THE UNIVERSITY of York

RTS York

Compensating Adaptive Mixed Criticality Scheduling

Robert Davis, Alan Burns, Iain Bate

Real-Time Systems Research Group, University of York, UK

Background: Mixed Criticality Systems

- **Mixed Criticality Systems**
  - Criticality is a designation of the level of assurance against failure needed for a system component
  - A mixed criticality system is one that has two or more distinct *criticality levels*
  - Tasks are characterized by their criticality level either HI or LO
  - LO-criticality tasks have a single LO-criticality estimate of their WCET, $C_i(LO)$
  - HI-criticality tasks have an additional HI-criticality estimate, $C_i(HI)$

- **Academic perspective**
  - Most academic work assumes that if a HI-criticality task executes for $C_i(LO)$ without completing, then only jobs of HI-criticality tasks need meet their deadlines, and so jobs of LO-criticality tasks can be dropped in a *degraded mode* in order to ensure that HI-criticality tasks meet their deadlines
- **Industry perspective**
  - However industry takes a differ view of the importance of LO-criticality tasks, with many practical systems unable to tolerate their abandonment

Introduction: Compensating Adaptive Mixed Criticality (C-AMC)

- **This work**
  - Takes the view that abandoning new jobs of LO-criticality tasks is not acceptable, and so jobs of both HI- and LO-criticality tasks must *always* meet their deadlines
  - We propose the **Compensating Adaptive Mixed Criticality (C-AMC)** scheduling scheme that meets these stricter requirements
  - C-AMC ensures that both HI- and LO-criticality tasks meet their deadlines in normal (or LO-criticality) mode and also in degraded (or HI-criticality) mode
  - Once degraded mode is entered, newly released jobs of LO-criticality tasks execute *imprecise versions* that provide essential functionality and outputs of sufficient quality, while reducing overall workload via smaller $C_i(HI)$ execution time budgets ($C_i(HI) \leq C_i(LO)$)
  - This adaptive behavior compensates, at least in part, for the longer execution times ($C_k(HI) \geq C_k(LO)$) that may be exhibited by jobs of HI-criticality tasks, for example executing error handling code that is not expected to execute during normal mode

System Model

- **Mixed criticality system**
  - Set of N sporadic (or periodic) tasks that execute on a single-core processor
  - Tasks have constrained deadlines that do not exceed their periods ($D_i \leq T_i$)
  - Each task has a unique priority
  - Two criticality levels: LO and HI
  - Each task has two execution time budgets $C_i(LO)$ and $C_i(HI)$
  - For a HI-criticality task $C_k(LO)$ and $C_k(HI)$ are the low assurance and the high assurance estimates of the WCET of its *primary version*, which is the only version that it executes, hence $C_k(HI) \geq C_k(LO)$
  - For a LO-criticality task $C_i(LO)$ and $C_i(HI)$ are the low assurance estimates of the WCET of, respectively, its *primary version* and its *imprecise version*, hence $C_i(LO) \geq C_i(HI)$

