Utilization bounds for Multiprocessor Rate-Monotonic Scheduling

J.M. López, M. García, J.L. Díaz and D.F. García
Departamento de Informática, Universidad de Oviedo, Gijón 33204, Spain

Abstract. In this paper we extend Liu & Layland's utilization bound for fixed priority scheduling on uniprocessors to homogeneous multiprocessor systems under a partitioning strategy. Assuming that tasks are pre-emptively scheduled on each processor according to fixed priorities assigned by the Rate-Monotonic policy, and allocated to processors by the First Fit algorithm, we prove that the utilization bound is $(n-1)(2^{1/2}-1)+(m-n+1)(2^{1/(m-n+1)}-1)$, where m and n are the number of tasks and processors respectively. This bound is valid for arbitrary utilization factors. Moreover, if all the tasks have utilization factors under a value α, the previous bound is raised and the new utilization bound considering α is calculated. Finally, simulation provides the average-case behaviour.

Keywords: hard real-time, multiprocessor scheduling, partitioning, rate monotonic scheduling, utilization bound.

1. Introduction

Liu & Layland proved the optimality of the Rate Monotonic (RM) priority assignment for pre-emptive uniprocessor scheduling with fixed priorities, where task deadlines are equal to task periods (Liu and Layland, 1973). Throughout this paper, this scheduling policy will be referred to as *RM scheduling*. In addition, they derived the utilization bound $m(2^{1/m}-1)$, for RM scheduling of m tasks on uniprocessors. This bound represents the value to be exceeded by the total utilization of any task set before any task can miss its deadline. The objective of our paper is to extend the bound $m(2^{1/m}-1)$ to homogeneous multiprocessor systems by adding a new parameter, n, indicating the number of processors.

A new issue arises in multiprocessor scheduling with regard to the uniprocessor case; that is which processor executes each task at a given time. There are two major strategies to deal with this issue: *partitioning strategies*, and *non-partitioning strategies* (Oh and Son, 1995). In a partitioning strategy, once a task is allocated to a processor, it executes exclusively on that processor. In a non-partitioning strategy any instance of a task can execute on a different processor, or even be pre-empted and moved to a different processor before it is completed.

Non-partitioning strategies have several disadvantages versus partitioning strategies. Firstly, the scheduling overhead associated with a

non-partitioning strategy is greater than the overhead associated with a partitioning strategy. Secondly, partitioning strategies allow us to apply well-known uniprocessor scheduling algorithms to each processor.

In this paper, we follow the partitioning strategy, and we assume that all the tasks allocated to a processor are pre-emptively scheduled using fixed priorities defined by RM (as it is the optimal priority assignment for uniprocessors). Subsequently, the only degree of freedom is the allocation algorithm.

The problem of allocating a set of tasks to a set of processors is analogous to the bin-packing problem, where the set of processors is regarded as a set of bins. A bin-packing algorithm is said to be optimal if it finds a feasible allocation of items to bins whenever a feasible allocation exists. The capacity of the processor (bin) depends on the schedulability condition that is being used. Using Liu & Layland's schedulability condition for RM scheduling, the capacity of a processor is $m(2^{1/m} - 1)$, where m is the number of tasks allocated to the processor. The capacity of the processor is not constant, as it depends on m, and so the optimal allocation problem is as least as hard as the bin-packing problem, which is known to be NP-hard in the strong sense (Garey and Johnson, 1979). Thus, searching for optimal allocation algorithms is not practical. Several heuristic allocation algorithms have been proposed in the literature (Dall and Liu, 1978; Garey and Johnson, 1979; Burchard et al., 1995; Oh and Son, 1995; S. Sáez and Crespo, 1998).

Most works about RM scheduling on multiprocessors focus on searching for heuristic allocation algorithms which are compared to each other using the metric (N_A/N_{opt}), where N_A is the number of processors required to schedule a task set using a given allocation algorithm, A, and N_{opt} is the number of processors needed by the optimal allocation algorithm (Dall and Liu, 1978; Burchard et al., 1995; Oh and Son, 1995). This metric is useful to compare the performance of different allocation algorithms, but not to establish the schedulability of the system. There are several reasons:

- In general, the number N_{opt} can not be obtained in polynomial time.

- Even if N_{opt} were known, the utilization bound derived from the metric is too pessimistic, as is shown by Oh and Baker (1998).

The objective of this paper is not to investigate new allocation algorithms. The objective is to obtain the utilization bound for multiprocessor systems using RM scheduling and well-known allocation

algorithms. With this purpose a new parameter, n, indicating the number of processors will be added to the utilization bound $m(2^{1/m} - 1)$ given by Liu & Layland for uniprocessor systems.

The only result known by the authors related to the utilization bound using allocation and RM scheduling on each processor is that given by Oh and Baker (1998). They provide the interval (1) for the utilization bound $U_{\text{wc}}^{\text{RM-FF}}$, using First Fit (FF) allocation and RM scheduling on a homogeneous multiprocessor system.

$$n(2^{1/2} - 1) < U_{\text{wc}}^{\text{RM-FF}}(n) \leq (n + 1)/(1 + 2^{1/(n+1)}) \tag{1}$$

The practical implication of equation (1) is that any task set of total utilization less than or equal to $n(2^{1/2} - 1) \approx 0.414n$ is schedulable using FF allocation and RM scheduling.

Our paper proves that the utilization bound for FF allocation and RM scheduling takes the value

$$\begin{aligned} U_{\text{wc}}^{\text{RM-FF}}(m, n) = (n - 1)(2^{1/2} - 1) \\ + (m - n + 1)(2^{1/(m-n+1)} - 1) \end{aligned} \tag{2}$$

The difference between the utilization bound given by equation (2) and the expression $n(2^{1/2} - 1)$ given by equation (1) is particularly significant in systems with a small number of processors.

If all the tasks have a utilization factor under a value α, the utilization bound is proved to be

$$\begin{aligned} U_{\text{wc}}^{\text{RM-FF}}(m, n, \alpha) = (n - 1)(2^{1/(\beta+1)} - 1)\beta \\ + (m - \beta(n - 1))(2^{1/(m-\beta(n-1))} - 1) \end{aligned} \tag{3}$$

where

$$\beta = \lfloor 1/\log_2(\alpha + 1) \rfloor$$

Equation (3) represents the general case from which equation (2) is obtained making $\alpha = 1$, and therefore $\beta = 1$. As α decreases, both β and the bound given by equation (3) increase. In the limit, when $\alpha \to 0$, then $\beta \to \infty$, and the bound is $n \ln 2$. Therefore in the case of tasks with "low" utilization factors, the multiprocessor performance is close to that of an uniprocessor n-times faster than each of its processors.

The rest of the paper is organized as follows. Section 2 defines the system we deal with. The expression (3) of the utilization bound is proved in Section 3. Section 4 analyzes the expression of the utilization bound. Section 5 provides by means of simulation the average-case behaviour of RM scheduling with FF allocation. Allocation heuristics other than FF are considered in Section 6. Finally, Section 7 presents our conclusions.

2. System definition

The task set model consists of m independent periodic tasks $\{\tau_1, \ldots, \tau_m\}$ of computation times $\{C_1, \ldots, C_m\}$, periods $\{T_1, \ldots, T_m\}$, and hard deadlines equal to the task periods. The utilization factor u_i of any task τ_i, defined as $u_i = C_i/T_i$, is assumed to be $0 < u_i \leq \alpha \leq 1$, where α is the maximum value that can be taken by the utilization factor of any task. Thus, the total utilization of the task set defined as $U = \sum_{i=1}^{m} u_i$ is less than or equal to $m\alpha$. No particular order is assumed among the utilization factors.

Tasks are allocated to an array of n identical processors $\{P_1, \ldots, P_n\}$, which execute independent of each other. Once a task is allocated to a processor, it executes only on that processor. Within each processor, tasks are scheduled pre-emptively using fixed priorities defined by the RM priority assignment. This paper focuses basically on the *First Fit* (FF) allocation heuristic. Other allocation heuristics are also considered in Section 6.

