Semi-Federated Scheduling of Mixed-Criticality System for Sporadic DAG Tasks

Tao Yang\textsuperscript{1}, Yue Tang\textsuperscript{2}, Xu Jiang\textsuperscript{2}, Qingxu Deng\textsuperscript{1}, Nan Guan\textsuperscript{2}

\textsuperscript{1}Northeastern University, China, \textsuperscript{2}The Hong Kong Polytechnic University, Hong Kong
Outline

- Background & Contributions
- Preliminaries
- Semi-Federated Mixed-Criticality Algorithm
- Evaluation
Outline

- Background & Contributions
- Preliminaries
- Semi-Federated Mixed-Criticality Algorithm
- Evaluation
Mixed-Criticality Systems

Sub-systems with different criticality levels (e.g. avionics):

- Monitoring and control sub-system
- Anti-collision sub-system
- Navigation sub-system
- ...  
- Infotainment sub-system
Mixed-Criticality Systems

Characteristics:
- More complex functions of system
- No reduction in system security requirements

Challenges:
- Multicore: no enough analytical techniques
- Task parallelization: dependencies and competitions among tasks
Facing the problem

How to assign computation resources to parallel tasks on a multicore platform while guaranteeing security?
Propose a semi-federated mixed-criticality scheduling algorithm

- Propose a mapping algorithm: from mixed-criticality tasks to a middleware layer (mixed-criticality container tasks)
- Prove the schedulability of our proposed algorithm
Outline

- Background & Contributions
- Preliminaries
- Semi-Federated Mixed-Criticality Algorithm
- Evaluation
Mixed-Criticality Task Model

• Implicit-deadline sporadic parallel
• Each task is denoted by a directed acyclic graph
  ✓ Vertices: sequential subtasks
  ✓ Edges: dependencies
• Two types: low-criticality and high-criticality
Mixed-Criticality Task Model

- **WCET** of a vertex (subtask): $c^N$ and $c^O$
- **WCET** of a task: $C^N$ and $C^O$
- The longest chain of a task: $L^N$ and $L^O$
- **Deadline** of a task: $D$
- **Virtual deadline** of a task: $D'$

- **Task utilization**: $u^N = \frac{C^N}{D'}$  $u^O = \frac{C^O}{D - D'}$
Existing work: Federated Mixed-Criticality Scheduling

- Task types: LH, HVH, HMH
- System states: normal state and critical state
- Assigning strategy: independently usable cores to each task in the normal and critical states.

- In the worst case, half of the resources are wasted

\[ 1 + \epsilon \quad 1 + \epsilon \quad 1 + \epsilon \ldots \]
Outline

- Background & Contributions
- Preliminaries
- Semi-Federated Mixed-Criticality Algorithm
- Evaluation
Architecture of Our Algorithm

Semi-Federated Mixed-Criticality Algorithm:

- Two-level hierarchical scheduling framework
Semi-Federated Mixed-Criticality Algorithm:

- Two-level hierarchical scheduling framework

- **Top-level** calculates the mapping from mixed-criticality tasks to mixed-criticality container-tasks.
Architecture of Our Algorithm

**Semi-Federated Mixed-Criticality Algorithm:**

- **Two-level hierarchical scheduling framework**
- **Top-level** calculates the mapping from mixed-criticality tasks to mixed-criticality container-tasks.
- **Bottom-level** schedules mixed-criticality container-tasks on physical processors.
Mixed-Criticality Container Task

- Processor resources interface
Mixed-Criticality Container Task

- Processor resources interface
- Calculate number of mixed-criticality container tasks in both normal states and critical states
Mixed-Criticality Container Task

• Processor resources interface

• Semi-federated: Mixed-criticality container tasks provide virtual resources, so the number of mixed-criticality task can be decimal, avoiding resource waste
Mixed-Criticality Container Task

- Processor resources interface

- If the WCET of a task equals $c$ when it executes on a unit speed processor, then its WCET on a container-task with speed $\sigma$ is:

$$t = \frac{c}{\sigma}$$
Mapping Algorithm

A mixed-criticality task

\[ c^N_{v0} = 1 \quad c^O_{v0} = 2, \quad c^N_{v1} = 1 \quad c^O_{v1} = 3 \]
\[ c^N_{v2} = 1 \quad c^O_{v2} = 3, \quad c^N_{v3} = 1 \quad c^O_{v3} = 3 \]
\[ c^N_{v4} = 1 \quad c^O_{v4} = 3, \quad c^N_{v5} = 1 \quad c^O_{v5} = 2 \]
\[ D = 13, \quad D' = 5 \]
Mapping Algorithm

A mixed-criticality task

\[
\begin{align*}
C_N &= 6, \quad C_O = 16, \quad L_N = 3, \quad L_O = 7 \\
D &= 13, \quad D' = 5
\end{align*}
\]
Mapping Algorithm

A mixed-criticality task

\[ u^N = \frac{C^N}{D'} = \frac{6}{5} > 1 \]

\[ c_{v0}^N = 1, c_{v0}^O = 2, \quad c_{v1}^N = 1, c_{v1}^O = 3 \]
\[ c_{v2}^N = 1, c_{v2}^O = 3, \quad c_{v3}^N = 1, c_{v3}^O = 3 \]
\[ c_{v4}^N = 1, c_{v4}^O = 3, \quad c_{v5}^N = 1, c_{v5}^O = 2 \]

\[ D = 13, \quad D' = 5 \]
\[ C^N = 6, \quad C^O = 16, \quad L^N = 3, \quad L^O = 7 \]
A mixed-criticality task

• Calculating the mapping

\[
C^N = 6, C^O = 16, L^N = 3, L^O = 7, D = 13, D' = 5
\]