HI-criticality task

$C_i(LO)$  Primary version

$C_i(HI)$  Primary version

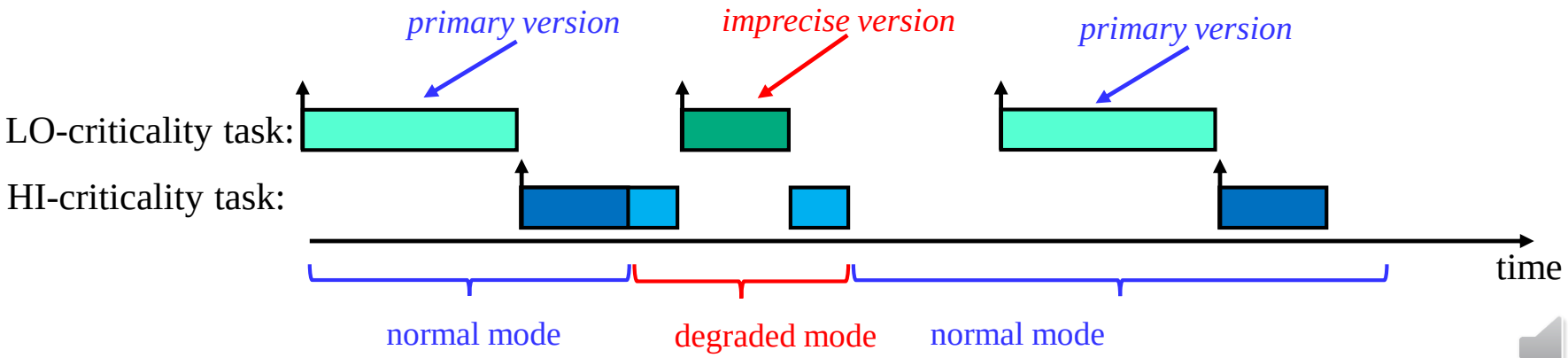
LO-criticality task

$C_i(LO)$  Primary version

$C_i(HI)$  Imprecise version

System Model

- **RTOS**
  - Executes tasks under fixed-priority pre-emptive scheduling
  - Responsible for transitioning the system between normal and degrade mode
  - The system switches to degraded mode when a HI-criticality task executes for its $C_k(LO)$ without completing and switches back to normal mode on an idle instant
  - Ensures that all jobs of HI-criticality tasks execute their primary version, and that jobs of LO-criticality tasks released in normal mode also execute their primary version, while jobs of LO-criticality tasks released in degraded mode execute their imprecise version



Analysis for C-AMC

- **Schedulability tests**
  - We derive two schedulability tests for C-AMC based on the AMC-rtb and AMC-max tests for the AMC scheme. (Recall that the AMC scheme drops LO-criticality jobs in degraded mode)
  - In contrast to AMC, the tests for C-AMC check schedulability of both LO- an HI-criticality tasks in both normal and degraded modes, thus guaranteeing that all jobs of LO- and HI- criticality tasks meet their deadlines
  - The C-AMC scheme provides the flexibility to have LO-criticality tasks with $0 \leq C_j(HI) \leq C_j(LO)$, so some LO-criticality tasks could drop jobs ($C_j(HI) = 0$), others could run imprecise versions, while others could continue to run primary versions ($C_j(HI) = C_j(LO)$). This is all covered by the analysis

Analysis for C-AMC: C-AMC-rtb schedulability test

- **Normal mode**

$$R_i(LO) = C_i(LO) + \sum_{j \in \mathbf{hp}(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil C_j(LO)$$

2. Assumes that jobs of all higher priority tasks can cause interference of $C_j(HI)$ throughout the entire time interval

- **Degraded mode**

$$R_i(HI) = \max(C_i(LO), C_i(HI)) + \sum_{j \in \mathbf{hp}(i)} \left\lceil \frac{R_i(HI)}{T_j} \right\rceil C_j(HI)$$

1. Accounts for the larger of the two execution time budgets for each task

$$+ \sum_{j \in \mathbf{hpL}(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil (C_j(LO) - C_j(HI))$$

3. Adjusts for the fact that higher priority LO-criticality tasks released by $R_i(LO)$ can cause interference of $C_j(LO) \geq C_j(HI)$

Analysis for C-AMC: C-AMC-max schedulability test

- **Avoiding pessimism**
  - C-AMC-rtb test is somewhat pessimistic, since it includes the larger contribution of $C_j(HI)$ from jobs of HI-criticality tasks over the entire response time $R_i(HI)$ (effectively assuming a mode change at time $s = 0$) and also the larger contribution of $C_j(LO)$ from jobs of LO-criticality tasks over $R_i(LO)$ (effectively assuming a mode change at time $s = R_i(LO)$); however, the mode change cannot simultaneously be both as early as possible and as late as possible
  - The C-AMC-max test seeks to eliminate this pessimism by considering the various different times at which the mode change could take place and taking the maximum response time over all of those values of s

- **Normal mode**
  - Same as C-AMC-rtb

$$R_i(LO) = C_i(LO) + \sum_{j \in \mathbf{hp}(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil C_j(LO)$$

Analysis for C-AMC: C-AMC-max schedulability test

- **Degraded mode**

$$R_i^s(HI) = \max(C_i(HI), C_i(LO)) + I_L(i, s, R_i^s(HI)) + I_H(i, s, R_i^s(HI))$$

  - Interference from higher priority LO-criticality tasks

1. Larger $C_j(LO)$ contribution up to the mode change at time s

$$I_L(i, s, t) = \sum_{j \in \mathbf{hpL}(i)} \left(\left\lceil \frac{t}{T_j} \right\rceil C_j(HI) + \left(\left\lfloor \frac{s}{T_j} \right\rfloor + 1 \right) (C_j(LO) - C_j(HI)) \right)$$

  - Interference from higher priority HI-criticality tasks

$$I_H(i, s, t) = \sum_{k \in \mathbf{hpH}(i)} \left\lceil \frac{t}{T_k} \right\rceil C_k(LO) +$$

2. Larger $C_j(HI)$ contribution after the mode change at time s

$$\sum_{k \in \mathbf{hpH}(i)} \min \left\{ \left\lceil \frac{t - s + D_k}{T_k} \right\rceil, \left\lceil \frac{t}{T_k} \right\rceil \right\} (C_k(HI) - C_k(LO))$$

