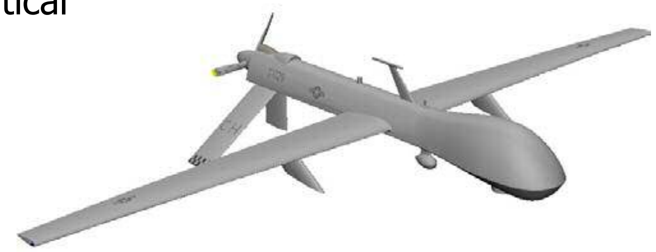
Adaptive Mixed Criticality Scheduling with Deferred Pre-emption

Alan Burns and Robert Davis

Real-Time Systems Research Group, University of York

Mixed Criticality Systems

- MCS
  - Applications of different criticality levels on the same HW platform
    - E.g. Safety Critical, Mission Critical, Non-critical
  - Driven by SWaP and cost requirements



- Examples
  - Aerospace: e.g. UAVs
    - Flight Control Systems v. Surveillance
  - Automotive:
    - Electronic Power Assisted Steering v. Cruise Control

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

Mixed Criticality Systems

- Key requirements
  - *Separation* – must ensure 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)
  - System starts in LO-criticality mode
  - 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
- Scheduling based on fixed priorities
- Sporadic task sets
  - T_i – Period or minimum inter-arrival time (sporadic behaviour)
  - D_i – Relative deadline
  - L_i – Criticality level (LO or HI)
  - HI-criticality tasks have both $C_i(HI)$ and $C_i(LO)$ worst-case execution time estimates with $C_i(HI) > C_i(LO)$
  - LO-criticality tasks need only have $C_i(LO)$

Adaptive Mixed Criticality (AMC)

- 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

- Analysis of AMC
  1. Check all tasks are schedulable in LO-criticality mode
  2. Check HI-criticality tasks are schedulable in HI-criticality mode
  3. Check HI-criticality tasks are schedulable over the mode change

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

Analysis for AMC

- LO-criticality mode

$$R_i(LO) = C_i(LO) + \sum_{\forall j \in hpH(i)} \left\lceil \frac{R_i(LO)}{T_j} \right\rceil C_j(LO) + \sum_{\forall k \in hpL(i)} \left\lceil \frac{R_i(LO)}{T_k} \right\rceil C_k(LO)$$

- HI-criticality mode ………… and mode transition

$$R_i(HI) = C_i(HI) + \sum_{\forall j \in hpH(i)} \left\lceil \frac{R_i(HI)}{T_j} \right\rceil C_j(HI) + \sum_{\forall k \in hpL(i)} \left\lceil \frac{R_i(LO)}{T_k} \right\rceil C_k(LO)$$

Interference from higher priority LO-criticality tasks up to $R(LO)$

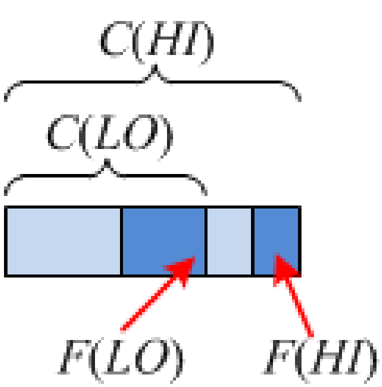
How to improve upon AMC?

- Focus on unwanted interference from LO-criticality tasks in HI-criticality mode

$$R_i(HI) = C_i(HI) + \sum_{\forall j \in hpH(i)} \left\lceil \frac{R_i(HI)}{T_j} \right\rceil C_j(HI) + \sum_{\forall k \in hpL(i)} \left\lceil \frac{R_i(LO)}{T_k} \right\rceil C_k(LO)$$

How to reduce this?

- Final non-pre-emptive regions
  - A non-pre-emptive region $F(LO)$ at the end of $C(LO)$ can reduce $R(LO)$
  - No interference due to LO-criticality jobs released during non-pre-emptive region $F(LO)$ as they cannot start prior to HI-criticality mode
  - Trade-off is blocking higher priority tasks by $F(LO)$
  - Non-pre-emptive region $F(HI)$ at the end of $C(HI)$ comes for free if $F(HI) \leq F(LO)$

