Scheduling Algorithms in Mixed-Criticality Systems

Jaewoo Lee

WPE-II Presentation
May. 1. 2015
Introduction

• Safety-critical real-time systems
  – Timing correctness is important

• Trends
  – Complex and multi-functional
  – Constrained by SWaP (Size, Weight and Power)
A Trend in Safety-Critical RTS

• Mixed-Criticality (MC) system
  – A shared platform with multiple components of different criticality
  – Ex) US FAA, RTCA DO-178B (S/W standard)
    • Specifies criticality of aircraft component

<table>
<thead>
<tr>
<th>Criticality</th>
<th>Failure Condition</th>
<th>Example result</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Catastrophic</td>
<td>Crash</td>
</tr>
<tr>
<td>B</td>
<td>Hazardous</td>
<td>People injury</td>
</tr>
<tr>
<td>C</td>
<td>Major</td>
<td>People discomfort</td>
</tr>
<tr>
<td>D</td>
<td>Minor</td>
<td>A routine flight plan change</td>
</tr>
<tr>
<td>E</td>
<td>No effect</td>
<td>No impact on safety</td>
</tr>
</tbody>
</table>
MC Systems: Importance

• Design of MC systems: practically important
  – US Air Force Research Laboratory
    • Mixed Criticality Architecture Requirement (MCAR) program (2009)

  – A workshop report (2012) by European Commission
    • MC research challenges on wide applications
      – Aerospace, automotive, health-care, and building-automation
MC Systems: Challenge

• Research challenge
  – Validation of timing correctness in multiple levels
  – An example
    • Unmanned Aerial Vehicle (UAV)
      1) Flight-critical functions (e.g. engines)
      2) Mission-critical functions (e.g. cameras)
    • System designers concern engines and cameras under normal condition
    • Certification Authorities (CAs) concern only engines under pessimistic condition (e.g. extreme weather)
MC Systems: A Question

MC scheduling problem

Schedule tasks of different criticality resource-efficiently (due to SWaP constraint) while satisfying multiple levels of timing requirements

Ex)
- Low-criticality level: all tasks under normal condition
- High-criticality level: high-critical tasks under pessimistic condition
MC Systems: Revisit Real-Time Techniques

• Traditional Real Time (RT) approaches
  – All tasks have equal criticality

• What if the RT approaches on the MC problem?

<table>
<thead>
<tr>
<th>Low-criticality (normal condition)</th>
<th>High-criticality (pessimistic condition)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex) Parameters (e.g. period, WCET) from empirical measurement</td>
<td>Ex) Parameters from conservative static analysis</td>
</tr>
<tr>
<td>The failure of high-critical tasks</td>
<td>Too pessimistic system design</td>
</tr>
<tr>
<td>→ Violation on high-criticality req.</td>
<td>→ Inefficiency in resource</td>
</tr>
</tbody>
</table>

• Revisit RT techniques on the MC problems
  – To deal with the MC problem w/ multiple criticality, revisit RT techniques (w/ single criticality)
Current status of MC scheduling

- Vestal (2007) first considered MC scheduling problem
  - Proposed MC system model

- The unicore MC scheduling
  - Studied extensively

- The multicore MC scheduling
  - Little work exists

- Open challenges on MC scheduling
  1. the multicore MC scheduling problem is largely open
  2. to close gaps between theory and practice, refine MC system modelling
This Survey

• Understand existing MC scheduling
  1. a seminal work on MC scheduling (MC task model)
  2. a state-of-the-art MC scheme (high resource-efficiency)
     Ekberg and Yi, *Bounding and shaping the demand of generalized mixed-criticality sporadic task systems*, RTS, 2014
  3. an initial work on multicore MC scheduling (extension to multicore)
     Li and Baruah, *Global mixed-criticality scheduling on multiprocessors*, ECRTS, 2012

• Compare MC scheduling algorithms

• Discuss possible research directions
Outline

• Introduction
• **System Model**
• Three MC approaches
• Evaluation
• Research directions
• Conclusion
System Model

