Article:
Burns, Alan, Fleming, Thomas David, Baruah, S et al. (2017) Corrections to and Discussion of "Implementation and Evaluation of Mixed-criticality Scheduling Approaches for Sporadic Tasks". ACM Transactions on Embedded Computing Systems (TECS). 77:1-&&:4.

https://doi.org/10.1145/2974020

Corrections to and Discussion of "Implementation and Evaluation of Mixed-criticality Scheduling Approaches for Sporadic Tasks"

Tom Fleming, University of York
Huang-Ming Huang, Washington University in St. Louis
Alan Burns, University of York
Chris Gill, Washington University in St. Louis
Sanjoy Baruah, University of North Carolina - Chapel Hill
Chenyang Lu, Washington University in St. Louis

The AMC-IA mixed-criticality scheduling analysis was proposed as an improvement to the AMC-MAX adaptive mixed-criticality scheduling analysis. However, we have identified several necessary corrections to the AMC-IA analysis. In this letter we motivate and describe those corrections, and discuss and illustrate why the corrected AMC-IA analysis cannot be shown to outperform AMC-MAX.

Additional Key Words and Phrases: Adaptive mixed-criticality scheduling, real-time systems

ACM Reference Format:
Tom Fleming, Huang-Ming Huang, Alan Burns, Chris Gill, Sanjoy Baruah, and Chenyang Lu, 2016. Corrections to and Discussion of the AMC-IA Mixed-Criticality Scheduling Algorithm. *ACM Trans. Embedd. Comput. Syst.* , , Article (July 2016), 4 pages.
DOI: 0000001.0000001

1. INTRODUCTION

The AMC-IA mixed-criticality scheduling analysis [Huang et al. 2014] was proposed as an improvement to the AMC-MAX adaptive mixed-criticality scheduling analysis [Baruah et al. 2011]. AMC-IA uses two definitions of a function $n(s)$ to represent the number of releases of a task by time s which is defined as '*the last deadline before a criticality change*'. For low-criticality tasks n is defined by:

$$n_j(s) \;=\; \left\lceil \frac{s}{T_j} \right\rceil . \tag{1}$$

For high-criticality tasks a different definition for n is given:

$$n_k(s) \;=\; \max(\left\lfloor \frac{s - D_k}{T_k} \right\rfloor + 1, 0). \tag{2}$$

Note that subscript j is used in the first definition and k in the second.

Work described in this note was supported in part by an EPSRC(UK) Tempo grant; by US National Science Foundation grants CNS-0834755, CNS-0834270, CNS-0834132, CNS-1016954, CCF-1136073, and CNS-1329861; by US ARO grant W911NF-09-1-0535; by US AFOSR FA9550-09-1-0549; and by AFRL grant FA8750-11-1-0033.
Author's addresses: Tom Fleming and Alan Burns, Computer Science Department, University of York; Huang-Ming Huang and Chris Gill and Chenyang Lu, Department of Computer Science and Engineering, Washington University in St. Louis; Sanjoy Baruah, Computer Science Department, University of North Carolina - Chapel Hill.

DOI: 0000001.0000001

The response-time of a two-criticality system, during the change from low-criticality mode LO to high-criticality mode HI (0 to 1), is given by the following equation (from [Huang et al. 2014])[1]:

$$R_i^s = C_i(0) + \sum_{\tau_j \in H_i} n_j(s)C_j(0) + \sum_{\tau_k \in HHC_i} \left(\left\lceil \frac{R_i^s}{T_k} \right\rceil - n_k(s) \right) C_k(1), \quad (3)$$

where H_i is the set of task with higher priority than τ_i (of either criticality) and HHC_i is the set of task with higher priority and higher (or equal) criticality than τ_i.

2. CORRECTIONS TO AMC-IA

The first correction to the AMC-IA analysis is that the initial computation time (for τ_i) itself should be $C_i(1)$ as it executes for its maximum value. The second correction to AMC-IA, to remove confusion as to which of the n functions should be used, is to rewrite the above equations in an equivalent but more obvious form. We still define n by equation (1) and introduce a new function m to encode equation (2):

$$m_k(s) \; = \; \max(\left\lfloor \frac{s - D_k}{T_k} \right\rfloor + 1, 0). \quad (4)$$

We then rewrite equation (3):

$$R_i^s = C_i(1) + \sum_{\tau_j \in HLC_i} n_j(s)C_j(0) + \sum_{\tau_k \in HHC_i} m_k(s)C_k(0) + \sum_{\tau_k \in HHC_i} \left(\left\lceil \frac{R_i^s}{T_k} \right\rceil - m_k(s) \right) C_k(1), \quad (5)$$

where HLC_i is the set of low-criticality tasks with higher priority than τ_i, and HHC_i is the set of high-criticality tasks with higher priority than τ_i.

Following this notational clarification, we then modify the definition of AMC-IA itself. As the third correction, the '+1' is removed from equation (4):

$$m_k(s) \; = \; \max(\left\lfloor \frac{s - D_k}{T_k} \right\rfloor, 0). \quad (6)$$

The fourth correction to the AMC-IA analysis is that each 's' point is now defined to be the *'first deadline after a criticality mode change'*.

