Semi-partitioned Model on Dual-core Mixed Criticality System

Hao Xu, Alan Burns

RTS York

The University of York

Background

- Mixed Criticality System
  - Task has different Worst Case Execution Time (C) for different criticality levels (L)
  - For the same task:

    $L_m > L_n \Rightarrow C_m > C_n$
- Two criticality levels (HI, LO)
  - HI > LO

Background

- Vestal's Approach[1]
  - All tasks keeps schedulable
  - Pessimistic
- Runtime monitors
  - LO-crit task -> prevent from execution
  - HI-crit task -> mode change
- AMC[2] and EDF-VD[3] etc.

[1] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International*, pages 239–243. IEEE, 2007.
[2] S. K. Baruah, A. Burns, and R. I. Davis. Response-time analysis for mixed criticality systems. In *Real-Time Systems Symposium*, pages 34–43. IEEE, 2011.
[3] S. Baruah, H. Li, and L. Stougie. Towards the design of certifiable mixed-criticality systems. In *Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 13–22. IEEE, 2010.

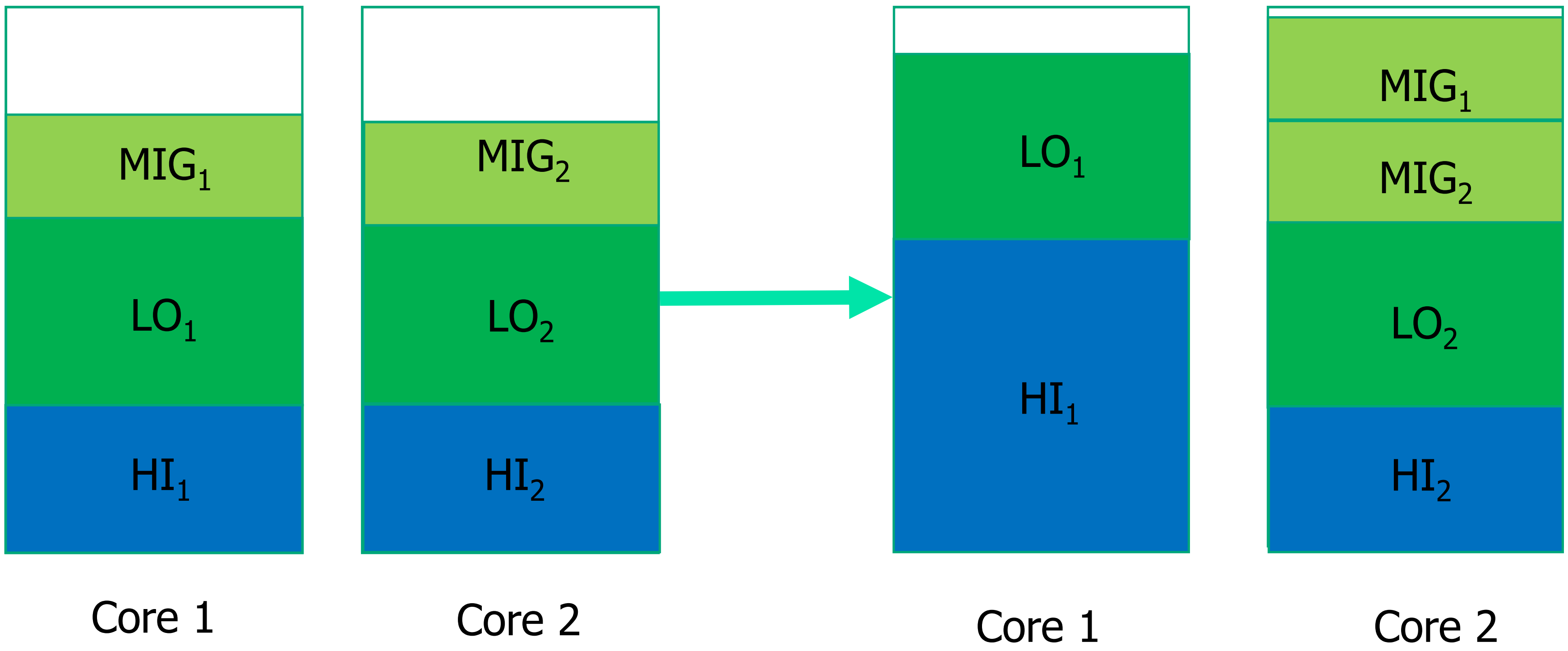
Motivation

- Explore a way that all of the tasks remain schedulable throughout the criticality level changes.
- Multi-core platform provides the option of migration