Analysis for C-AMC: C-AMC-max schedulability test

- **Degraded mode**
  - Putting it all together

$$R_i^s(HI) = \max(C_i(HI), C_i(LO)) +$$

$$\sum_{j \in \text{hpL}(i)} \left(\left\lceil \frac{R_i^s(HI)}{T_j} \right\rceil C_j(HI) + \left(\left\lfloor \frac{s}{T_j} \right\rfloor + 1 \right) (C_j(LO) - C_j(HI)) \right) +$$

$$\sum_{k \in \text{hpH}(i)} \left\lceil \frac{R_i^s(HI)}{T_k} \right\rceil C_k(LO) +$$

$$\sum_{k \in \text{hpH}(i)} \min \left\{ \left\lceil \frac{R_i^s(HI) - s + D_k}{T_k} \right\rceil, \left\lceil \frac{R_i^s(HI)}{T_k} \right\rceil \right\} (C_k(HI) - C_k(LO))$$

  - Take maximum over all values of *s* corresponding to releases of higher priority LO-criticality tasks before $R_i(LO)$

$$R_i(HI) = \max_{\forall s, s < R_i(LO)} \left\{ R_i^s(HI) \right\}$$

Experimental Evaluation

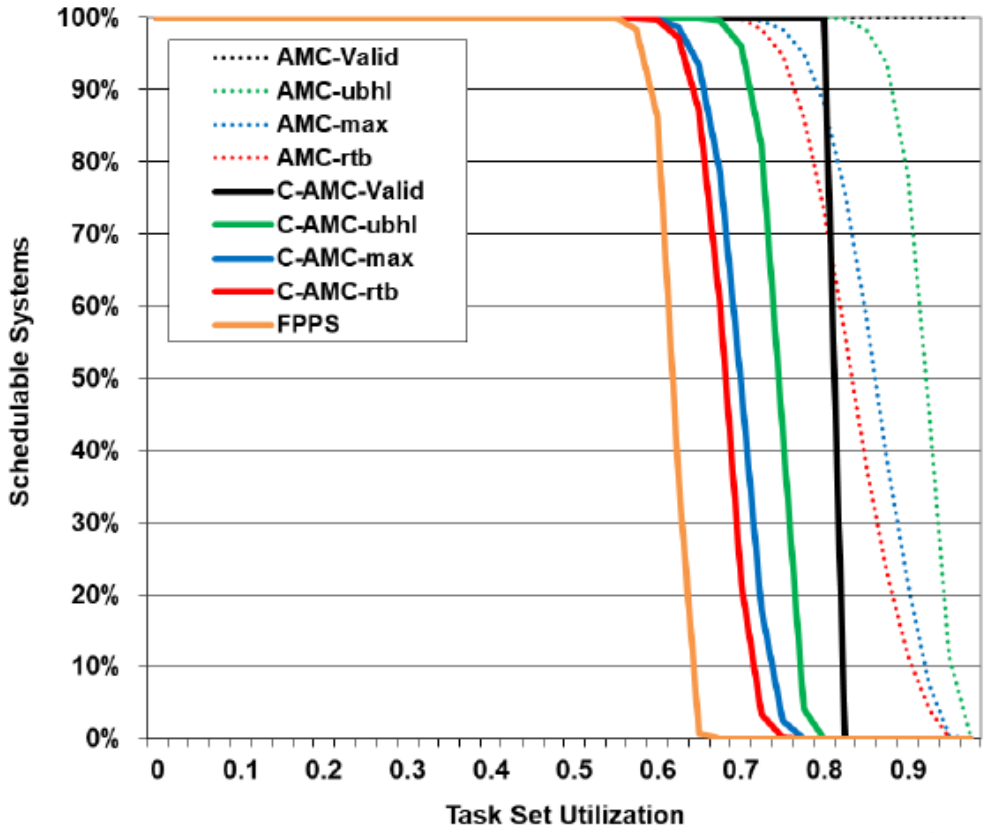
- **C-AMC tests**
  - **C-AMC-valid** necessary feasibility test that checks that utilization does not exceed 1 in either normal or degraded mode, and ignores the mode change transition
  - **C-AMC-ubhl** feasibility condition that checks schedulability in both normal and degraded mode assuming FPPS, ignoring the mode change transition
  - **C-AMC-max** schedulability test
  - **C-AMC-rtb** schedulability test
- **AMC tests**
  - **AMC-valid** necessary feasibility test that checks that utilization does not exceed 1 in either normal or degraded mode, and ignores the mode change transition
  - **AMC-ubhl** feasibility condition that checks schedulability in both normal and degraded mode assuming FPPS, ignoring the mode change transition
  - **AMC-max** schedulability test
  - **AMC-rtb** schedulability test
- **Optimal Priority Assignment**
  - Using Audsley's algorithm, which is optimal for all of these tests