The FF algorithm assigns any periodic task, τ_i, to the first processor, P_j, with enough capacity. The capacity is given by Liu & Layland's schedulability condition for RM scheduling. Thus, the task is allocated to the first processor fulfilling $(u_i + U_j) \leq (m_j + 1)(2^{1/(m_j+1)} - 1)$, where m_j is the number of tasks previously allocated to processor P_j, and U_j is the total utilization of these tasks. Processors are visited in the order $P_1, P_2, \ldots, P_n$. If no processor has enough capacity to hold τ_i, then we can not guarantee the schedulability of the periodic task set (at least using Liu & Layland's schedulability condition).

3. Calculation of the utilization bound

In this section we obtain the utilization bound $U_{\text{wc}}^{\text{RM-FF}}$ for RM scheduling and FF allocation on multiprocessors, which is defined as follows.

DEFINITION 1. *The utilization bound for RM scheduling and FF allocation is defined as the real number $U_{wc}^{RM\text{-}FF}$, fulfilling the following properties.*

- *Any periodic task set of total utilization $U \leq U_{wc}^{RM\text{-}FF}$ fits into the processors, using Liu & Layland's schedulability condition for RM scheduling, and the allocation policy FF. Therefore the periodic task set is schedulable.*

- *For any total utilization $U > U_{wc}^{RM\text{-}FF}$, it is always possible to find a periodic task set, which does not fit into the processors using*

Liu & Layland's schedulability condition for RM scheduling and the allocation policy FF. In this case, the periodic task set may be or may not be schedulable.

In other words, the utilization bound is the maximum total utilization guaranteeing the schedulability of the task set even in the worst-case.

The rest of this section is structured as follows.

- The existence of the utilization bound $U_{\text{wc}}^{\text{RM-FF}}$ is proved (Lemma 1).
- A new parameter β is defined as a function of α (Lemma 2). This parameter is a key concept in the derivation of $U_{\text{wc}}^{\text{RM-FF}}$. In addition, it provides a simple schedulability condition, which states that any task set made up of $m \leq \beta n$ tasks is schedulable. It is not worth obtaining the utilization bound when $m \leq \beta n$, as in this case the task set is directly schedulable.
- The utilization bound for task sets with $m > \beta n$ tasks is calculated. This last step is relatively complex, so further on it is divided into five substeps.

Next, Lemma 1 is presented, which proves the existence of the utilization bound for RM scheduling and the FF allocation algorithm.

LEMMA 1. *There exists one utilization bound for RM scheduling and FF allocation, which is a function of the number of tasks, m, the number of processors, n, and the maximum reachable utilization factor, α.*

Proof. Let $\Pi(m, n, \alpha)$ be the set of all the positive real numbers, π, fulfilling the following condition: any task set made up of m tasks, of utilization factors $0 < u_i \leq \alpha$, and total utilization $U \leq \pi$ fits into n processors, using Liu & Layland's schedulability condition for RM scheduling, and FF allocation. The set $\Pi(m, n, \alpha)$ is not empty, as any task set of total utilization $\ln 2$ or less fits into one processor, and therefore also fits into n processors using FF allocation. In addition, all the elements of $\Pi(m, n, \alpha)$ are less than or equal to the finite value n, as any task set of total utilization greater than n does not fit into n processors. Therefore, a maximum in $\Pi(m, n, \alpha)$ exists, termed $\pi_{\max}(m, n, \alpha)$, which is a function of m, n and α. Next, we will prove that $\pi_{\max}(m, n, \alpha)$ is the utilization bound, i.e, it fulfills the two properties given in Definition 1.

Any task set of total utilization less than or equal to $\pi_{\max}(m, n, \alpha)$ fits into n processors, as $\pi_{\max}(m, n, \alpha)$ is an element of $\Pi(m, n, \alpha)$. Furthermore, being $\pi_{\max}(m, n, \alpha)$ the maximum of Π implies that at least one set of m tasks exists, of total utilization $\pi_{\max}(m, n, \alpha) + \epsilon$,

with $\epsilon \to 0^+$, which does not fit into the processors.[1]. If this were not so, $\pi_{\max}(m,n,\alpha)+\epsilon$ would be an element of Π greater than the maximum, $\pi_{\max}(m,n,\alpha)$, which is not possible. Subsequently, for any total utilization of value $\pi_{\max}(m,n,\alpha)+\epsilon$, at least one set of m tasks which does not fit into n processors exists. For any total utilization greater than $\pi_{\max}(m,n,\alpha)+\epsilon$, it is even easier to find a set of m tasks which does not fit into n processors. This proves the last property of Definition 1.

We conclude that the utilization bound exists, and it is equal to $\pi_{\max}(m,n,\alpha)$. □

At this point, we introduce a new parameter β, defined as *the maximum number of tasks of utilization factor α, which fit into one processor*. This parameter is a key concept in the derivation of $U_{\text{wc}}^{\text{RM-FF}}$, and gives rise to a simple schedulability condition.

From the above definition it is clear that β is a function of the maximum utilization factor, α. This function is given by Lemma 2.

LEMMA 2.

$$\beta = \left\lfloor \frac{1}{\log_2(\alpha+1)} \right\rfloor \tag{4}$$

Proof. From the definition of β, β tasks of utilization factor α fit into one processor. Applying Liu & Layland's bound for RM scheduling this means that $\beta\alpha \leq \beta(2^{1/\beta}-1)$. Finding β we obtain $\beta \leq 1/\log_2(\alpha+1)$. Since β is an integer value we get

$$\beta \leq \left\lfloor \frac{1}{\log_2(\alpha+1)} \right\rfloor \tag{5}$$

Since β is the maximum number of tasks of utilization factor α that fit into one processor, $(\beta+1)$ tasks of utilization factor α do not fit into one processor. Thus, $(\beta+1)\alpha > (\beta+1)(2^{1/(\beta+1)}-1)$. Finding β we obtain $\beta > 1/\log_2(\alpha+1)-1$. Since β is an integer value we get

$$\beta \geq \left\lfloor \frac{1}{\log_2(\alpha+1)} \right\rfloor \tag{6}$$

The lemma is proved from (5) and (6). □

The value of β can be used to establish the schedulability of some task sets. From the definition of β, β tasks of utilization factor α fit into each processor. Since all the tasks have utilization factors less than or equal to α, at least β tasks of arbitrary utilization factors

[1] The expression $\epsilon \to 0^+$ is equivalent to $\epsilon \to 0$, and $\epsilon > 0$.

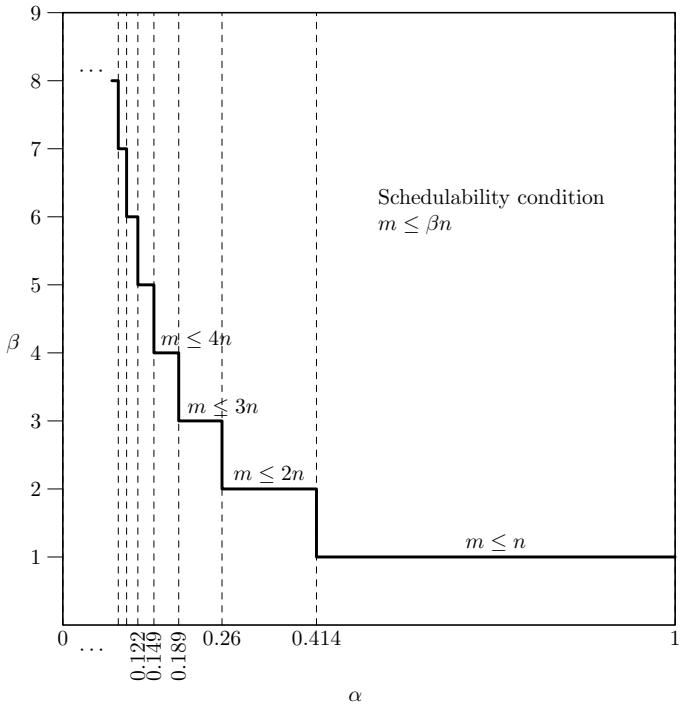

Figure 1. Representation of the function $\beta(\alpha)$, and the associated schedulability condition.

