Mixed Criticality Systems with Weakly-Hard Constraints

Oliver Gettings
University of York
oliver@cs.york.ac.uk

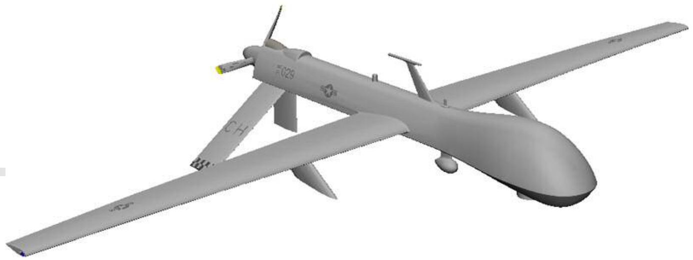
Sophie Quinton
INRIA Grenoble
sophie.quinton@inria.fr

Rob Davis
University of York
rob.davis@york.ac.uk

RTS York

THE UNIVERSITY of York

Mixed Criticality Systems



- Mixed Criticality
  - Criticality is the required level of assurance against failure
  - Mixed Criticality Systems contain applications of at least two criticality levels
  - Examples: Aerospace – Flight Control Systems v. Surveillance
    Automotive – Electric Power Steering v. Cruise Control

- Motivation for MCS
  - Driven by Size, Weight and Power (SWaP) and cost requirements
  - Applications with different criticalities (safety critical, mission critical etc.) on the same HW platform

- This research:
  - Dual-Criticality - Applications of HI and LO criticality

Mixed Criticality Systems

- Key requirements
  - *Separation* – must ensure that LO-criticality applications cannot impinge on those of HI-criticality
  - *Sharing* – want to allow LO- and HI-criticality applications to use the same resources for efficiency
- Real-Time behaviour
  - Concept of a criticality mode (LO or HI)
  - LO and HI-criticality applications must meet their time constraints in LO-criticality mode
  - **Only HI-criticality applications need meet their time constraints in HI-criticality mode (?)**
- Initial Research (Vestal 2007)
  - Idea of different LO- and HI-criticality WCET estimates for the same code
  - Certification authority requires pessimistic approach to C^{HI}
  - System designers take a more realistic approach to C^{LO}

System Model

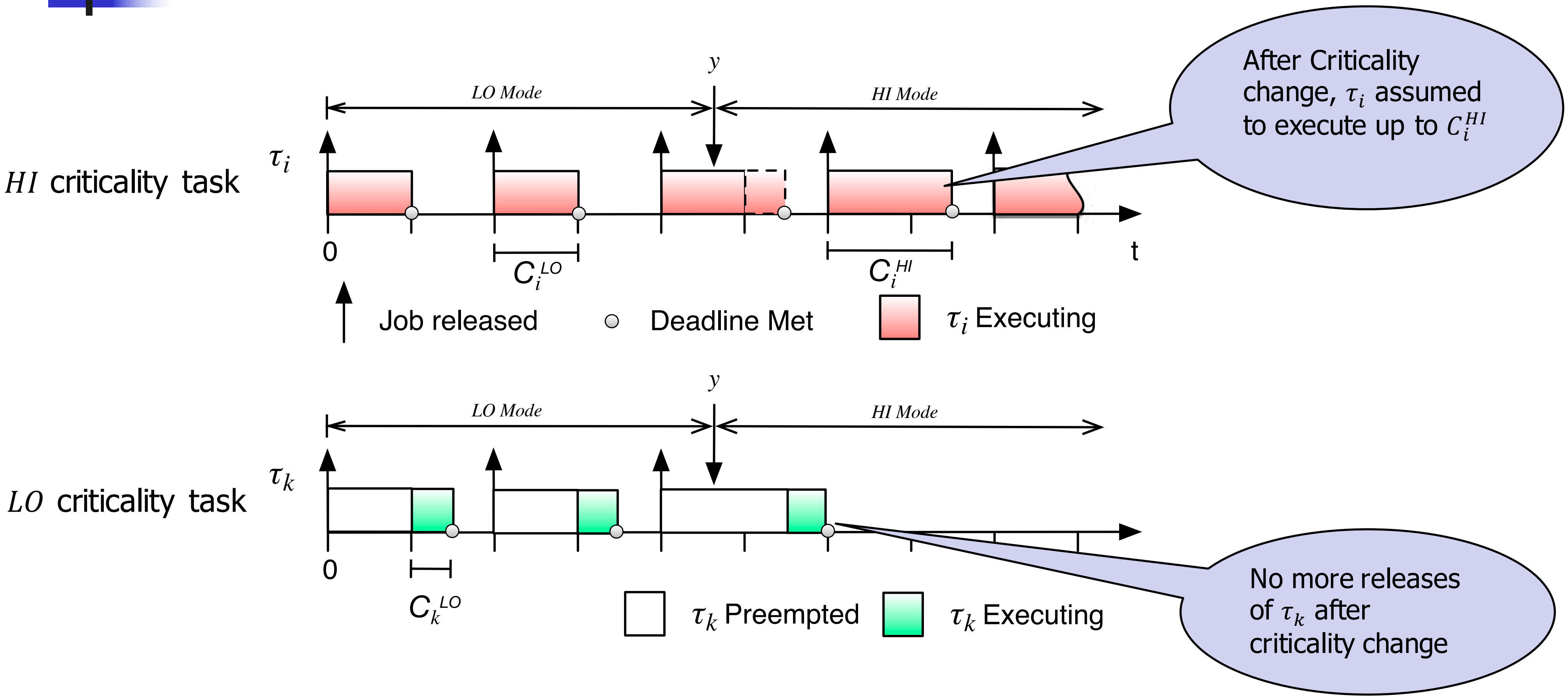
- Uniprocessor, fixed priority pre-emptive scheduling
- Sporadic task sets where a task, $\tau_i = (T_i, D_i, \overrightarrow{C_i}, L_i)$
  - T_i - Task period or minimum inter-arrival time
  - D_i - Relative deadline
  - C_i^l - WCET of τ_i at criticality level l
  - L_i - Designated criticality level for τ_i
- $hp(i)$ - Set of higher priority tasks (than τ_i)
- $hpHI(i)$ - Set of higher priority, HI criticality tasks
- $hpLO(i)$ - Set of higher priority, LO criticality tasks