Experimental Evaluation

- **Task parameters**
  - Varied over a wide range of different settings using a set of controlling parameters
  - **CP** is the **Criticality Proportion** – the proportion of tasks that were designated HI-criticality (default 0.5)
  - **CF** is the **Criticality Factor** – the ratio between the utilization of HI-criticality tasks assuming their $C_j(HI)$ values versus assuming their $C_j(LO)$ values (default 2.0)
  - **XF** is the **Compensating Factor –** the ratio between the utilization of LO-criticality tasks assuming their $C_j(HI)$ values versus assuming their $C_j(LO)$ values (default 0.5)
  - See the paper for full details of task set generation

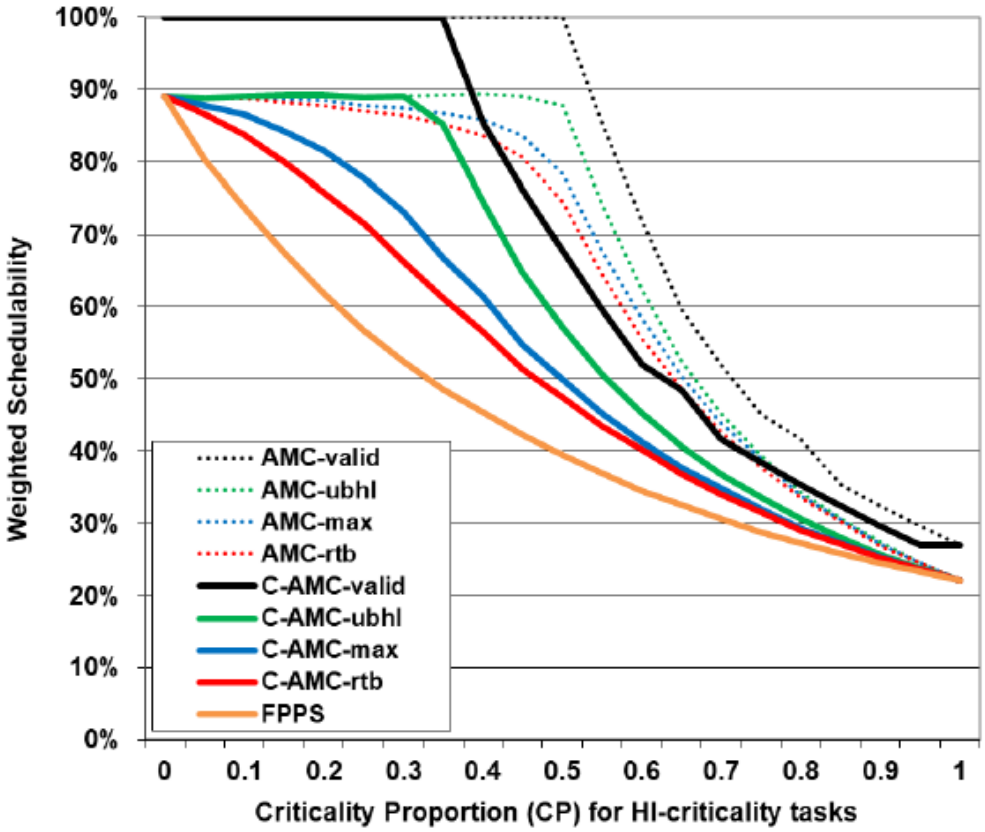
Results: Success ratio

- **Varying task set utilization**
  - C-AMC-max has a small but useful advantage over C-AMC-rtb
  - The C-AMC scheme provides substantial improvements over the single criticality default (FPPS)
  - Performance of the tests follows the dominance relations between them
  - Comparison with AMC shows that as expected there is a performance penalty in providing truly graceful degradation