Using our mapping equation:

\[
S_i^N = \begin{cases} 
  u_i^N & \text{when } u_i^N < 1 \\
  \frac{C_i^N - L_i^N}{D_i' - L_i^N} & \text{when } u_i^N \geq 1 
\end{cases}
\]

\[
S_i^O = \begin{cases} 
  0 & \text{if task is LO} \\
  \frac{C_i^O - S_i^N D_i' - L_i^O}{D_i - D_i' - L_i^O} & \text{if task is HI}
\end{cases}
\]
A mixed-criticality task
• Calculating the mapping

\[ S^N = \frac{C^N - L^N}{D' - L^N} = \frac{6 - 3}{5 - 3} = 1.5 \]
\[ S^O = \frac{C^O - S^N D' - L^O}{D - D' - L^O} = \frac{16 - 1.5 \times 5 - 7}{13 - 5 - 7} = 1.5 \]
A mixed-criticality task

Mapping Algorithm

Normal State

Critical State

\( \sigma = 1 \)

\( \sigma = 0.5 \)
Runtime of a Mixed-Criticality Task

• Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks

• Scheduling these mixed-criticality container-tasks to physical processors with a partitioned or global algorithm
Runtime of a Mixed-Criticality Task

• Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks
  ➢ Encapsulating the vertex v0

\[
\begin{align*}
\sigma &= 0.5 \\
\sigma &= 1 \\
\text{Normal State}
\end{align*}
\]
Runtime of a Mixed-Criticality Task

• Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks
  ➢ Setting the deadline of the mixed-criticality container-task

\[
\begin{align*}
  c_{v0}^N &= 1, \quad c_{v0}^O = 2, \\
  c_{v1}^N &= 1, \quad c_{v1}^O = 3, \\
  c_{v2}^N &= 1, \quad c_{v2}^O = 3, \\
  c_{v3}^N &= 1, \quad c_{v3}^O = 3, \\
  c_{v4}^N &= 1, \quad c_{v4}^O = 3, \\
  c_{v5}^N &= 1, \quad c_{v5}^O = 2.
\end{align*}
\]
Runtime of a Mixed-Criticality Task

- Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks

  ➢ Encapsulating the vertex v1

\[
c_{v0}^N = 1 \quad c_{v0}^O = 2, \quad c_{v1}^N = 1 \quad c_{v1}^O = 3
\]
\[
c_{v2}^N = 1 \quad c_{v2}^O = 3, \quad c_{v3}^N = 1 \quad c_{v3}^O = 3
\]
\[
c_{v4}^N = 1 \quad c_{v4}^O = 3, \quad c_{v5}^N = 1 \quad c_{v5}^O = 2
\]

![Diagram](image)
Runtime of a Mixed-Criticality Task

- Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks
  - Encapsulating the vertex $v_5$

$$
\begin{align*}
\sigma &= 0.5 \\
\sigma &= 1 \\
\text{deadline} &= 5
\end{align*}
$$

$$
\begin{align*}
c_{v_0}^N &= 1 & c_{v_0}^O &= 2, \\
c_{v_1}^N &= 1 & c_{v_1}^O &= 3, \\
c_{v_2}^N &= 1 & c_{v_2}^O &= 3, \\
c_{v_3}^N &= 1 & c_{v_3}^O &= 3, \\
c_{v_4}^N &= 1 & c_{v_4}^O &= 3, \\
c_{v_5}^N &= 1 & c_{v_5}^O &= 2
\end{align*}
$$
• Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks

➢ State transition: normal -> critical

\[ \sigma = 0.5 \]

\[ \sigma = 1 \]

\[ c_{v0}^N = 1 \quad c_{v0}^O = 2, \quad c_{v1}^N = 1 \quad c_{v1}^O = 3 \]

\[ c_{v2}^N = 1 \quad c_{v2}^O = 3, \quad c_{v3}^N = 1 \quad c_{v3}^O = 3 \]

\[ c_{v4}^N = 1 \quad c_{v4}^O = 3, \quad c_{v5}^N = 1 \quad c_{v5}^O = 2 \]
Runtime of a Mixed-Criticality Task

\[ \sigma = 0.5 \]

\[ \sigma = 1 \]

Normal State

Critical State
Runtime of a Mixed-Criticality Task

• Mixed-Criticality tasks are encapsulated into mixed-criticality container-task
  ➢ In critical states

\[
\begin{aligned}
c^N_{v0} &= 1, \quad c^O_{v0} = 2, \\
c^N_{v1} &= 1, \quad c^O_{v1} = 3, \\
c^N_{v2} &= 1, \quad c^O_{v2} = 3, \\
c^N_{v3} &= 1, \quad c^O_{v3} = 3, \\
c^N_{v4} &= 1, \quad c^O_{v4} = 3, \\
c^N_{v5} &= 1, \quad c^O_{v5} = 2
\end{aligned}
\]
Runtime of a Mixed-Criticality Task

- Mixed-Criticality tasks are encapsulated into mixed-criticality container-tasks

- Scheduling these mixed-criticality container-task to physical processor with a partitioned or global algorithm
  - First, system is in normal state and only normal mixed-criticality container-tasks are scheduled
  - When the system experiences state transition, all normal mixed-criticality container-tasks are banned from scheduling and only critical mixed-criticality container-tasks can run
Outline

- Background & Contributions
- Preliminaries
- Semi-Federated Mixed-Criticality Algorithm
- Evaluation
Evaluation

• Our task sets are generated based on OpenMP benchmarks and ompTG tool.

• Evaluate the acceptance ratio of task sets with different normalized utilizations.

• We compare our results with those in existing work for federated mixed-criticality scheduling.
Evaluation

Semi-federated mixed-criticality algorithm has better schedulability performance
Thanks for your attention!