Applied Mathematics & Information Sciences
An International Journal

RT-DBP: A Multi-Criteria Priority Assignment Scheme For Real-Time Tasks Scheduling

L. Baccouche [1] *and H. Eleuch*[2]

[1] INSAT, National Institute of Applied Science and Technology, C.U. Nord, B.P 676, Tunis Cedex 1080, TUNISIA
[2] Institute for Quantum Studies and Department of Physics, Texas A&M University, TX 77843, USA

Received: Jul 18, 2010; Revised Apr. 24, 2011; Accepted May. 21, 2011
Published online: 1 May 2012

Abstract: Several real-time applications, such as E-commerce, multimedia and process control are subject to multiple criteria scheduling problems. In such systems service differentiation and quality of service (QoS) guarantees are necessary. Often the multi-class task model can be associated to multiple (m,k)-firm constraints indicating the degree of missed deadlines that the system can tolerate for each class. In this paper we study (m,k)-firm task scheduling and we propose a priority assignment scheme that takes into consideration various scheduling constraints. The approach is validated through simulation under various configuration patterns.

Keywords: Real-time systems, (m,k)-firm constraints, multi-criteria scheduling.

1. Introduction

Traditionally real-time tasks are classified as hard and soft tasks. Hard tasks are those for which deadlines miss is catastrophic and may have serious damage. Soft real-time tasks are used in soft real-time applications where time is important and should be considered during execution but no serious problem occurs if some deadlines are missed. E-commerce applications, stock trading, internet bids, media servers, process control and robotics are examples of soft real-time applications.

However, these systems need that the percentage of tolerated missed deadlines (loss rate) be clearly specified. The solution is given using Weakly Hard Real-Time Systems (WHRT). In this new real-time model, Bernat [1] specify that a real-time system can tolerate some deadline misses provided that their number is bounded and precisely distributed. A suitable approach is to use a window based loss rate, defined by using two constants. The most famous WHRT constraint is the (m,k)-firm constraint applied by Hamdaoui in [2]. It is based on two constants *m* and *k* and its principle is to guarantee that *m* tasks respect their deadlines among *k* consecutive tasks.

When the system becomes overloaded, in addition to the tolerated missed deadline, it can be useful to provide the application developer with a way to correctly point out, how much it is critical to the system that some tasks in particular meet their deadlines. Often developers use priority or importance attributes. Such systems have multi-class tasks model and need service differentiation oriented scheduling protocols. In such multi-class models, each class may have its own (m_i, k_i) parameters. Many problems concentrating in tasks schedulability under these constraints were investigated in the last ten years. However, little work has been done in combining the optimisation of the QoS and task scheduling subject to both real-time and (m,k)-firm constraints [4],[5],[6],[7],[9].Suitable approaches to real-time systems that can tolerate occasional deadline misses fall into two categories: static and dynamic and are dedicated to real-time networks. In the static algorithms, the priority is determined off line while using a stationary parameter, for example the ratio of success *m/k*. With dynamic algorithms, the priority is determined according to the state of the system. Most famous algorithms are DBP (Distance Based Priority) [2], Matrix-DBP [8], Extended-DBP [11] and DWCS (Dynamic Window-Constrained Scheduling) [12]. Dynamic approaches are interesting because they allow the scheduling algorithm to compute tasks priorities during execution. In this paper we give an adaptation of DBP and we propose a new formula for the assignment of the priority that takes into consideration mul-

* Corresponding author: e-mail: leila.baccouche@free.fr

tiple scheduling criteria. The proposed equation combines all the objectives that the scheduling algorithm wants to meet.

This paper is organised as follows. The application and the system model is presented in the following section. In section 3 we introduce the DBP algorithm which we have adapted to schedule real-time tasks. Section 4 presents RT-DBP the proposed algorithm and we develop the global index of priority, a novel dynamic priority assignment scheme. Evaluation is presented in Sections 5. Section 6 concludes this paper.