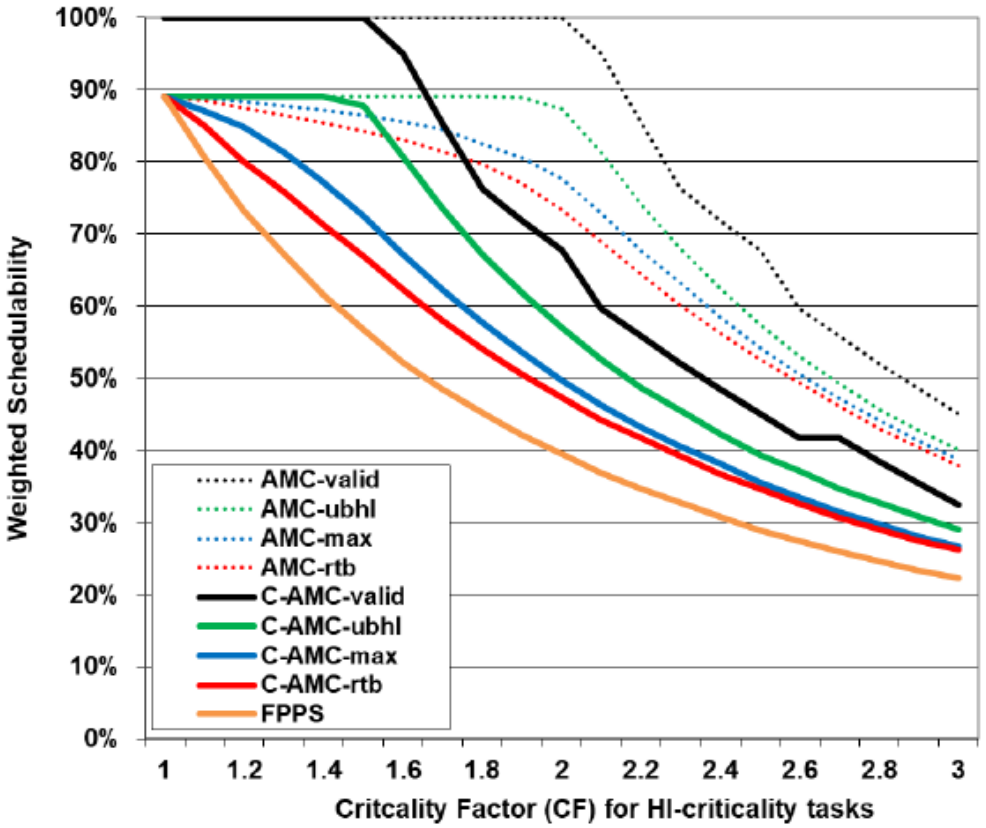
Results: Weighted schedulability

- **Varying Criticality Proportion**
  - CP is the proportion of HI-criticality tasks, so CP=0 and CP=1 are single criticality systems where the schedulability tests reduce to those for FPPS
  - For CP in the range [0.1,0.4], the C-AMC-rtb and C-AMC-max tests provide significant gains over a single criticality approach. This is a result of the workload reduction due to executing imprecise versions