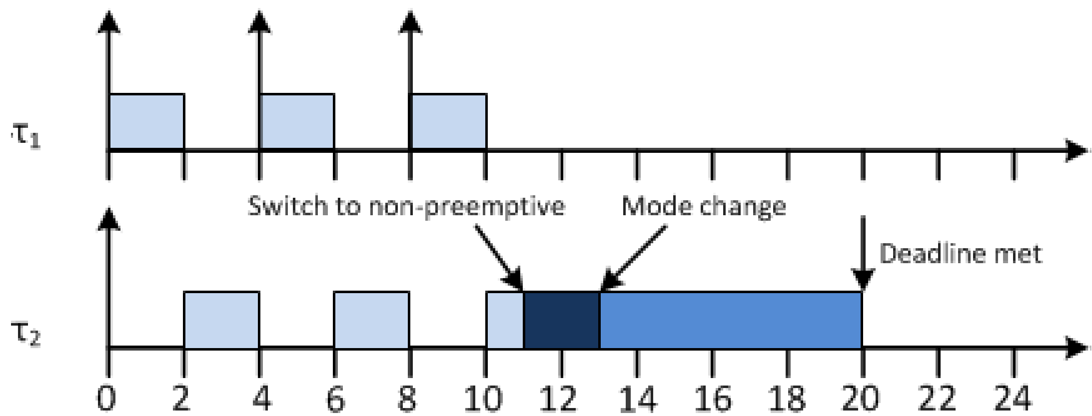


Example of AMC and AMC-NPR

- Task set

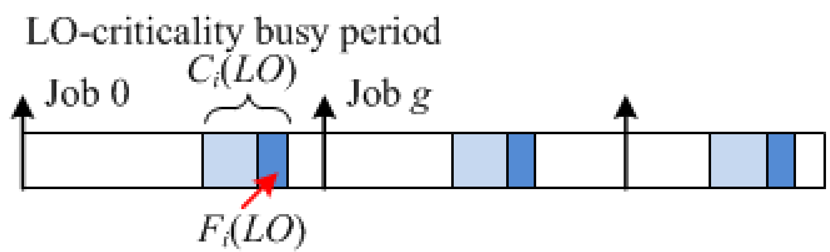
Task	L	$C(LO)$	$C(HI)$	$D = T$
τ_1	LO	2	-	4
τ_2	HI	7	14	20

- AMC-NPR



Analysis for AMC-NPR

- For LO-criticality behaviour
  - With final non-pre-emptive regions need to examine all jobs g in the longest busy period for task τ_i (due to push-through blocking effects)



  - Start of final non-pre-emptive region of job g w.r.t. start of busy period

$$R^s_{i,g}(LO) = B_i(LO) + (g+1)C_i(LO) - F_i(LO) + \sum_{\tau_j \in hp(i)} \left(\left\lfloor \frac{R^s_{i,g}(LO)}{T_j} \right\rfloor + 1 \right) C_j(LO)$$

