2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)

The Improved Earliest Deadline First with Virtual Deadlines Mixed-Criticality Scheduling Algorithm

Xuejing Zhao, Yehua Wei
School of Physics and Information Science
Hunan Normal University
Changsha, China
985047950@qq.com, yehuahn@163.com

Wenjia Li
Department of Computer Science
New York Institute of Technology
New York, USA
wli20@nyit.edu

***Abstract*—The current trend in embedded systems towards integrating multiple functionalities with different degrees of criticalities on a shared computing platform. The concept of mixed-criticality system has been presented and is becoming common in embedded systems. The scheduling problem for mixed-criticality systems has become a hot research topic. A algorithm called EDF-VD was proposed for mixed- criticality systems, it can make a better performance guarantee. In order to increase the number of schedulable tasks, an improved mixed-criticality scheduling algorithm based on EDF-VD is proposed. In the proposed algorithm, a probabilistic parameter is introduced to represent the probability that the execution time of high criticality task exceeding the worst execution time at its low criticality mode, and sufficient conditions that the mixed-criticality task instances can be scheduled are derived. Simulation results show that the number of schedulable tasks in the proposed algorithm is more than the EDF-VD algorithm.**

Keywords-mixed criticality;task scheduling; virtual deadline

I. INTRODUCTION

With the increasing complexity of embedded system and embedded hardware itself in terms of volume, function, energy consumption and other constraints, integrating multiple functionalities subsystems with different criticalities on a same hardware platform, is becoming current trend [1], which means that not all functionalities are equally critical to the overall system, such systems are called mixed-criticality system (MC system).

The worst-case execution time(WCET) is used for task to represent upper bound of its duration time needed on a specified platform. To ensure the reliability and safety of a MC system, some safety-critical functionalities are generally required to be certified by statutory certification authorities (CAs). CAs usually tend to be conservative, the WCET estimate is more pessimistic than the system designer, the pessimism will increase with the increase of critical level of a task. Once a task executes beyond its WCET at current criticality level, the system criticality level is immediately increased. Then, the low-criticality tasks will be immediately discarded or indefinitely hanged to meet the deadlines of the high criticality tasks, which is very pessimistic and may affects the data consistency and integrity of the active task[2].

Existing MC scheduling algorithms usually make the most pessimistic assumption that each HI-criticality task may execute beyond its low WCET and reach its high criticality simultaneously. However, for many safety-critical tasks, strict upper bounds on their WCETs are extremely pessimistic and the represented scenarios that are indeed impossible to occur in practice[3-5,15]. Moreover, even when severe restrictions are placed upon a piece of code, it is still extremely difficult to determine the absolute WCET.

In another hand, CAs only concerns the correctness of the safety-critical functions of the system. The system designers need to ensure the overall system function to be executed correctly, including non-criticality parts. So, from the point of view of designers and practical applications, in order to reduce the pessimistic level of assumption, we can consider more realistic system model which includes faulty conditions, and an improved scheduling algorithm based on EDF-VD(Earliest Deadline First with Virtual Deadline) is presented.

II. RELATE WORK

Mixed-criticality task scheduling is current research focus in mixed-criticality systems. Li and Baruah [6] design a priority-based algorithm for scheduling mixed-criticality sporadic task systems and derive a sufficient condition for determining whether any given task set can be correctly scheduled by the presented algorithm. EDF-VD is a classic mixed-criticality scheduling algorithm, proposed in [7,13], which also makes pessimistic assumption that high criticality tasks may execute beyond their WCET estimates at the low criticality level simultaneously. Su and Zhu [8] present an elastic mixed-criticality task model, the variable periods is used for low-criticality tasks so that the execution frequencies of low-criticality tasks can be improved.

Some works have attempted to introduce probability parameter to task model. Maxim and Cucu-Grosjean [9] propose a definition of the probabilistic deadlines. A model is proposed in [10], which faces probabilities and probabilistic real-time systems unifying in the same framework probabilistic scheduling techniques and compositional guarantees for probabilistic real-time systems. A probabilistic scheduling framework for the design and development of mixed-criticality systems is proposed in [11], the probabilistic real-time guarantees on low criticality tasks are provided.

Most studies don't apply the probabilistic information to concrete mixed-criticality scheduling strategy. The work is

0-7695-6329-5/17/31.00 ©2017 IEEE
DOI 10.1109/ISPA/IUCC.2017.00072

presented in [12] introduces a new parameter to the traditional high criticality tasks model, the parameter represents the probability of no job of the tasks exceeding its low criticality WCET, and a novel scheduling algorithm is presented for a given MC task sets and allowed system failure probability. But, the algorithm introduces extra preemption for its server scheme. In order to increase the number of schedulable tasks, referring to the MC model with probabilistic WCET presented in [12], we propose an improved scheduling algorithm based on EDF-VD, called PEDF-VD(Probabilistic Earliest Deadline First With Virtual Deadlines).