($\leq \alpha$) fit into each processor. Therefore, a multiprocessor made up of n processors can allocate at least βn tasks. Subsequently, if a task set is made up of m tasks and $m \leq \beta n$, the task set is schedulable by RM and any *reasonable* allocation algorithm on n processors. By a *reasonable* allocation algorithm, we mean one which fails to allocate a task only when there is no processor in the system with enough remaining capacity to receive the task.

Figure 1 depicts β as a function of α, and also shows the sufficient schedulability condition $m \leq \beta n$. For example, if α is in the interval $(2^{1/3} - 1, 2^{1/2} - 1] \approx (0.26, 0414]$ then $\beta = 2$. In this case, the task set is schedulable if it has $2n$ tasks or less.

Another consequence of the schedulability condition $m \leq \beta n$ is that it is worthwhile to obtain the value of the utilization bound only for the case $m > \beta n$. Otherwise, the task set is directly schedulable. From now we assume $m > \beta n$, so the above schedulability condition can not be applied.

The rest of this section is devoted to the calculation of the utilization bound for RM scheduling and FF allocation, under the restriction $m > \beta n$. The strategy in the calculation is the following:

1. Some mathematical relationships used in the proofs are presented.
2. Theorem 1 gives an upper limit on the utilization bound.
3. Lemma 3 is proved. This lemma is necessary in order to prove Theorem 2.
4. Theorem 2 proves an expression which relates the utilization bound for m tasks and n processors, with the utilization bound for $(m-\beta)$ tasks and $(n-1)$ processors.
5. From the result given in step 4, and Liu & Layland's bound for uniprocessors, Theorem 3 obtains a lower limit on the utilization bound. The upper and lower limits on the utilization bound given in steps 2 and 5 are the same. Finally, Theorem 3 gives the exact value of the utilization bound, which coincides with the upper and lower limits.

Before calculating $U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$, some relationships of the positive integer numbers, $\mathbb{Z}^+$, are presented without proof.

(i) $x+1 \leq y \quad \forall x,y \in \mathbb{Z}^+ \mid x<y$

(ii) $(2^{1/x}-1)x > (2^{1/y}-1)y > \ln 2 \quad \forall x,y \in \mathbb{Z}^+ \mid x<y$

(iii) $(2^{1/(x+1)}-1)x < (2^{1/(y+1)}-1)y < \ln 2 \quad \forall x,y \in \mathbb{Z}^+ \mid x<y$

These will be referred to as Relationship (i), Relationship (ii), and Relationship (iii).

The following theorem gives an upper limit on the utilization bound using RM scheduling and FF allocation. The proof is based on finding a task set which does not fit into the processors.

THEOREM 1. *If $m > \beta n$ then*

$$\begin{aligned} U_{wc}^{RM\text{-}FF}(m,n,\alpha) \leq\ & (n-1)(2^{1/(\beta+1)}-1)\beta \\ & + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1) \end{aligned} \tag{7}$$

Proof. Let us define

$$\begin{aligned} g(m,n,\alpha) =\ & (n-1)(2^{1/(\beta+1)}-1)\beta \\ & + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1) \end{aligned}$$

We will prove that there exists a set of m tasks, $\{\tau_1,\ldots,\tau_m\}$, with utilization factors $0 < u_i \leq \alpha$ for all $i = 1,\ldots,m$, and total utilization $U = g(m,n,\alpha)+\epsilon$, given $\epsilon \to 0^+$, which does not fit into n processors, using FF allocation and Liu & Layland's bound for RM scheduling.

This set of m tasks is made up of two subsets: a first subset with $(m - \beta n)$ tasks, and a second subset with βn tasks.

All the tasks of the first subset have the same utilization factor of value

$$u_i = \frac{(m - \beta(n-1))(2^{1/(m-\beta(n-1))} - 1) - (2^{1/(\beta+1)} - 1)\beta}{(m - \beta n)} \quad (8)$$

where $i = 1, \ldots, (m - \beta n)$.

All the tasks of the second subset have the same utilization factor of value

$$u_i = (2^{1/(\beta+1)} - 1) + \frac{\epsilon}{\beta n}$$

where $i = (m - \beta n + 1), \ldots, m$.

It can be easily checked that the total utilization of the whole task set is $g(m, n, \alpha) + \epsilon$.

Firstly, it is necessary to prove that the utilization factors of both subsets are valid, i.e, $0 < u_i \leq \alpha$ for all $i = 1, \ldots, m$.

Check of the utilization factors of the first subset.

By hypothesis, $m > \beta n$, so $m - \beta(n-1) > \beta$. Applying Relationship (i) we get $m - \beta(n-1) \geq \beta + 1$. Now applying Relationship (ii) makes $(m - \beta(n-1))(2^{1/(m-\beta(n-1))} - 1) \leq (\beta + 1)(2^{1/(\beta+1)} - 1)$. Considering this expression and equation (8) we get

$$u_i \leq \frac{(2^{1/(\beta+1)} - 1)}{(m - \beta n)} \quad (9)$$

On one hand, Lemma 2 provides the value $\beta = \lfloor 1/\log_2(\alpha + 1) \rfloor$. Thus, $(\beta + 1) > 1/\log_2(\alpha + 1)$, and finding α

$$\alpha > (2^{1/(\beta+1)} - 1) \quad (10)$$

On the other hand $m > \beta n$ by hypothesis, so $(m - \beta n) > 0$, and applying Relationship (i) we get

$$m - \beta n \geq 1 \quad (11)$$

Substituting (10), and (11) into (9) proves that $u_i < \alpha$ for all the tasks of the first subset.

Next we will prove that all the utilization factors of the first subset are greater than zero. From Relationships (ii), (iii), and equation (11) we get

$$(m - \beta(n-1))(2^{1/(m-\beta(n-1))} - 1) > \ln 2$$
$$(2^{1/(\beta+1)} - 1)\beta < \ln 2$$
$$m - \beta n \geq 1$$

Substituting the above expressions into equation (8) gives $u_i > 0$ for all the tasks of the first subset.

Check of the utilization factors of the second subset.

It is always possible to find one real number between two real numbers. Hence, from equation (10), a positive value $\epsilon/(\beta n)$ must exist such that

$$\alpha > (2^{1/(\beta+1)} - 1) + \frac{\epsilon}{\beta n} = u_i \tag{12}$$

which proves that the utilization factors of the second subset are less than α when $\epsilon \to 0^+$. In addition, the utilization factors of the second subset are obviously greater than zero.

From the above results, we conclude that the proposed task set is valid. Next we prove that it does not fit into n processors, using Liu & Layland's bound for RM scheduling and FF allocation.

The first subset of tasks, $\{\tau_1, \ldots, \tau_{m-\beta n}\}$, and the first β tasks of the second subset, $\{\tau_{m-\beta n+1}, \ldots, \tau_{m-\beta n+\beta}\}$, do not fit into processor P_1, since the total utilization of these tasks is over Liu & Layland's bound.

$$\begin{aligned}\sum_{i=1}^{m-\beta n+\beta} u_i &= \sum_{i=1}^{m-\beta n} u_i + \sum_{i=m-\beta n+1}^{m-\beta n+\beta} u_i \\ &= (m-\beta(n-1))(2^{1/(m-\beta(n-1))} - 1) + \frac{\epsilon}{n} \\ &> (m-\beta(n-1))(2^{1/(m-\beta(n-1))} - 1)\end{aligned}$$

However, from the above expression it can be proved that if task $\tau_{m-\beta n+\beta}$ is removed, then the first subset of tasks, and the first $(\beta-1)$ tasks of the second subset do fit into processor P_1.

