Incorporating Robustness and Resilience into Mixed-Criticality Scheduling Theory

Sanjoy Baruah, Washington University in St. Louis, USA
Alan Burns, University of York, UK

8th May 2019

Mixed-Criticality Systems, MCS

- Vestal's 2007 paper advocated a multi-model view of task scheduling
- Key parameters of tasks were open to more than one interpretation
- In particular the estimated worst-case execution time of each task was no longer a single C value, but was given a vector of estimates, one per criticality level
- So if there were two criticality levels, $\chi_i \in \{\mathsf{LO}, \mathsf{HI}\}$ there would be two estimates of WCET: C^L and C^H i (with $C^H \geq C^L$)

Mixed-Criticality Scheduling

- There has been a flood of papers addressing many different aspects of scheduling for MCS
- There has also been a certain amount of criticism of the ‘Vestal Model’
- Much of this criticism is due to a misunderstanding of what was the focus of Vestal’s paper
- Safety-critical (high-integrity) system require Verification and Survivability
- But Vestal’s paper only really concerned Verification

Vestal's Correctness Criteria

a. if every job of every task τ_i completes execution within C_i^L units of execution then all jobs should meet their deadlines; and

b. if a job of some task τ_i fails to complete execution despite having executed for C_i^L time units, then all jobs of each HI-criticality task τ_i should receive up to C_i^H units of execution by their respective deadlines

Since b violates the assumptions under which LO-criticality verification is required to be performed, no requirements are placed upon the execution of jobs of LO-criticality tasks

Survivability

- In this paper we focus not on correctness but on Survivability - also termed Fault Tolerance
- Survivability addresses expectations of system behavior in the event that the assumptions for correctness fail to fully hold
- We utilise the notions of Robustness and Resilience
- Informally, the *robustness* of a system is a measure of the degree of fault it can tolerate without compromising on the quality of service it offers
- *resilience*, by contrast, refers to the degree of fault for which it can provide degraded yet acceptable (i.e. safe) quality of service
- Resilience is also know as Graceful Degradation

Robustness and Resilience

- We seek to define quantitative metrics of robustness and resilience that are applicable to the Vestal model, and to correlate these metrics to the resulting run-time survivability guarantees of the system
- To illustrate our approach we shall use the MC-Fluid scheduling algorithm, but for just a single processor

System Model

- We consider the scheduling of systems of independent dual-criticality implicit-deadline sporadic tasks upon a shared preemptive processor
- We assume that a dual-criticality implicit-deadline sporadic task τ_i is characterized by the parameters $(T_i, C_i^L, C_i^H, \chi_i)$, where $\chi_i \in \{\text{LO}, \text{HI}\}$ denotes its criticality, C_i^L and C_i^H its LO and HI criticality WCETs, and T_i its period
- We require that $C_i^L \leq C_i^H$
- Some additional notation: we let $u_i^L \stackrel{\text{def}}{=} (C_i^L / T_i)$ and $u_i^H \stackrel{\text{def}}{=} (C_i^H / T_i)$ denote the LO-criticality and HI-criticality *utilizations* of task τ_i

Additional Notation

Various system utilization parameters are defined:

$$U_L^L \stackrel{\text{def}}{=} \sum_{\tau_i \in \tau_L} u_i^L$$

$$U_H^L \stackrel{\text{def}}{=} \sum_{\tau_i \in \tau_H} u_i^L$$

$$U_H^H \stackrel{\text{def}}{=} \sum_{\tau_i \in \tau_H} u_i^H$$

where τ_L is the set of LO-criticality task, and τ_H is the set of HI-criticality tasks (τ is the full set of tasks)

Basic MC-Fluid Algorithm

1. Each τ_i initially executes at a constant rate θ_i^L. That is, at each time-instant it is executing upon θ_i^L fraction of a processor
2. If a job of any task τ_i does not complete despite having received C_i^L units of execution (equivalently, having executed for a duration (C_i^L/θ_i^L)), then
   - All LO-criticality tasks are immediately discarded, and
   - Each HI-criticality task henceforth executes at a constant (higher) rate θ_i^H

Rates of Progress for MC-Fluid

1. Define ρ as follows:

$$\rho \leftarrow \max\left\{ U_L^L + U_H^L, U_H^H \right\} \tag{1}$$

2. If $\rho > 1$ then declare failure; else assign values to the execution-rate variables as follows:

$$\theta_i^H \leftarrow u_i^H / \rho \text{ for all } \tau_i \in \tau_H \tag{2}$$

$$\theta_i^L \leftarrow \begin{cases} \frac{u_i^L \theta_i^H}{\theta_i^H - \left(u_i^H - u_i^L\right)}, & \text{if } \tau_i \in \tau_H \\ u_i^L, & \text{else (i.e., if } \tau_i \in \tau_L) \end{cases} \tag{3}$$

