01 Dec 2015

Mixed-Criticality Job Models: A Comparison

Sanjoy K. Baruah

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

Recommended Citation

S. K. Baruah and Z. Guo, "Mixed-Criticality Job Models: A Comparison," *Proceedings of the 3rd Workshop on Mixed-Criticality Systems (2015, San Antonio, TX)*, Workshop on Mixed-Criticality Systems (WMC), Dec 2015.

Mixed-criticality job models: a comparison

Sanjoy Baruah Zhishan Guo
Department of Computer Science
The University of North Carolina at Chapel Hill.
{baruah,zsguo}@cs.unc.edu

***Abstract*—The Vestal model [6] in widely used in the real-time scheduling community for representing mixed-criticality real-time workloads. This model requires that multiple WCET estimates – one for each criticality level in a system – be obtained for each task. Burns suggests [3] that being required to obtain too many WCET estimates may place an undue burden on system developers, and proposes a simplification to the Vestal model that makes do with just two WCET estimates per task. Burns makes a convincing case in favor of adopting this simplified model; here, we report on our attempts at comparing the two models – Vestal's original model, and Burns' simplification – with regards to expressiveness, as well as schedulability and the tractability of determining schedulability.**

I. Introduction

In the model for real-time mixed-criticality (MC) workloads that was proposed by Vestal [6] and forms the basis of a significant fraction of the research being conducted within the mixed-criticality real-time scheduling community, each job in an MC system with L distinct criticality levels is characterized by L worst-case execution time (WCET) estimates, one corresponding to each criticality level in the system under analysis. Burns recently proposed (in, e.g., the addendum [3] to his keynote presentation at the Dagstuhl Seminar *Mixed Criticality on Multicore/Manycore Platforms*) a simplification to this model, in which each job J_i is characterized by just two WCET estimates regardless of the number of distinct criticality levels in the system. One, denoted $C_i(\text{SELF})$ or $C_i(\mathsf{SF})$, is determined at a level of assurance that is consistent with its own criticality level (denoted χ_i); a second, denoted $C_i(\text{NORMAL})$ or $C_i(\mathsf{NL})$, is determined at a level of assurance that is consistent with the *lowest* (i.e., least critical) criticality level in the entire system. (For jobs of criticality equal to the lowest criticality level in the system, these two estimates are the same.) The run-time behavior desired of the system is as follows:

- If each job J_i executes for no more than its $C_i(\mathsf{NL})$ value then all jobs' deadlines are met; intuitively, this represents the "normal" behavior of the system.
- Each job J_i is prevented, by run-time monitoring, from executing for a duration greater than $C_i(\mathsf{SF})$.
- If any job J_i of criticality level χ_i executes for more than $C_i(\mathsf{NL})$, then
  - jobs that are less critical than J_i are no longer guaranteed.
  - the remaining jobs all complete by their deadlines, provided each such job J_j executes for no more than $C_j(\mathsf{SF})$ if $\chi_j = \chi_i$, and for no more than $C_j(\mathsf{NL})$ if χ_j denotes a greater criticality level than χ_i (i.e., J_j is more critical than J_i.)

  In other words, the only jobs that are guaranteed to complete execution by their deadlines are those of criticality greater than, or equal to, the criticality of the greatest-criticality job J_i to execute beyond its $C_i(\mathsf{NL})$ value.

Burns [3] makes a strong and convincing case justifying his simplification of the Vestal model from a pragmatic implementation-oriented perspective. Burns' model turns out to bear some similarities with an earlier model proposed by de Niz *et al*. [4], which, too, was inspired by the experience of de Niz *et al*. in implementing mixed-criticality systems. The evidence is thus strong that this model is a very reasonable and potentially useful one, meriting deeper analysis. We have initiated such an analysis from a scheduling-theoretic perspective; in this paper, we report on out initial findings. We restrict attention here to the scheduling of mixed-criticality systems that are modeled as collections of independent jobs executing upon a preemptive uniprocessor. Our findings thus far may be summarized as follows.

The Burns model is strictly less expressive than the Vestal model. Determining whether a given instance can be scheduled correctly remains NP-hard in the strong sense. Lower bounds on schedulability, as quantified using the speedup factor metric, are no better for the Burns model than for the Vestal model.