Recap: Adaptive Mixed Criticality

- AMC scheduling scheme
  - If a HI-criticality task executes for its C^{LO} without signalling completion then no further jobs of LO-criticality tasks are started[1] and the system enters HI-criticality mode
  - This frees up processor bandwidth to ensure that HI-criticality tasks can meet their deadlines in HI-criticality mode
  - **But, … it has the drawback that LO-criticality functionality is completely abandoned**

[1]Any partially executed job of each LO-criticality task may complete

Recap: Adaptive Mixed Criticality



Recap: AMC-rtb Analysis

LO-criticality mode

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

HI-criticality mode

$$R_i^{HI} = C_i^{HI} + \sum_{j \in \boldsymbol{hpHI}(i)} \left\lceil \frac{R_i^{HI}}{T_j} \right\rceil C_j^{HI}$$

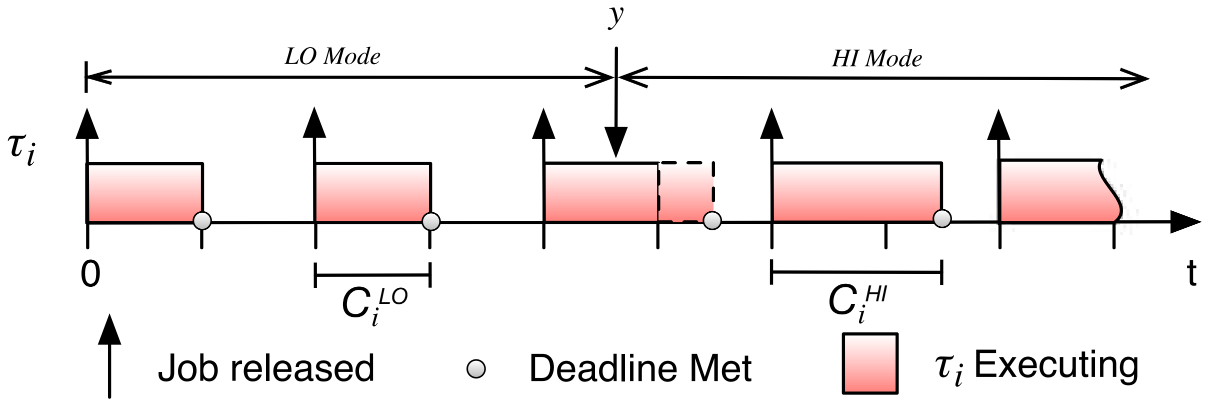
Interference from higher priority LO-criticality tasks only up to R^{LO}

Mode change transition

$$R_i^{*} = C_i^{HI} + \sum_{j \in \boldsymbol{hpHI}(i)} \left\lceil \frac{R_i^{*}}{T_j} \right\rceil C_j^{HI} + \sum_{k \in \boldsymbol{hpLO}(i)} \left\lceil \frac{R_i^{LO}}{T_k} \right\rceil C_k^{LO}$$

Recap: AMC-max Analysis

- AMC-rtb analysis assumes (pessimistically) that **all** jobs of HI-criticality tasks execute with their C^{HI} values
- AMC-max removes this pessimism



Calculates number of releases after criticality change up to t

$$M(i, y, t) = min\left\{\left\lceil\frac{t + y + D_i}{T_i}\right\rceil, \left\lceil\frac{t}{T_i}\right\rceil\right\}$$

Recap: AMC-max Analysis

AMC-max Criticality Mode Change $(LO \rightarrow HI)$ at time y

$$R_i^y = C_i^{HI} + \sum_{k \in \boldsymbol{hpLO}(i)} \left(\left\lfloor \frac{y}{T_k} \right\rfloor + 1 \right) C_k^{LO} + \sum_{j \in \boldsymbol{hpHI}(i)} \left(M(j, y, R_i^y) C_j^{HI} + \left(\left\lceil \frac{R_i^y}{T_j} \right\rceil - M(j, y, R_i^y) \right) C_j^{LO} \right)$$