Assumptions

- Independent tasks
- Deadline < period
- Migration cost
- Mode change frequency and isolation

Semi-partitioned Model



Semi-partitioned Model

- If all tasks execute within their LO-crit budget then all deadlines are met and no tasks migrate.
- No LO-crit task is allowed to exceed its LO-crit budget.
- If HI-crit tasks on one core exceed their LO-crit budget, then some LO-crit tasks will migrate, but ALL LO-crit tasks and HI-crit tasks remain schedulable.
- If HI-crit tasks on more than one core exceed their LO-crit budget, then some LO-crit tasks will be abandoned, but all HI-crit tasks remain schedulable (without migration).

State View

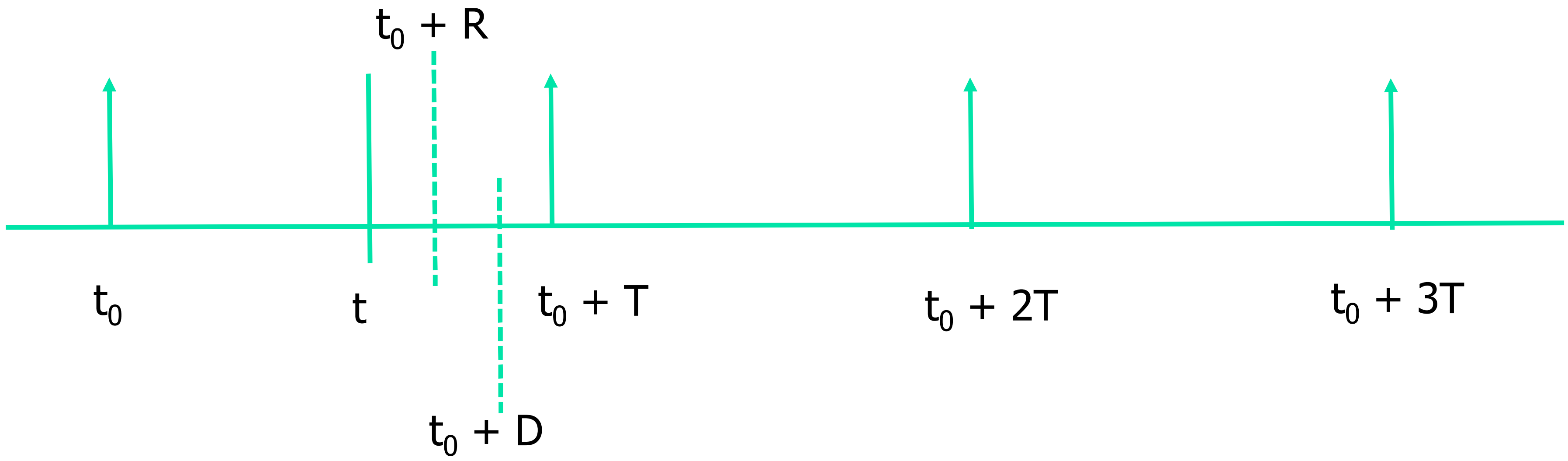
- Task set S to represent the collection of all tasks sets:
  - $S = (LO_1 \cup LO_2) \cup (HI_1 \cup HI_2) \cup (MIG_1 \cup MIG_2)$
- State X to represent the normal execution state:
  - $X_1 = LO_1 \cup HI_1 \cup MIG_1$
  - $X_2 = LO_2 \cup HI_2 \cup MIG_2$
  - $S = X_1 \cup X_2$

One core enters HI-crit Mode

- State *Y(1)* to represent the state that Core c_1 enters HI-crit mode.
  - $Y(1)_1 = LO_1 \cup HI_1$
  - $Y(1)_2 = LO_2 \cup HI_2 \cup MIG_1 \cup MIG_2$
  - $S = Y(1)_1 \cup Y(1)_2$
- Release Jitter and Reduced deadline issues for migrating tasks (MIG_1)