2. Task MIQSS Model

We adopt a multiple input queues single server (MIQSS) model. This model is useful when executing tasks with different priorities to be served by one server. We assume a non preemptive service for the server (which applies the scheduling algorithm). In our model, a task is described by an importance attribute which provides the application developer with a way to correctly point out, how much it is critical to the system that the task meets its deadline. Tasks are organised in N queues according to their importance imp_i by the following way:

–the highest importance corresponding to *imp* value 1 means that the task is very important. The next importance corresponding to *imp* value 2 means that the task should satisfy its deadline and so on. The lowest importance corresponding to *imp* value *N* is the default importance and means that a deadline miss for this task is not so important.

In addition, a task is characterized by timing attributes and (m,k)-firm constraints. The timing requirements of a task T_i are generally given by the following attributes:

–r_i the ready time, when the task arrives to the system.
–d_i the deadline, it indicates the requirement to complete the task before d_i.
–we_i the worst-case estimated execution time.
–st_i the slack time of T_i. It represents the maximum amount of time during which the task can be delayed and still satisfy its deadline. d_i, st_i and r_i are related by, st_i=d_i - r_i-we_i. Initially the slack time is computed using r_i but this attribute is dynamic and on time t, st_i = d_i - *t* - we_i.

Upon its arrival, a task is inserted in the corresponding queue according to its importance. All queues are sorted using the EDF policy [5] so that the task at the head is the one with the closest deadline. In addition, the application developer specifies the (m,k)-firm constraints of the task. We assume that within a single queue, all tasks have the same (m,k)-firm values. The closer *m* is to *k* the more priority the queue has.

3. DBP (Distance Based Priority) outline

This section briefly describes the DBP scheduling algorithm. We refer the reader to [2] for more details. Among the algorithms presented in the introduction, DBP is the one we retained for the scheduling of real-time tasks. Indeed, DBP is dynamic what makes it possible to calculate the priority during the execution and to take into account other scheduling criteria. DBP uses the history of the execution to calculate the priority DBP of each queue and determine the queue which is going to miss its (m,k)-firm requirements and be in dynamic failure state (less than *m* tasks among *k* consecutive tasks respect their deadlines). The selected queue is considered of high priority DBP and the task at the head of the queue is extracted and served.

3.1. Computing priority

DBP saves the history of execution in a structure named *k-sequence* which is a sequence of *k* bits updated after task execution (1 indicates the respect of deadline and 0 the opposite). The priority (distance) is computed by DBP in the following way:

$$P_{DBP} = k - l(m, s) + 1. \quad (1)$$

Where *l(m,s)* is the position leaving from the right of the m^{th} success (1) in the *k-sequence s* (the state of the queue). If there are less than *m* byte 1 in the *k-sequence s* then *l(m,s) = k+1*. Each queue has its own (mi,ki) constraints and its own k-sequence. The queue with the weakest priority is the closest one to dynamic failure (0 being the top priority). Let's take the example of Fig. 1 in which we represent a 3-importance model, (m,k)-firm requirements are respectively (4,4), (2,4) and (1,4) for $queue_1$, $queue_2$ and $queue_3$. Let us suppose that $queue_2$ has the least priority and is the closest one to the dynamic failure state. Table 1 shows the computation of DBP for each queue and fig. 1 illustrates the principle of the algorithm.

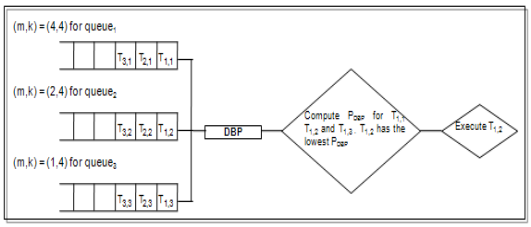

Figure 1 Execution of DBP

3.2. Drawbacks of DBP

DBP was proposed for the network field and is not suitable for the execution of real-time tasks, indeed DBP does

