Response-Time Analysis for Mixed Criticality Systems

S.K. Baruah, A. Burns and R.I. Davis
Background

- Mixed criticality systems are becoming a distinct focus for research and industrial application

- Two key issues:
  1. Run-time Robustness
  2. Static Verification

- This paper focuses on the latter
Requirements

- In any multi-application system, failures must be confined to the application experiencing the fault.

- In particular, in mixed criticality systems, failure of a low criticality application must not compromise higher criticality applications.

- But the over provision of resources to high criticality tasks could lead to poor schedulability.
Constraints of the work

- Uni-processor
- Sporadic task model
- No shared resources/blocking
- No overhead costs
- Fixed Priority Scheduling
- Two Criticality Levels
Each task, $\tau_i$, is defined by its period (minimum arrival interval), deadline, computation time and criticality level:

$T_i, D_i, C_i, L_i$

but worst-case computation time is a function of criticality, so:

$T_i, D_i, \vec{C}_i, L_i$
Criticality Level

- High criticality tasks use WCET estimation techniques that are inheritably more conservative than those for low criticality tasks.

- So, $L_1 > L_2 \Rightarrow C(L_1) \geq C(L_2)$ for any two criticality levels $L_1$ and $L_2$.
Criticality Level

- Task $\tau_i$ with criticality level $L_i$ will have one value from its $\vec{C}_i$ vector that defines its representative computation time.

- This is the value corresponding to $L_i$, i.e., $C_i(L_i)$.

- This will be given the normal symbol $C_i$. 

Response-Time Analysis for Mixed Criticality Systems – p. 7/30
Implementation Schemes

- Partitioned Criticality (PC) – a standard scheme sometimes called *criticality monotonic priority assignment*
- Static Mixed Criticality (SMC)
- Adaptive Mixed Criticality (AMC)
Partitioned

- Priorities are assigned according to criticality, so all jobs of criticality L1 have a higher priority than all jobs of criticality L2 if L1 \( \geq \) L2
- No run-time monitoring is required
- Poor schedulability
Static - SMC

- All jobs can execute up to their representative execution time $C_i$ (and possible beyond)
- Priorities are assigned (via Audsley’s algorithm) to maximise schedulability
- As a result a task with low criticality can have a high priority
Response Time Analysis

For a single criticality system

\[ R_i = C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j \]

This is solved using standard techniques for recurrence relations.
If there is no run-time monitoring

\[ R_i = C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j(L_i) \]
If there is run-time monitoring

\[ R_i = C_i + \sum_{\tau_j \in \text{hp}(i)} \left\lfloor \frac{R_i}{T_j} \right\rfloor C_j(\min(L_i, L_j)) \]

So a low criticality task \( \tau_j \) only needs its low-crit WCET \( C_j \) but is prevented from executing beyond \( C_j \).
Adaptive Mixed Criticality (AMC)

- Now if a high-crit task executes for more than its low-crit WCET, all low crit tasks are abandoned.
- This significantly increases schedulability.
  - By utilising the ‘reserved’ capacity of high-crit tasks.
RTA for AMC

Three stages to the analysis:

1. Verifying the schedulability of the LO-criticality mode,
2. Verifying the schedulability of the HI-criticality mode,
3. Verifying the schedulability of the criticality change itself.
Stage 1 - All tasks

\[ R_{i}^{LO} = C_{i}(LO) + \sum_{j \in h_{p}(i)} \left\lceil \frac{R_{i}^{LO}}{T_{j}} \right\rceil C_{j}(LO) \]

where \( h_{p}(i) \) is the set of all tasks with priority higher than that of task \( \tau_{i} \).
Stage 2 - HI only

\[ R_{i}^{HI} = C_i + \sum_{j \in hpH(i)} \left\lceil \frac{R_{i}^{HI}}{T_j} \right\rceil C_j \]

where \( hpH(i) \) is the set of HI-critical tasks with priority higher than, or equal to, that of task \( \tau_i \)
Stage 3 - HI only

- Need to analysis the worst-case change from LO to HI behaviour
- Similar problem to mode change analysis
- Worst-case may not be when sporadic tasks arrive at their worst-case, hence tractable analysis is unlikely to exist
Stage 3 - HI only

- Only care about HI-crit tasks
- For HI-crit task $\tau_i$ behaviour change must occur between release and the completion time in LO ‘mode’
- So in the interval: $[0, R_i^{LO})$
- We consider two methods for determining sufficient schedulability
For a HI-crit task, the earlier equation becomes

\[ R_i = C_i + \sum_{\tau_j \in hpH(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j + \]

\[ \sum_{\tau_k \in hpL(i)} \left\lceil \frac{R_i}{T_k} \right\rceil C_k \]

But the final term is bounded by \( R_{i}^{LO} \), so
We refer to this method as AMC-rtb
Here we examine all intermediate points $s$ in $[0, R_{i}^{LO})$ and take the maximum (interference).

Only values of $s$ at which a LO-crit task is released need to be considered.

We use analysis that is compatible with Audsley’s priority assignment algorithm.

At time $s$ we conservatively assume ‘active’ HI-crit task consume $C(HI)$ and ‘active’ LO-crit task complete and use $C(LO)$.
Example

Consider an example task system $\tau$ comprised of three tasks, as follows:

<table>
<thead>
<tr>
<th>$\tau_i$</th>
<th>$L$</th>
<th>$C_i(LO)$</th>
<th>$C_i(HI)$</th>
<th>$D_i$</th>
<th>$T_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_1$</td>
<td>LO</td>
<td>1</td>
<td>-</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$\tau_2$</td>
<td>HI</td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>$\tau_3$</td>
<td>HI</td>
<td>20</td>
<td>20</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Deemed unschedulable by either partitioned priorities or SMC, AMC(1) gives $R_3$ as 85, AMC(2) gives 59
AMC and SMC

- AMC method 2 dominated method 1
- AMC method 1 dominates SMC
- SMC dominates SMC-no
- SMC-no dominated partitioned priorities
Evaluation: \( N=20, 50\% \text{ HI, C}^*2 \)
Evaluation: weighted
Evaluation: weighted

![Graph showing weighted response-time analysis for mixed criticality systems. The graph plots weighted schedulability against the percentage of tasks with high criticality. Different lines represent various scheduling algorithms, including UB-H&L, AMC-max, AMC-rtb, SMC, SMC-NO, and CrMPO. The graph illustrates the trade-offs between schedulability and the proportion of high-critical tasks.]
Evaluation: weighted

![Graph showing weighted schedulability vs taskset size]

- **UB-H&L**: Black dashed line
- **AMC-max**: Red line
- **AMC-rtb**: Blue line
- **SMC**: Green line
- **SMC-NO**: Orange line
- **CrMPO**: Brown line

Taskset size ranges from 8 to 88.
Evaluation: D<T
Conclusion

- Mixed Criticality systems are becoming increasingly important.
- Smart scheduling can significantly increase resources usage.
- The proposed AMC scheme is a significant improvement on SMC.
- Simple AMC analysis gets a long way to towards the optimal.
- Sensitivity analysis could be used to increase C(LO) values.