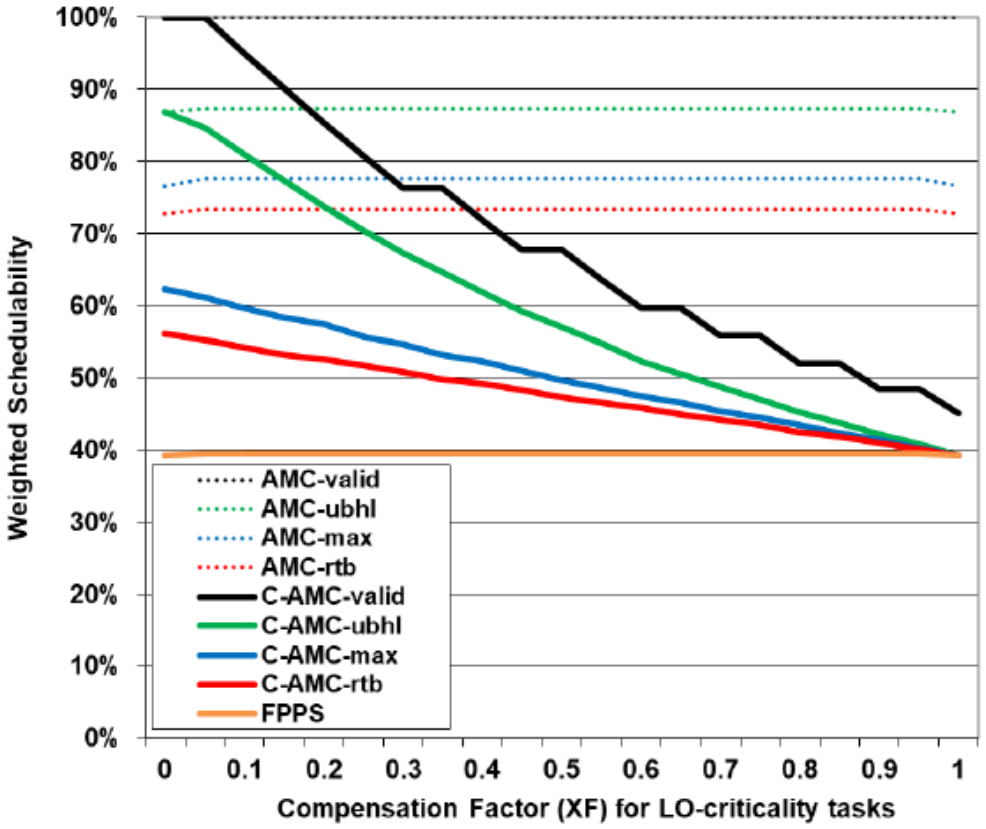
Results: Weighted schedulability

- **Varying Criticality Factor**
  - $CF = U_{HI}^{HI} / U_{LO}^{HI}$
  - A smaller value of CF in the range [1.1,1.8] results in a smaller workload from HI-criticality tasks in degraded mode that can be more effectively compensated for by the reduction in workload of LO-criticality tasks
  - CF =1 means that the workload from HI-criticality tasks does not increase in degraded mode, and hence that mode is never actually entered, so effectively we have a single criticality system



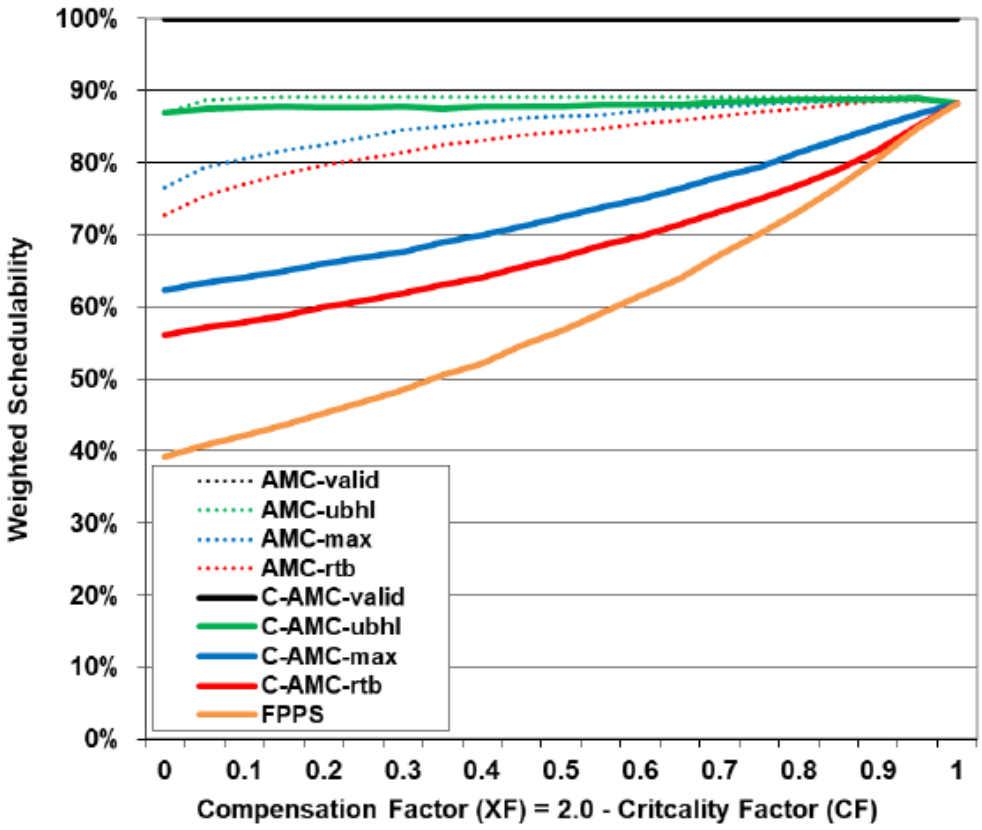
Results: Weighted schedulability

- **Varying Compensating Factor**
  - $XF = U_{LO}^{HI} / U_{LO}^{LO}$
  - AMC drops LO-criticality jobs in degraded mode, hence XF has no impact on AMC schedulability (horizontal lines on the graph)
  - When XF=0, C-AMC still guarantees schedulability of LO-criticality jobs that execute across the mode change, whereas AMC does not. This explains the difference in **-rtb** and **-max** schedulability test performance at this point. By contrast the **-valid** and **-ubhl** feasibility tests ignore the transition and so are the same in this case
  - As expected, the smaller the value of XF, the greater the improvement in performance for C-AMC compared to the single criticality baseline