  - Blocking

$$B_i(LO) = \max_{\tau_j \in lp(i)} (F_i(LO) - 1)$$

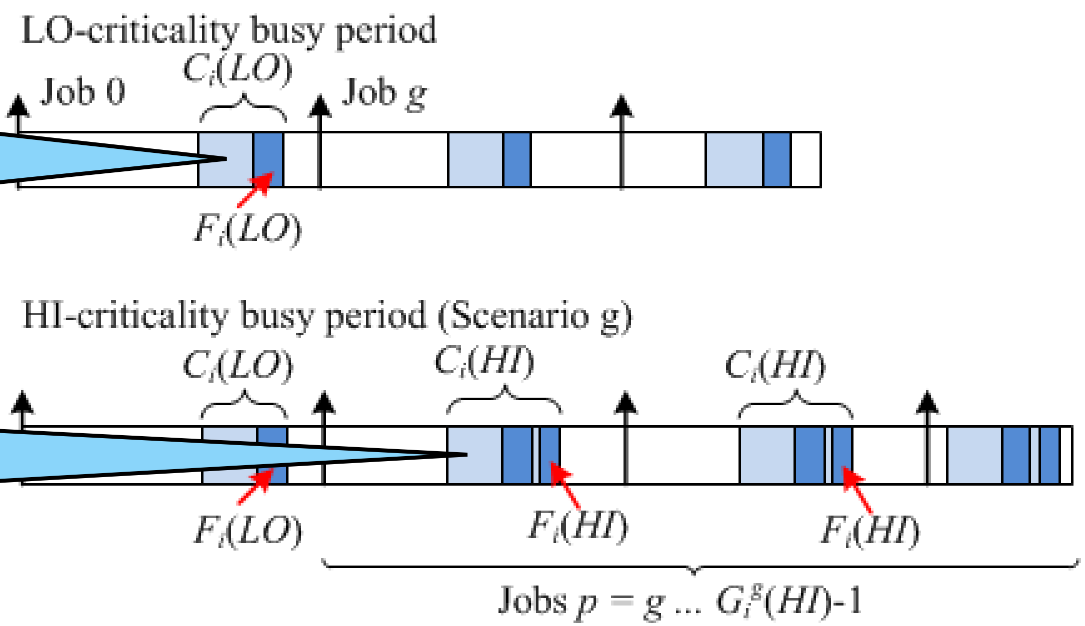
  - Response time of task τ_i

$$R_i(LO) = \max_{g=0 \ldots G_i(LO)-1} (R^s_{i,g}(LO) + F_i(LO) - gT_i)$$

Analysis for AMC-NPR

- For HI-criticality behaviour
  - Need to consider that HI-criticality behaviour could start with any of the jobs g in the LO-criticality busy period and then continue for a number of jobs p in HI-criticality mode

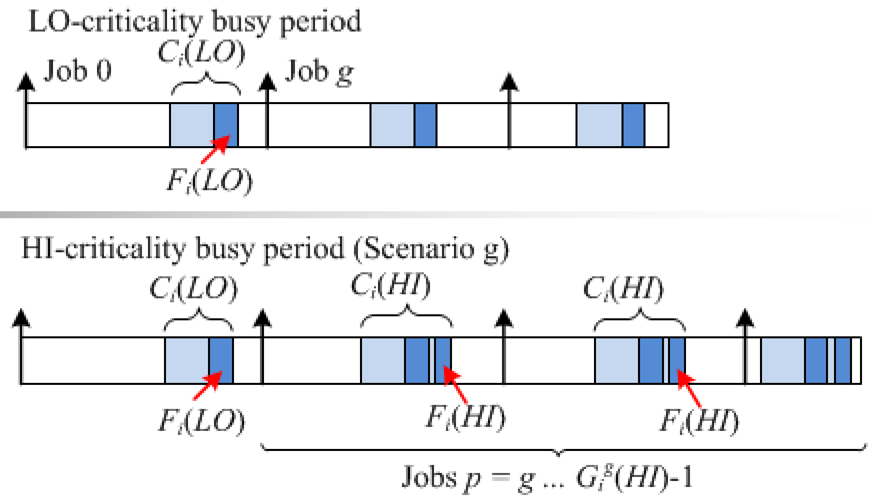
Outer iteration:
For all jobs g in LO-criticality busy period assume g is the first to exhibit HI-criticality behaviour



Inner iteration:
For all jobs p from g onwards check response time.
Assuming these jobs have HI-criticality behaviour

Analysis for AMC-NPR

- HI-criticality behaviour



  - Start of HI-criticality non-pre-emptive region $F(HI)$ of job p w.r.t. start of busy period for the scenario where gth job is the first to have HI-criticality behaviour

g jobs of this task with LO-criticality execution

p+1-g jobs of this task with HI-criticality execution

$$R^s_{i,g,p}(HI) = B_i + gC_i(LO) + (p+1-g)C_i(HI) - F_i(HI)$$
$$+ \sum_{\tau_j \in hpH(i)} \left(\left\lfloor \frac{R^s_{i,g,p}(HI)}{T_j} \right\rfloor + 1 \right) C_j(HI) + \sum_{\tau_k \in hpL(i)} \left\lceil \frac{R^s_{i,g}(LO)}{T_k} \right\rceil C_k(LO)$$

LO-criticality execution only as far as start of FNR of job g

  - Blocking $B_i = B_i(LO)$ since $F_i(LO) \geq F_i(HI)$
  - Response time of task τ_i

$$R_i(HI) = \max_{g=0 \ldots G_i(LO)-1} \left(\max_{p=g \ldots G_i^g(HI)-1} \left(R^s_{i,g,p}(HI) + F_i(HI) - pT_i \right) \right)$$

  - See paper for details

Priority & NPR length assignment

- Optimal Final Non-pre-emptive region length and Priority Assignment
  - For this problem an assignment algorithm is optimal if it finds a schedulable set of final non-pre-emptive region lengths and task priorities whenever such an assignment exists

- An Optimal assignment for AMC-NPR?
  - Assume the restriction that $F(HI) \leq F(LO)$
  - Provided that $F(HI) \leq F(LO)$ blocking is unaffected by $F(HI)$ hence making $F(HI)$ as large as possible subject to constraints gives the best schedulability
  - Hence can set $F(HI) = \max(C(HI)\text{-C(LO)}, \text{F(LO)})$
  - Now only need to determine $F(LO)$ and priorities

