Schedulability analysis of multiple criticality real-time tasks

François DORIN[1], Joël GOOSSENS[2], Pascal RICHARD[1] and Michaël RICHARD[1]

[1] LISI / ENSMA, France
{francois.dorin, pascal.richard, michael.richard}@lisi.ensma.fr
[2] Université Libre de Bruxelles, Belgium
joel.goossens@ulb.ac.be

Keywords: Multiple criticality model, sensitivity analysis, schedulability analysis

1 Introduction

Real-time applications are modeled using the task model introduced in Liu and Layland's seminal paper (Liu *et al.* 1973). A task is periodically released and generates an infinite set of jobs. A job of a task τ_i that is released at time t has a worst-case execution time C_i and must be completed in the time window $[t, t + D_i]$, where D_i is the deadline relatively to the job release. The next job of τ_i will be released at time $t + T_i$, where T_i is time interval between two successive releases of τ_i (i.e. the period). Since only a bound of tasks execution requirement are known, tasks are usually scheduled using *online* scheduling strategies: EDF (Earliest Deadline First) or FPP (Fixed-Priority Policy). Next, we consider FPP that defines one static priority to every task and at any time the highest ready priority task is executed. Preemption is allowed at no cost. We consider the *schedulability analysis problem*, that consists on checking that deadlines for all jobs will be met at run-time. A classical approach to perform a schedulability analysis is to compute the worst-case response time of a task, that is the maximum interval of time between the release and the completion of a task job while considering all the task jobs (Joseph *et al.* 1986).

In avionic systems, numerous tasks having different criticality levels are executed on a same processor (i.e., as specified by the international norm DO-178B on the design of avionic softwares). The multiple criticality model is a recent model of tasks introduced by Vestal (Vestal 2007) which allows to take into account different worst-case execution times for every task.

Task	D_i	T_i	L_i	$C_i(1)$	$C_i(2)$
1	4	4	2	2	2
2	7	7	1	2	5

Fig. 1. Example of multiple criticality task system

In the example given in the Figure 1, classical characteristics are defined like the deadline D_i or the period T_i. The notion of criticality level introduced by Vestal is denoted by L_i, and for this system having 2 criticality levels ($\{1, 2\}$), 2 execution times per task are defined, $C_i(\ell)$ corresponding to the execution time when considering the criticality level ℓ. An important assumption in the model is:

$$C_i(\ell - 1) \leq C_i(\ell), \forall \ell \tag{1}$$

Using this model is quite simple. Let us suppose that we want to compute the response times of the tasks 1 and 2. Since the task 1 has a criticity level 2, we only consider the $C_i(2)$ while computing its response time. Doing this, we can compute the response time of the task 1 by considering only a classical system with $C_1 = 2$ and $C_2 = 5$ and using classical response time computations (Joseph *et al.* 1986). It is the same principle for the task 2, but since task 2 has a criticity level of 1, we consider the execution time $C_i(1)$.

The interest of the method is to reduce the pessimism. For example, if we consider task 1 having the highest priority (and so task 2 has the lowest one), then the system is schedulable if we take into account the criticity levels, but is not schedulable if we only consider a classical Liu and Layland task system using the worst-case execution times (i.e., $C_1 = 2$ and $C_2 = 5$).

The main results about this new model were first performed by Vestal, which introduced the model and defined a priority assignment algorithm (Vestal 2007) that is based on Audsley's one by introducing a tie breaking rule at each priority level. Then, Baruah and Vestal (Baruah *et al.* 2008) provided several important results:

- they showed that EDF and Fixed-Task Priority (FTP) algorithms are not comparable,
- they introduced an hybrid algorithm which can schedule any task systems which can be schedule either by EDF or by a FTP algorithm.

In our works, we are interested by the optimality of Audsley's algorithm, the properties of the Vestal's algorithm (Vestal 2007) and by adapting a sensitivity analysis developed by Bini (Bini *et al.* 2006) to this new task model.