- Values of y that need to be assessed are bounded by 0 and R^{LO}.
- Values of y at which response time may change correspond to releases of higher priority, LO-criticality tasks:

$$R_i^* = \max(R_i^y) \forall y \text{ where } y \in kT_j \mid \forall j \in hpLO(i) \wedge y \leq R_i^{LO} \mid \forall k : \mathbb{N}$$

AMC Abandonment Problem

- Abandoning all LO-criticality jobs
  - Is not acceptable in many real systems
  - May lead to loss of important functionality as LO-criticality tasks are still critical (not non-critical)

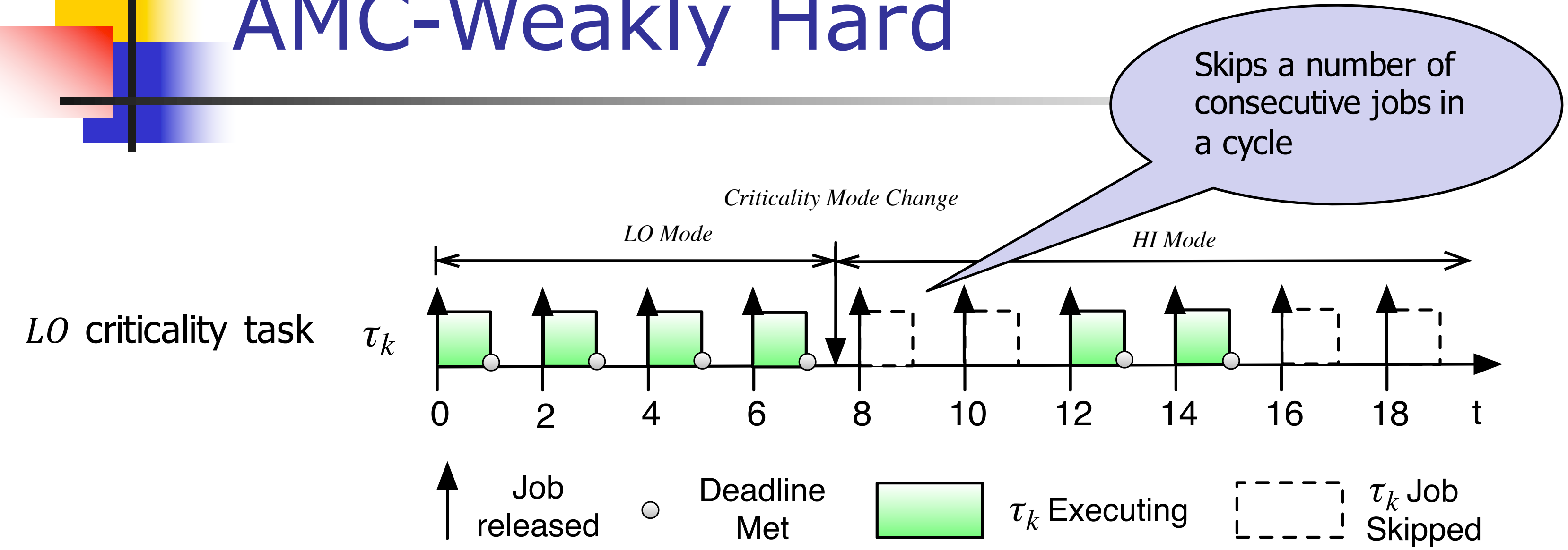
- This work:
  - Aims to address the abandonment problem by combining AMC with an existing concept called *Weakly-Hard*
  - Provides a guaranteed minimum quality of service for LO-criticality tasks in HI-criticality mode – graceful degradation

AMC-Weakly Hard

- Weakly Hard Model
  - Proposed in 2001 by Guillem Bernat *et al.*
  - Guarantees that $(\mathrm{m} - s)$ out of any m deadlines are met via (somewhat complex) offline analysis

- AMC-Weakly Hard
  - Combines a simple interpretation of the weakly-hard concept with existing AMC policy and schedulability analysis
  - Allows s out of m LO-criticality jobs to be skipped in HI-criticality mode to reduce the load on the system
  - Still provides a level of service to LO-criticality applications, since $(\mathrm{m} - s)$ out of m deadlines are met
  - Gives system designer flexibility to provide graceful degradation for LO-criticality applications