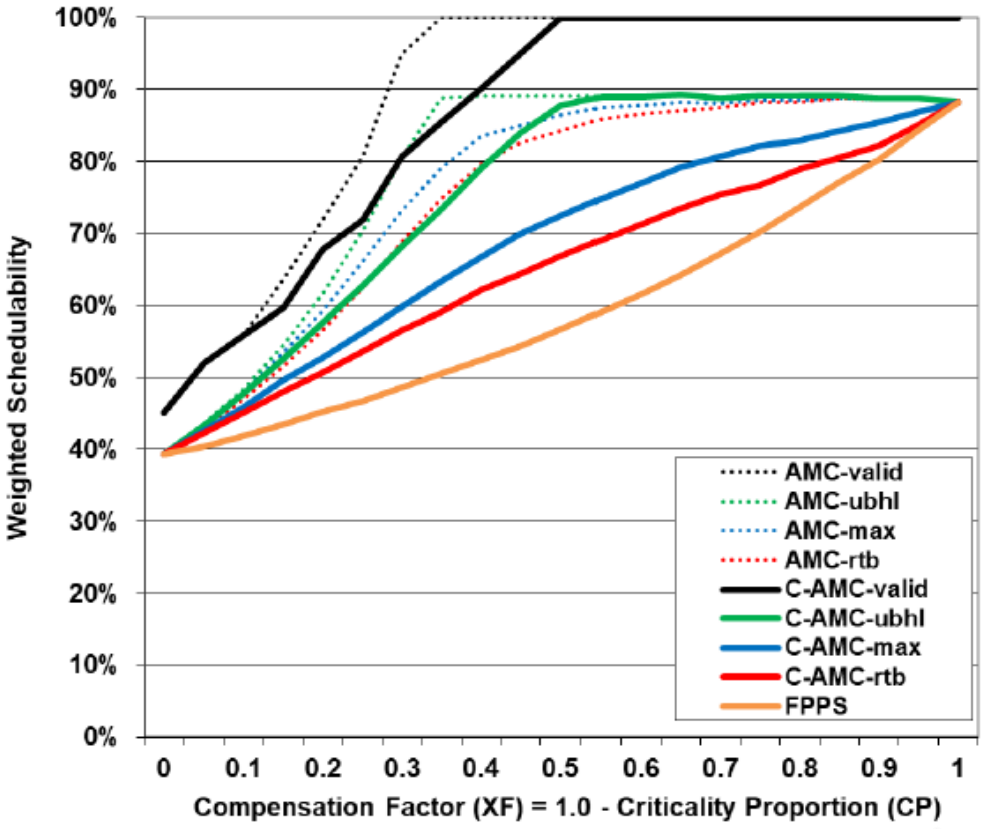
Results: Weighted schedulability

- **Varying both XF and CF**
  - XF and CF varied in opposition to each other such that the utilization in degraded mode is held constant
  - When XF =1 and CF =1 the workload for both LO- and HI-criticality tasks remains constant and hence we have a single criticality system
  - As CF increases and XF decreases the workload change across the transition from normal to degraded mode becomes larger and harder to schedule. In this case C-AMC substantially improves upon the single criticality baseline
  - Note the **-ubhl** feasibility tests only consider schedulability in normal and degraded mode and not across the transition, hence the horizontal lines on the graph