2 Vestal's algorithm

Vestal's algorithm is a particular case of the algorithm of Audsley. The priority assignments are realized from the lowest priority to the highest. But when Audsley's algorithm assigns at a level the first task which can be scheduled at this given priority level, the algorithm will use a tie breaker, based on the notion of critical scaling factor as introduced by Lehoczky (Lehoczky *et al.* 1989).

The critical scaling factor is the maximum reduction factor which can be applied to the processor speed in such a way the system is still schedulable (i.e., task execution times are stretched). By extension, we define the critical scaling factor of a task, which corresponds in this case to the scaling factor when considering only a subset of tasks, composed of a given task i and all the tasks with a higher priority than the task i.

Vestal chooses the task having the highest critical scaling factor for a given level (cf. Figure 2(a) and 2(b)). The critical scaling factor of the tasks can be computed since we only need to know the set of higher priority tasks and not their relative priority order. The priority column in Figure 2(a) denotes the priority of the tasks computed by the algorithm and a value of 0 corresponds to the highest priority.

For example, to compute the scaling factor of the task 1 at the priority level 2, the set of higher priority task is composed of the remaining tasks 2 and 3 (cf. Figure 2(b)).

2.1 Optimality of Audsley's algorithm

We have proved that Audsley's algorithm, known to be optimal for fixed-priority tasks system, is still optimal when considering the multiple criticality task model. Thus, the tie breaking rule introduced in Vestal's algorithm is not necessary to find a priority assignment leading to a feasible schedule if one exists.

The proof sketch is the following: based on the proof provided by Audsley, we made another one using the notion of scaling factor as defined by Lehoczky instead of the notion

Task	P_i	D_i	L_i	$C_i(1)$	$C_i(2)$	Priority
1	164	104	1	7	17	2
2	89	44	2	4	4	0
3	191	80	1	12	16	1
4	283	283	2	85	85	3

(a) Example of a priority assignement

	Task 1	Task 2	Task 3	Task 4
Level 3	–	–	–	**1.69**
Level 2	**3.87**	1.19	3.47	
Level 1		2.2	**5**	
Level 0		**11**		

(b) Trace of the assignment

Fig. 2. Vestal's algorithm

of interference. We proved that a scaling factor of a task can only be increased if the priority of this task is increased. Please note that this new proof is also valid for showing the optimality of the algorithm of Vestal, since it is a particular case of the algorithm of Audsley.

2.2 Maximization of the scaling factor

Since the algorithm of Audsley is already optimal, we can think that the tie breaker introduced by Vestal to choose which task has to be assigned at a given level is useless. We shown that using Vestal's tie breaker provides a very interesting property to the algorithm: it maximizes the critical scaling factor. In other word, it does not exist any valid schedule which can have a higher critical scaling factor than the critical scaling factor of the schedule returns by Vestal's algorithm. This property is very interesting in the context of embedded systems since it allows to define the optimal processor speed modulation for reducing power consumption of the system.

The proof sketch is to introduce a transformation which will modify a valid schedule in another one in such a way that the new schedule has a higher critical scaling factor than the previous one. The transformation is simple and consists in reducing the priority of one task under condition. Applying the transformation until the system remains unchanged will conduce to the schedule returned by the Vestal's algorithm.

3 Sensitivity analysis

Sensitivity analysis consists in studying the modification which can be made to a system while still meeting all deadlines. A basic example is to find how the execution time of a task can be increased in such a way the whole system is still schedulable.

One of the first work about sensitivity analysis was realized by Lehoczky (Lehoczky *et al.* 1989) with the notion of critical scaling factor. The approach was extended by Vestal (Vestal 1994) by introducing slack variation in the result of Lehoczky. Later, Bini performed (Bini *et al.* 2006) a deeper analysis which introduces the notion of feasibility region and which generalizes the previous works, but for the Liu and Layland's task model. Our work was about adapting the works of Bini for the multiple criticality tasks model.