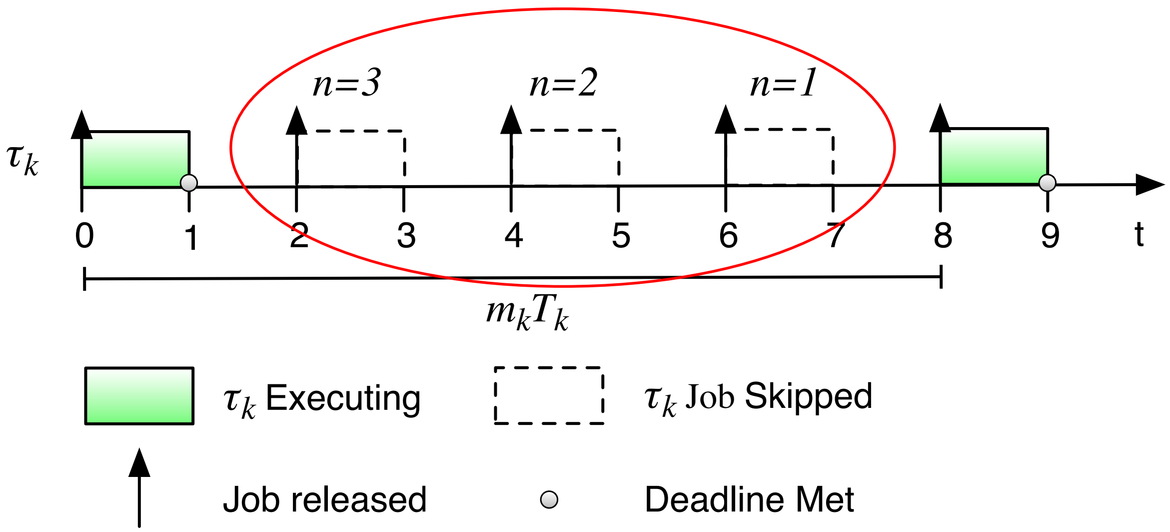
AMC-Weakly Hard



- After criticality mode change:
  - Skip s jobs in next m releases
  - Repeat this cycle indefinitely in HI-criticality mode
  - Number of skipped jobs is strictly bounded $(\mathrm{m} - s)$ out of m deadlines met

THE UNIVERSITY of York

AMCrtb-WH Analysis



$$\left(\left\lceil\frac{t}{T_k}\right\rceil - \sum_{n=1}^{s_k}\left\lceil\frac{t-(m_k-n)T_k}{m_k T_k}\right\rceil\right) C_k$$

$$\tau_i = (T_i, D_i, \overrightarrow{C_i}, L_i, s_i, m_i)$$

m is length of a cycle

s is number of skipped jobs in a cycle

n is index of a skipped job

AMCrtb-WH Analysis

LO Criticality Mode

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

Worst case assumes skips are at the end of each cycle

HI Criticality Mode

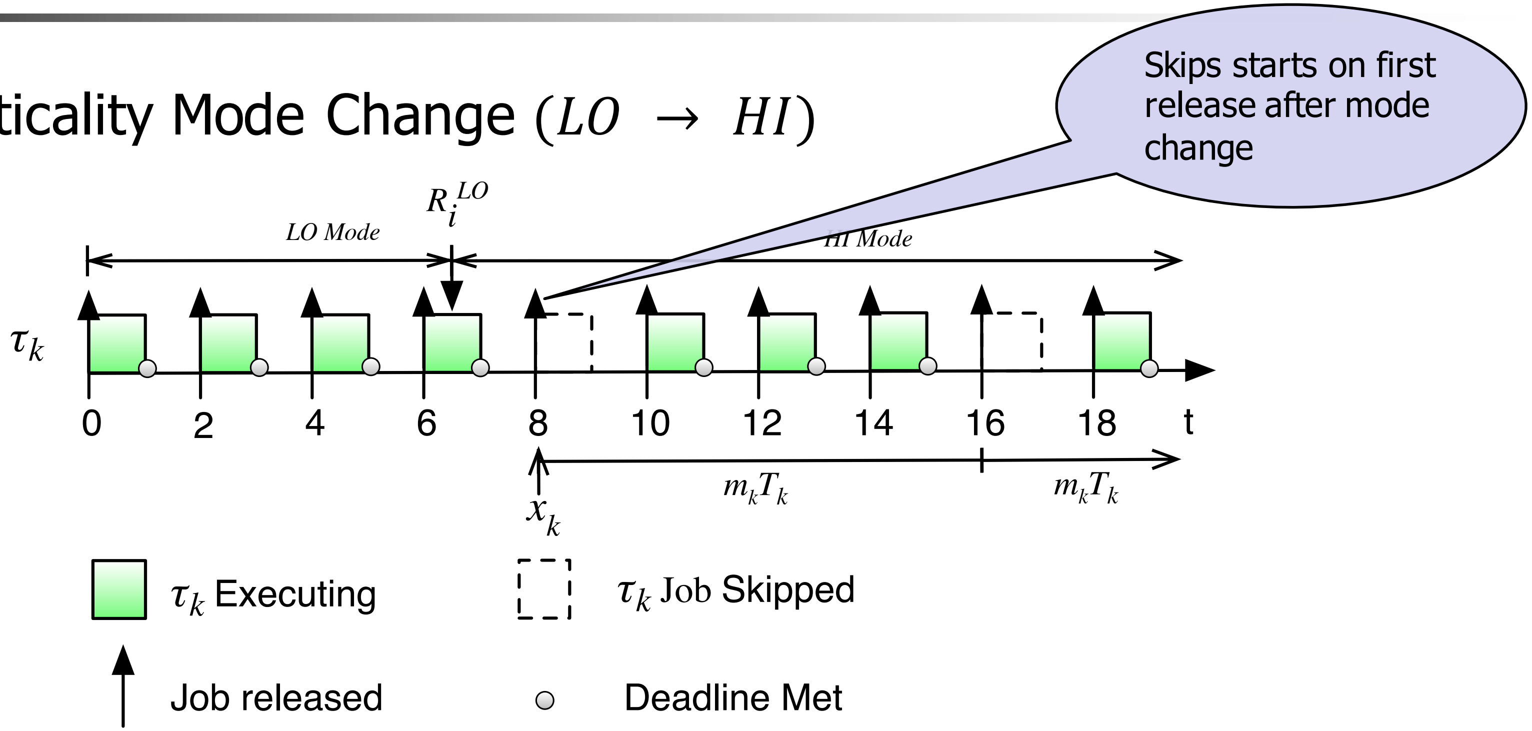
$$R_i^{HI} = C_i^{L_i} + \sum_{j \in \boldsymbol{hpHI}(i)} \left\lceil \frac{R_i^{HI}}{T_j} \right\rceil C_j^{HI} + \sum_{k \in \boldsymbol{hpLO}(i)} \left(\left\lceil \frac{R_i^{HI}}{T_k} \right\rceil - \sum_{n=1}^{s_k} \left\lceil \frac{R_i^{HI} - (m_k - n)T_k}{m_k T_k} \right\rceil_0 \right) C_k^{LO}$$

