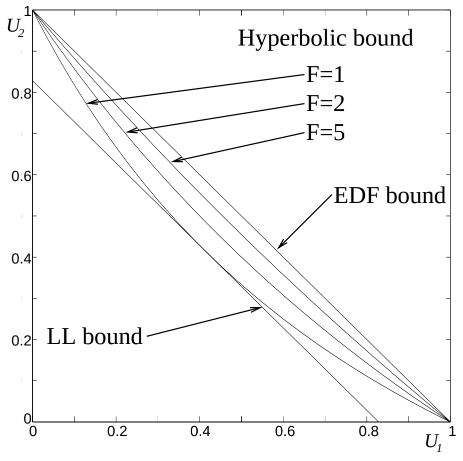

Figure 2. Hyperbolic RM bound for different values of F.

Unfortunately, generalizing equation (8) to the case of n tasks is not trivial, and will be investigated as a future work.

3 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:

$$L_n(A) = \left\{ x \in \mathbf{R}^n : \ x_i \geq 0, \ \sum_{i=1}^{n} x_i \leq A \right\}. \qquad (9)$$

In the following we denote by $|E|$ the n-dimensional measure of a subset E of $\mathbf{R}^n$.

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

$$|L_n(A)| = \frac{A^n}{n!}. \qquad (10)$$

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 (10) 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

$$\begin{aligned} |L_n(A)| &= \int_0^A dx_n \int_{L_{n-1}(A-x_n)} dx_1 \cdots dx_{n-1} \\ &= \int_0^A \frac{(A-x_n)^{n-1}}{(n-1)!} dx_n \end{aligned}$$

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

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

as required. □

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

$$H_n(A) = \left\{ x \in \mathbf{R}^n : \ x_i \geq 0, \ \prod_{i=1}^{n} (1 + x_i) \leq A \right\}. \qquad (11)$$

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

$$|H_n(A)| = (-1)^n \left[1 - A \sum_{k=0}^{n-1} \frac{(-\ln A)^k}{k!} \right]. \qquad (12)$$

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

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

Therefore

$$\begin{aligned} |H_n(A)| &= \int_0^{A-1} dx_n \int_{H_{n-1}(\frac{A}{1+x_n})} dx_1 \cdots dx_{n-1} \\ &= \int_0^{A-1} (-1)^{n-1} \left[1 - \frac{A}{1+x_n} \sum_{k=0}^{n-2} \frac{1}{k!} \left(-\ln \frac{A}{1+x_n} \right)^k \right] dx_n \end{aligned}$$

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

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

as required. □

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

$$\rho_n = \frac{|H_n(2)|}{\left| L_n\left(n(2^{1/n} - 1)\right) \right|}. \qquad (13)$$

In fact,

- according to equation (11), $|H_n(2)|$ represents the hypervolume in the U-space of the task sets found schedulable by the hyperbolic test;
- according to equation (9), $\left| L_n\left(n(2^{1/n} - 1)\right) \right|$ 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:

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

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

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

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

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

which gives

$$\rho_n=\sqrt{2}+O\left(n^{-1}\right) \qquad (14)$$

as required. □

4 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 [5] 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 2 *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* $T_s \le \min(T_1,\ldots,T_n)$ *(that is, S is assigned the highest priority). Then,* Γ *is schedulable with the RM algorithm in the presence of server S if*

$$\prod_{i=1}^{n}(U_i+1)\le\frac{2}{U_s+1}. \qquad (15)$$

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

$$\begin{cases} T_s < T_i < 2T_s \quad \forall i=1,\ldots,n\\ C_s \;=\; T_1 - T_s\\ C_1 \;=\; T_2 - T_1\\ C_2 \;=\; T_3 - T_2\\ \cdots\\ C_{n-1} \;=\; T_n - T_{n-1}\\ C_n \;=\; T_s - C_s - \sum_{i=1}^{n-1} C_i = 2T_s - T_n.\end{cases} \qquad (16)$$

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

$$C_n \le 2T_s - T_n$$

or (dividing both sides by T_n) as

$$(U_n+1)\le 2\frac{T_s}{T_n}. \qquad (17)$$

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 (17) can be written as

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

which leads to

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

as required. $\Box$

For a Deferrable Server, the analysis performed in [5] by Lehoczky, Sha, and Strosnider can also be expressed in the hyperbolic form. By following the same reasoning presented in [5], in the presence of a high priority Deferrable Server, the constraint on C_n can be written as:

$$C_n \ \leq \ KT_1 - T_n.$$

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

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

Analogous considerations can be done for the more precise analysis presented in [12], 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 [10], can be expressed as follows:

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

5 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 by equation (9). 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 3 reports such ratios with respect to the total number of generated

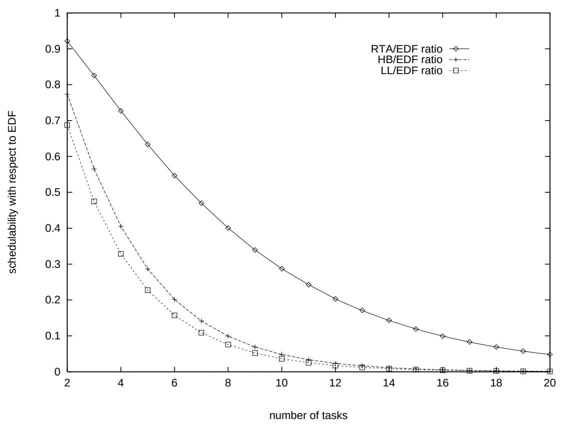

Figure 3. Feasibility ratios with respect to EDF as a function of n.