Table 1 Example of DBP computation

Queue	k-sequence	Priority	State
$queue_1$	0 0 1 1 1 1	(4-4+1)=1	$m = 4$ is respected
$queue_2$	0 0 0 0 1 0	(4-5 + 1) = 0	Queue in a dynamic failure state, it will be served next step
$queue_3$	0 1 1 0 0 0	(4-4 + 1) = 1	m=1 is respected

not take into account any timing attribute like the execution time e_i or the deadline d_i. Hence it is necessary for the proposed algorithm to check that the selected task can meet its deadline using these two attributes. Besides the algorithm should also check the deadline meet for higher importance tasks. In the example of Fig. 1 before extracting $T_{1,2}$, the algorithm should first check the deadline for $T_{1,1}$. The proposed algorithm should integrate a priority assignment step based on the distance that separates the queue from the dynamic failure state (a), an implicit deadline check in order not to launch the execution of a task which in any case will miss its deadline (b) and a formula allowing to take account of the task importance (c) and timing attributes of higher importance tasks (d).

4. RT-DBP : The Real-Time Distance Based Priority algorithm

In this section we propose to optimize the selection of the task to be extracted by operating a multi-criteria choice. Our objective is to combine several criteria which are not necessary compatible. We introduce RT-DBP Real-time Distance Based Priority, a new priority assignment scheme, which takes into account the criteria quoted above and which calculates a new priority parameter GIP (Global Index of Priority).

Based on response time analysis and the integration of the criteria quoted above in the computation of GIP, RT-DBP computes the priority DBP P_{DBP}, and the time necessary to carry out the task (response time) and then computes the global priority GIP. The task with the largest GIP is extracted and executed by the algorithm. From now on the queue from which the extraction is done is not anymore the one nearest to dynamic failure but that for which the task at the head presents the largest GIP. The index of priority calculated by RT-DBP during the extraction is the following,

$$GIP = (\frac{D}{1+\beta P_{DBP}} + \frac{F}{imp^{\alpha}}) * e^{\frac{-(\frac{d_i}{rt_i}-1)^2}{\sigma 0^2}} * \sigma(\frac{d_i}{rt_i} - 1). \quad (2)$$

D and F represent weight factors and α is the non linear importance coefficient. Next sections explain how we built the GIP equation.

4.1. Meeting (m,k)-firm constraints

DBP algorithm which we adapted looks after the respect of (m,k)-firm constraints by calculating for each queue the distance which separates it from the dynamic failure state. By integrating a parameter P_{DBP} (calculated according to 1) in the GIP computation, we take into consideration the state of the queues, close or far from a dynamic failure state.

4.2. Highlighting the task importance

Each time the application developer specifies a high importance for a given task, the algorithm must give the highest interest to that task during extraction. However, as explained above, the dispatching algorithm can proceed to the extraction from a different importance queue if this latter is in dynamic failure. Let us suppose that $queue_1$ has a ratio m_h/k_h = 1, the other queues will often fall in dynamic failure, which leads the RT-DBP algorithm to extract starting from these queues instead of queue1. This procedure is logical in (m,k)-firm context, however in our context we expect that the algorithm gives the same attention to the two parameters imp_i and P_{DBP_i}. For this purpose, the GIP equation should integrate the task's importance. Both imp_i and P_{DBP_i} should be associated with a weight in order to control their influence on the GIP value.

4.3. Checking the deadline

A task meets its timing requirements if its response time is less than its deadline. In order to check the deadline meet of a task T_i, RT-DBP computes its response time. Thus a transaction can be extracted only if,

$$rt_i = t + we_i < d_i. \quad (3)$$

Where *t* is present time, we_i its worst-case execution time and d_i its deadline. In order to penalize the task which cannot meet its deadline, we propose to divide the deadline by the maximum response time. The obtained factor d_i/rt_i is lower than 1 when the deadline is exceeded. By integrating the cut function (4) in the computation of GIP, we make this component and also GIP null for the task that cannot meet its deadline. As RT-DBP extracts the task with the greatest GIP, this latter cannot be extracted. The cut function can be described as,

$$Cut - f(\frac{d_i}{rt_i}) = \sigma(\frac{d_i}{rt_i} - 1). \quad (4)$$

Where σ is the Heaviside function defined as sygma = 1 if x $\geq$ 0 and 0 if x<0

