Towards Compositional Mixed-Criticality Real-Time Scheduling in Open Systems

Invited Paper

Jaewoo Lee
University of Pennsylvania

Hoon Sung Chwa
KAIST

Arvind Easwaran
Nanyang Technological University

Insik Shin
KAIST
insik.shin@cs.kaist.ac.kr

Insup Lee
University of Pennsylvania

ABSTRACT

Although many cyber-physical systems are both mixed-criticality system and compositional system, there are little work on intersection of mixed-criticality system and compositional system. We propose novel concepts for task-level criticality-mode and reconsider temporal isolation in terms of compositional mixed-criticality scheduling.

1. INTRODUCTION

As computer technology advances, traditional real-time systems evolve into cyber-physical systems (CPS), which are intelligent systems sensing and actuating physical environment. There are two increasing trends in CPS. One trend is resource sharing under a shared platform for efficient resource usage. Resource efficiency is important because modern CPS have a large number of functions under Size, Weight, and Power (SWaP) constraint. Another trend is partitioning for safety assurance. To minimize interference from other parts of the system, we need to partition functions of CPS.

One may achieve partitioning/isolation through worst-case resource reservation while highly reliable (and potentially pessimistic) Worst-Case Execution Time (WCET) estimates are used for reservation. This is ideal for achieving isolation, but not very efficient in resource utilization.

The concept of *mixed-criticality* has been introduced to resolve effectively those two seemingly conflicting requirements. In 2007, a Mixed-Criticality (MC) task model based on multiple execution time estimates was introduced [12] with a less reliable estimate for a common scenario and a more reliable estimate for a critical scenario. Several studies have used (and extended) the above model to develop mixed-criticality scheduling theory [3, 4, 6, 8, 12], showing that it is possible to achieve partitioning for higher-criticality tasks even in critical scenarios, while improving resource utilization in common scenarios.

Yet, there are many issues to explore to improve resource utilization. For example, many previous studies share the assumption that all high-criticality tasks will simultaneously exhibit critical scenarios, and consequently suspend all low-criticality tasks. However, such an assumption is very unlikely to happen in practice. Recently, a few studies considered this issue [5, 7, 9, 11] and proposed approaches to improve resource utilization for low-criticality tasks. We plan to consider this issue for compositional mixed-criticality scheduling in open systems.

In practice, many large-scale complex CPS are constructed under open system design, where individual components are constructed independently from different vendors and integrated into the systems (e.g., ARINC [1] and AUTOSAR [2] standards). Little attention has made to efficiently scheduling mixed-criticality components in a compositional manner in open systems, except recent studies [7, 9]. Although traditional temporal isolation only consider strict isolation between components, we plan to consider a diverse levels (degrees) of temporal isolation in consideration of mixed-criticality.

2. SYSTEM MODEL

We consider dual criticality: high (HI) and low (LO). We consider a MC task set consisting of n MC tasks. Each MC task τ_i is characterized by $(T_i, C_i^L, C_i^H, \chi_i)$, where T_i is task period, C_i^L is LO-criticality WCET (LO-WCET), C_i^H is HI-criticality WCET (HI-WCET), and $\chi_i \in \{HI, LO\}$ is task criticality level. Depending on χ_i, a task is either a LO-criticality task (LO-task) or a HI-criticality task (HI-task).

A component consists of a set of component or a set of tasks. The system has a special component which represents workload of the system.

3. MC SCHEDULING ON INDUSTRIAL PLATFORMS

A demand of industry is to optimize the real-time performance of both HI- and LO-tasks in MC systems. For an example of the automotive system with high-critical engine function and lower-critical communication function, the system designer wants to execute communication function as much as possible except emergency situation on the engine.

However, existing MC schemes show low performance of LO-tasks in terms of the ratio of meeting deadline (RMD)[1]. It is because existing MC schemes rely on extremely pessimistic assumption of system-mode, which represents runtime system-level criticality-mode[2]. In practice, it is a rare case that all HI-tasks execute for their HI-WCETs simultaneously[3].

If we can capture a fine-grained runtime status of the system, we can reduce the pessimism of system-mode. For this purpose, we propose a task-level criticality-mode (called *Task-mode*), which represents runtime criticality status of every task.

Definition 1 (Task-mode)**.** *For each task, task-mode represents its runtime criticality mode as follows:*

- *When the task-mode of the task is HI, current job of the task executes up to HI-WCET.*
- *When the task-mode of the task is LO, current job of the task executes up to LO-WCET.*

Some initial works [5, 11] improved RMD of LO-tasks under pessimistic system mode assumption. Overcoming the limitation of system mode, compositional MC approaches [7, 9] introduced a concept of component-mode, which represents runtime criticality mode for a component. Using component-mode, they identify fine-grained execution behaviors of jobs in a component. However, they still have pessimism that all components request up to their HI-criticality budgets when a single component requests its HI-criticality budget [7] or all LO-tasks in a component are dropped at component mode-switch [9]. Our task-mode identifies more fine-grained execution behaviors in the system. In task-mode approach, each task has independent criticality mode and the system drop only minimal LO-tasks at task-mode switch.