RTS York

THE UNIVERSITY of York

AMCrtb-WH Analysis

Criticality Mode Change $(LO \rightarrow HI)$



First release of job after Criticality Mode Change $x_k = \left\lceil \frac{R_i^{LO}}{T_k} \right\rceil T_k$

AMCrtb-WH Analysis

Criticality Mode Change $(LO \rightarrow HI)$: HI Criticality Tasks

$$R_i^* = C_i^{HI} + \sum_{j \in \boldsymbol{hpHI}(i)} \left\lceil \frac{R_i^*}{T_j} \right\rceil C_j^{HI} + \sum_{k \in \boldsymbol{hpLO}(i)} \left(\left\lceil \frac{R_i^*}{T_k} \right\rceil - \sum_{n=s_k}^{m_k} \left\lceil \frac{R_i^* - (m_k - n)T_k - x_k}{m_k T_k} \right\rceil_0 \right) C_k^{LO}$$

Assumes skips are at the start of each cycle

Criticality Mode Change $(LO \rightarrow HI)$: LO Criticality Tasks

$$R_i^* = C_i^{LO} + \sum_{j \in \boldsymbol{hpHI}(i)} \left\lceil \frac{R_i^*}{T_j} \right\rceil C_j^{HI} + \sum_{k \in \boldsymbol{hpLO}(i)} \left\lceil \frac{R_i^*}{T_k} \right\rceil C_k^{LO}$$

No skipping assumed for higher priority LO-criticality task.

AMCmax-WH Analysis

- AMCrtb-WH criticality mode change analysis is **pessimistic**
  - Analysing HI-criticality: Assumes all HI-criticality jobs up to R^* execute with their C^{HI} values

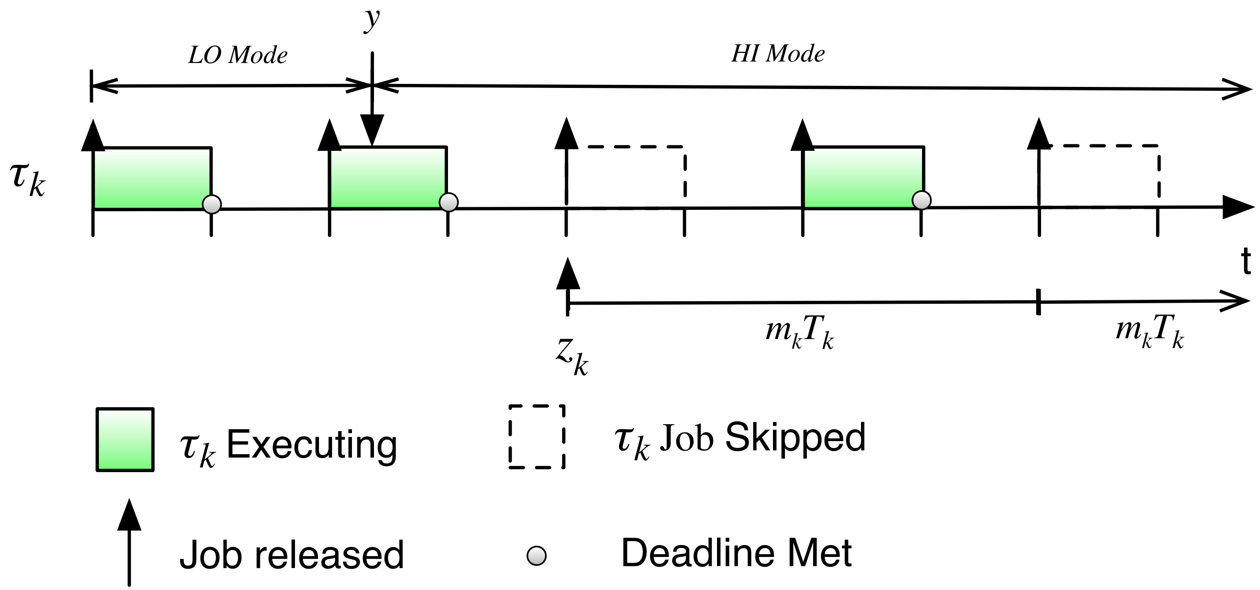
  AND
  - Analysing LO-criticality: Assumes no skipping of LO-criticality jobs up to R^*.