That is, although the reduced expressiveness of the Burns model makes it easier to use in many practical contexts, it does not reduce the inherent intractability of schedulability analysis, nor make the scheduling problem any easier.

Organization. The remainder of this paper is organized as follows. We formally describe the Vestal and Burns models, and state some more-or-less obvious facts concerning the relationship between them, in Section II. In Section III we show that scheduling instances specified using the simpler Burns model appears to be as difficult as scheduling instances specified using the Vestal model. We conclude in Section IV with some pointers to future work.

II. Model

In this section, we start out in Section II-A briefly reviewing the Vestal model [6], and provide definitions of the major concepts – *behavior*, *criticality level* of a behavior, *correctness* criteria, *clairvoyant schedulability*, *MC schedulability*, etc. —

of mixed-criticality scheduling, and summarize some prior results concerning the preemptive uniprocessor scheduling of mixed-criticality systems that are modeled as collections of independent jobs executing upon a preemptive uniprocessor. Next, we briefly describe the Burns model [3] in Section II-B, explaining how the concepts of MC scheduling are adapted to apply to the Burns model. In Section II-C, we make some rather straightforward observations concerning the Burns model, and its relationship with the Vestal model, with regards to the preemptive uniprocessor scheduling of collections of independent jobs.

A. The Vestal model

In the Vestal model [6], a mixed-criticality (MC) *job* is characterized by a 4-tuple of parameters: $J_i = (A_i, D_i, \chi_i, C_i)$, where

- $A_i \in R^+$ is the release time.
- $D_i \in R^+$ is the deadline. We assume that $D_i \geq A_i$.
- $\chi_i \in N^+$ denotes the criticality of the job, with a larger value denoting higher criticality.
- $C_i : N^+ \rightarrow R^+$ specifies the worst case execution time (WCET) estimate of J_i for each criticality level. (It is reasonable to assume that $C_i(\ell)$ is monotonically non-decreasing with increasing ℓ.)

An MC *instance* is specified as a finite collection of such MC jobs: $I = \{J_1, J_2, \ldots, J_n\}$. Given such an instance, we are concerned here with determining how to schedule it to obtain correct behavior; in this document, we restrict our attention to scheduling on preemptive uniprocessor platforms.

Behaviors. The MC job model has the following semantics. Each job J_i is released at time-instant A_i, needs to execute for some amount of time γ_i, and has a deadline at time-instant D_i. The values of A_i and D_i are known from the specification of the job. However, the value of γ_i is not known from the specifications of J_i, but only becomes revealed by actually executing the job until it *signals* that it has completed execution. γ_i may take on very different values during different execution runs: we will refer to each collection of values $(\gamma_1, \gamma_2, \ldots, \gamma_n)$ as a possible *behavior* of instance I.

The *criticality level* of the behavior $(\gamma_1, \gamma_2, \ldots, \gamma_n)$ of I is the smallest integer ℓ such that $\gamma_i \leq C_i(\ell)$ for all $i, 1 \leq i \leq n$. (If there is no such ℓ, then we define that behavior to be *erroneous*.)

Scheduling strategies. A *scheduling strategy* for an instance I specifies, in a completely deterministic manner for all possible behaviors of I, which job (if any) to execute at each instant in time. A *clairvoyant* scheduling strategy knows the behavior of I — i.e., the value of γ_i for each $J_i \in I$ — prior to generating a schedule for I. By contrast, an *on line* scheduling strategy does not have a priori knowledge of the behavior of I: for each $J_i \in I$, the value of γ_i only becomes known by executing J_i until it signals that it has completed execution. Since these actual execution times – the γ_i's – only become revealed during run-time, an on-line scheduling strategy does not a priori know what the criticality level of any particular behavior is going to be; at each instant, scheduling decisions are made based only on the partial information revealed thus far.

Correctness. A scheduling strategy is *correct* if it satisfies the following criterion for each $\ell \geq 1$: when scheduling any behavior of criticality level ℓ, it ensures that every job J_i with $\chi_i \geq \ell$ receives sufficient execution during the interval $[A_i, D_i)$ to signal that it has completed execution.