To develop a new MC scheduling algorithm with task-mode, we utilize an existing MC scheduling algorithm. EDF-VD [3] in unicore MC has a simple algorithm with the concept of virtual deadlines and has the speedup factor of 4/3, which is optimal. We propose *EDF-VD-TM* scheduling algorithm extending EDF-VD with task-mode. We schedule a LO-task with its real deadline. We schedule a HI-task as follows:

- If task-mode of the task is LO, the task is scheduled with its virtual deadline.
- If the task execute more than its LO-WCET, the system switches task-mode of the task to HI and drops LO-tasks that are necessary to schedule all HI-tasks.
- If task-mode of the task is HI, the task is scheduled with its real deadline.

Difference between EDF-VD and EDF-VD-TM is handling the situation when a task executes more than its LO-WCET. Although EDF-VD switches system-mode to HI and drops all LO-tasks, EDF-VD-TM switches only task-mode of the task and drops only minimal LO-tasks for schedulability of HI-tasks.

We plan to derive schedulability analysis for EDF-VD-TM. We also plan to evaluate EDF-VD-TM in terms of RMD of LO-tasks in comparison with existing algorithms.

4. MC SCHEDULING ON OPEN INDUSTRIAL PLATFORMS

Many CPS applications have characteristics of both MC systems and compositional systems. For example, avionic system development is subject to ARINC standard [1] (composability) and DO-178B standard (MC characteristic). Recent MC schemes support open system design [7, 9]. In this section, we will extend EDF-VD-TM considering open system.

To extend MC scheduling into open systems, we need to revisit temporal isolation between components[4]. In traditional compositional scheduling [10], compositional real-time guarantee is built on temporal isolation and computation of component interface. In MC systems, however, traditional isolation is inefficient and overly-pessimistic [7]. If a HI-task in a component executes more than its LO-WCET, it is allowed to drop LO-tasks inside the component (naturally) or even LO-tasks in other components (by mixed-criticality principle). Then, what is proper concept of isolation for MC components? We propose the definition of MC isolation (Def. 2).

Definition 2 (MC isolation)**.** *Consider any two components:*

- *Components are* **weakly isolated** *if HI-tasks in one component may affect the correctness of LO-tasks in the other component.*
- *Components are* **strongly isolated** *if no tasks in one component may affect the correctness of tasks in the other component (the same as traditional isolation).*

Recent compositional MC works also considered their isolation concepts [7, 9]. Ren and Phan [9] only considered strong isolation in their MC scheme. Gu et al. [7] introduced tolerance limit (TL) to determine degree of isolation

[1] RMD is the reverse of deadline miss ratio (DMR): $RMD = 1 - DMR$

[2] If the system mode is HI (LO), current jobs of all tasks may execute up to their HI-WCETs (LO-WCETs).

[3] Its example can be found in Gu et al. [7].

[4] Any component cannot affect the correctness of other components.

between weak isolation and strong isolation. Although Gu et al. [7] consider TL as a given parameter for a component, we propose a different perspective with the concept of MC isolation.

We will seek a proper MC isolation for any two components. If components are designed in the same standard (specially in recognition of criticality), we can maximize RMD by using weak isolation. However, it is possible that different vendors develop different components in different standard. Then, we may not allow for HI-tasks in one component to affect the correctness of LO-tasks in the other component. In this case, we can say that two components are *incompatible* and need to enforce strong isolation between components.

To implement MC isolation, we propose a component group, which consists of compatible components. Within a component group, the system provides weak isolation between components. If two component belong to different component groups, the system provides strong isolation between components. When a new component is added, the system seeks its compatible components. If it is not found, the system creates a new component group. As the number of component groups increases, resource efficiency of the system decreases while the level of safety assurance (from partitioning) increases.

We plan to extend EDF-VD-TM with the MC isolation concept (Def. 2) and component group. We also need to derive schedulability analysis for the extended algorithm.

5. REFERENCES

[1] AEEC. Avionics Application Software Standard Interface: Part 1 - required services (ARINC specification 653-2), 2006.

[2] AUTOSAR. AUTomotive Open System ARchitecture. www.autosar.org.

[3] S. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, S. van der Ster, and L. Stougie. The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems. In *ECRTS*, 2012.

[4] S. Baruah, A. Burns, and R. Davis. Response-time analysis for mixed criticality systems. In *RTSS*, 2011.

[5] A. Burns and S. Baruah. Towards a more practical model for mixed criticality systems. In *WMC*, 2013.

[6] P. Ekberg and W. Yi. Bounding and shaping the demand of mixed-criticality sporadic tasks. In *ECRTS*, 2012.

[7] X. Gu, A. Easwaran, K.-M. Phan, and I. Shin. Resource efficient isolation mechanisms in mixed-criticality scheduling. In *ECRTS*, 2015.

[8] J. Lee, K.-M. Phan, X. Gu, J. Lee, A. Easwaran, I. Shin, and I. Lee. Mc-fluid: Fluid model-based mixed-criticality scheduling on multiprocessors. In *RTSS*, 2014.

[9] J. Ren and L. T. X. Phan. Mixed-criticality scheduling on multiprocessors using task grouping. In *ECRTS*, 2015.

[10] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *RTSS*, 2003.

[11] H. Su and D. Zhu. An elastic mixed-criticality task model and its scheduling algorithm. In *DATE*, 2013.

[12] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *RTSS*, 2007.