Hence, there are $\beta(n-1)+1$ tasks left of utilization factor, $(2^{1/(\beta+1)} - 1) + \frac{\epsilon}{\beta n}$, which FF tries to allocate to the last $(n-1)$ processors, $\{P_2, \ldots, P_n\}$.

No processor in the set $\{P_2, \ldots, P_n\}$ can allocate $(\beta+1)$ or more tasks of the second subset, since $(\beta+1)$ of these tasks together have a utilization over Liu & Layland's bound.

$$(\beta+1)\left((2^{1/(\beta+1)} - 1) + \frac{\epsilon}{\beta n}\right) > (\beta+1)(2^{1/(\beta+1)} - 1)$$

However, each processor in $\{P_2, \ldots, P_n\}$ can allocate β tasks, as by the definition of β, at least β tasks can be allocated to each processor.

Subsequently, tasks $\{\tau_{m-\beta n+\beta}, \ldots, \tau_{m-1}\}$ are allocated to processors, but the last one, τ_m, can not be allocated to any processor.

We conclude that the proposed task set of total utilization $g(m,n,\alpha)+\epsilon$ does not fit into n processors when $\epsilon \to 0^+$, so the utilization bound, $U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$, must be less than or equal to $g(m,n,\alpha)$. □

The proof of Theorem 2 requires Lemma 3, which is proved below. It relates the utilization bound for the same number of processors, but a different number of tasks.

LEMMA 3.

$$U_{wc}^{RM\text{-}FF}(q,n,\alpha) \geq U_{wc}^{RM\text{-}FF}(m,n,\alpha) \quad \textit{for all } q < m$$

Proof. This lemma will be proved by contradiction.

Let us suppose that a pair of integers q and m exist, such that $q < m$, and $U_{\text{wc}}^{\text{RM-FF}}(q,n,\alpha) < U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$. Between two real numbers, it is always possible to find another real number, so we can find an $\epsilon > 0$ such that

$$U_{\text{wc}}^{\text{RM-FF}}(q,n,\alpha) < U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) - \epsilon < U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$$

By the definition of utilization bound, there exists at least one set of q tasks, $\{\tau_1,\ldots,\tau_q\}$, of total utilization

$$\sum_{i=1}^{q} u_i = U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) - \epsilon$$

which does not fit into n processors. Next, we prove that this gives rise to a contradiction.

If we add to this task set $(m-q)$ new tasks, $\{\tau_{q+1},\ldots,\tau_m\}$, each of utilization factor $\epsilon/(m-q)$, we obtain a task set made up of m tasks of total utilization $\sum_{i=1}^{m} u_i = U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$, which fits into n processors. Hence, the first q tasks fit into n processors, which is a contradiction. □

Next, we prove an expression which relates the utilization bound of multiprocessors with n and $(n-1)$ processors. This will allow us to obtain a lower limit for the utilization bound, going from the case $n=1$ (uniprocessor case) to a general multiprocessor case with an arbitrary n.

THEOREM 2. *If $m > \beta n$ then*

$$U_{wc}^{RM\text{-}FF}(m,n,\alpha) \geq (2^{1/(\beta+1)}-1)\beta + U_{wc}^{RM\text{-}FF}(m-\beta,n-1,\alpha)$$

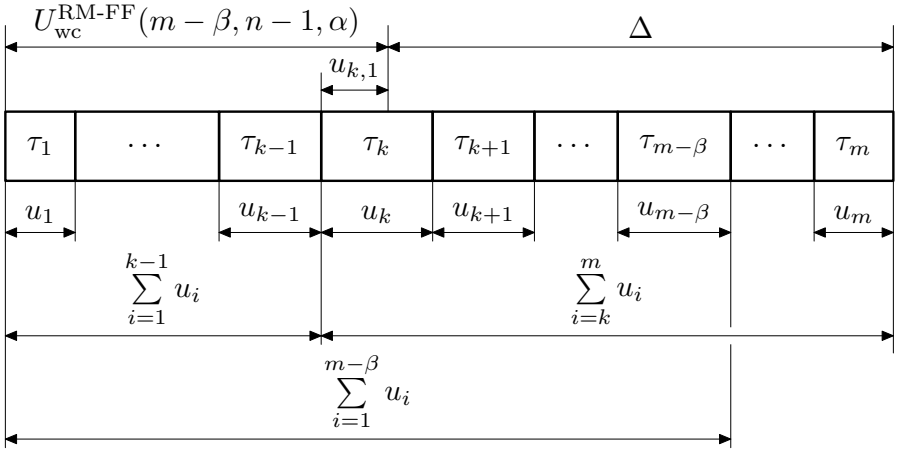

Figure 2. General situation in *case 2* of Theorem 2.

Proof. We will prove that any set of m tasks $\{\tau_1, \ldots, \tau_m\}$, with utilization factors $0 < u_i \leq \alpha$ for all $i = 1, \ldots, m$, and total utilization less than or equal to

$$(2^{1/(\beta+1)} - 1)\beta + U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$$

fits into n processors using Liu & Layland's bound for RM scheduling and FF allocation.

There are two possible cases:

Case 1: The first $(m-\beta)$ tasks have a total utilization less than or equal to $U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$, that is, $\sum_{i=1}^{m-\beta} u_i \leq U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$. In this case the whole set of m tasks always fits into n processors, because the first $(m-\beta)$ tasks fit into the first $(n-1)$ processors (since its utilization is below the bound), and the remaining β tasks fit into the last processor, since the definition of β implies that a least β tasks always fit into one processor.

Case 2: The first $(m-\beta)$ tasks have a total utilization greater than $U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$, that is, $\sum_{i=1}^{m-\beta} u_i > U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$. In this case we will prove that the whole set of m tasks still fits into n processors if the total utilization is equal to $U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)+\Delta$, provided $\Delta \in \mathbb{R}$, and $\Delta \leq (2^{1/(\beta+1)} - 1)\beta$.

A task τ_k must exist, whose u_k added to the previous utilizations u_i, causes the bound $U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$ to be exceeded. This situation is depicted in Figure 2, which is a graphical representation of the utilization factors of each task and the relationships between several quantities and summations used through this proof, for a generic set of m tasks in *Case 2*. It is important to observe that each rectangle in Figure 2 represents one of the m tasks making up the set, and the

horizontal dimension of each rectangle gives the utilization factor of the task it represents.

The value of k is obtained as the integer which fulfills:

$$\sum_{i=1}^{k-1} u_i \leq U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) < \sum_{i=1}^{k} u_i$$

Note that $k \leq m - \beta$ (if $k > m - \beta$ we would be in case 1).

It can be seen that the first $(k-1)$ tasks fit into the first $(n-1)$ processors. The total utilization of the first $(k-1)$ tasks fulfills

$$\sum_{i=1}^{k-1} u_i \leq U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha)$$

Bearing in mind that $k-1 < m-\beta$ in *Case 2*, and applying Lemma 3, we get

$$U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) \leq U_{\text{wc}}^{\text{RM-FF}}(k-1, n-1, \alpha)$$

and thus

$$\sum_{i=1}^{k-1} u_i \leq U_{\text{wc}}^{\text{RM-FF}}(k-1, n-1, \alpha)$$

Therefore, the first $(k-1)$ tasks fit into the first $(n-1)$ processors. We only have to prove that the remaining $(m-k+1)$ tasks fit into the last processor.

The worst situation in terms of schedulability appears when all the tasks τ_i in $\{\tau_k, \ldots, \tau_m\}$ fulfill $u_i > u_{k,1}$, where

$$u_{k,1} = U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) - \sum_{i=1}^{k-1} u_i$$

as shown in Figure 2. Note that if there were a task τ_i in $\{\tau_k, \ldots, \tau_m\}$ with $u_i \leq u_{k,1}$, we could always allocate this task to the first $(n-1)$ processors (since the addition of this new task does not cause the total utilization to exceed the bound), and the result would be a situation analogous to the current one, with k one unit greater. This reasoning can be repeated until no task τ_i with $u_i \leq u_{k,1}$ exists among the last $(m-k+1)$ tasks, or until we are in Case 1.