Priority & NPR length assignment

- Weakly Optimal Final Non-pre-emptive Region length and Priority Assignment Algorithm

```
for each priority level i, lowest first {
    for each unassigned task τ {
          determine the minimum value of F (if any) that makes the
          task schedulable at priority i with F(LO) = min(F, C(LO))
          and F(HI) = min(F(LO), C(HI) – C(LO)) assuming that all
          unassigned tasks have higher priorities
    }
    if no tasks are schedulable at priority i {
          return unschedulable
    }
    else {
          assign the schedulable task with the minimum value of F at
          priority i to priority i. Assume the values of F(LO) and
          F(HI) for that task
    }
}
return schedulable
```

- Variation on Davis and Bertogna's algorithm for the FNR-PA problem (RTSS 2012) which is based on Audsley's optimal priority assignment algorithm

Evaluation

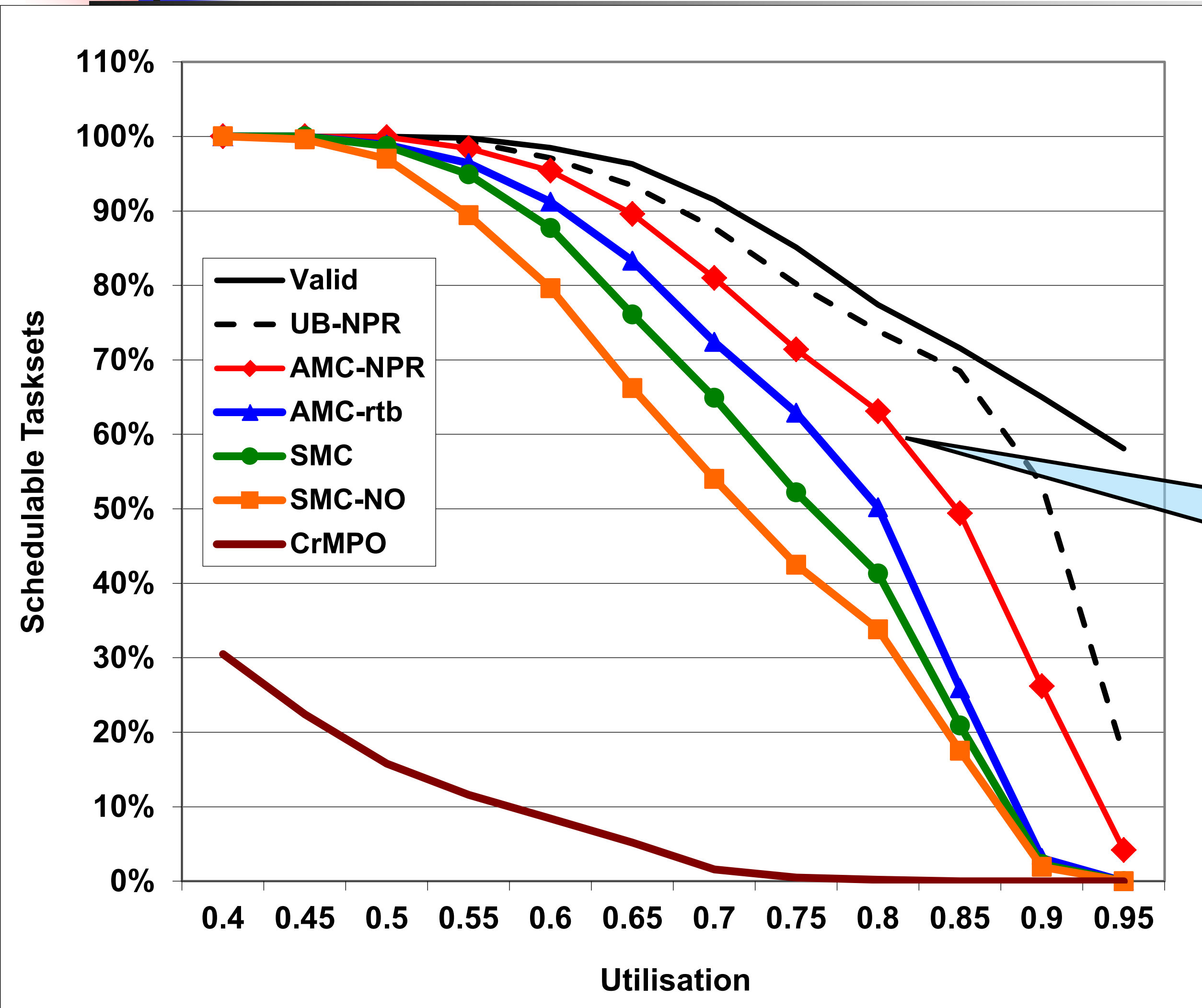
- Compared the following schemes:
  - **CrMPO**: Criticality monotonic priority assignment – all HI-criticality tasks have higher priorities than LO-criticality tasks (with Deadline Monotonic Priority Order used within the subsets)
  - **SMC-NO**: Vestal's original scheme [31]
  - **SMC**: Vestal's scheme with budget enforcement at C(LO) for LO-criticality tasks [3]
  - **AMC-rtb**: Adaptive Mixed Criticality scheduling [5]
  - **AMC-NPR**: The scheme described in this talk
  - **UB-NPR**: A composite upper bound obtained using the FNR-PA algorithm to *independently* check schedulability in LO- and HI-criticality modes ignoring the mode change itself. UB-NPR is a necessary condition rather than a schedulability test
  - **VALID**: Task sets with total HI-criticality utilisation ≤ 1 (and total LO-criticality utilisation ≤ 1)