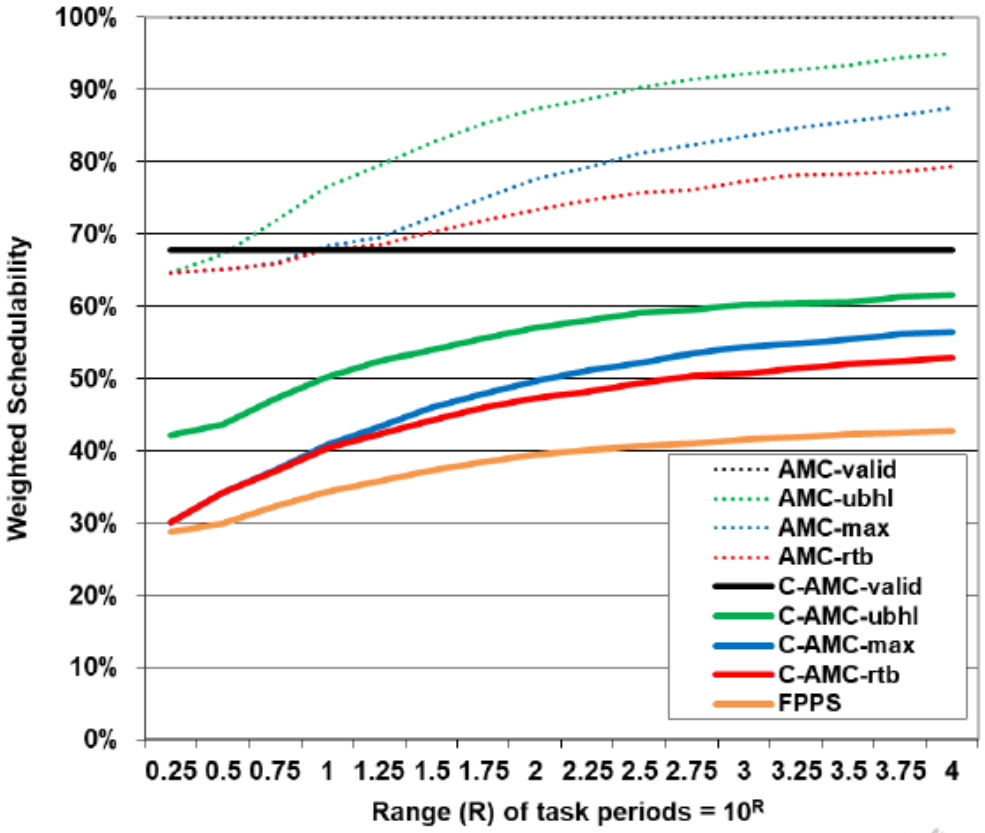
Results: Weighted schedulability

- **Varying both XF and CP**
  - XF and CP varied in opposition to each other.
  - Here, the utilization in degraded mode is held constant for XF > 0.5 (CF was set to 1.5 to achieve this), but then increases as XF gets smaller and CP larger, due to an increased proportion of HI-criticality tasks that cannot be fully compensated for
  - At either extreme, we have a single criticality system (CP=0 or CP=1)
  - At intermediate values, the decrease in workload due to running imprecise versions of LO-criticality tasks compensates at least partially for the increase in workload of HI-criticality tasks in degraded mode. Thus C-AMC improves upon the performance of the single criticality baseline



Results: Weighted schedulability

- **Varying range of task periods**
  - Range of task periods 10^R varied from $10^{0.25}$=1.77 to 10^4=10,000
  - As expected for schemes based on FPPS schedulability improves as the range of task periods increases
  - For small ranges ≤10, the **-max** test shows minimal if any improvement over the **-rtb** test. The reason for this is that when all tasks have roughly the same period, both tests include just one job of each higher priority task at its larger execution time
  - For larger ranges ≥1,000 the **-max** test provides a larger improvement over the **-rtb** test



Conclusions: Summary

- **Perspectives**
  - Academic research often assumes that jobs of LO-criticality tasks can be dropped in order to ensure that HI-criticality tasks will meet their deadlines
  - Industry, however, takes a different view of the importance of LO-criticality tasks, with many practical systems unable to tolerate their abandonment
- **This work**
  - Introduced the **Compensating Adaptive Mixed Criticality (C-AMC) scheduling scheme**
  - Under C-AMC, jobs of LO-criticality tasks that are released in degraded mode execute imprecise versions. These imprecise versions are able to provide outputs of sufficient quality, while also reducing the overall workload
  - This compensates, at least in part, for the overload due to HI-criticality tasks, which while always executing their primary versions, may also run error handling code that is not expected to execute during normal operation

Conclusions: Contribution

- **Compensating Adaptive Mixed Criticality (C-AMC)**
  - Ensures that both HI- and LO-criticality tasks meet their deadlines in both normal and degraded modes
  - Supports a form of degradation that is *genuinely* graceful, while reducing LO-criticality workload to compensate for unexpected increases in HI-criticality workload
  - The C-AMC-rtb and C-AMC-max tests substantially improve schedulability compared to the single criticality baseline that is common practice in industry
  - The C-AMC scheme provides a viable migration path for industry to make an evolutionary transition from current practice, which is predominantly based on fixed-priority pre-emptive scheduling
  - C-AMC addresses one of the key open issues identified in the survey of research into mixed criticality systems by adding *"support for limited low-criticality functionality in higher criticality modes, avoiding the abandonment problem."*

RTS York

Discussion and Questions?

$$R_i(LO) = C_i(LO) + \sum_{j \in hp(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil C_j(LO)$$

2. Assumes that jobs of all higher priority tasks can cause interference of $C_j(HI)$ throughout the entire response time

$$R_i(HI) = \max(C_i(LO), C_i(HI)) + \sum_{j \in hp(i)} \left\lceil \frac{R_i(HI)}{T_j} \right\rceil C_j(HI) + \sum_{j \in hpL(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil (C_j(LO) - C_j(HI))$$

1. Accounts for the larger of the two execution time budgets for each task

3. Adjusts for the fact that higher priority LO-criticality tasks released by $R_i(LO)$ can cause interference of $C_j(LO) \geq C_j(HI)$

Compensating Adaptive Mixed Criticality (C-AMC)

rob.davis@york.ac.uk