3. If

$$\sum_{\tau_i \in \tau} \theta_i^L \leq 1 \tag{4}$$

then declare success else declare failure

Approach Employed

- We assume HI-criticality tasks never execute for more than their HI-criticality bound
- Hence Robustness and Resilience is all about what can be given to the LO-criticality tasks
- Within the context of MC-Fluid scheduling this means what rates can be assigned to LO-criticality tasks

Illustrative Example

	T_i	C_i^L	C_i^H	χ_i	u_i^L	u_i^H
τ_1	10	2	–	LO	0.2	–
τ_2	20	6	–	LO	0.3	–
τ_3	30	3	18	HI	0.1	0.6

$$
\begin{aligned}
U_L^L &= 0.2 + 0.3 = \mathbf{0.5} \\
U_H^L &= 0.1 \\
U_H^H &= 0.6
\end{aligned}
$$

Applying Basic MC-Fluid Scheme to the Example

$$\rho \leftarrow \max\{0.5 + 0.1, 0.6\} = \mathbf{0.6}$$

Consequently, $\theta_3^H = 0.6/0.6$ or 1.0, and

$$\begin{aligned}
\theta_1^L &= u_1^L = 0.2 \\
\theta_2^L &= u_2^L = 0.3 \\
\theta_3^L &= \frac{u_3^L \theta_3^H}{\theta_3^H - (u_3^H - u_3^L)} = \frac{0.1 \times 1}{1 - (0.6 - 0.1)} = \frac{0.1}{0.5} = 0.2
\end{aligned}$$

Since

$$\theta_1^L + \theta_2^L + \theta_3^L = 0.2 + 0.3 + 0.2$$

which is clearly ≤ 1, Algorithm MCF declares success

Incorporating Robutness

- Scheme is schedulable but not robust or resilient
- Working through the equations, the system remains correctly schedulable as long as $u_3^L \leq 0.4$; equivalently: $C_3^L \leq 0.4 \times T_3 = 0.4 \times 30 = 12$
- The system can therefore be scheduled in a robust manner by terminating τ_1 and τ_2 only upon some job of τ_3 executing beyond 12 time units (rather than $C_3^L = 3$ time units)
- A reasonable quantitative metric of the robustness of this schedule is the ratio of these two quantities: $12/3$, or **4.**

Incorporating Resilience

- This maximum robustness comes at the cost of no further resilience being possible
- Resilience involves reducing the rate of progress of the LO-criticality tasks
- It is then application specific how these tasks cope with having less load than required for normal behaviour

Incorporating Resilience

- Working through the example again it is easy to show that if we were to reduce the sum of the rates of the LO-criticality tasks τ_1 and τ_2 to $3/8$ (i.e., 0.375) from 0.5 — a reduction to $\frac{3}{4}$ of the desired level of service — upon some job of τ_3 executing for beyond 3 time units, we would not need to degrade service to τ_1 and τ_2 any further as long as τ_3's jobs do not exceed their HI-criticality WCET estimate of 18 time units.
- This factor of $\frac{3}{4}$ may be considered a quantitative metric of the resilience of this schedule.

Robustness and Resilience

- In reality we would want both robustness and resilience
- But we cannot maximise both
- We can however choose an attainable robustness value and then maximise the resilience metric
- With the example we can achieve, for example, $(\mathbf{2}, \frac{2}{3})$ as apposed to $(\mathbf{4}, 0)$ or $(\mathbf{1}, \frac{3}{4})$
- In general $(\mathbf{r}, \frac{4-r}{5-r})$, for a valid $\mathbf{r}$

Conclusions

- Pre-runtime verification and run-time survivability are two distinct aspects of correctness in safety-critical systems
- Mixed-criticality scheduling theory (MCSh) has, thus far, focused almost exclusively on the verification aspect
- In this paper we have described some of our ongoing efforts at extending MCSh to incorporate survivability considerations
- We have proposed quantitative metrics of both aspects of survivability – robustness and resilience – for mixed-critical task systems that are represented using the Vestal model

Future Work

- As future work we plan to subject other mixed-critical scheduling algorithms that have been proposed (such as AMC, EDF-VD, etc.) to the same form of analysis as we have done here with MC-Fluid, and thereby develop survivable implementations of systems that are based upon these non-fluid mixed-criticality scheduling algorithms
- Also as future work, we plan to revisit some mixed-criticality scheduling algorithms that have previously been proposed for addressing the non-survivability of traditional mixed-criticality scheduling algorithms. We will seek to characterize the robustness and resilience properties of these algorithms on the basis of the metrics that we have proposed in this paper