4.4. Guarantying high importance tasks

Let us begin by presenting a solution to check that high importance tasks are guaranteed. We denote by T_i the task at the head of $queue_{priority}$ the closest queue to dynamic failure. To achieve the success requirements of high importance tasks when $queue_{priority}$ is the queue of importance *i* the algorithm should check if the slack time st_j of each task in the higher importance queues, is sufficient to execute the selected task and the higher importance tasks themselves. Hence the response time (5) should include execution times of tasks with higher importance (the set $hi(T_i)$),

$$rt_i = t + we_i + \Sigma_{j \epsilon hi(T_i)} we_j. \quad (5)$$

Another way to achieve this goal is that low importance tasks should be extracted only if the slack time of the higher importance task is sufficient and there is no risk to miss its deadline. To achieve this, we adopt a Gaussian distribution centered in the ratio $(\frac{d_i}{rt_i} = 1)$. Indeed the GIP should have low values for tasks with high values of the ratio and be maximised each time a high importance task is close to its deadline. The distribution function (6) has the following expression,

$$dist - f(\frac{d_i}{rt_i}) = e^{\frac{-(\frac{d_i}{rt_i}-1)^2}{\sigma 0^2}}. \quad (6)$$

The integration of the Gaussian component makes it possible to give to the curve illustrating the GIP a form which reaches its maximum when the task is close to its deadline. Moreover using the cut function (4) the left side of the curve is brought back to zero as soon as the deadline is exceeded what draws aside the task. Fig. 2 illustrates components (4) and (6) with different values of the ratio d_i/rt_i.

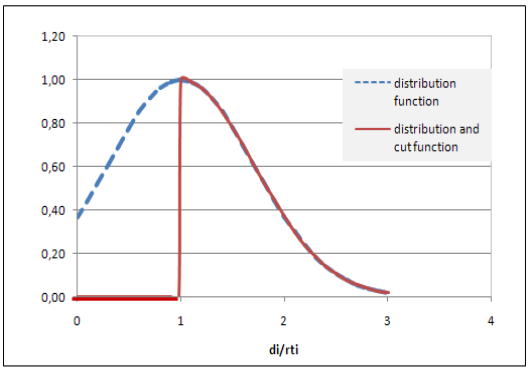

Figure 2 The cut and the distribution function with different values of d_i/rt_i

5. Evaluation of RT-DBP

In order to validate the equation applied by RT-DBP to assign the global index of priority, we built validation scenarii established starting from the objectives to respect. We restricted our evaluation to a 3-queues model. N=3 queues is a very realistic value indeed many applications can be structured using 3 kinds of tasks : tasks with high importance for which the deadline must be met, tasks with medium importance that are important to the system but whose deadlines can sometimes be missed and tasks with low importance that may not have deadlines.

5.1. Validation Scenarii

There are 3 validation scenarii described below:

–A task T_i that cannot meet its deadline because its response time is greater than its deadline is not extracted. In that case T_i has a null GIP.
–The task which has the lowest priority DBP is extracted if tasks with higher importance have sufficient slack times.
–When a task of high importance has a weak slack time, it is this one which is extracted regardless of the priority DBP of the other queues.

Table 2 shows different generated values for the ratio di/rti. Each scenario was validated by varying the priority DBP and the ratio di/rti. For all the scenarii the values of the parameters in equation (2) are α=0.7,β=1,D=2,F=1.