  - AMCmax-WH analysis remove these sources of pessimism by taking into account the points at which a criticality mode change could occur

  - Analysis for LO- and HI-criticality modes is same as AMCrtb-WH

AMCmax-WH Analysis

Criticality Mode Change $(LO \rightarrow HI)$ at time y



First release of job after Criticality Mode Change $z_k = \left\lceil \frac{y}{T_k} \right\rceil T_k$

AMCmax-WH Analysis

Criticality Mode Change $(LO \rightarrow HI)$: All Tasks

$$R_i^y = C_i^{L_i} + \sum_{k \in \boldsymbol{hpLO}(i)} \left(\left\lceil \frac{R_i^y}{T_k} \right\rceil - \sum_{n=s_k}^{m_k} \left\lceil \frac{R_i^y - (m_k - n)T_k - z_k}{m_k T_k} \right\rceil_0 \right) C_k^{LO}$$

$$+ \sum_{j \in \boldsymbol{hpHI}(i)} \left(M(j, y, R_i^y) C_j^{HI} + \left(\left\lceil \frac{R_i^y}{T_j} \right\rceil - M(j, y, R_i^y) \right) C_j^{LO} \right)$$

Jobs of LO-criticality task k skipped after the criticality mode change at time y

Jobs of HI-criticality task k only take C^{HI} values after the criticality mode change at time y

$$R_i^* = \max(R_i^y) \; \forall_y \text{where } y \in kT_j \mid \forall j \in hpLO(i) \wedge y \leq R_i^{LO} \mid \forall k : \mathbb{N}$$

- For HI-criticality tasks, y checked for values up to R^{LO}
- For LO-criticality tasks y is increased until R^* converges below the current value of y

Evaluation

- Compared existing policies:
  - **UB-H&L** - Composite upper-bound on schedulability
  - **AMC-max** – Baruah et al. 2011 [3]
  - **AMC-rtb** - Baruah et al. [3]
  - **SMC** – SMC-NO with budget enforced execution for *LO*-criticality tasks [3]
  - **SMC-NO** - Vestal's original analysis [29]
  - **AMCmax-WH** - Weakly-Hard version of AMC-max
  - **AMCrtb-WH** - Weakly-Hard version of AMC-rtb
  - **FPPS** – Fixed priority preemptive scheduling with run-time monitoring to prevent *LO*-criticality tasks overrunning
  - **CrMPO** – Criticality Monotonic Priority Ordering. Tasks ordered by criticality then by DMPO within the two partitions

Evaluation

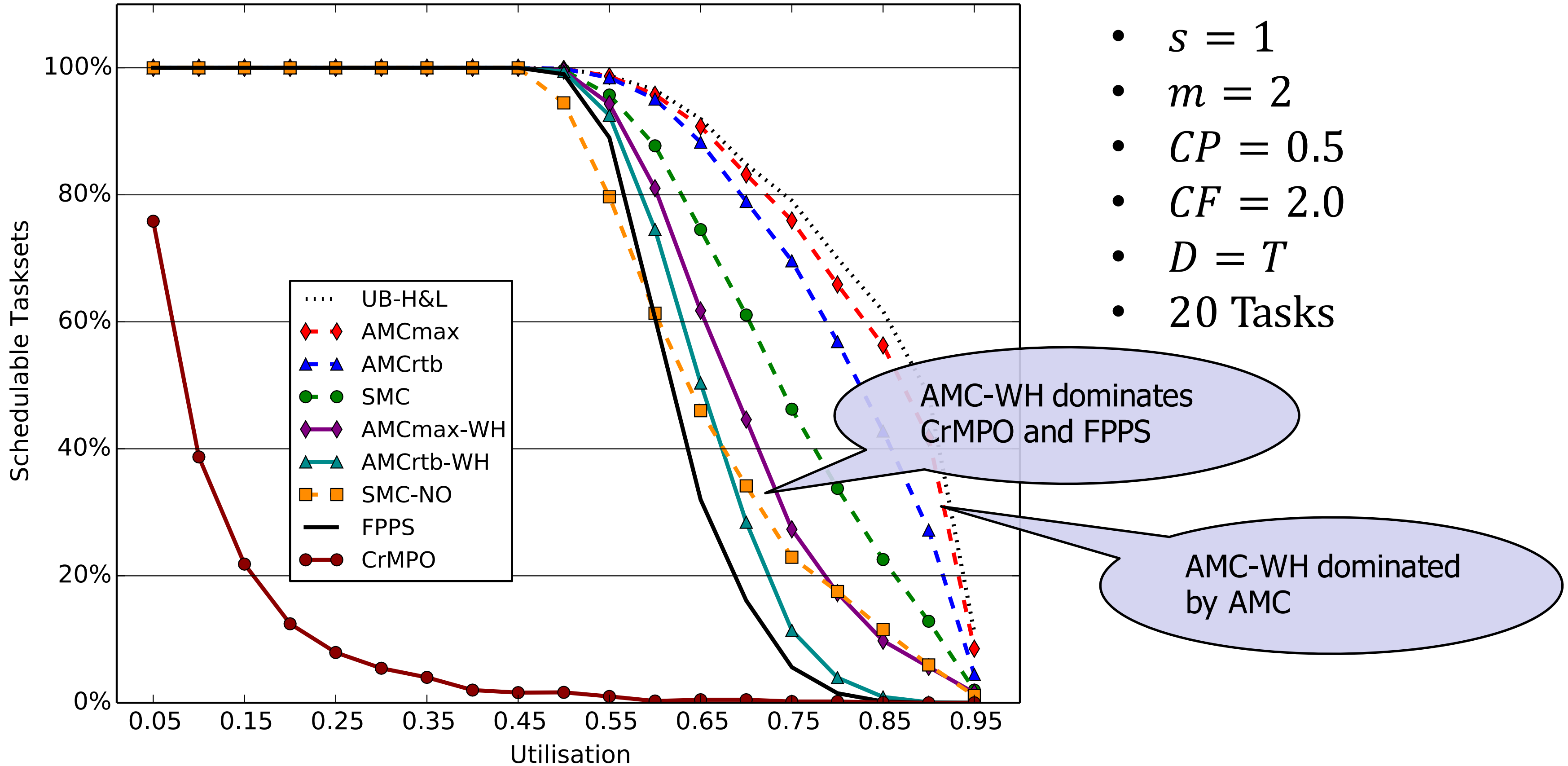
- Taskset generation:
  - Uniformly distributed utilisation values generated with UUnifast
  - T randomly assigned from a Log uniform distribution between 10 and 1000
  - $C_i^{LO} = U_i / T_i$
  - Criticality Factor (CF)
  - $C_i^{HI} = C_i^{LO} * CF$
  - Criticality Probability (CP) - probability that a task will be HI-criticality