- **Dual Criticality** (for ease of presentation)
  - HI- and LO-criticality level

- **Sporadic task** $\tau_i = (T_i, D_i, C_i)$

  - $T_i$: Period
  - $D_i$: (relative) Deadline
  - $C_i$: Worst Case Execution Time (WCET)
System Model

- **Dual Criticality** (for ease of presentation)
  - HI- and LO-criticality level

- **Sporadic MC task** \( \tau_i = (T_i, D_i, C_i^L, C_i^H, X_i) \)

  - \( C_i^L \): LO-criticality Worst Case Execution Time (LO-WCET)
  - \( C_i^H \): HI-WCET
  - \( D_i \): (relative) Deadline
  - \( T_i \): Period
  - Task-criticality (HI or LO)

- **Task classification**
  - HI-task: a task with HI task-criticality
  - LO-task: a task with LO task-criticality
Outline

• Introduction
• System Model
• Three MC approaches
  – Classic MC
  – EDF-based MC
  – Multicore MC
• Evaluation
• Research directions
• Conclusion
Classic MC: Motivation

• Motivation
  – Problematic assumption: WCET (Worst Case Execution Time)
    • Static analysis → overly-pessimistic
    • Empirical measurement → underestimating
  – Different requirement depending on criticality
    • CAs (HI-crit.) require high confidence on WCET
    • System designers (LO-crit.) allow occasional failures

• MC task model by Vestal in 2007
  – Multiple WCETs for a task by different measurement
  – Different roles consider exec. time of tasks differently
    • CAs: each task has HI-WCET
    • System designers: each task has LO-WCET
Classic MC: Scheduling Requirement

- **System behavior** (assumption under criticality level)
  - **LO-behavior**: tasks are finished within LO-WCETs
  - **HI-behavior**: tasks are finished within HI-WCETs
Classic MC: Scheduling Requirement

- **System behavior** *(assumption under criticality level)*
  - **LO-behavior**: tasks are finished within **LO-WCETs**
  - **HI-behavior**: tasks are finished within **HI-WCETs**

- **Scheduling requirement**
  - **LO-criticality req.**
    - All **HI-** and **LO-tasks** are schedulable under **LO-behavior**

![Diagram](image)
Classic MC: Scheduling Requirement

• System behavior (assumption under criticality level)
  – LO-behavior: tasks are finished within LO-WCETs
  – HI-behavior: tasks are finished within HI-WCETs

• Scheduling requirement
  – LO-criticality req.
    • All HI- and LO-tasks are schedulable under LO-behavior
  – HI-criticality req.
    • Only HI-tasks are schedulable under HI-behavior

<table>
<thead>
<tr>
<th>HI-tasks</th>
<th>LO-tasks</th>
<th>HI-WCET</th>
<th>Schedulable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>√</td>
</tr>
</tbody>
</table>
Classic MC: Overview

• Scheduling algorithm
  – The Fixed-Priority (FP) algorithm

• Schedulability analysis
**Classic MC: Scheduling Algorithm**

- The Fixed-Priority (FP) scheduling algorithm
  - All jobs of a task share the same priority
  - Cf.) Dynamic priority scheduling algorithm

![Diagram of FP and DP scheduling algorithms]
Classic MC: Schedulability Analysis

• Response Time Analysis (RTA) for **non-MC tasks**
  – Response time is calculated by the recurrence relation:

\[
R_i = \sum_{\tau_k \in HP(\tau_i)} \left[ \frac{R_i}{T_k} \right] C_k + C_i
\]

  – The task is schedulable if response time \( \leq \) deadline
  – Complexity: **pseudo-polynomial**

• Vestal’s RTA
  – Extend RTA for **MC tasks**

For a **HI-task**,\[R_i = \sum_{\tau_k \in HP(\tau_i)} \left[ \frac{R_i}{T_k} \right] C_k^{H} + C_i^{H},\]