III. Model and Definitions

A MC real-time system τ consists of a finite collection of independent implicit-deadline sporadic tasks, each of which may generate a potentially infinite sequence of MC jobs. In the work, we only focus on the system with two criticality levels that we will call LO(Low) and HI(High). Such system is called dual-criticality system. HI-criticality level tasks need certification, LO-criticality level tasks not need.

MC tasks. Different with traditional MC task models, a parameter f_i is introduced to the HI-criticality task model, which can be characterized by a 5-tuple: $\tau_i=(c_i(LO), c_i(HI), \chi_i, T_i, f_i)$, where

-$c_i(LO)$ is the worst case execution time(WCET) estimate of jobs of τ_i, the estimate is at the LO-criticality level.

-$c_i(HI)$ is WCET estimate of jobs at the HI-criticality level.

-χ_i denotes the criticality level.

-T_i is the minimum interarrival time (or period) between two jobs of task τ_i. Each job has a deadline that is T_i time units after its release.

-f_i is the probability of any job of task τ_i has an execution greater than $c_i(LO)$, but not greater than $c_i(HI)$ in certain time interval.

Same as existing works of MC systems, jobs of LO-criticality tasks only have one WCET estimate, the value of $c(LO)$ is equal to $c(HI)$. In addition, a system failure probability threshold $F_s \in (0,1)$ is introduced, it may be very close to zero, not equal to 0, e.g., 10^{-12} for some safety critical avionics functionalities.

MC jobs. Each job is characterized by a 5-tuple: $J_k=(r_k, d_k, \chi_k, c_k(LO), c_k(HI))$, where r_k is the release time, d_k is the deadline, $\chi_k \in \{LO,HO\}$ represents the criticality level of the job. Jobs generated by τ_i normally will execute the same piece of code and share the failure probability f_i.

Utilization parameters. For implicit deadline sporadic task systems, the period of each task equals its relative deadline. So the utilization of task τ_i can be defined as the ratio of its WCET to its period:

$$u_i(k)=c_i(k)/T_i \quad i=1,\ldots,n. \quad k \in \{LO,HI\} \tag{1}$$

And, the total utilization at level k of tasks that each of them has criticality level γ can be defined:

$$U_\gamma(k) = \sum_{i\in[n]\wedge\chi_i=\gamma} c_i(k)/T_i \quad \gamma,k \in \{LO,HI\} \tag{2}$$

Thus for example, $U_{HI}(LO)$ represents the sum of utilizations of HI-tasks when the system is in LO-criticality mode. Similar to [12], a parameter of additional utilization is introduced to quantify the additional processor requirement by HI-criticality task τ_i when it executes beyond its LO-WCET:

$$\theta_i = (c_i(HI) - c_i(LO))/T_i \tag{3}$$

IV. Example Analysis

In real applications, the industry standards usually only require the expected probability of missing deadlines within a specified duration to be below some specified small value. Thus, it may sometimes be overly pessimistic by completely discarding low-criticality tasks after observing a high-criticality behavior of system, here is a simple example:

TABLE I. MC Tasks Set with Probability Parameter

τ_i	χ_i	$c_i(HI)$	$c_i(LO)$	$d_i=T_i$	f_i
τ_1	HI	2	1	5	0.001
τ_2	HI	6	2	10	0.003
τ_3	LO	1	1	8	

In Table I, the task set τ consists of three tasks with independence between their WCETs, the permitted failure probability F_s is assumed to be 10^{-5}. According to (1) and (2), the sum of utilizations of HI-criticality tasks in τ is 1. If any HI-criticality job exceeds its LO-WCET, the system will switch to high criticality mode, the jobs of task τ_3 are immediately discarded in traditional MC scheduling algorithm.

However, the probability of jobs from both HI-criticality tasks exceeding their LO-WCETs simultaneously is $0.001*0.003= 0.3*10^{-5}$, which is smaller than the specified value of F_s. Thus, the case that the jobs of τ_1 and τ_2 exceed their LO-WCETs simultaneously can be ignored. Thus, the utilization of τ_1 and τ_2 will be less than 1 if only one task's job execute beyond its LO-WCET.