Evaluation

- Task set generation:
  - Number of tasks (Default n = 20)
  - Periods followed a Log-uniform distribution (Default 10ms – 100ms)
  - Implicit Deadlines
  - Utilisation values U_i generated using Uunifast
  - LO-criticality execution times set via $C_i(LO) = U_i\, T_i$
  - HI-criticality execution times $C(HI) = CF.\, C(LO)$ where CF is the criticality factor (Default CF = 2.0)
  - Probability CP of a task being HI-criticality (Default CP = 0.5)

- Note about graphs
  - Plotted against total LO-criticality utilisation
  - **VALID** line is needed to show when a proportion of the generated task sets have a total HI-criticality utilisation > 1 and could not possibly be schedulable

Success ratio



20 tasks
CP = 0.5
CF = 2.0

Significant improvement over standard AMC

Weighted schedulability

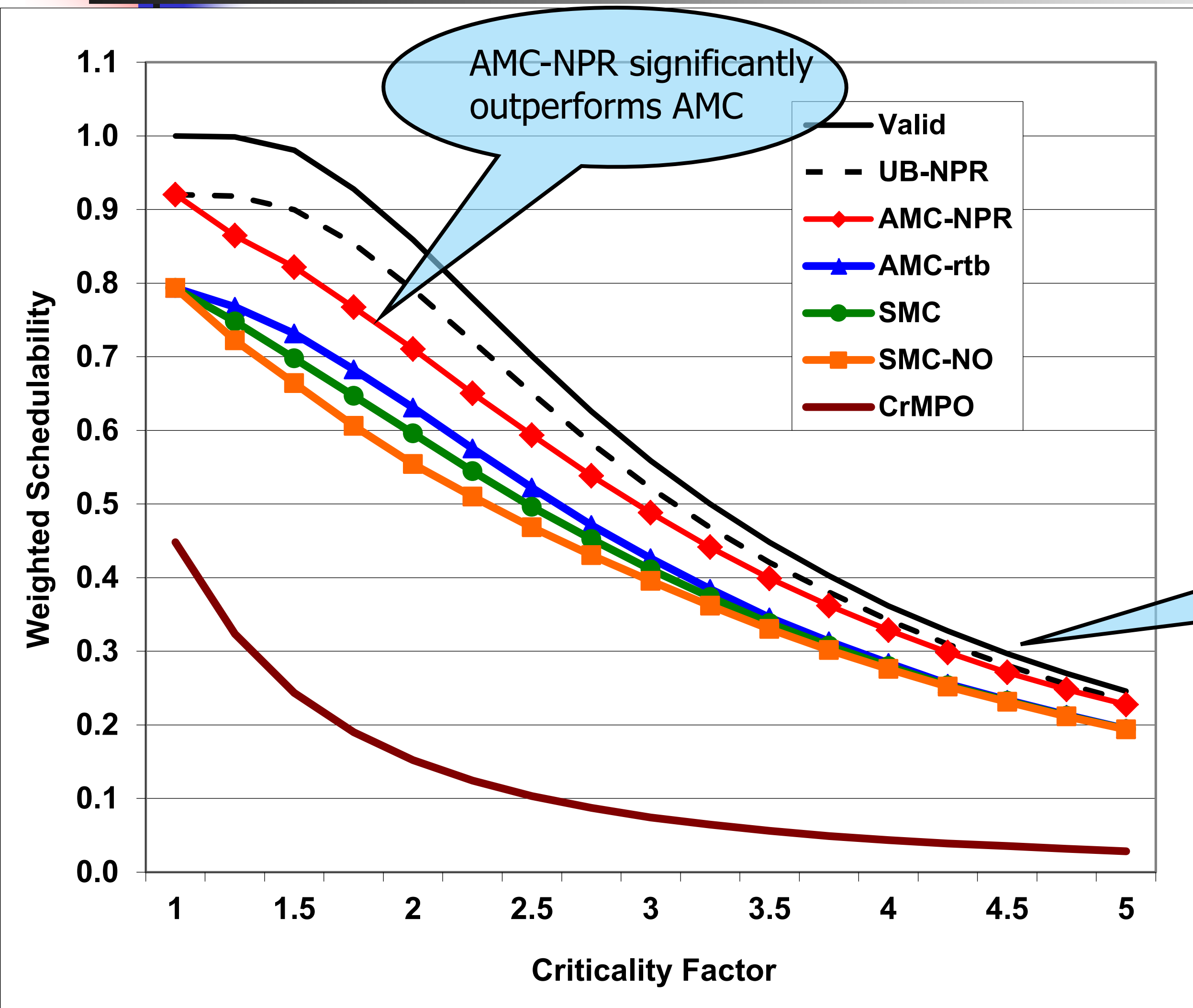
- Weighted schedulability
  - Enables overall comparisons when varying a specific parameter (not just utilisation)
  - Combines results from all of a set of equally spaced utilisation levels
  - Weighted schedulability:

