Regarding the Optimality of Speedup Bounds of Mixed-Criticality Schedulability Tests

Zhishan Guo
Missouri University of Science and Technology
guozh@mst.edu

Keywords: mixed-criticality, speedup bounds, optimality, clairvoyance

1 Introduction

Much existing research on Mixed-Criticality (MC) scheduling (see [7] for a review) has focused on dealing with the Vestal model [15], where different WCET estimations of a single piece of code are provided. This is typically a consequence of different tools for determining worst case execution time (WCET) bounds being more or less conservative than each other. It is known [1] that mixed criticality (MC) scheduling under such model is highly intractable, such that polynomial-time optimal solution is impossible unless $P = NP$. As a result, *speedup bound* is widely used in MC scheduling for measuring how close to optimal is a given schedulability analysis.

- A schedulability test has *speedup factor* of $s(s \geq 1)$, if any task set that is schedulable by any algorithm on a given platform with processing speed of 1, it will be deemed schedulable by this test upon a platform that is s times as fast.

Of course when deriving MC schedulers and associated schedulability tests, one of the goals is to identify/prove a relative small speedup bound (that is closer to 1). A minimum possible speedup is often presented as the "*optimal* speedup bound" of a given MC scheduling problem. However, we would like to point out that:

- Optimality of scheduler should not be derived against optimal speedup bounds.

2 Non-Optimal Schedulers with Optimal Speedup Bounds

For scheduling (dual-criticality) Vestal job set on a uniprocessor platform, it has been shown [2] that OCBP algorithm (following the idea of Audsley's priority assignment mechanism) has an optimal speedup bound of $(\sqrt{5} - 1)/2$. However, several algorithms has been identified to *strictly dominate* OCBP; e.g., Lazy Priority Adjustment [10], LE-EDF [12] [11] — they have the same speedup, yet the latter has better schedulability at *all* time. Similar results can be observed when we consider the scheduling of Vestal task as well. It has been shown that $4/3$ is the best speedup that any non-clairvoyant scheduler can achieve. Upon proposing a speedup-optimal uniprocessor scheduler named EDF-VD [3], improvements on the schedulability can still be made, e.g., [9] [8]. As for the multiprocessor case, it is proved [4] that both MC-Fluid [13] and MCF [4] achieve the optimal speedup of $4/3$. However, MCF is a simplified version of (and is dominated by) MC-Fluid. Moreover, improvements on schedulability can be further made to MC-Fluid [14].

3 Speedup over Non-Clairvoyance?

When deriving speedup bounds, in most of the existing works of the community, the proposed algorithm is compared with a *clairvoyant optimal scheduler*, and adapts the necessary conditions for MC schedulability. This may not be a very fair way of comparison, since the penalty for unawareness of the future is applied into the speedup bounds. Following the varying-speed MC model [6] [5], we have identified an on-line optimal[1] scheduler in [11] that has a speedup factor significantly greater than 1 when comparing to an optimal clairvoyant algorithm. However, such a speedup factor only reflects the price one must pay for not knowing the future (or the difficulty of the scheduling problem itself) — it has nothing to do with the MC scheduler design any more.

Since MC schedulability analysis is for off-line verification of correctnesses of real-time systems, all possible scenarios should be taken into consideration (which is non-clairvoyance). We believe speedup results comparing to optimal non-clairvoyance schedule may be worth investigating for MC systems.

References

[1] S. Baruah. Mixed criticality scheduling is highly intractable. http://www.cs.unc.edu/ baruah/Submitted/02cxty.pdf.

[2] S. Baruah, H. Li, and L. Stougie. Towards the design of certifiable MC systems. IEEE RTAS 2010.

[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 MC implicit-deadline sporadic task systems. ECRTS 2012.

[4] S. Baruah, A. Easwaran, and Z. Guo. MC-Fluid: simplified and optimally quantified. RTSS 2015.

[5] S. Baruah and Z. Guo. Scheduling mixed-criticality implicit-deadline sporadic task systems upon a varying-speed processor. RTSS 2014. IEEE Computer Society Press.

[6] —. Mixed-criticality scheduling upon varying-speed processors. IEEE RTSS 2013.

[7] A. Burns and R. Davis. Mixed-criticality systems: A review. http://www-users.cs.york.ac.uk/burns/review.pdf.

[8] A. Easwaran. Demand-based MC scheduling of sporadic tasks on one processor. IEEE RTSS 2013.

[9] P. Ekberg, W. Yi. Bounding and shaping the demand of generalized mixed-criticality sporadic task systems. Real-Time Systems, 50(1): 48-86, 2014.

[10] C. Gu, N. Guan, Q. Deng, and W. Yi. Improving OCBP-based scheduling for MC sporadic task systems. RTCSA 2013.

[11] Z. Guo and S. Baruah. The concurrent consideration of uncertainty in WCETs and processor speeds in mixed criticality systems. IEEE RTNS 2015.

[12] —. Mixed-criticality scheduling upon varying-speed multiprocessors. Leibniz Transactions on Embedded Systems, 1(2): 3:1 - 3:19, 2014.

[13] 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.RTSS 2015.

[14] S. Ramanathan and A. Easwaran. MC-fluid: rate assignment strategies. WMC 2015.

[15] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. IEEE RTSS 2007.

[1] By on-line optimal, if our algorithm returns unschedulable for an MC instance, then no algorithm can guarantee correctness without making lucky guesses to the future.