Using (3), we can get $\theta_1=0.2$, $\theta_2=0.4$. A virtual service can be defined, it can be regard as a high criticality task, and its utilization is 0.2 and period is 1. When system switch to HI-criticality mode, the service will be used to run HI-criticality task whose job exceeds its LO-WCET. For example, if only the job of task τ_1 exceeds its LO-WCET, the total system utilization for the HI-criticality tasks is $0.4+0.2+0.2=0.8$, plusing the utilization of τ_3, the value becomes 0.925, is still less than 1. The LO-criticality task does not need to be dropped.

V. PEDF-VD Scheduling Algorithm

As illustrated in the example of section IV, if the constraint of system failure probability F_s can be satisfied, task τ_3 can be successfully scheduled. So, based on the probability idea, an improved EDF-VD algorithm is proposed to increase the number of schedulable tasks.

A. EDF-VD Algorithm

EDF-VD [7] is a classic scheduling algorithm for MC implicit-deadline sporadic task sets. The algorithm makes a schedulability test to determine whether task set τ can be successfully scheduled. If τ is schedulable, the period T_i of

each HI-criticality task τ_i is modified into $\hat{T}_i = xT_i$,called virtual deadline, the x can be computed with (4). Using the virtual deadlines, run-time scheduling is done with EDF algorithm. If any job exceeds its LO-WCET, all LO-criticality jobs are immediately dropped. The HI-criticality tasks continue to be executed, but the actual job deadlines are used.

$$x = U_{HI}(LO)/(1-U_{LO}(LO)) \tag{4}$$

In the algorithm, the sufficient conditions for ensuring the system to be successfully scheduled are:

$$\frac{U_{HI}(LO)}{1-U_{LO}(LO)} \le x \tag{5}$$

$$xU_{LO}(LO)+U_{HI}(HI) \le 1 \tag{6}$$

B. Schedulability Analysis

We first analyze schedulability conditions before describing the scheduling strategy. Similar to [12], the HI-criticality tasks are divided to M clusters: $G_1,\ldots,G_M$. One task can exceed its LO-criticality WCET in every cluster. Failure probability of a cluster can be defined as follows:

$$g_m = 1-\prod_i (1-f_i) - \sum_j f_j \frac{\prod_j (1-f_i)}{1-f_i} \tag{7}$$

where the second item represents the probability of no task exceeding its LO-WCET, the third item denotes the probability of only one task exceeding its LO-WCET. Meanwhile, it is proved that the failure probability of any cluster is less than F_s if $g_m<F_s/M$ holds for any cluster G_m in [12]. Thus, the probabilistic constraint can be used directly.

In order to derive our schedulability conditions, the additional utilization of a cluster G_m and tasks set are defined respectively:

$$\lambda_m = \max_{\tau_i \in G_m} \theta_i \tag{8}$$

$$\lambda = \sum_{m=1}^{M} \lambda_m \tag{9}$$

Now, we can modify the utilization of HI-criticality tasks using (2) and (9):

$$U_{HI}(HI) = U_{HI}(LO) + \lambda \tag{10}$$

According to (4),(6),(10),we can derive (11):

$$\begin{aligned} &\frac{U_{HI}(LO)}{1-U_{LO}(LO)} \le \frac{1-(U_{HI}(LO)+\lambda)}{U_{LO}(LO)} \\ &U_{HI}(LO)U_{LO}(LO) \le (1-U_{HI}(LO)-\lambda)(1-U_{LO}(LO)) \\ &U_{HI}(LO) \le 1-\lambda-U_{LO}(LO)+\lambda U_{LO}(LO) \\ &U_{HI}(LO) \le (1-\lambda)(1-U_{LO}(LO)) \end{aligned} \tag{11}$$

The formula 5 and formula 11 are taken as the sufficient conditions that the tasks set can be scheduled by our algorithm when system is in LO-criticality mode and HI-criticality mode.

On the basis of the above analysis, the pseudo-code description is proposed for schedulability test in Fig.1. The clustering steps are described in lines 4-14, and the lines 15-19 are used to determine whether the tasks set is schedulable.

```
Input: F_s, τ
Output: value of parameter x, schedulability result
1:Compute U_HI(HI), U_HI(LO), U_LO(LO), θ_i with (1)-(3);
2:Compute the value of x using (4);
3: Sort the tasks in descending order of θ_i ;
4: m=1, Cno_i=0 i=1,…,n, M=num_HI
5: While
6:   λ_m=0
7:   for i=1 to num_HI
8:     if Cno_i>0 : continue;
9:     if g_m < Fs / M
10:      Cno_i=m , M=M-1;
11:  end for
12:  Compute λ_m using (8);
13:  m=m+1;
14: end while
15: Compute λ using (9);
16:if U_HI(LO)/(1-U_LO(LO))≤x, U_HI(LO)≤(1-λ)(1- U_LO(LO))
17:   return sucess;
18:else return failure;
19:end if
```

