Mixed-Criticality Scheduling for Parallel Real-Time Tasks with Resource Reclamation

Qingqiang He\textsuperscript{1}, Nan Guan\textsuperscript{2}, Xu Jiang\textsuperscript{3}

\textsuperscript{1} The Hong Kong Polytechnic University, China
\textsuperscript{2} City University of Hong Kong, China
\textsuperscript{3} Northeastern University, China
Summary

• federated scheduling of parallel real-time tasks
  o each task is scheduled independently on a set of dedicated cores
  o so we only need to consider the scheduling of one task on multi-cores

• target of our approach
  o guarantee the deadline for hard real-time tasks
  o reclaim computing resources for soft real-time tasks

• the proposed approach
  o online monitor the execution of hard real-time tasks
  o dynamically adjust the allocated number of cores for hard real-time tasks
The Parallel Real-Time Task

• for the scheduling
  o volume $vol(G)$: the total workload in this task
  o length $len(G)$: the workload in the longest path
  o these parameters can be measured without knowing the structure of the DAG

• for the analysis
  o the DAG task model

The longest path is $\lambda = (v_0, v_1, v_4, v_5)$

$len(G) = 6 \quad vol(G) = 10.$
The Scheduling

- work-conserving
- identical multi-core platform
Motivation

• Graham’ bound in federated scheduling

\[ R \leq \text{len}(G) + \frac{\text{vol}(G) - \text{len}(G)}{m} \]
\[ m = \left\lceil \frac{\text{vol}(G) - \text{len}(G)}{D - \text{len}(G)} \right\rceil \]

• type 1 : analysis pessimism
  - assuming that vertices not in the longest path do not execute in parallel with the execution of the longest path.
Motivation

• type 2: execution pessimism
  o volume and length are based on the worst case execution time (WCET).
  o the actual execution time of vertices can be far less than the WCET

• due to these two types of pessimism, the federated scheduling can waste a large amount of computing resources.
The Proposed Approach

- online monitor the execution of hard real-time tasks
  - the volume of the executed workload \( w(t_i) \)
  - the length of time intervals during which at least one core is idle \( l(t_i) \)
- adjust the allocated number of cores for hard real-time tasks

**Definition 1** (Allocation Vector). For a parallel real-time task \((G, D, T)\), the allocation vector \(\Phi\) is a set of time points \(\{t_0, \cdots, t_k\} (k \geq 0)\) satisfying all of the following conditions.

1) \( \forall i \in [0, k], 0 \leq t_i < D. \)
2) \( \forall i, j \in [0, k] \) and \( i < j \), \( t_i < t_j \).
The Proposed Approach

• online monitor the execution of hard real-time tasks
  o the volume of the workload executed $w(t_i)$
  o the length of time intervals during which at least one core is idle $l(t_i)$

• adjust the allocated number of cores for hard real-time tasks

$$m = \left\lceil \frac{vol(G) - len(G)}{D - len(G)} \right\rceil$$

$$vol(G') = vol(G) - w(t_i)$$
$$len(G') = len(G) - l(t_i)$$
$$D' = D - t_i$$

$$m_i = \left\lceil \frac{vol(G') - len(G')}{D' - len(G')} \right\rceil$$
The Proposed Approach

• the critical path
  o the length of time intervals during which at least one core is idle $l(t_i)$
  o the length of the remaining graph is bounded by

$$\text{len}(G') = \text{len}(G) - l(t_i)$$

• our approach dominates the original federated scheduling
  o the adjusted number of cores cannot increase (Corollary 1)
The Proposed Approach

• design principle for soft real-time tasks
  o how to determine the allocation vector
• monitor the execution satisfying both of the following conditions
  o at least one core is idle
  o more than one core are busy
An Example

\[ \text{len}(G) = 6 \quad \text{vol}(G) = 10 \]

\[ D = 7 \]

\[
m = \left\lfloor \frac{\text{vol}(G) - \text{len}(G)}{D - \text{len}(G)} \right\rfloor = 4
\]

\[ m \times D = 4 \times 7 = 28 \]

allocation vector

\[ \Phi = \{ t_0 = 2, t_1 = 3 \} \]

\[ t_0 = 2 \quad w(t_0) = 4 \quad l(t_0) = 2 \]

\[ \text{vol}(G') = \text{vol}(G) - w(t_i) = 6 \]

\[ \text{len}(G'') = \text{len}(G) - l(t_i) = 4 \]

\[ D' = D - t_i = 5 \]

federated scheduling

our approach
Conclusion

• the proposed approach
  o online monitor the execution of hard real-time tasks
  o dynamically adjust the allocated number of cores for hard real-time tasks

• allocation vector as interface
  o for hard real-time tasks, a schedulability test under the interface
  o for soft real-time tasks, the design principle of how to determine the interface to reclaim computing resources

• using an example to illustrate the effectiveness of the proposed approach.

Thank you and Questions?