- Notes about graphs
  - Plotted against LO-criticality utilisation
  - Solid lines represent policies that guarantee some LO-criticality task deadlines are met in HI-criticality mode.
  - Dashed lines represent polices that de-schedule or permit deadline misses of LO-criticality tasks in HI criticality mode.

RTS York

THE UNIVERSITY of York

1: Percentage of Schedulable Tasksets



- $s = 1$
- $m = 2$
- $CP = 0.5$
- $CF = 2.0$
- $D = T$
- 20 Tasks

RTS York

THE UNIVERSITY of York

Weighted Schedulability

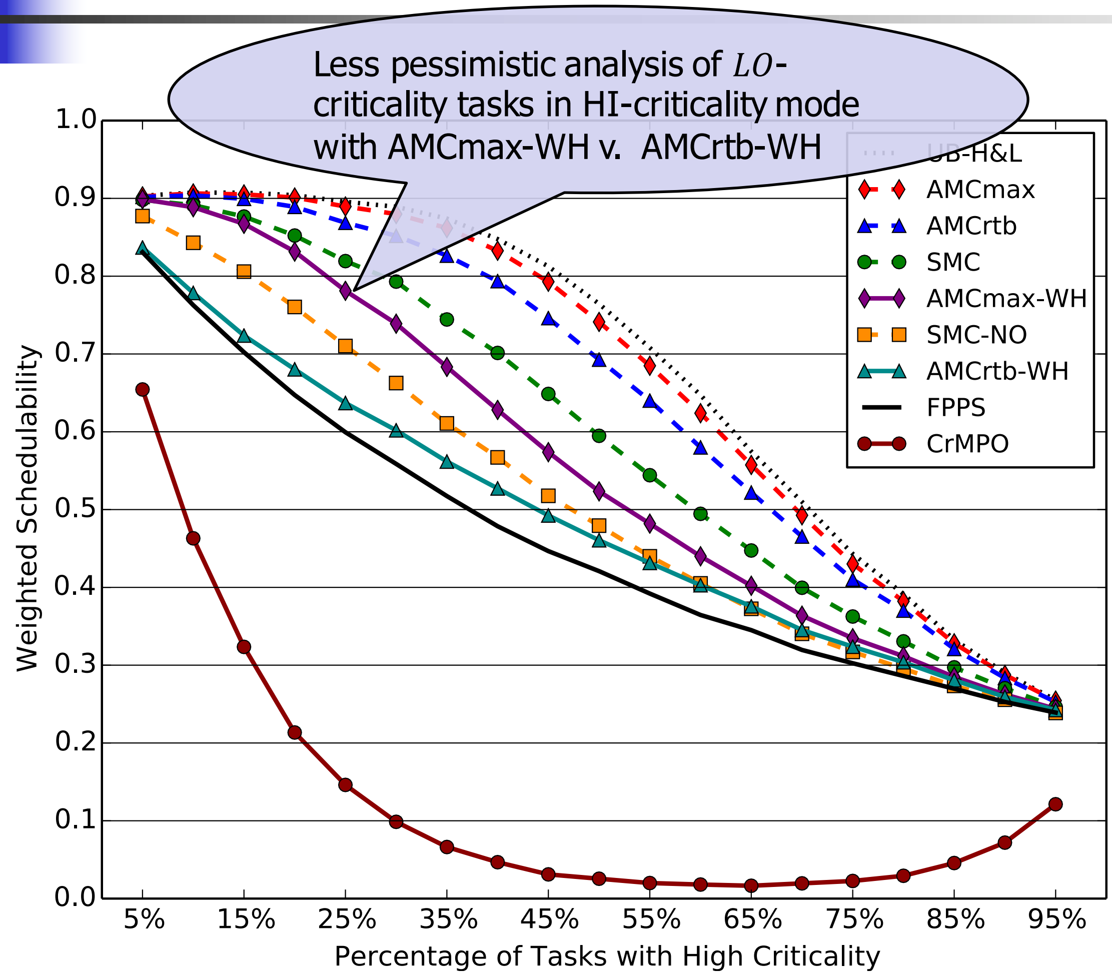
- Weighted Schedulability
  - Enables overall comparisons when varying a specific parameter (not just utilisation)
  - Combines results form of a set of equally spaced utilisation levels

$$W_{\phi}(p) = \frac{\sum_{\forall \tau} U(\tau) * S_{\phi} \ (\tau, P)}{\sum_{\forall \tau} U(\tau)}$$