3.1 C-space

The analysis on the C-Space developed by Bini define a scalar λ for evaluating how can vary processing times. Precisely, λ is defined by the following formula:

$$\lambda = \min_{i=1,\dots,n} \max_{t \in \text{sched}(P_i)} \frac{t - n_i(t)\mathbb{C}_i}{n_i(t)d_i} \tag{2}$$

In the above equation, n is the number of the tasks of the system, $\mathbb{C}_i$ is a vector which is equal to $\mathbb{C}_i = (C_1, C_2, \ldots, C_i)$, $\text{sched}(P_i)$ is the set of the scheduling point of the task i, $n_i(t) = \left(\left\lceil \frac{t}{T_1} \right\rceil, \left\lceil \frac{t}{T_2} \right\rceil, \ldots, \left\lceil \frac{t}{T_{i-1}} \right\rceil, 1\right)$ and d_i is the direction on which the sensitivity analysis is performed. For example, if we want to perform schedulability analysis on the task k only, then d_i must be equal to 0 except for the k^{th} component that will be set to 1.

The λ factor is such that replacing $\mathbb{C}_i$ by $\mathbb{C}_i + \lambda\mathbb{C}_i$ conduces to a schedulable system, but any higher value of λ will lead to a non schedulable system.

3.2 Adaptation for multi-criticality task systems

The main idea is to perform a sensitivity analysis per criticality level, and so having one factor per level of criticity instead of one factor for the whole system.

$$\lambda(\ell) = \min_{i=1,\ldots,n \wedge L_i = \ell} \max_{t \in \text{sched}(P_i)} \frac{t - n_i(t)\mathbb{C}_i(\ell)}{n_i(t)d_i} \tag{3}$$

Attention must be paid when performing this analysis, since it is possible to break the basic rule introduce with the model and stated by the Equation 1. If the rule is broken then a normalization step must be perform. This step consists in copying the value of the execution of the higher criticality level to the lower criticality level in such a way that the basic rule is satisfied.

4 Conclusion

We presented several results on multi-criticality task systems to be schedule upon a fixed-priority on-line scheduling algorithm. We first establish that the tie breaking rule introduced by Vestal in his scheduling algorithm is not useful for computing a feasible schedule (i.e., so that all deadlines will be met at run-time). Nevertheless, we proved that the tie breaking rule is useful if processor speed modulation is allowed(e.g., for reducing energy consumption). We then extend an existing sensibility analysis algorithm to multi-criticality task systems. In a further work, we shall study some extensions of our sensibility analysis algorithm.

References

Bini E., Di Natale M. and Buttazzo G., 2006, "Sensitivity analysis for fixed priority real-time systems", *Real-Time Systems*, Vol. 39, pp. 5-30.

Joseph M. and Pandya P., 1986, "Finding Response Times in a Real-Time System", *The Computer Journal*, Vol. 29, pp. 390-395.

Lehoczky J.P., Sha L. and Ding Y., 1989, "The rate-monotonic scheduling algorithm: exact characterization and average case behavior", *10th IEEE Real-Time Systems Symposium*, pp. 166-171.

Liu C.L. and Layland J.W., 1973, "Scheduling algorithms for multiprogramming in a hard real-time environment",*Journal of the Association for Computing Machinery*, Vol. 20, pp. 46-61.

Vestal S., 1994, "Fixed-Priority Sensitivity Analysis for Linear Compute Time Models", *IEEE Transactions on Software Engineering*, Vol. 20, pp. 308-317.

Vestal S., 2007, "Preemptive Scheduling of Multi-criticality Systems with Varying Degrees of Execution Time Assurance", *RTSS'07*

Vestal S. and Baruah S., 2008, "Schedulability Analysis of Sporadic Tasks with Multiple Criticality Specifications", *ECRTS'08*