3. DISCUSSION

The two significant corrections described above are to the definitions of 's' and the equation for $m_k(s)$. The change to 's' is necessary because the initial definition could underestimate the impact of high priority low criticality tasks on other tasks. This is easy to see with a simple system that has one high priority low-criticality task τ_1 with $D_1 = T_1 = 10$, and a set of high-criticality tasks with $D > 12$. If all tasks are released at time 0, and a mode change occurs at time 12 (in some high criticality task), then the 'last deadline before 12' is at time 10; equation (1) gives $n(10) = 1$, but it should be 2 as releases at times 0 and 10 will impact tasks at time 12. To avoid underestimating the interference give by equation (1) it is necessary to define 's' as the *'first deadline after a criticality mode change'*. Unfortunately this can now underestimate the interference of high-criticality tasks, as illustrated in figure 1.

[1] In [Huang et al. 2014], LO and HI criticalities are denoted as 0 and 1 respectively; thus for example, the LO-criticality WCET of τ_i is denoted $C_i(0)$ (rather than $C_i(LO)$ – the notation used in [Baruah et al. 2011]).

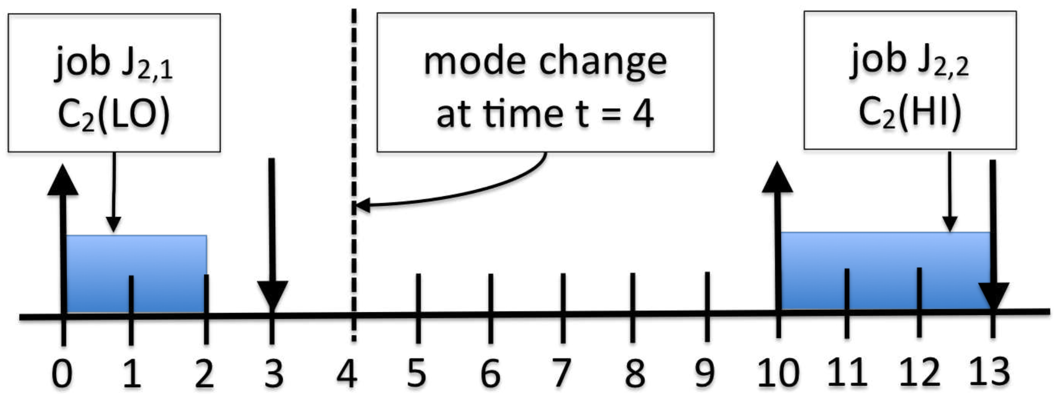

Fig. 1. High Criticality Task Example for Discussion

In this example consider a high-criticality task τ_2 with $C_2(LO) = 2$, $C_2(HI) = 3$, $D_2 = 3$ and $T_2 = 10$. If a criticality mode change occurred at time 4 (with all tasks released at time 0, and all other tasks having deadline later than time 13) then the new definition of 's' would give a point at time 13 (as 13 is the next deadline after time 4). At this point equation (4) gives a value of 2. This implies that in any interval after 13 there will be two executions of the task with $C(LO)$. It is easy to see that this is wrong, as at time 13 there may be only one execution with $C(LO)$ and hence one with $C(HI)$. As $C(HI) \geq C(LO)$ this could lead to an underestimation of this task's interference. To correct for this error, the '+1' in equation (4) is removed to give equation (6). Now $m(13) = 1$ which is a sufficient correction.

If these two corrections are applied then it is not clear how equation (5) can lead to tighter analysis than AMC-MAX. With the removal of '+1' a task with $D = T$ will assume no $C(LO)$ interference unless 's' is $2T$ or greater. AMC-MAX will assume (for $s < 2T$) either 0 or 1 $C(LO)$ hits - depending on the value of R. Hence they will often give the same result, but AMC-MAX can in some situations deliver (correctly) less interference.

It remains an open question whether there is a definition of AMC-IA that lies between the incorrect published one and the one given above that is both sufficient and 'better' than AMC-MAX, where 'better' means that it will deem more task sets to be schedulable. Certainly AMC-MAX is not exact, so such a scheme is possible. However with our current investigation it appears so far that in order for AMC-IA to be made sufficient it may not be able to outperform AMC-MAX.

4. CONCLUSIONS

In this letter we have described two necessary corrections to the AMC-IA analysis, and have shown that with those corrections AMC-IA cannot be shown to outperform AMC-MAX. It would be helpful to examine further whether the approach taken by AMC-IA offers potential insights into how AMC-MAX could be improved, though as we have demonstrated in this letter such improvement remains future work.

REFERENCES

S.K. Baruah, A. Burns, and R. I. Davis. 2011. Response-Time Analysis for Mixed Criticality Systems. In *IEEE Real-Time Systems Symposium (RTSS)*. 34–43.

Huang-Ming Huang, Christopher Gill, and Chenyang Lu. 2014. Implementation and Evaluation of Mixed-criticality Scheduling Approaches for Sporadic Tasks. *ACM Trans. Embed. Comput. Syst.* 13, 4s, Article 126 (April 2014), 25 pages. DOI: http://dx.doi.org/10.1145/2584612