  - Collapses all data on a success ratio plot for a given method, into a single point on a weighted schedulability graph

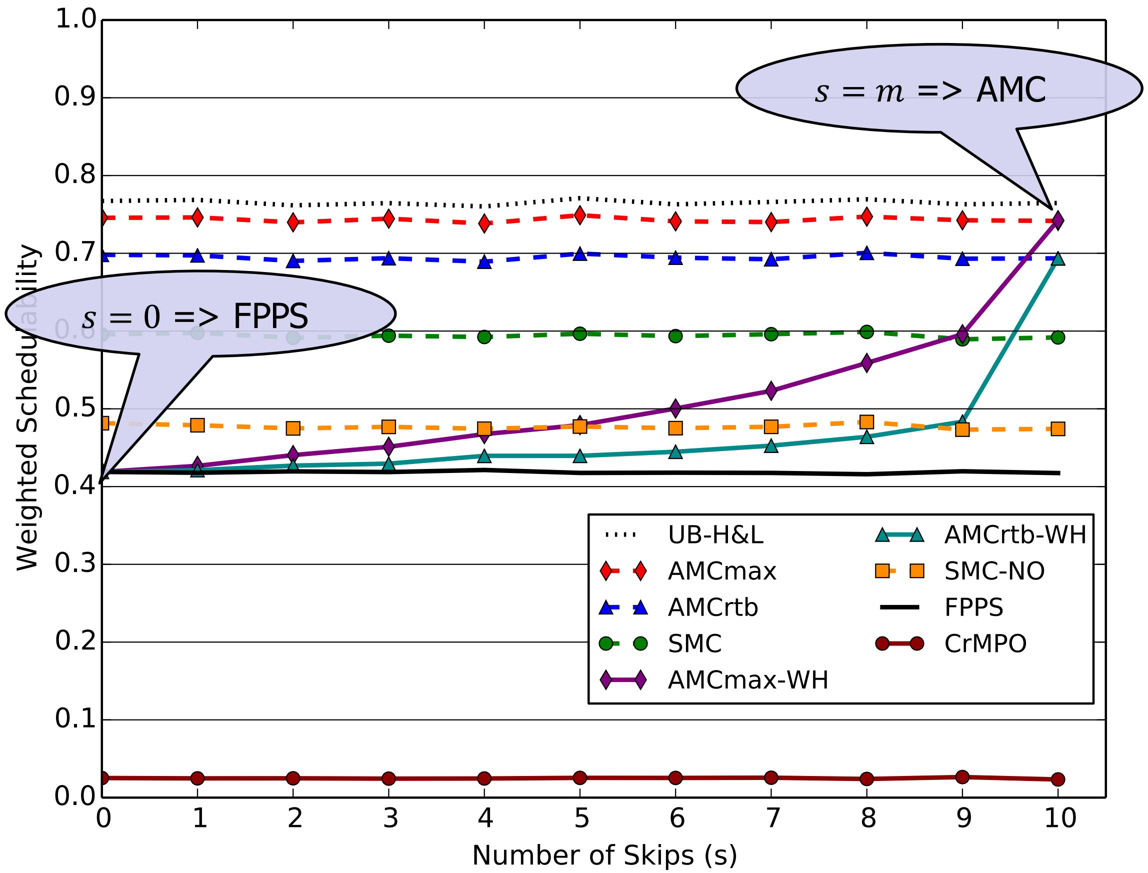
Weighted schedulability is effectively a weighted version of the area under a success ratio curve biased towards scheduling higher utilisation message sets

2: Varying the Criticality Mix



- $s = 1$
- $m = 2$
- $CP = 0.05\ to\ 0.95$
- $CF = 2.0$
- $D = T$
- 20 Tasks

3: Varying the Number of Skips (fixed cycle)



- $s = 0\ to\ 10$
- $m = 10$
- $CP = 0.5$
- $CF = 2.0$
- $D = T$
- 20 Tasks

RTS York

THE UNIVERSITY of York

Summary and Conclusions

- AMC-WH
  - Combines AMC protocol, with a simple interpretation of Weakly Hard constraints
  - Provides guaranteed minimum Quality of Service (QoS) for LO-criticality tasks HI-criticality mode, meet $(m - s)$ out of m deadlines
  - Performance scales between AMC and FPPS
- Schedulability tests developed based on AMC-rtb and AMC-max.
- Scope for future work:
  - Permit weakly-hard behaviour in any criticality mode, where each task is assigned a set of weakly hard constraints per criticality level
  - Investigate recovery to LO-criticality mode

Questions?