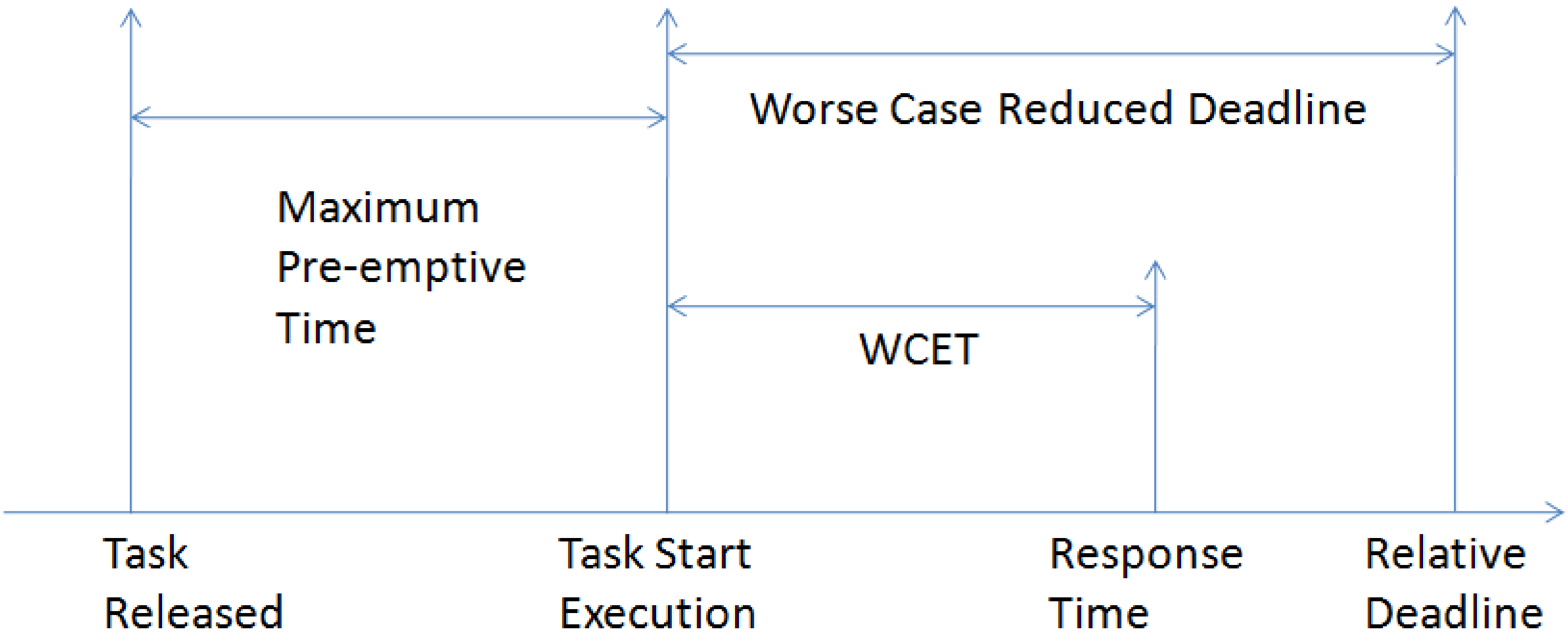
Release Jitter Issue

- Task released at time t_0 and migrates at time t



Reduced Deadline Issue

$$d_i'_{max} = d_i - (R_i(LO) - C_i)$$



Both Cores in HI-crit Mode (1)

- *BY(1)* to represent the case that core c_2 also enters HI-crit mode from state *Y(1)*
  - BY(1)1 = LO_1 U HI_1
  - BY(1)2 = HI_2
  - S = $BY(1)_1$ U $BY(1)_2$ U LO_2 U MIG_1 U MIG_2

Both Cores in HI-crit Mode (2)

- *BX* to represent the case that both cores enter HI-crit mode at the same time.
  - $BX1 = LO_1 \cup HI_1$
  - $BX2 = LO_2 \cup HI_2$
  - $S = BX_1 \cup BX_2 \cup MIG_1 \cup MIG_2$

Allocation

- Bin packing algorithms
  - Criticality aware utilisation decreasing
    g
- Non-migration
- Migration