For a **LO-task**,\[R_i = \sum_{\tau_k \in HP(\tau_i)} \left[ \frac{R_i}{T_k} \right] C_k^{L} + C_i^{L}.\]
Classic MC: Priority Assignment

- Goal: an optimal priority assignment for Vestal’s RTA

- Two simple priority assignment
  - Deadline Monotonic (DM)
    - Higher priority on the task with shorter deadline
  - Criticality Monotonic (CM)
    - Higher priority on the task with higher task-criticality
  - DM or CM is not optimal
Classic MC: Priority Assignment

• Goal: an optimal priority assignment for Vestal’s RTA

• Two simple priority assignment
  – Deadline Monotonic (DM)
    • Higher priority on the task with shorter deadline
  – Criticality Monotonic (CM)
    • Higher priority on the task with higher task-criticality
  – DM or CM is not optimal

• Algorithm OPA (Optimal Priority Assignment) [Audsley 01]
  – It is for non-MC RTA but applicable for Vestal’s RTA
  – Can achieve the goal by checking only $n^2$ combinations (polynomial)
Classic MC: Discussion

• **Strength**
  – MC task model (extending non-MC task)

• **Limitation**
  1. limited optimal under the **FP** algorithm
     • Better schedulability by using dynamic priority algorithms
  2. limited schedulability
     • Requirement for LO-tasks: schedulable **only under LO-behavior**
     • Better schedulability by **canceling jobs** of LO-tasks that execute more than LO-WCETs
## Classic MC: Summary

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Classic MC: Related Work

- Better schedulability with dynamic priority scheduling
  - [Baruah08]: a dynamic priority algorithm that dominates Vestal’s approach

- Better schedulability within fixed-priority scheduling
  - [Burns 12]: utilize canceling over-execution of LO-tasks
# Runtime Monitoring

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td>X</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Runtime Monitoring

• Runtime Monitoring
  – Platform support to monitor current execution times of jobs
  – Common on many real time systems
    • Ex) Sporadic server monitors budget consumption

• Runtime Monitoring Consideration
  – Key feature in recent MC approaches
  – Asymmetric protection by task-criticality
    • LO-tasks are prevented to execute more than LO-WCET
      Requirement of LO-tasks: schedulable only under LO-behavior
    • HI-tasks are allowed to execute more than LO-WCET
Runtime Monitoring: Scheduling Requirement

• Change of MC tasks
  – No jobs of LO-tasks can execute more than LO-WCET
    • Runtime monitoring can cancel over-executed jobs of LO-tasks
Runtime Monitoring: Scheduling Requirement

- **Change of MC tasks**
  - No jobs of LO-tasks can execute more than LO-WCET
  - Runtime monitoring can cancel over-executed jobs of LO-tasks

- **System mode** (depending on behaviors of HI-tasks)
  - LO-mode: no task executes more than LO-WCET
  - HI-mode: some HI-task executes more than LO-WCET
Runtime Monitoring: Scheduling Requirement

• Change of MC tasks
  – No jobs of LO-tasks can execute more than LO-WCET
    • Runtime monitoring can cancel over-executed jobs of LO-tasks

• System mode (depending on behaviors of HI-tasks)
  – LO-mode: no task executes more than LO-WCET
  – HI-mode: some HI-task executes more than LO-WCET

• MC scheduling requirement with runtime monitoring
  – In LO-mode, all tasks are schedulable
  – In HI-mode, only HI-tasks are schedulable
Runtime Monitoring: MC System Scenario

- **MC system scenario**
  - Starts in **LO-mode**
  - If a **HI-task** executes more than **LO-WCET**, mode-switch (LO-mode $\rightarrow$ HI-mode)
  - At mode switch, drop all **LO-tasks** (not considered in HI-mode)

  ![Diagram](image)

  - **LO-WCET**
    - HI-task
    - LO-task
  - **Runtime Mode Indicator**
    - LO
    - HI