MC schedulability. Let us define an instance I to be MC schedulable if there exists a correct on-line scheduling strategy for it. The *MC schedulability problem* then is to determine whether a given MC instance is MC schedulable or not.

Some prior results. In the following, let s_L denote the root of the equation

$$x^L = (1 + x)^{L-1}. \tag{1}$$

For $L \leftarrow 2$, this is root of the equation $x^2 = x + 1$; it takes on the value $(\sqrt{5} + 1)/2$ and is commonly called the Golden Ratio or the Divine Proportion, notated Φ.

- Determining whether a given instance is MC-schedulable is NP-hard in the strong sense [2]. This holds even if all the jobs in the instance have the same release date, and there are just two distinct criticality levels in the instance.
- It was also shown [2] that there are instances with L distinct criticality levels that are clairvoyantly schedulable upon a unit-speed processor but not scheduled correctly upon a speed-s processor by any fixed-priority (FP) algorithm[1], for each $s < s_L$.
- An FP algorithm called OCBP was defined [1] for scheduling MC instances upon a preemptive uniprocessor. It was shown [2] that any instance with L distinct criticality levels that is MC-schedulable upon a unit-speed processor is scheduled correctly by OCBP upon a speed-s_L processor. This speedup bound for OCBP was shown to be tight: there are instances with L distinct criticality levels that are MC-schedulable upon a unit-speed processor but not scheduled correctly upon a speed-s processor by OCBP for each $s < s_L$.

B. The Burns model

In the Burns model [3], a mixed-criticality (MC) *job* is characterized by a 5-tuple of parameters: $J_i = (A_i, D_i, \chi_i, C_i(\mathsf{NL}), C_i(\mathsf{SF}))$, where A_i, D_i, and χ_i have exactly the same interpretation as in the Vestal model, and

- $C_i(\mathsf{NL}) \in R^+$ specifies the WCET estimate of J_i at criticality level 1 (the lowest criticality level)
- $C_i(\mathsf{SF}) \in R^+$ specifies the WCET estimate of J_i at the criticality level χ_i. (It is reasonable to assume that $C_i(\mathsf{NL}) \leq C_i(\mathsf{SF})$.

[1] An FP algorithm determines, prior to run-time, a total ordering of the jobs in a priority list and during run-time executes at each moment in time the currently active job with the highest priority. Note that EDF is an FP algorithm according to this definition.

The notion of *instance* and *behavior* is the same for the Burns and the Vestal models. The *criticality level* of the behavior $(\gamma_1, \gamma_2, \ldots, \gamma_n)$ is defined as follows:

- If $\gamma_j > C_j(\mathsf{SF})$ for any j, $1 \leq j \leq n$, then the behavior is erroneous.
- Else, the criticality level of the behavior is defined to be the criticality level of the greatest-criticality job J_j with execution exceeding its $C_j(\mathsf{NL})$ value:

$$\max_{j=1}^{n}\{\chi_j \mid \gamma_j > C_j(normal)\}$$

The notions of *scheduling strategy*, clairvoyance, correctness, and MC-schedulability are identical for the Vestal and Burns models.

C. Some observations

Since correctness requirements (i.e., which jobs are required to complete execution by their deadlines for the execution to be considered correct) for mixed-criticality instances are specified in a manner that depends upon the criticality level assigned to behaviors, we first investigate, in Propositions 1 and 2 below, whether the Vestal and Burns models assign behaviors the same criticality level or not.

Proposition 1: Any instance represented in the Burns model can be represented exactly in the Vestal model.

Proof: A job J_i that is specified according to the Burns model can be completely represented in the Vestal model by setting the WCET parameter values as follows:

$$C_i(\ell) \leftarrow \begin{cases} C_i(\mathsf{NL}) & \text{if } \ell < \chi_i \\ C_i(\mathsf{SF}) & \text{otherwise (i.e., if } \ell \geq \chi_i) \end{cases}$$

Consider any instance I in the Burns model, and let I' denote the instance in the Vestal model that is obtained by applying the above transformation t each job in I. Consider any behavior $(\gamma_1, \gamma_2, \ldots, \gamma_n)$ of instance I; this can also be considered a behavior of the Vestal instance I'. It follows from the definitions in Sections II-A and II-B above that this behavior is assigned exactly the same criticality level for I and I'; hence, the correctness requirements for both I and I' are identical. ■

Proposition 2: Instances represented in the Vestal model *cannot* always be represented exactly in the Burns model.