Figure 1. Schedulability analysis

C. Run-time Strategy

After performing schedulability test, we can obtain the values of parameters of λ and x. The jobs of tasks will be dispatched according to the following description:

1) The system is initialized to LO-criticality mode:

①Assume that a job arrives at time r. If the job generated by LO-criticality tasks, its deadline is set to $r+T_i$. Its deadline is set to $r+\hat{T}_i = r+x\cdot T_i$ if the job generated by HI-criticality tasks.

②At each instant, the job with the earliest deadline is selected from ready jobs list to execute.

③If the currently-executing job executes beyond its LO-WCET, the system switches to HI-criticality mode.

2) The system enters HI-criticality mode:

①The deadline of currently active job or a future job is modified to arrive time plus the original relative deadline of the task to which it belongs, that is, its deadline is $r+p_i$.

②As shown in the above example, a virtual service τ_s is introduced, its utilization is λ and period is 1. It is used to run the HI-criticality task whose job may execute for more than its LO-criticality WCET. Moreover, we don't need to drop all LO-criticality jobs, the rest processor capacity 1-λ can be used to run some LO-criticality jobs. Thus, the number of LO-criticality jobs that can be scheduled will increase.

VI. Performance Evaluation

Simulation experiments have been conducted to evaluate the efficiency of the proposed algorithm. In our experiments, the tasks sets are randomly generated using the algorithm UUniFast[14]. Each task set contains 20 tasks, the probability of which is HI-criticality task is p. The value of F_s is set to 10^{-5}. We mainly compared our algorithm with EDF-VD algorithm.

A. Impact of Parameters on Performance

The impact of parameters on performance of our algorithm is depicted in Fig.2 and Fig.3, where the ratio of

schedulable task sets is depicted on the y-axis, the utilization on the x-axis. Fig.2 shows that the ratio of schedulable sets increases with the decrease of the failure probability parameter *f*. The smaller the value of *f* is, the probability of HI-criticality jobs exceeding their LO-WCETs simultaneously is smaller, and more task sets will be scheduled.

From Fig.3, we can observe that the smaller the value of *p*, the higher the ratio of schedulable task sets. This is because if the value of *p* is small, the number of HI-criticality tasks will be less, the LO-criticality tasks have more opportunities to execute.

B. Comparison of Algorithm Performance

Fig.4 shows the schedulable ratio of EDF-VD and the proposed algorithm. It can be observed that there is no schedulable task for EDF-VD algorithm when the utilization at LO-criticality mode is in (0.7, 1). The schedulable ratio of our algorithm falls from about 60% of all task sets to 0, which indicating that our algorithm can schedule more tasks under same simulation conditions.