5.2. Validation of scenario 1

Scenario 1 checks that a task that cannot meet its deadline is not extracted. Scenario 1 is the simplest to evaluate indeed if the ratio d_i/rt_i falls below 1, the value of the Heaviside function is 0. This brings back the value of the GIP to 0. It is thus useless to vary the values of the ratio or those of priority DBP. From Table 2, we can see each time d_i/rt_i is less than 1 that the value of GIP is zero. We also observe that these tasks preserve a null GIP even if this queue is in dynamic failure.

5.3. Validation of scenario 2

Scenario 2 checks that the task which has the lowest priority DBP is extracted. In order to validate this scenario we must compare GIP values for tasks with different values of P_{DBP}. From Table 2, we extracted the following values showing in Table 3 that the task at the head of the queue in dynamic failure always have the highest GIP if the higher importance tasks are not close to their deadlines (i.e $d_i/rt_i > 1.4$). We also observe when there is no queue in dynamic failure, that the nearest queue to dynamic failure has the largest GIP whatever the importance of the task. The extracted values are for the ratio d_i/rt_i = 1.5, 1.7 and 2.

Task importance	P_{DBP}	GIP					
		$\frac{d_i}{rt_i} < 1$	$\frac{d_i}{rt_i} = 1.0$	$\frac{d_i}{rt_i} = 1.4$	$\frac{d_i}{rt_i} = 1.5$	$\frac{d_i}{rt_i} = 1.7$	$\frac{d_i}{rt_i} = 2$
$Imp = 1$	0	0.000	1.500	2.556	2.336	1.838	1.104
$Imp = 2$		0.000	1.308	2.229	2.037	1.602	0.962
$Imp = 3$		0.000	1.232	2.099	1.919	1.509	0.906
$Imp = 1$	1	0.000	1.000	1.704	1.558	1.225	0.736
$Imp = 2$		0.000	0.808	1.377	1.258	0.990	0.594
$Imp = 3$		0.000	0.732	1.247	1.140	0.897	0.538
$Imp = 1$	2	0.000	0.833	1.420	1.298	1.021	0.613
$Imp = 2$		0.000	0.641	1.093	0.999	0.786	0.472
$Imp = 3$		0.000	0.565	0.963	0.880	0.692	0.416
$Imp = 1$	3	0.000	0.750	1.278	1.168	0.919	0.552
$Imp = 2$		0.000	0.558	0.951	0.869	0.683	0.410
$Imp = 3$		0.000	0.482	0.821	0.750	0.590	0.354

Figure 3 Values of GIP with d_i/rt_i less than 2.

Scenario	Task at the head of the queue	$P_{DBP}(i)$	GIP		
			$\frac{d_i}{rt_i} = 1.5$	$\frac{d_i}{rt_i} = 1.7$	$\frac{d_i}{rt_i} = 2$
$queue_1$ has the highest P_{DBP}	$T_{1,1}$	0	2.336	1.838	1.104
	$T_{1,2}$	1	1.258	0.990	0.594
	$T_{1,3}$	2	0.880	0.692	0.416
	$T_{1,1}$	1	1.558	1.225	0.736
	$T_{1,2}$	2	1.070	0.683	0.472
	$T_{1,3}$	3	0.839	0.488	0.354
$queue_2$ has the highest P_{DBP}	$T_{1,1}$	1	1.558	1.225	0.736
	$T_{1,2}$	0	2.037	1.602	0.962
	$T_{1,3}$	2	0.880	0.692	0.416
	$T_{1,1}$	2	1.298	0.919	0,613
	$T_{1,2}$	1	1.329	0.990	0,594
	$T_{1,3}$	3	0.839	0.488	0.354
$queue_3$ has the highest P_{DBP}	$T_{1,1}$	2	1.298	1.021	0.613
	$T_{1,2}$	1	1.258	0.990	0.594
	$T_{1,3}$	0	1.919	1.509	0,906
	$T_{1,1}$	3	1.168	0.817	0,552
	$T_{1,2}$	2	1.070	0.683	0,472
	$T_{1,3}$	1	1.228	0.488	0,538