  *Cancel current job/no job release of the task*
# EDF-based MC

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td>X</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EDF-based MC

- The EDF (earliest deadline first) scheduling algorithm
  - **Optimal** for non-MC tasks
  - Schedule the task with the earliest deadline

- EDF is **not directly applicable** to MC tasks

- MC characteristic: mode-switch
  - LO-tasks are dropped
  - The execution requirement of HI-tasks: LO-WCET → HI-WCET
EDF-based MC: Mode Switch

- Carry-over jobs
  - Jobs (of HI-tasks) released in LO-mode and finished in HI-mode
  - Only jobs that experience mode-switch
  - May execute some amount of works in LO-mode

- To analyze mode-switch, consider schedulability of carry-over jobs
EDF-based MC: Carry-over Jobs

- What if we apply EDF for carry-over job?
  - Consider mode-switch just before deadline
    - A job of a HI-task will execute LO-WCET by deadline
    - When the job is not finished at MS: the job → **carry-over** job
    - The requirement of **carry-over** job is HI-WCET → **deadline miss**
EDF-based MC: The Virtual Deadline

- What if we apply EDF for carry-over jobs?
  - Consider mode-switch just before deadline
    - A job of a HI-task will execute LO-WCET by deadline
    - When the job is not finished at MS, the job → carry-over job
    - The requirement of carry-over job is HI-WCET → deadline miss

- Virtual Deadlines (VDs) for HI-tasks in LO-mode
  - VD is an artificial deadline ≤ real deadline
  - The use of VD in LO-mode → execution of LO-WCET by VD
    → even if mode-switch, execution of HI-WCET by real deadline
EDF-based MC: The Choice of Virtual Deadline

- The choice of virtual deadline
  - **Shorter** virtual deadline
    - Smaller remaining time to schedule other HI-tasks and LO-tasks in LO-mode
EDF-based MC: The Choice of Virtual Deadline

- The choice of virtual deadline
  - **Shorter** virtual deadline
    - *Smaller* remaining time to schedule other HI-tasks and LO-tasks in LO-mode
  - **Larger** virtual deadline
    - *Smaller* remaining time to schedule other carry-over jobs in HI-mode
  - A **proper** choice of virtual deadline is important
    - *Schedulability analysis* for given VD assignment
    - *VD assignment* algorithm

![Diagram showing time intervals for scheduling tasks](chart.png)
EDF-based MC: Scheduling Algorithm

- MC-EDF: an EDF-based scheduling algorithm with VDs
  - In LO-mode, schedule HI-tasks with virtual deadlines and LO-tasks with real deadlines by EDF
  - In HI-mode, schedule HI-tasks with real deadlines by EDF
EDF-based MC: Schedulability Analysis

• Demand analysis [Baruah 90] for non-MC tasks
  – Demand Bound Function (DBF)
    • An exact demand (execution requirement) of a task set for a given time interval
  – Determine schedulability of EDF with pseudo-polynomial complexity
EDF-based MC: Schedulability Analysis

• Demand analysis [Baruah 90] for non-MC tasks
  – Demand Bound Function (DBF)
    • An exact demand (execution requirement) of a task set for a given time interval
  – Determine schedulability of EDF with pseudo-polynomial complexity

• Extension of demand analysis for MC tasks
  – Virtual deadlines are given
  – Derive DBF for MC tasks in LO-mode and HI-mode considering carry-over jobs
  – Determine schedulability of MC-EDF with pseudo-polynomial complexity
EDF-based MC: Virtual Deadline Assignment

• Goal: VD assignment for MC-Demand analysis

• *GreedyTuning* VD assignment algorithm
  – A heuristic algorithm that tunes VD of each task individually with *pseudo-polynomial* complexity
EDF-based MC: Discussion

• Strength
  1. consideration of **carry-over jobs**
  2. **task-level** VD tuning
  3. **demand analysis** for MC-EDF