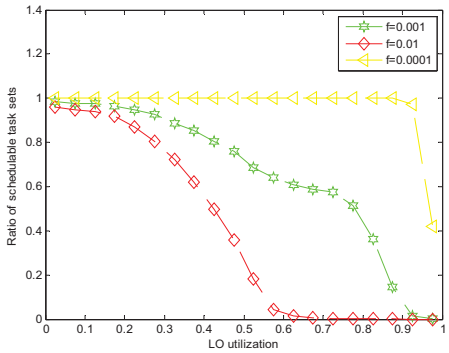

Figure 2. Comparison of schedulability under different f values

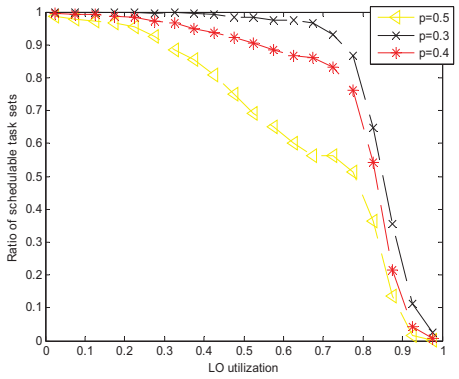

Figure 3. Comparison of schedulability under different mixed rate

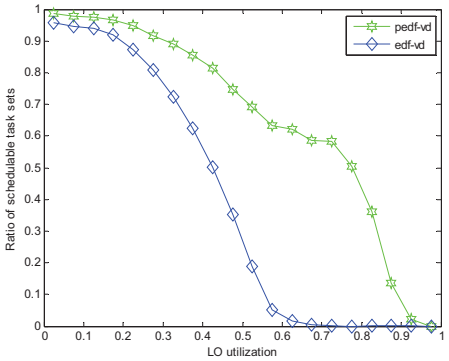

Figure 4. Comparison of schedulable ratio of EDF-VD and our algorithm

VII. CONCLUSION

In order to increase the number of schedulable tasks, PEDF-VD algorithm was presented for scheduling implicit-deadline sporadic task sets. We have evaluated the effectiveness of our algorithm by simulation experiments. In our work, only two criticality levels for task sets are considered. In fact, there are more than two criticality levels in many safety-critical applications, which will be concerned in our further research.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China(No.61402170), and the Research Foundation of Education Bureau of Hunan Province of China(No.17A130).

REFERENCES

[1] C.C. Gu ,N. Guan ,J.M. Yu ,Y. Wang and Q.X. Deng, "Partitioned Scheduling Policies on Multi-Processor Mixed Criticality Systems," Journal of Software, vol.25, no.2, 2014, pp.284-297.

[2] K.N.G. Chetan, S. Vyas, R.K. Cytron, C.D. Gill , J. Zambreno and Philip H. Jones, "Scheduling Challenges in Mixed Critical Real-Time Heterogeneous Computing Platforms," Procedia Computer Science, vol.18, 2013, pp.1891-1898.

[3] E. Quinones,E. Mezzetti,J. Abella,L.s Kosmidis,T. Vardanega,C. Lo, et al., "Measurement-Based Probabilistic Timing Analysis for Multi-path Programs," Proc. the 24th Euromicro Conference on Real-Time Systems, 2012, pp.91-101.

[4] S.K. Baruah, A. Burns and R.I. Davis, "Response-Time Analysis for Mixed Criticality Systems," Proc. IEEE Symposium on Electromagnetic Compatibility, 2011, pp.34-43.

[5] P. Huang, G. Giannopoulou, N. Stoimenov and L. Thiele, "Service adaptions for mixed criticality systems," Proc. the 19th Asia and South Pacific Design Automation Conference, Jan. 2014, PP.125-130.

[6] H.H. Li and S. Baruah, "An Algorithm for Scheduling Certifiable Mixed Criticality Sporadic Task System," Proc. IEEE Real-Time Systems Symposium.,IEEE Computer Society, 2010, pp.183-192.

[7] S. Baruah, V. Bonifaci, G. Dangelo and H.H. Li, "The Preemptive Uniprocessor Scheduling of Mixed Criticality Implicit-Deadline Sporadic Task Systems," Real-time Systems, vol.62, no.2, 2012, pp.145-154.

[8] H. Su and D. Zhu, "An elastic mixed criticality task model and its scheduling algorithm," Proc. the conference on Design, Automation and Test in Europe, Mar. 2013, pp.147-152.

[9] D. Maxim and L. Cucu-Grosjean, “Towards an analysis framework for tasks with probabilistic execution times and probabilistic inter-arrival times,” ACM SIGBED Review, vol.9, no.4, 2012, pp.33-36.

[10] L. Santinelli and L. Cucu-Grosjean, “A Probabilistic Calculus for Probabilistic Real-Time Systems,” ACM Transactions on Embedded Computing Systems, vol.14, no.3, 2015, pp.1-30.

[11] A. Masrur, “A probabilistic scheduling framework for mixed-criticality systems,” Proc. the 53rd Annual Design Automation Conference(DAC'16), no.132, Jun. 2016.

[12] Z. Guo, L. Santinelli and K. Yang, “EDF Schedulability Analysis on Mixed-Criticality Systems with Permitted Failure Probability,” Proc. IEEE Embedded and Real-Time Computing Systems and Applications, Aug. 2015, pp.187-196.

[13] S. Baruah, V. Bonifaci, G. Dangelo, H.H. Li, A. MARCHETTI-SPACCAMELA, S. VAN DER, et al., “Preemptive Uniprocessor Schedul- ing of Mixed-Criticality Sporadic Task Systems,” Journal of the Acm, vol.62,no.2, 2015,article no.14.

[14] E. Bini and G.C. Buttazzo, “Measuring the Performance of Schedulability Tests,” Real-Time Systems,vol.30, no.1-2, pp.129-154.

[15] A. Burns and B. Littlewood, “Reasoning about the reliability of multi-version, diverse real-time systems,” Proc. IEEE Real-Time Systems Symposium, IEEE Computer Society Press, Dec. 2010, pp.73‑81.