Figure 4 GIP values for queues in dynamic failure or close to dynamic failure

Scenario	Task	P_{DBP}	$\frac{d_i}{rt_i}$	GIP	$\frac{d_i}{rt_i}$	GIP	$\frac{d_i}{rt_i}$	GIP	$\frac{d_i}{rt_i}$	GIP
$T_{1,1}$ has a close deadline	$T_{1,1}$	2	1.1	1.650	1.4	1.420	1.7	1.021	2	0.613
	$T_{1,2}$	1	2	0.594	2	0.594	2	0.594	2	0.594
	$T_{1,3}$	0	2	0.906	2	0.906	2	0.906	2	0.906
$T_{1,2}$ has a close deadline	$T_{1,1}$	1	2	0.736	2	0.736	2	0.736	2	0.736
	$T_{1,2}$	2	1.1	1.269	1.4	1.093	1.7	0.786	2	0.472
	$T_{1,3}$	0	2	0.906	2	0.906	2	0.906	2	0.906
$T_{1,3}$ has a close deadline	$T_{1,1}$	0	2	1.104	2	1.104	2	1.104	2	1.104
	$T_{1,2}$	1	2	0.594	2	0.594	2	0.594	2	0.594
	$T_{1,3}$	2	1.1	1.119	1.4	0.963	1.7	0.692	2	0.416

Figure 5 GIP values for transactions close to their deadlines

5.4. Validation of scenario 3

In scenario 2, we have showed that the task at the head of the queue which is close to dynamic failure is extracted. We have varied the ratio d_i/rt_i and we have observed that (m,k)-firm constraints are privileged when tasks have sufficient slack times. In scenario 3, we consider tasks with few slack times and we show through Table 4 that the behaviour of the algorithm is inversed in such situations. Thanks to the formula of GIP when the queue in dynamic failure is that of importance 3 whereas the $queue_2$ or $queue_1$ have at the head a task $T_{1,2}$ (resp. $T_{1,1}$) with a close deadline, the GIP of task $T_{1,2}$ (resp. $T_{1,1}$) will be higher than the GIP of task $T_{1,3}$. The first two scenarii illustrate a configuration in which the $queue_3$ is in dynamic failure. In Table 5.4, d_i/rt_i values of 1.1 and 1.4 are considered close deadlines. Tasks associated to values beyond are considered with large slack times. We observe many interesting behaviours:

–When $queue_3$ is in dynamic failure, its head task is extracted only if the high importance task has a large slack time (greater than 1.7)
–When $queue_3$ is in dynamic failure and there is a medium importance task (imp = 2) close to its deadline, this last is extracted.

It would be inconceivable that the GIP is the largest for this reversed scenario in which the task $T_{1,3}$ has a weak latency time and $queue_1$ is in dynamic failure. The last scenario shows that the algorithm distinguishes between the tasks' importance when those have a weak latency time. Only the tasks of high importance are then privileged (values of d_i/rt_i 1.4, 1.7 and 2). The exception obtained for the value 1.1 can be adjusted with appropriate values of the weight parameters D and F in order to give more weight to the parameter using the importance than the one using the priority DBP. Besides, it is important to notice that the distribution function enables to modify the urgency of the task by acting on the centered value 1. By adopting a value of 1.1 we sanction the tasks for which it remains hardly more than their execution time before the deadline. This has the advantage of not launching tasks which would finish in extremis. On another side this distinction is significant in the sense that one could need to exploit it in order to include or to draw aside the tasks close to their deadlines.

6. Conclusion

In this paper, we have introduced a new scheduling algorithm for Real-time applications tolerating occasional deadline misses. RT-DBP proposes a new priority assignment scheme where the history of tasks' execution is no longer the most important criteria. During the priority computing, RT-DBP takes into account many scheduling parameters, namely the task (m,k)-firm constraints, its deadline, execution time, importance and also the timing parameters of tasks with higher importance. All the criteria are associated with weight parameters in order to give more weight to one criterion in particular depending on the application requirements. The algorithm is validated using scenario established starting from the objectives to respect. Each scenario is tested with various values of tasks' priority DBP, importance and timing parameters. The GIP scheme allows to efficiently check if the task to extract can meet its deadline otherwise it will be rejected upon its arrival. In that case an admission controller can be associated to the scheduling algorithm. In the near future, we plan to

deploy the GIP scheme and determine empirically the best values that should take the parameters alpha, beta, D and F.