In order to prove that the last $(m-k+1)$ tasks fit into the last processor we have to prove that the total utilization of these tasks does not exceed Liu & Layland's bound, that is,

$$\sum_{i=k}^{m} u_i \leq (m-k+1)(2^{1/(m-k+1)} - 1)$$

Figure 2 shows that

$$\sum_{i=k}^{m} u_i = u_{k,1} + \Delta \tag{13}$$

As already stated, all the utilization factors u_i in this summation are greater than $u_{k,1}$, so

$$\begin{aligned}(m-k+1)u_{k,1} &< u_{k,1} + \Delta \\ &< u_{k,1} + (2^{1/(\beta+1)} - 1)\beta\end{aligned}$$

by the definition of Δ. Finding $u_{k,1}$ from the above equation

$$u_{k,1} < \frac{(2^{1/(\beta+1)} - 1)\beta}{m-k} \tag{14}$$

Substituting the value of $u_{k,1}$ from (14) into (13) we obtain

$$\begin{aligned}\sum_{i=k}^{m} u_i &< \frac{(2^{1/(\beta+1)} - 1)\beta}{m-k} + \Delta \\ &< \frac{(2^{1/(\beta+1)} - 1)\beta}{m-k} + (2^{1/(\beta+1)} - 1)\beta \qquad \text{by def. of } \Delta \\ &= \frac{(m-k+1)(2^{1/(\beta+1)} - 1)\beta}{m-k}\end{aligned}$$

We know that $m - k \geq \beta$ in Case 2, so applying Relationship (iii) we get

$$(2^{1/(\beta+1)} - 1)\beta \leq (2^{1/(m-k+1)} - 1)(m-k) \tag{15}$$

and,

$$\sum_{i=k}^{m} u_i \leq (m-k+1)(2^{1/(m-k+1)} - 1)$$

This equation shows that the last $(m - k + 1)$ tasks meet Liu & Layland's schedulability condition, so they are schedulable on the last processor.

We have proved that any task set with m tasks and a total utilization

$$\begin{aligned}U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) + \Delta \leq \\ U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) + (2^{1/(\beta+1)} - 1)\beta\end{aligned}$$

fits into n processors, so the utilization bound, $U_{\text{wc}}^{\text{RM-FF}}(m, n, \alpha)$, must be greater than or equal to $U_{\text{wc}}^{\text{RM-FF}}(m-\beta, n-1, \alpha) + (2^{1/(\beta+1)} - 1)\beta$, and the theorem is proved. □

The utilization bound for RM scheduling and FF allocation, given by Theorem 3, is obtained from Theorem 1 and Theorem 2.

THEOREM 3. *If $m > \beta n$ then*

$$U_{wc}^{RM\text{-}FF}(m,n,\alpha) = (n-1)(2^{1/(\beta+1)}-1)\beta + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1) \quad (16)$$

Proof. Firstly, we obtain a lower limit for the utilization bound for a set of m tasks on a multiprocessor with n processors.

Theorem 2 relates the utilization bound of sets of m tasks on multiprocessors of n processors, to the utilization bound of sets of $(m-\beta)$ tasks on multiprocessors with $(n-1)$ processors.

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) \geq (2^{1/(\beta+1)}-1)\beta + U_{\text{wc}}^{\text{RM-FF}}(m-\beta,n-1,\alpha) \quad (17)$$

Theorem 2 also relates the utilization bound of sets of $(m-\beta)$ tasks on multiprocessors of $(n-1)$ processors, to the utilization bound of sets of $(m-2\beta)$ tasks on multiprocessors of $(n-2)$ processors.

$$U_{\text{wc}}^{\text{RM-FF}}(m-\beta,n-1,\alpha) \geq (2^{1/(\beta+1)}-1)\beta + U_{\text{wc}}^{\text{RM-FF}}(m-2\beta,n-2,\alpha) \quad (18)$$

Substituting (18) into (17) we get

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) \geq 2(2^{1/(\beta+1)}-1)\beta + U_{\text{wc}}^{\text{RM-FF}}(m-2\beta,n-2,\alpha)$$

This procedure can be repeated until finally relating the utilization bound of sets of m tasks on multiprocessors of n processors, with the utilization bound of sets of $(m-\beta(n-1))$ tasks on a uniprocessor.

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) \geq (n-1)(2^{1/(\beta+1)}-1)\beta + U_{\text{wc}}^{\text{RM-FF}}(m-\beta(n-1),1,\alpha) \quad (19)$$

The utilization bound for $(m-\beta(n-1))$ tasks and one processor coincides with Liu & Layland's bound, which does not depend on the value of α.

$$U_{\text{wc}}^{\text{RM-FF}}(m-\beta(n-1),1,\alpha) = (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1) \quad (20)$$

Substituting (20) into (19) gives a lower limit on the utilization bound of m tasks on n processors.

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) \geq (n-1)(2^{1/(\beta+1)}-1)\beta + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1)$$

Theorem 1 proved that

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) \leq (n-1)(2^{1/(\beta+1)}-1)\beta + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1)$$

Thus, we finally conclude that

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha) = (n-1)(2^{1/(\beta+1)}-1)\beta + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1)$$

□

Therefore, any set of m tasks, with task utilization factors less than or equal to α, and total utilization less than or equal to

$$(n-1)(2^{1/(\beta+1)}-1)\beta + (m-\beta(n-1))(2^{1/(m-\beta(n-1))}-1)$$

is feasibly scheduled by RM on n processors using FF allocation, where $\beta = \lfloor 1/\log_2(\alpha+1) \rfloor$. This is a sufficient condition, analogous to that given by Liu & Layland for uniprocessor systems. For any value of total utilization greater than $U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$ it is possible to find a task set with this utilization such that it does not fit into the processors using FF allocation and Liu & Layland's schedulability condition for RM scheduling. In this case, the task set may be or may not be schedulable.

Having calculated the general expression of the utilization bound, and making $\alpha = 1$, we can remove the restriction $u_i \leq \alpha$, since the utilization factors of the tasks can be now in the interval $(0, 1]$. In this case $\beta = 1$, and the utilization bound is given by

$$U_{\text{wc}}^{\text{RM-FF}}(m,n,1) = (n-1)(2^{1/2}-1) + (m-n+1)(2^{1/(m-n+1)}-1) \quad (21)$$

Notice that α need not to be equal to the maximum utilization factor of the task set. All is needed to have a valid election of α is to fulfill $u_i \leq \alpha \leq 1$ for all the tasks. Thus, $\alpha = 1$ is always a valid election. However in practice, α should be chosen as the maximum utilization factor in order to maximize the utilization bound.

4. Analysis of the theoretical results

In this section, we analyze the expression of the utilization bound $U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha)$, for RM scheduling and FF allocation, given by equation (16).

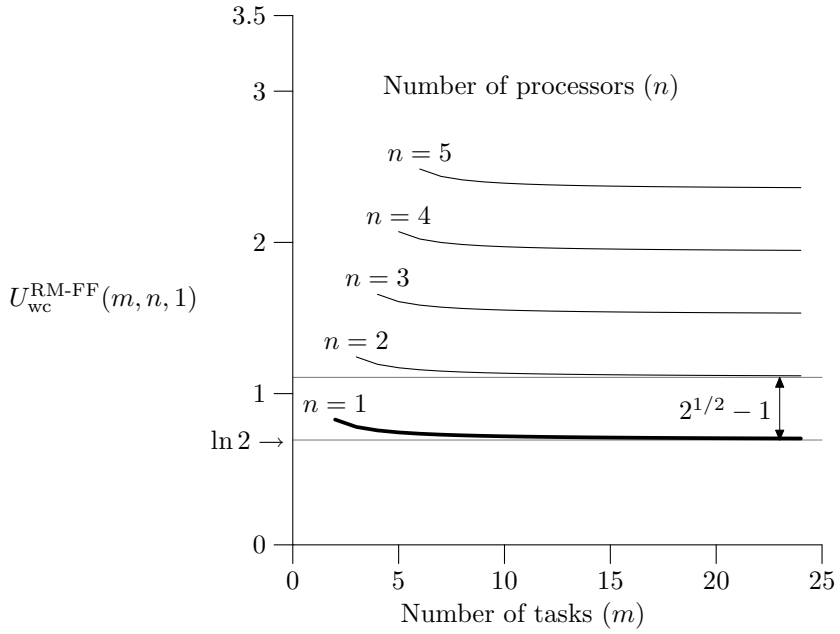

Figure 3. Comparison between the utilization bound of uniprocessors and multiprocessors, for the case $\alpha = 1$, i.e, $\beta = 1$.