$$Z_y(p) = \frac{\sum_{\forall \tau} S_y(\tau).U(\tau)}{\sum_{\forall \tau} U(\tau)}$$

  - 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

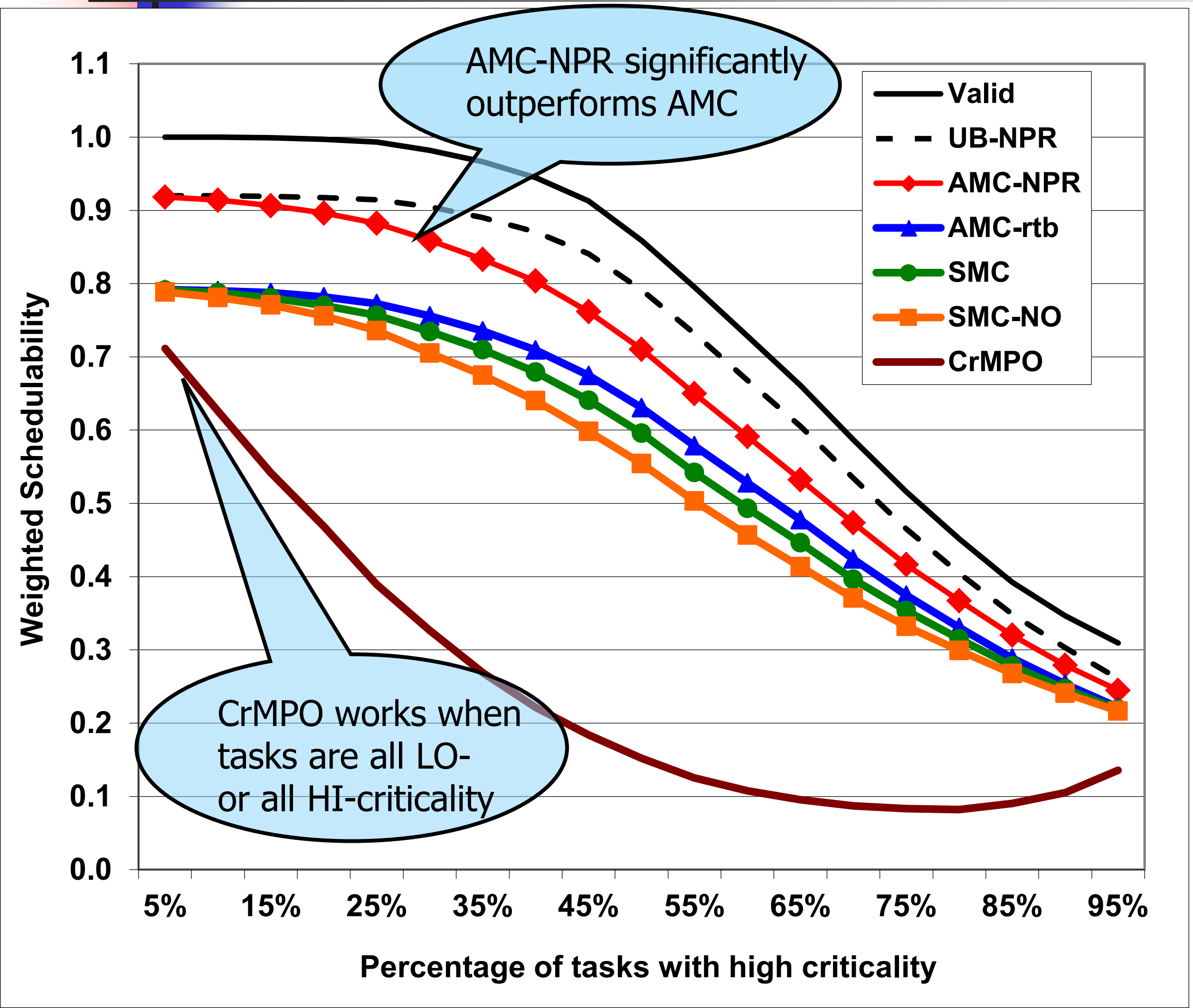
Weighted schedulability: Criticality Factor