References

[1] G. Bernat, A. Burns and A. Llamosi, Weakly-hard real-time systems, *IEEE transactions on Computers.* 50(4), 2001, 308-321.

[2] M. Hamdaoui and P. Ramanathan, A dynamic priority assignment technique for streams with (m,k)-firm deadlines, *IEEE transactions on Computers*, 44(12)(1995), 1443-1451.

[3] A. Koubaa, Y-Q. Song, J-P. Thomesse,Integrating (m,k)-firm real-time guarantees into the internet QoS Model", *Lecture notes in computer science*, 2004.

[4] J. Lin, A. M. K. Cheng, *Maximizing Guaranteed QoS in (m,k)-firm Real-time Systems*, 12th IEEE International Conf. on Embedded and Real-Time Computing Systems and Applications, 2006, 402-410.

[5] C. L. Liu, and J. Layland, Scheduling algorithms for multiprogramming in hard real-time environments, *Journal of the ACM*. 20(1)(1973), 46-61.

[6] D. Liu, X. Sharon Hu, M. Lemmon and Q. Ling , *Firm Real-Time System Scheduling Based on a Novel QoS Constraint* ,in the 24th IEEE International Real-Time Systems Symposium (RTSS2003), 2003 Cancun, Mexico.

[7] A. Mittal, G. Manimaran, C. Siva Ram Murthy, Integrated Dynamic Scheduling of Hard and QoS Degradable Real-Time Tasks in Multiprocessor Systems, *Journal of Systems Architecture*. 46 (2000), 793-807.

[8] E.-M. Poggi, Y.-Q. Song, A. Koubaa and Z. Wang, *Matrix-DBP For (m,k)-firm Real-Time Guarantee*, Proc. of Real-Time Systems, 2003, 457-482.

[9] G. Quan and X. Hu, *Enhanced fixed-priority scheduling with (m,k)-firm guarantee*, Real-Time Systems Symposium, 2000.

[10] A. Striegel, G. Manimaran, Best-effort scheduling of (m,k)-firm real-time streams in multihop networks, *Computer Comm.*.23(13)(2000),1292-1300.

[11] J. Z. Wang, Y. Song and Y. Sun, *Extended DBP for (m,k)-firm Based QoS*, LNCS Proc. Network and Parallel Computing NPC 2004, IFIP International Conf., 2004, 357-365.

[12] R. West, Y. Zhang, K. Schwan, and C. Poellabauer, Dynamic window-constrained scheduling of real-time streams in media servers, *IEEE transactions on Computers*. 53(6)(1994), 744-759.

L. Baccouche received her Ph. D. in computer science from the National Polytechnic Institute of Grenoble in France in 1996. Sheös an assistant professor at the National Institute of Applied Science and Technology in Tunisia in the computer science and mathematics department. Her research interest is related to real-time systems and real-time databases and includes scheduling, quality of service and feedback control.

H. Eleuch received his Ph. D. in Quantum Physics from Kastler Brossel Laboratory at Ecole Normale Suprieure de Paris and Universit Pierre-et-Marie-Curie in France in 1998. Recently, he spent two years in USA as researcher at the Institute for Quantum Science and Engineering Institute for Quantum Studies, Texas A&M University as well as visiting scientist at Princeton University. Now he is a Guest Scientist at Max Planck Institute of Physics for Complex Systems and a professor at National Institute of Applied Science and Technology (University of Carthage, Tunis). Furthermore, he is an associate member at the International Centre of Theoretical Physics in Trieste. His research interest is related to quantum optics, nonlinear optics, stochastic processes and Complex Systems.