In spite of m and n being integers, the function $U_{\text{wc}}^{\text{RM-FF}}(m, n, \alpha)$ is represented as continuous function in Figures 3, 4, 5, and 6, with the aim of improving its visualization.

Figure 3 depicts the utilization bound for RM scheduling and FF allocation as a function of the number of tasks, for different numbers of processors, and $\alpha = 1$. The bold line provides the utilization bound for the uniprocessor case, which coincides with Liu & Layland's result $U_{\text{wc}}^{\text{RM-FF}}(m, 1, \alpha) = m(2^{1/m} - 1)$. The addition of each new processor increments the utilization bound at least in $(2^{1/2} - 1) \approx 0.414$.

Figure 4 is analogous to Figure 3, but shows the situation when all the tasks have utilization factors less than or equal to $(2^{1/3} - 1) \approx 0.26$. We observe a substantial improvement with regard to the case $\alpha = 1$. Now, adding a new processor increments at least in $(2^{1/4} - 1)3 \approx 0.57$ the utilization bound.

Figure 5 depicts the utilization bound for RM scheduling and FF allocation as a function of the number of processors, for a different number of tasks, and $\alpha = 1$. In addition, it shows the expression $(2^{1/2} - 1)n$, proposed by Oh and Baker (1998) as a dashed line, in order to compare this expression with the utilization bound. The improvement is substantial, particularly if the number of processors is small.

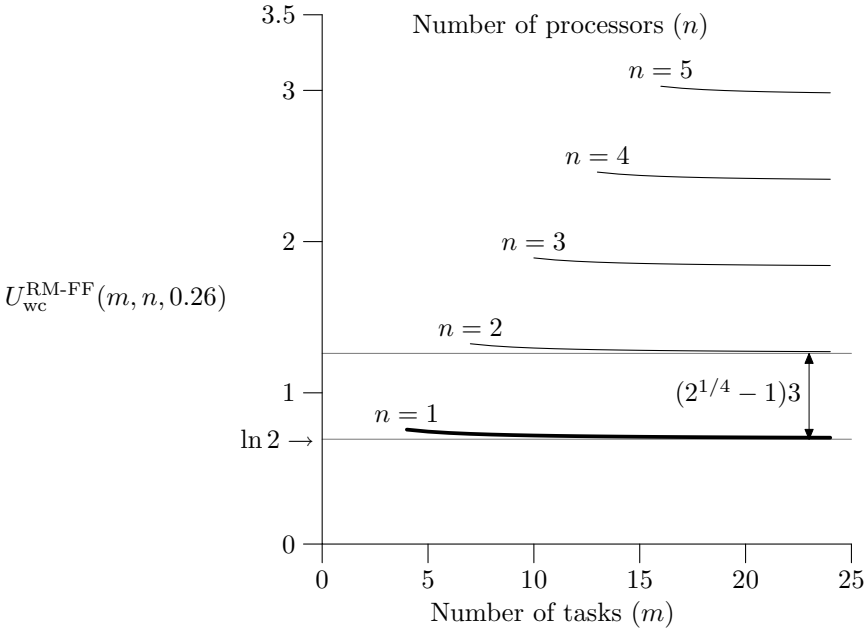

Figure 4. Comparison between the utilization bound of uniprocessors and multiprocessors, for the case $\alpha = (2^{1/3} - 1) \approx 0.26$, i.e, $\beta = 3$.

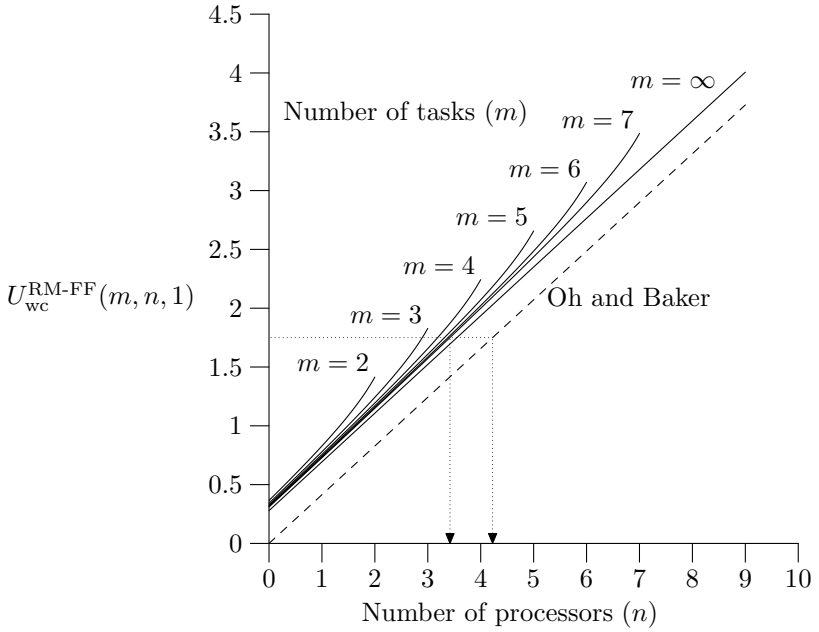

Figure 5. Utilization bound for the case $\alpha = 1$, i.e, $\beta = 1$, in function of n for different values of m.

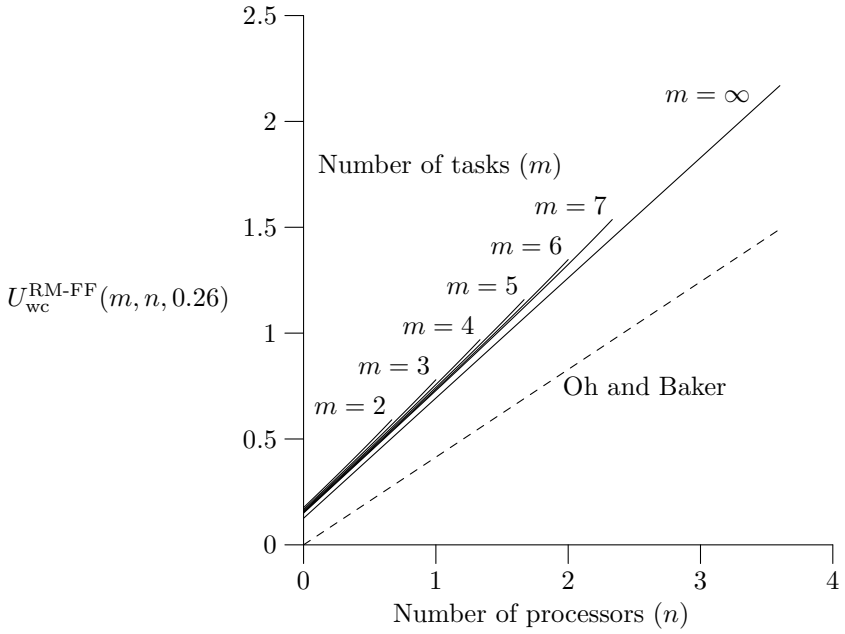

Figure 6. Utilization bound for the case $\alpha = (2^{1/3} - 1) \approx 0.26$, i.e, $\beta = 3$, in function of n for different values of m.

Figure 6 is analogous to Figure 5 for $\alpha = 0.26$, and again shows the improvement with regard to the case $\alpha = 1$.