Proof: We illustrate this by an example. Consider the following instance $I = \{J_1, J_2, J_3\}$ represented in the Vestal model:

J_i	A_i	D_i	χ_i	C_i
J_1	0	3	1	$\langle 1, 1, 1\rangle$
J_2	0	3	2	$\langle 1, 1, 1\rangle$
J_3	0	3	3	$\langle 1, 2, 3\rangle$

Its representation in the Burns model would be as follows:

J_i	A_i	D_i	χ_i	$C_i(\mathsf{NL})$	$C_i(\mathsf{SF})$
J_1	0	3	1	1	1
J_2	0	3	2	1	1
J_3	0	3	3	1	3

Under the Vestal model, a behavior of the instance with $\gamma_1 = \gamma_2 = 1$, $\gamma_3 = 2$ has criticality level equal to 2 and hence requires that jobs J_2 and J_3 both complete by their deadlines. Under the Burns model, however, this same behavior has a criticality level equal to 3, and requires only that J_3 complete by its deadline: this is a *weaker* requirement than was mandated in the original (i.e., in the Vestal model). ■

It is evident that the Vestal model requires more parameters than the Burns model in order to specify an instance. What Proposition 2 illustrates is that these additional parameters in the Vestal model do indeed allow for the specification of a more nuanced set of requirements for a given instance. Taken together, Propositions 1 and 2 above consequently yield the (not unexpected) conclusion that *the Vestal model is strictly more expressive than the Burns model*.

Next, we explore whether this reduced expressiveness buys us anything in terms of tractability of analysis with respect to determining whether a given instance is MC-schedulable or not; Proposition 3 reveals that it does not:

Proposition 3: Determining whether a given instance specified according to the Burns model is MC-schedulable is NP-hard in the strong sense. This holds even if all the jobs in the instance have the same release date, and there are just two distinct criticality levels in the instance.

Proof Sketch: It may be verified that the intractability proof for the Vestal model [2, Theorem 1] only involves instances with just two criticality levels, in which all jobs have the same release date. Since the Vestal and Burns models are identical for two criticality levels, this proof hold unchanged for the Burns model as well, and its conclusion continues to hold for the Burns model. ■

III. Priority-based scheduling

As a consequence of Proposition 3, we are unlikely to be able to design an exact schedulability test to efficiently determine whether a given instance specified in the Burns model is MC-schedulable or not. But what about *sufficient* schedulability tests? Here, Proposition 1 means that we may use prior results that were developed for instances represented using the Vestal model to schedule instances that are specified using the Burns model as well. In particular, prior algorithms such as OCBP [1], MC-EDF [5], etc. may continue to be used for scheduling MC instances specified using the Burns model; their performance metrics are guaranteed to be no worse for Burns instances than for Vestal instances. In particular, we may conclude from prior results [2] that OCBP has a speedup bound no worse than s_L (recall that s_L is defined to be the root of Equation 1) in scheduling any instance with L distinct criticality levels.

A natural question to ask at this point in time is, do these algorithms offer better performance guarantees when scheduling instances specified using the Burns model than they do when scheduling instances specified using the more expressive Vestal model? Somewhat surprisingly, the answer

turns out to be "no." A close examination of the proofs of the analogous results in [2] reveal that

1 There are instances with L distinct criticality levels that are MC-schedulable upon a unit-speed processor but not scheduled correctly upon a speed-s processor by OCBP for each $s < s_L$.
2 There are instances with L distinct criticality levels that are clairvoyantly schedulable upon a unit-speed processor but not scheduled correctly upon a speed-s processor by *any* fixed-priority (FP) scheduling policy, for each $s < s_L$.

Both these results may be proved using techniques essentially identical to the ones used in proving the corresponding results in [2] for instances specified using the Vestal model; for the sake of completeness, we formally present the second result as Theorem 1 below, and provide a complete proof.