• Limitation
  – **Limited** VD assignment
## EDF-based MC: Summary

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td>MC-EDF</td>
<td></td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td>Demand analysis</td>
<td></td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td>A heuristic VD assignment</td>
<td></td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td>Pseudo-polynomial</td>
<td></td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td>Carry-over jobs</td>
<td></td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td>Limited VD assignment</td>
<td></td>
</tr>
</tbody>
</table>
EDF-based MC: Related Work

• OCBP [Baruah10]
  – First proposed runtime monitoring consideration

• EDF-VD [Baruah11, Baruah12]
  – First proposed MC-EDF algorithm

• GreedyTuning [Ekberg 12, Ekberg 14] (this section)

• ECDF [Easwaran 13]
  – Proposed tighter analysis than GreedyTuning
# Multicore MC

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td>X</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td>MC-EDF</td>
<td></td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td>Demand analysis</td>
<td></td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td>A heuristic VD assignment</td>
<td></td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td>Pseudo-polynomial</td>
<td></td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td>Carry-over jobs</td>
<td></td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td>Limited VD assignment</td>
<td></td>
</tr>
</tbody>
</table>
Multicore MC

- Multicore platforms are dominant
  - Limitations of Unicore platforms
    - Power consumption and heat dissipation

- Multicore MC systems
  - MC systems are migrating to multicore platforms
  - Generally, unicore scheduling algorithm is not applicable to multicore
    - Limitation of sequential task on multiple cores
Multicore MC: Extension of MC-EDF

- Multicore extension of MC-EDF [Li 12]
  - Extend unicore MC-EDF with global multicore scheduling
    - Need an EDF-based global scheduling algorithm

- The fpEDF algorithm for non-MC tasks
  - An EDF-based global multicore scheduling algorithm
  - Schedulability condition:

$$\sum_{\tau_i \in \tau} u_i \leq \frac{m + 1}{2}$$

where task utilization

$$u_i = \frac{C_i}{T_i}$$

# of cores
Multicore MC: Algorithm & Analysis

• Unicore EDF-VD [Baruah11]
  – Utilization-based analysis for MC-EDF
  – VD assignment algorithm by a single system-level VD tuning parameter

• Algorithm fpEDF-VD: MC-EDF with fpEDF
  – In LO-mode, schedule HI-tasks with virtual deadlines and LO-tasks with real deadlines by fpEDF
  – In HI-mode, schedule HI-tasks with real deadlines by fpEDF

• Analysis of fpEDF-VD
  – Another utilization-based analysis by extending EDF-VD analysis with fpEDF condition
Multicore MC: Discussion

• Strength
  – Extended MC-EDF into multicore

• Limitation
  – Poor schedulability for larger number of cores
# Multicore MC: Summary

<table>
<thead>
<tr>
<th></th>
<th>Classic MC</th>
<th>EDF-based MC</th>
<th>Multicore MC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors</strong></td>
<td>Vestal</td>
<td>Ekberg and Yi</td>
<td>Li and Baruah</td>
</tr>
<tr>
<td><strong># of cores</strong></td>
<td>1</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td><strong>Runtime Monitoring</strong></td>
<td>X</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td><strong>Scheduling Algorithm</strong></td>
<td>FP</td>
<td>MC-EDF</td>
<td>MC-EDF</td>
</tr>
<tr>
<td><strong>Schedulability Analysis</strong></td>
<td>RTA</td>
<td>Demand analysis</td>
<td>Utilization analysis</td>
</tr>
<tr>
<td><strong>Priority Assignment</strong></td>
<td>OPA</td>
<td>A heuristic VD assignment</td>
<td>VD tuning by a system parameter</td>
</tr>
<tr>
<td><strong>Test Complexity</strong></td>
<td>Pseudo-polynomial</td>
<td>Pseudo-polynomial</td>
<td>Polynomial</td>
</tr>
<tr>
<td><strong>Strength</strong></td>
<td>MC task model</td>
<td>Carry-over jobs</td>
<td>Multicore extension</td>
</tr>
<tr>
<td><strong>Limitation</strong></td>
<td>Limited schedulability</td>
<td>Limited VD assignment</td>
<td>Poor schedulability for larger cores</td>
</tr>
</tbody>
</table>
Multicore MC: Related Work