In general, any decrement of α produces an increment of β and an increment of the utilization bound. In the limit, if $\alpha \to 0$ then $\beta \to \infty$, and it follows

$$\lim_{\alpha \to 0} U_{\text{wc}}^{\text{RM-FF}}(m, n, \alpha) = n \ln_2$$

Hence, if the utilization factor of all the tasks is low, the multiprocessor system using RM scheduling and FF allocation, approximates an ideal uniprocessor n-times faster than each processor of the multiprocessor.

We can obtain a utilization bound independent of the number of tasks making $m = \infty$ in equation (16).

$$U_{\text{wc}}^{\text{RM-FF}}(m, n, \alpha) \geq U_{\text{wc}}^{\text{RM-FF}}(\infty, n, \alpha) = \ln 2 + (n-1)(2^{1/(\beta+1)} - 1)\beta$$

Figures 5 and 6 are also useful in the design stage to obtain the worst-case number of processors, $n_{\text{wc}}^{\text{RM-FF}}(m, U, \alpha)$, required to feasibly schedule a task set using RM scheduling and FF allocation. A multiprocessor made up of $n_{\text{wc}}^{\text{RM-FF}}(m, U, \alpha)$ processors can allocate any set of m periodic tasks, of utilization factors $0 < u_i \leq \alpha \leq 1$, and total utilization less than or equal to U. Furthermore, $n_{\text{wc}}^{\text{RM-FF}}(m, U, \alpha) - 1$

processors may not be enough, and it is possible to find at least one set of m periodic tasks, of utilization factors $0 < u_i \leq \alpha \leq 1$, and total utilization less than or equal to U, which can not be allocated.

For example, consider a task set made up of $m = 7$ tasks, of total utilization $U = 1.75$, and arbitrary utilization factors. From Figure 5, applying the expression proposed by Oh and Baker (1998) five processors are needed. However, applying the utilization bound presented in this paper, we only need $n_{\text{wc}}^{\text{RM-FF}}(7, 1.75, 1) = 4$ processors. If we use three processors it is possible to find a set of seven tasks which is not schedulable by RM and FF. For instance, the task set with utilization factors $\{u_1 = 0.01, u_2 = 0.01, u_3 = 0.01, u_4 = 0.43, u_5 = 0.43, u_6 = 0.43, u_7 = 0.43\}$ does not fit into three processors.

Figures 5 and 6 are only valid for $m > \beta n$. Therefore, if there is not a point in these figures for some pair of values (m, U), the worst-case number of processors is obtained as the minimum integer value greater than or equal to m/β.

$$n_{\text{wc}}^{\text{RM-FF}}(m, U, \alpha) = \left\lceil \frac{m}{\beta} \right\rceil \tag{22}$$

For example, for $m = 3$, $U = 2.5$, and $\beta = 1$, there is not a point in Figure 5, and so $n_{\text{wc}}^{\text{RM-FF}}(3, 2.5, 1) = \lceil 3/1 \rceil = 3$ processors.

5. Average-case behaviour

Section 3 provided the utilization bound for RM scheduling and FF allocation. Any task set of total utilization less than or equal to the utilization bound is schedulable. In addition, for any given value of total utilization greater than the utilization bound, task sets which do not fit into the processors using Liu & Layland's schedulability condition and FF allocation exist. However, tasks sets of total utilization greater than the utilization bound may be schedulable. In fact, the utilization bound is obtained for pessimistic task sets which might be very infrequent in practice.

In order to perceive the pessimism of the utilization bound, we will define the *average-case utilization bound*, $U_{\text{ac}}^{\text{RM-FF}}(m, n, p)$, for RM scheduling and FF allocation as follows.

DEFINITION 2. *A task set made up of m tasks, of total utilization $U = U_{ac}^{RM\text{-}FF}(m, n, p)$ is schedulable by RM and FF on n processors with a probability equal to $p\%$.*

If sets of m tasks with total utilization $U = U_{\text{ac}}^{\text{RM-FF}}(m, n, p)$ are randomly generated, $p\%$ of the task sets are schedulable by RM and

FF on n processors. The other $(100-p)\%$ corresponds to task sets which do not fit into the processors. Obviously, the average-case utilization bound must be greater than or equal to the (worst-case) utilization bound.

$$U_{\text{ac}}^{\text{RM-FF}}(m,n,p) \geq U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha=1)$$

The equality corresponds to the case $p = 100\%$.

$$U_{\text{ac}}^{\text{RM-FF}}(m,n,p=100\%) = U_{\text{wc}}^{\text{RM-FF}}(m,n,\alpha=1)$$

There are several problems in the definition of the average-case utilization bound. Firstly, the utilization of the tasks making up the set must follow some statistical distribution which must be defined. Secondly, the function $U_{\text{ac}}^{\text{RM-FF}}(m,n,p)$ must be calculated for any value of m, n and p from the statistical distribution.

The statistical distribution chosen to generate the task sets is the *beta distribution*. It is a continuous distribution of probability given by:

$$f(x;a,b) = \begin{cases} \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} x^{a-1}(1-x)^{b-1} & 0 < x < 1 \\ 0 & \text{for any other value} \end{cases}$$

This distribution has two positive parameters a, and b, which allow us to select the mean μ, and standard deviation σ of the distribution.

$$\mu = \frac{a}{(a+b)}; \quad \sigma^2 = \frac{ab}{(a+b)^2(a+b+1)}$$

The maximum σ for a given value of μ is $\sigma_{\max} = \sqrt{\mu(1-\mu)}$, which is obtained by considering the restriction $a > 0$.

The results obtained using the Beta distribution must be considered carefully, as this distribution need not to produce realistic task sets. However, it is useful to generate the utilization factors of the tasks, as it generates random values in the interval $(0,1)$.

Varying the value of the standard deviation from zero to the maximum, the utilization factors of the tasks vary from being equal to having unlike values. In order to assess the pessimism of the (worst-case) utilization bound $U_{\text{wc}}^{\text{RM-FF}}$, the average-case utilization bound was obtained by simulation, for standard deviations going from $\sigma = 0.001\sigma_{\max}$ to $\sigma = 0.9\sigma_{\max}$, and for $p = 99\%$. Standard deviations greater than $0.9\sigma_{\max}$ are not practical as all the utilization factors generated randomly are approximately zero or one. In any case, the average-case utilization bound for such high values of standard deviations is close to that given for $\sigma = 0.9\sigma_{\max}$.

Figure 7 depicts the average-case utilization bound for arbitrary standard deviation, which has been obtained as the minimum among

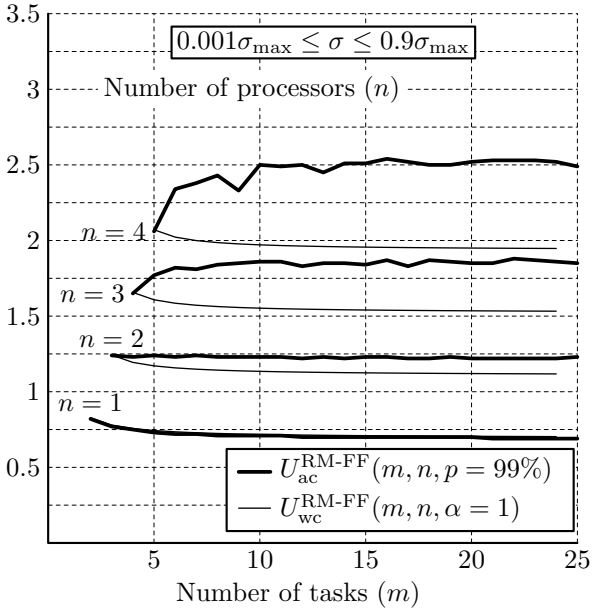

Figure 7. Average-case utilization bound for arbitrary standard deviation.

the standard deviations going from $\sigma = 0.001\sigma_{\max}$ to $\sigma = 0.9\sigma_{\max}$, of the associated curves of average-case utilization bound. Each point $\{m, n, U_{\text{ac}}^{\text{RM-FF}}(m, n, p = 99\%)\}$ in Figure 7 indicates that a set of m tasks of total utilization $U_{\text{ac}}^{\text{RM-FF}}(m, n, p = 99\%)$ can be feasibly scheduled by RM and FF on n processors with a probability greater than or equal to 99%, for arbitrary standard deviation.