Theorem 1: There are MC instances with L distinct criticality levels specified using the Burns model that are clairvoyantly-schedulable, but that are not Π-schedulable for any fixed priority policy Π on a processor that is less that s_L times as fast.

Proof: Consider an instance with L criticality levels and L jobs:

	A_i	D_i	χ_i	$C_i(\mathsf{NL})$	$C_i(\mathsf{SF})$
J_1	0	D_1	1	D_1	D_1
J_i $(\forall i \geq 2)$	0	D_i	i	$D_i - D_{i-1}$	D_i

where the values of the D_i's will be specified later and shown to satisfy $D_i > D_{i-1}$ for all i, $1 < i \leq L$.

For example, this instance would look as follows for $L \leftarrow 3$:

J_i	A_i	D_i	χ_i	$C_i(\mathsf{NL})$	$C_i(\mathsf{SF})$
J_1	0	D_1	1	D_1	D_1
J_2	0	D_2	2	$D_2 - D_1$	D_2
J_3	0	D_3	3	$D_3 - D_2$	D_3

The system is clairvoyantly schedulable since, for a behavior of criticality-level ℓ, a clairvoyant scheduler could have each job complete by its deadline by

- not executing jobs $J_1, \ldots, J_{\ell-1}$ at all;
- executing job J_ℓ for a duration $C_\ell(\mathsf{SF}) = D_\ell$ over the interval $[0, D_\ell)$; and
- executing each job $J_j \in \{J_{\ell+1}, \ldots, J_L\}$ for a duration $C_\ell(\mathsf{NL}) = D_j - D_{j-1}$ over the interval $[D_{j-1}, D_j)$.

In the remainder of this proof, we will derive values for the D_i parameters such that this instance cannot be scheduled correctly by any FP scheduling algorithm. That will serve to show that this instance is clairvoyantly schedulable but not FP-schedulable, and hence establish the correctness of the theorem.

In any FP algorithm, some job from amongst the L jobs $J_1, \ldots, J_L$ in the instance must be assigned the lowest priority. Suppose that that job were J_i, and consider a behavior of the instance of criticality level i in which

- each job J_j with criticality lower than that of J_i executes for an amount $C_j(\mathsf{SF}) = D_j$,
- each job J_j with criticality greater than that of J_i executes for an amount $C_j(\mathsf{NL}) = D_j - D_{j-1}$, and
- job J_i executes for an amount equal to $C_i(\mathsf{SF}) = D_i$.

Since J_i is the lowest-priority job, it will only complete after an amount of execution equal to

$$\begin{aligned} & \big(\sum_{j=1}^{i-1} D_j\big) + D_i + \big(\sum_{j=i+1}^{L} (D_j - D_{j-1})\big) \\ = \quad & \big(\sum_{j=1}^{i-1} D_j\big) + D_L \end{aligned}$$

has completed. For J_i to meet its deadline on a speed-s processor, we therefore need this amount to be $\leq s \times D_i$:

$$\begin{aligned} & s\, D_i \geq \big(\sum_{j=1}^{i-1} D_j\big) + D_L \\ \Leftrightarrow \quad & s \geq \frac{D_L + \sum_{j=1}^{i-1} D_j}{D_i} \end{aligned}$$

Since *some* job from amongst the L jobs $\{J_1, J_2, \ldots, J_L\}$ must be assigned lowest priority by a fixed-priority policy, it follows that

$$\min_{1 \leq i \leq L} \Big\{ \frac{D_L + \sum_{j=1}^{i-1} D_j}{D_i} \Big\} \tag{2}$$

is a lower bound on the speedup necessary for a fixed-priority scheduling policy to successfully guarantee to schedule the instance correctly. This minimum is maximized when all L of the terms are equal to each other (and thus define the minimum). Let x be this maximum value. Instantiating the term in Expression 2 for $i \leftarrow L - 1$, we have

$$\begin{aligned} & x = \frac{D_L + \sum_{j=1}^{L-2} D_j}{D_{L-1}} \\ \Leftrightarrow \quad & x D_{L-1} = D_L + \sum_{j=1}^{L-2} D_j \end{aligned} \tag{3}$$