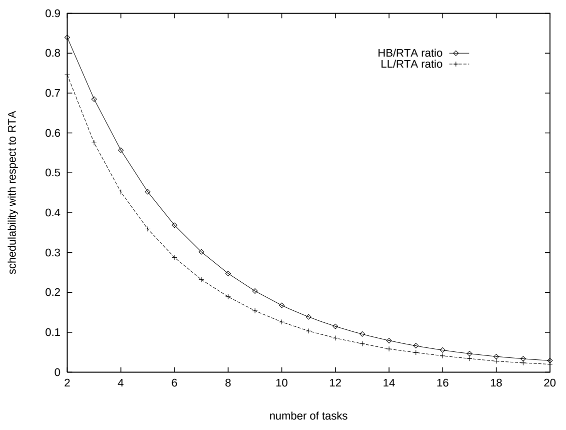

Figure 4. Feasibility ratios with respect to RTA as a function of n.

task sets (we recall that all task sets are generated to be feasible with EDF). Figure 4 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 5, 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 3.

6 Conclusions

In this paper we presented a hyperbolic schedulability bound for the Rate Monotonic algorithm and evaluated its effectiveness with respect to the classi-

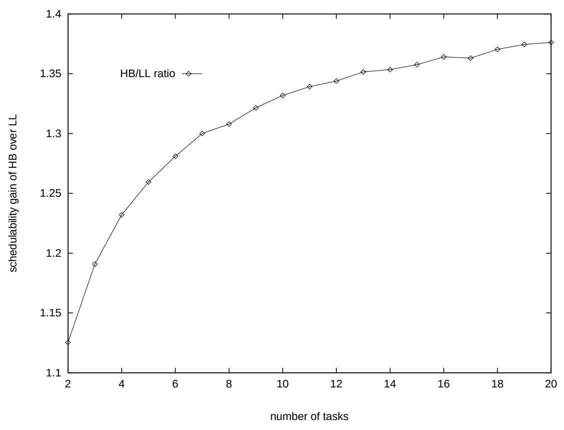

Figure 5. Hyperbolic test versus LL-test as a function of n.

cal Liu and Layland utilization bound and the necessary and sufficient condition computed through a response time analysis. The asymptotic behaviour 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 believe that the proposed analysis can be improved by relaxing other pessimistic assumptions typically made on the task set, and we are investigating the possibility of deriving a tighter schedulability condition with polynomial complexity as a function of arbitrary period relations.

References

[1] N.C. Audsley, A. Burns, M. Richardson, K. Tindell and A. Wellings, "Applying New Scheduling Theory to Static Priority Preemptive Scheduling", *Software Engineering Journal* 8(5), pp. 284-292, September 1993.

[2] A. Burns, R. Davis, and S. Punnekkat "Feasibility Analysis of Fault-Tolerant Real-Time Task Sets," *IEEE Proceedings of the Euromicro Workshop on Real-Time Systems,* pp. 29-33, June 1996.

[3] M.L. Dertouzos, "Control Robotics: the Procedural Control of Physical Processes," *Information Processing*, 74, North-Holland, Publishing Company, 1974.

[4] M. Joseph and P. Pandya, "Finding Response Times in a Real-Time System," *The Computer Journal*, 29(5), pp. 390-395, 1986.

[5] J. P. Lehoczky, L. Sha, and J. K. Strosnider, "Enhanced Aperiodic Responsiveness in Hard Real-Time Environments," *Proceedings of the IEEE Real-Time Systems Symposium*, pp. 261-270, 1987.

[6] J. P. Lehoczky, L. Sha, and Y. Ding, "The Rate-Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behaviour," *Proceedings of the IEEE Real-Time Systems Symposium*, pp. 166-171, 1989.

[7] C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard real-Time Environment," *Journal of the ACM* 20(1), 1973, pp. 40–61.

[8] C.D. Locke, "Software Architecture for Hard Real-Time Applications: Cyclic Executives vs. Fixed Priority Executives", *Real-Time Systems*, 4, pp. 37-53, 1992.

[9] Y. Oh and S. H. Son, "Allocating Fixed-Priority Periodic Tasks on Multiprocessor Systems," *Real-Time Systems*, 9, pp. 207-239, 1995.

[10] L. Sha, R. Rajkumar, and J. P. Lehoczky, "Priority Inheritance Protocols: An Approach to Real-Time Synchronization," *IEEE Transactions on Computers*, 39(9), pp. 1175-1185, 1990.

[11] B. Sprunt, L. Sha, and J. Lehoczky, "Aperiodic Task Scheduling for Hard Real-Time System," *Journal of Real-Time Systems*, 1, pp. 27-60, June 1989.

[12] 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*, 44(1), January 1995.