Figure 7 also represents the (worst-case) utilization bound to be compared with the average-case utilization bound. The difference is significant. Nevertheless, for the uniprocessor case, the average-case utilization bound coincides with the (worst-case) utilization bound, given by $m(2^{1/m} - 1)$.

6. Other allocation heuristics

This paper basically focuses on RM scheduling with FF allocation. However, other allocation heuristics exist, and it is interesting to calculate the utilization bound for RM scheduling and these heuristics.

One of the first allocation heuristics found in the literature is the Next Fit (NF) algorithm (Dall and Liu, 1978). It performs in much the same way as the FF algorithm, except when a task does not fit into a processor. In this case, the processor is discarded in the allocation of the following tasks. Therefore, a task can be said to be non-schedulable

even when it fits in one of the discarded processors. As a result, the NF algorithm presents poor performance with regard to the FF algorithm, which is expressed in terms of the metric $(N_{\mathrm{NF}}/N_{\mathrm{opt}})$. Dall and Liu (1978) obtained $(N_{\mathrm{NF}}/N_{\mathrm{opt}}) = 2.67$, compared with the value $(N_{\mathrm{FF}}/N_{\mathrm{opt}}) = 2.33$, obtained by Oh and Son (1995) for FF allocation.

Another common algorithm found in the literature is the Best Fit (BF) (Garey and Johnson, 1979). This algorithm assigns each task to the processor having the lowest remaining capacity among those processors with enough capacity. Intuitively, BF seems to be an improvement over the FF algorithm, so it should provide better performance. However, simulation experiments carried out by the authors with BF allocation gave almost identical results to those using FF allocation, so they have not been depicted. In addition, the utilization bound associated to RM scheduling with BF allocation is the same as that for FF allocation. The proof is analogous to that presented for FF allocation, so for the sake of brevity, we provide only the guidelines . The task set used in theorem 1 (to prove the upper limit on the utilization bound for RM-FF) does not fit into the processors using BF allocation, so the upper limit is also valid for BF allocation. Theorem 2 is also valid for RM-BF, but proving the statement "the worst situation in terms of schedulability appears when all the tasks τ_i in $\{\tau_k, \ldots, \tau_m\}$ fulfill $u_i > u_{k,1}$", requires some elaboration for BF allocation. Therefore, the utilization bound for BF allocation coincides with the upper and lower limits, an it is equal to the utilization bound for FF allocation.

Better approximation algorithms can be obtained by observing that the worst performance for both FF and BF occurs when tasks with low utilization factors appear before tasks greater utilization factors. This is the case of the task set used in theorem 1 to obtain the upper limit on the utilization bound. To improve the performance of the FF and BF algorithms, the tasks with the lowest utilization factors are allocated first. The resulting algorithms are called First Fit Decreasing (FFD) and Best Fit Decreasing (BFD) respectively.

The utilization bound for FFD or BFD can not be lower than that for FF or BF respectively. This can be proved with the following argument. Let $\mathcal{S}$ be the superset made up of all the task sets which do not fit into the processors using FF allocation. There must be a task set in $\mathcal{S}$, whose total utilization is the minimum among all the task sets in $\mathcal{S}$. The value obtained by subtracting an $\epsilon \to 0$ from this minimum is the utilization bound for RM-FF. Let $\mathcal{S}'$ be the superset made up of all the task sets ordered in decreasing utilization factors, which do not fit into the processors using FF allocation. There must exist a task set in $\mathcal{S}'$, whose total utilization is the minimum among all the task sets in $\mathcal{S}'$. The value obtained by subtracting an $\epsilon \to 0$ from this minimum is the

utilization bound for RM-FFD. Since $\mathcal{S}'$ is a subset of $\mathcal{S}$, its minimum can not be lower than the minimum of $\mathcal{S}$, and so the utilization bound for RM-FFD can not be lower than the utilization bound for RM-FFD. The same argument can be applied to BF and BFD allocations.

In addition, the task set proposed in theorem 1 does fit into the processors using FFD or BFD allocation. Thus, the utilization bound for FFD and BFD allocation may be greater than that for FF and BF allocation. Nevertheless, further investigation into this last part is required.

7. Conclusions and future work

Liu & Layland's schedulability bound for RM scheduling on multiprocessors has been extended to multiprocessors under a partitioning strategy and FF allocation. This bound is a function of the number of tasks, number of processors, and maximum reachable utilization factor of the task set. For the case of tasks with low utilization factors the utilization bound is significantly raised, reaching asymptotically the value $n \ln 2$ when all the utilization factors are close to zero. Allowing a low percentage of non-schedulable task sets, simulation has shown the pessimism of the utilization bound.

In general, the calculation of utilization bounds is a problem of importance in the real-time theory. Utilization bounds allow us not only to perform fast schedulability tests, but also to perform a schedulability analysis. That is, utilization bounds allow us to establish the influence of different parameters such as the number of tasks, task size, etc, on the schedulability of the system by considering the worst-case. In addition, utilization bounds indicate how far the system is from the ideal situation, in which, the total utilization equals the number of processors in the system.

Other allocation heuristics apart from FF have been dealt with briefly. The NF algorithm was shown to have worse performance than the FF algorithm. Since NF presents no practical advantage over FF, NF will not be considered in future works.

Simulation results have shown a behaviour of BF allocation almost identical to that of FF. This behaviour was previously observed by other authors (Garey and Johnson, 1979; Oh and Son, 1995). The same behaviour was also documented in terms of the metric (N_A/N_{opt}) by Oh and Son (1995). In addition, the utilization bound for RM-FF and RM-BF are the same. This point was not proved in this paper, although the guidelines of the proof were provided. The computational cost associated to the BF allocation is in general greater that that

associated to FF allocation. Thus, we can state that FF allocation is superior to BF allocation for multiprocessor RM scheduling.

In order to improve the utilization bound for RM-FF scheduling, we have investigated the utilization bound for FFD and BFD allocation. We suspect that it is greater, but we do not have proof of this point. The calculation of the utilization bound for the heuristics FFD and BFD will be performed in future work.

Finally, the utilization bound presented in this paper has a major restriction. It assumes the simple model of tasks defined by Liu & Layland. In future work, we will try to develop new utilization bounds considering extensions to the task model, such as access to shared resources, aperiodic tasks, release jitter, or mode changes.

8. Acknowledgments

We would like to thank the referees. Their remarks allowed us to improve the quality of this work.

References

Burchard, A., J. Liebeherr, Y. Oh, and S. Son: 1995, 'New Strategies for Assigning Real-Time Tasks to Multiprocessor Systems'. *IEEE Transactions on Computers* **44**(12).

Dall, S. and C. Liu: 1978, 'On a Real-Time Scheduling Problem'. *Operations Research* **6**(1), 127–140.

Garey, M. and D. Johnson: 1979, *Computers and Intractability*, pp. 121–127. New York: W.H. Freman.

Liu, C. L. and J. Layland: 1973, 'Scheduling Algorithms for Multiprogramming in a Hard-Real-time Environment'. *Journal of the ACM* **20**(1), 46–61.

Oh, D. and T. Baker: 1998, 'Utilization Bounds for N-Processor Rate Monotone Scheduling with Static Processor Assignment'. *Real-Time Systems* **15**(2), 183–193.

Oh, Y. and S. Son: 1995, 'Allocating Fixed-Priority Periodic Tasks on Multiprocessor Systems'. *Real-Time Systems* **9**(3), 207–239.

S. Sáez, J. V. and A. Crespo: 1998, 'Using Exact Feasibility Tests for Allocating Real-Time Tasks in Multiprocessor Systems'. In: *Proceedings of the 10th Euromicro Workshop on Real-Time Systems*. pp. 53–60.