Varying Criticality Factor and hence ratio of $C(HI)$ to $C(LO)$

20 tasks, $D = T$
$CP = 0.5$

Weighted schedulability: Percentage of HI-criticality tasks



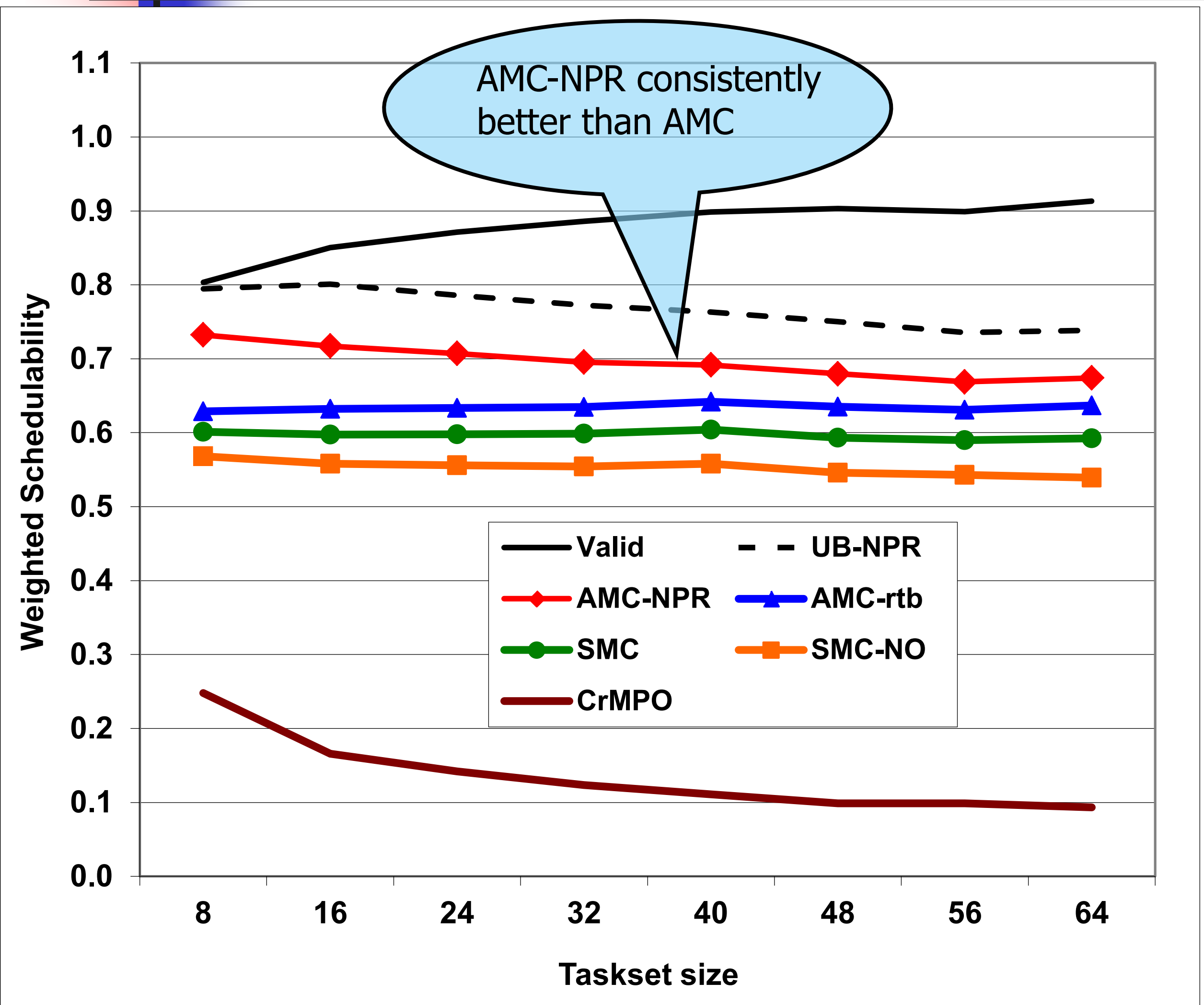
Varying Criticality Percentage (CP)