Choice of Migration

- Regarding to the equation:
  - Migrating LO-crit tasks with largest slack time

$$\forall \tau_i \in (Y(1)_1 \cup Y(2)_2):$$
$$R_i(HI)^* = C_i(L_i) + \sum_{j \in chpH(i)} \left\lceil \frac{R_i(HI)^*}{T_j} \right\rceil C_j(HI) + \sum_{k \in chpL(i)} \left\lceil \frac{R_i(HI)^*}{T_k} \right\rceil C_k(LO) + \sum_{l \in chpMIG(i)} \left\lceil \frac{R_i(LO)}{T_l} \right\rceil C_l(LO)$$

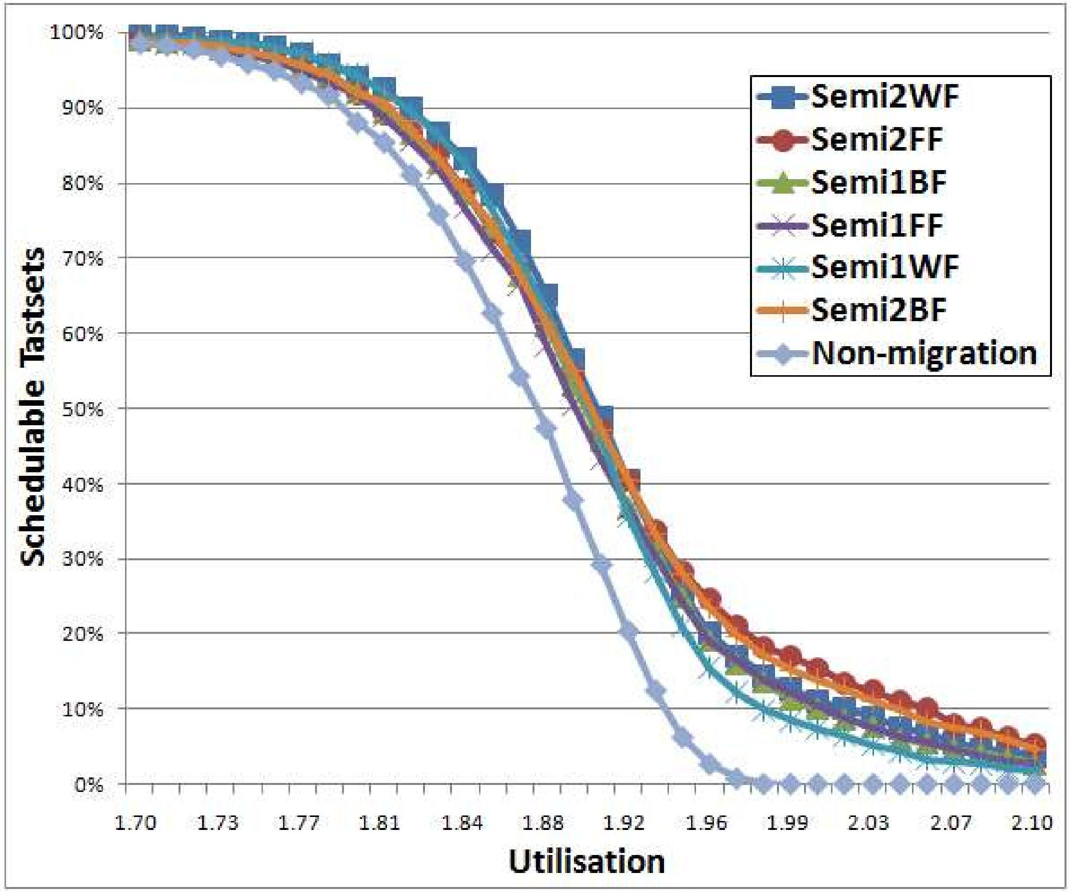
- Schedulability issues after migration

Semi-partitioned Model Configuration

- Two approaches on deciding migrating tasks
  - Semi1: task fetched
  - Semi2: task assigned
- Three bin-packing algorithms
  - FF, BF, WF
- Audeslay's priority assignment

Evaluation

- f = 3
- P = 0.5
- n = 12
- ts=10000



Evaluation

- Weighted graphs
  - Taskset size
  - Percentage of HI-crit tasks
  - Factor of WCET differences
- All algorithms behave better than the non-migration algorithm
- Semi2WF and Semi2FF

Conclusion

- Semi-partitioned model
- Six possible approaches
- Suggestion
- Future Work