Next instantiating the term in Expression 2 for $i \leftarrow L$, we have

$$\begin{aligned} x \quad = \quad & \frac{D_L + \sum_{j=1}^{L-1} D_j}{D_L} \\ = \quad & \frac{\big(D_L + \sum_{j=1}^{L-2} D_j\big) + D_{L-1}}{D_L} \quad \text{(Rearranging terms)} \\ = \quad & \frac{x D_{L-1} + D_{L-1}}{D_L} \quad \text{(By Eqn 3 above)} \\ = \quad & \frac{(1+x) D_{L-1}}{D_L} \end{aligned} \tag{4}$$

Hence we have

$$\begin{aligned} D_L &= \Big(\frac{1+x}{x}\Big) D_{L-1} \\ &= \Big(\frac{1+x}{x}\Big)^2 \times D_{L-2} \\ &= \Big(\frac{1+x}{x}\Big)^3 \times D_{L-3} \\ &\cdots \\ &= \Big(\frac{1+x}{x}\Big)^{L-1} \times D_1 \end{aligned} \tag{5}$$

Finally instantiating the term within Expression 2 for $i \leftarrow 1$, we have

$$x = \frac{D_L}{D_1} \tag{6}$$

From Equations 5 and 6 above, we are able to conclude that

$$\begin{aligned} & x = \Big(\frac{1+x}{x}\Big)^{L-1} \\ \Leftrightarrow \quad & x^L = (1+x)^{L-1} \end{aligned}$$

which is exactly Equation 1. It's solution is therefore equal to s_L, and the theorem is proved. ■

IV. Context and Conclusions

The Burns model for mixed-criticality workloads was proposed [3] as a simplification of the Vestal model [6] that has formed the basis of a large volume of research in real-time scheduling theory. From a pragmatic perspective and in terms of ease of use, there are undoubted benefits in using the Burns model in preference to the Vestal model — some of these benefits are persuasively articulated in [3]. However, this ease of use does come with some loss of expressiveness (as illustrated in Proposition 2). In our research, we are seeking to better understand whether this reduced expressiveness yields any analytical benefits in terms of reduced complexity of feasibility analysis, less schedulability loss, etc. Thus far, our results have been negative – we have not identified any such benefits.

In this paper, we have restricted attention to MC instances that are characterized as collections of independent jobs. In the future, we plan to study systems that are modeled as collections of recurrent tasks, as well as more general (e.g., multiprocessor) platforms.

Acknowledgements

We are grateful to Alan Burns and to the anonymous reviewers for detecting several typos in an earlier version of this paper.

This research was supported in part by NSF grants CNS 1115284, CNS 1218693, CNS 1409175, and CPS 1446631, AFOSR grant FA9550-14-1-0161, ARO grant W911NF-14-1-0499, and a grant from General Motors Corp.

References

[1] S. Baruah, H. Li, and L. Stougie. Towards the design of certifiable mixed-criticality systems. In *Proceedings of the IEEE Real-Time Technology and Applications Symposium (RTAS)*. IEEE, April 2010.

[2] S. K. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, N. Megow, and L. Stougie. Scheduling real-time mixed-criticality jobs. *IEEE Transactions on Computers*, 61(8):1140–1152, 2012.

[3] A. Burns. An augmented model for mixed criticality. In S. K. Baruah, L. Cucu-Grosjean, R. I. Davis, and C. Maiza, editors, *Mixed Criticality on Multicore/Manycore Platforms (Dagstuhl Seminar 15121)*, volume 5. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2015.

[4] D. de Niz, K. Lakshmanan, and R. R. Rajkumar. On the scheduling of mixed-criticality real-time task sets. In *Proceedings of the Real-Time Systems Symposium*, pages 291–300, Washington, DC, 2009. IEEE Computer Society Press.

[5] D. Socci, P. Poplavko, S. Bensalem, and M. Bozga. Mixed critical earliest deadline first. In *Proceedings of the 2013 25th Euromicro Conference on Real-Time Systems*, ECRTS '13, Paris (France), 2013. IEEE Computer Society Press.

[6] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *Proceedings of the Real-Time Systems Symposium*, pages 239–243, Tucson, AZ, December 2007. IEEE Computer Society Press.