20 tasks

$D = T$

$CF = 2.0$

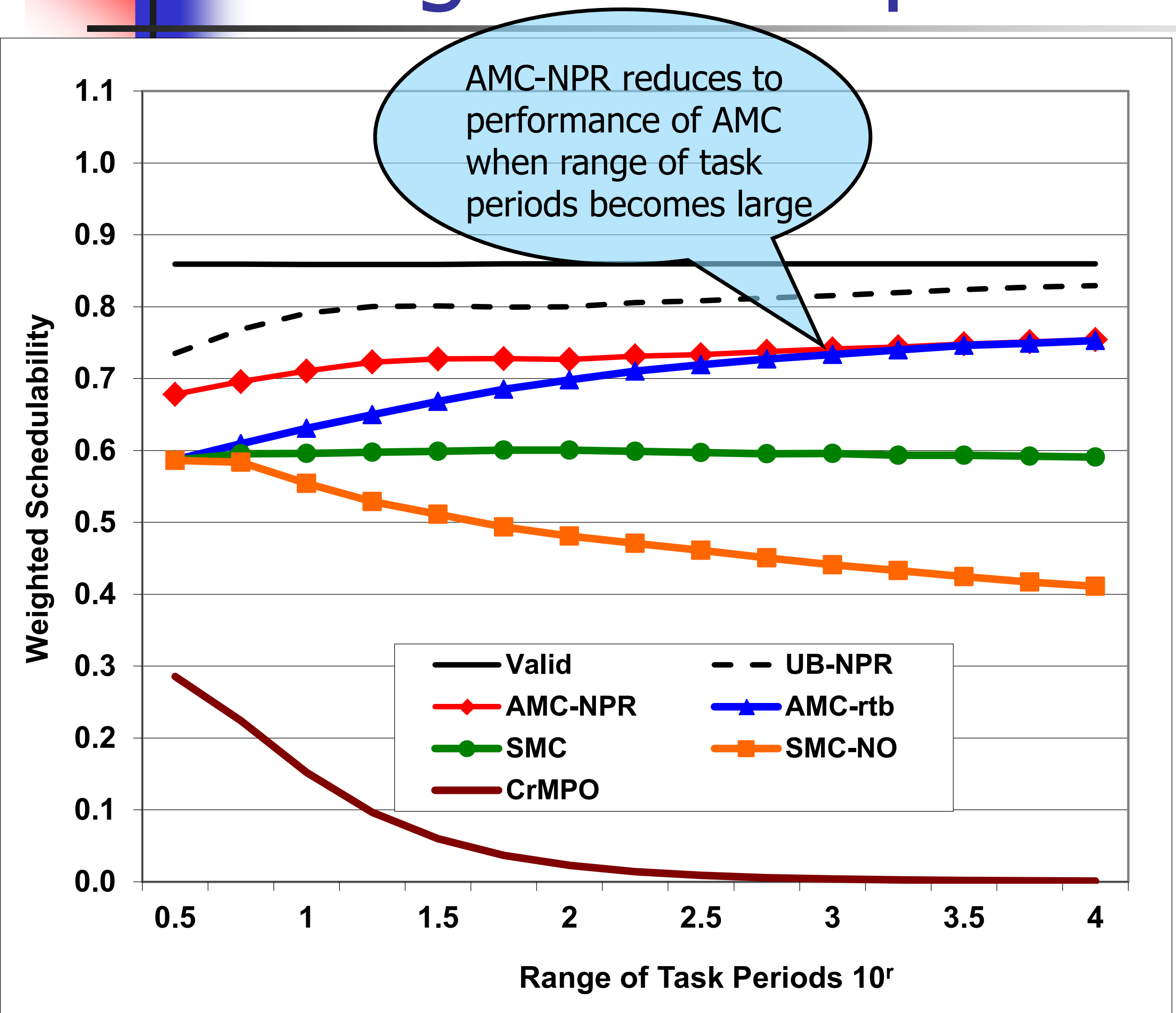
Weighted schedulability: Task set size



Varying task set size

$D = T$
$CF = 2.0$
$CP = 0.5$

Weighted schedulability: Range of task periods



Varying range of task periods (by orders of magnitude)

$D = T$
$CF = 2.0$
$CP = 0.5$

Summary and Conclusions

- Main contributions
  - Integration of research on final non-pre-emptive regions to improve schedulability in mixed criticality systems
  - Developed AMC-NPR scheme which dominates AMC
  - Evaluation shows a useful improvement in schedulability across a wide range of parameters

Questions?