• Initial works on multicore MC scheduling
  – P-EDF-VD [Baruah14]
    • Partitioned scheduling with MC-EDF
  – G-MC-FP [Pathan12]
    • Global FP scheduling algorithm
Outline

• Introduction
• System Model
• Three MC approaches
• Evaluation
• Research directions
• Conclusion
Evaluation

- We evaluate performance of MC algorithms

  1. compare unicore MC algorithms by simulations
     Please refer to my report

  2. compare multicore MC algorithms by simulations
     I conducted my own simulations
Evaluation: Multicore MC

• Performance comparison of multicore MC algorithms
  – Acceptance ratio for randomly-generated task sets

  For total task sets, the ratio of task sets schedulable by the algorithm

• Simulation on multicore platforms
  – Implement multicore MC algorithms
    • Global scheduling approaches:
      fpEDF-VD (reviewed), G-MC-FP [Pathan12]
    • Partitioned scheduling approach: P-EDF-VD [Baruah14]

• Focus
  1. fpEDF-VD shows poor schedulability for larger number of cores?
  2. any multicore MC algorithm w/ high schedulability?
Evaluation: Multicore MC Result

- Simulation varying normalized utilization bound
- Result:
  - Global scheduling algorithms have poor performance for larger number of cores
  - Base multicore algorithms have the same problem
Evaluation: Multicore MC Result

• Simulation varying task utilization
• Result
  – P-EDF-VD has poor performance for higher task-utilization due to bin-packing problem
Evaluation: Summary

• Summary of simulation for multicore MC systems

  – Global scheduling algorithms (fpEDF-VD, G-MC-FP)
    • Shows poor schedulability for larger number of cores

  – Partitioned scheduling algorithms (P-EDF-VD)
    • Shows poor schedulability for higher task utilization

  – There exist no multicore MC scheme with high schedulability
Outline

• Introduction
• System Model
• Three MC approaches
• Evaluation
• Research directions
• Conclusion
Direction: A Good Multicore MC Scheme

• Current status of MC scheduling
  – Unicore MC scheduling
    • GreedyTuning: high schedulability
  – Multicore MC scheduling
    • Poor schedulability for larger cores or higher task-utilization

• Direction: a good multicore MC scheduling scheme
  – Need a multicore MC scheme with high schedulability
Direction: A Good Multicore MC Scheme (a result)

- **Fluid scheduling model**
  - Overcome limitation of sequential task on multicore by large # of core migration

- **MC-Fluid [1]**
  - A first work to apply fluid model into multicore MC
  - Use virtual rate for HI-tasks in LO-mode

- **Simulation result**
  - High schedulability even for larger cores and high task-utilization

---

Direction: Another MC System Modelling

• Existing MC do not consider LO-task in HI-mode
  – CAs are not interested in LO-tasks

• LO-tasks are still important in practice
  – LO-tasks need to execute as much as possible
    • Ex) Surveillance mission in UAV

• Direction
  – Another scheduling requirement
    • Minimizing Deadline Miss Ratio (DMR) of LO-tasks
  – Another MC system modelling considering DMR of LO-tasks as well as existing scheduling requirement
Conclusion

- **MC systems**
  - An important trend in safety-critical RTS
- **Three different MC scheduling approaches**
  - MC task model
  - High schedulability in unicore MC domain
  - Extension to multicore MC domain
- **Current status**
  - Unicore MC scheduling: *extensively* studied
  - Multicore MC scheduling: *little* attention
- **Research directions**
  - A good *multicore* MC scheme
  - Another MC system modelling reducing *deadline miss ratio* of LO-task
Thank you

Any question?