The Feasibility Analysis of Mixed-Criticality Systems

Saravanan Ramanthan, Xiaozhe Gu, Arvind Easwaran

Nanyang Technological University, Singapore

July 5, 2016
Outline

1. Motivation
2. Open Problem
3. Possible Solution
4. Design Methodology
Motivation

- 2007: Vestal introduced MC system
- 2012: EDF-VD: Speed-up optimal single-core algorithm
- 2015: MC-Fluid/MCF: Speed-up optimal multi-core algorithm
- 2016: No optimal MC algorithm
  - No known tight feasibility bound
<table>
<thead>
<tr>
<th>1</th>
<th>Motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Open Problem</td>
</tr>
<tr>
<td>3</td>
<td>Possible Solution</td>
</tr>
<tr>
<td>4</td>
<td>Design Methodology</td>
</tr>
</tbody>
</table>
The open problem

What is a tight feasibility bound for Mixed-Criticality (MC) task systems?
Outline

1. Motivation
2. Open Problem
3. Possible Solution
4. Design Methodology
Possible Solution

Mixed-Criticality System:
- Single-core / Multi-core scheduling
- Dual-criticality / Multi-Criticality system
- Periodic / Sporadic task model
- Implicit / Constrained deadline
Possible Solution

Mixed-Criticality System:
- Single-core / Multi-core scheduling
- Dual-criticality / Multi-Criticality system
- Periodic / Sporadic task model
- Implicit / Constrained deadline
**Task Model**: Implicit-deadline dual-criticality (namely LO and HI) periodic task system is being considered.

\[ \tau_i = (T_i, \chi_i, C_{i}^{L}, C_{i}^{H}, D_i) \]

- \( T_i \in \mathbb{R}^+ \) is the period,
- \( \chi_i \in \{LO, HI\} \) is the criticality level,
- \( C_{i}^{L} \) and \( C_{i}^{H} \) are the LO- and HI-criticality Worst-Case Execution Time (WCET) values respectively; we assume \( C_{i}^{L} \leq C_{i}^{H} \) and,
- \( D_i = T_i \) is the relative deadline.
System-level utilizations are defined as

- $U_L^L \overset{\text{def}}{=} \sum_{\tau_i \in \tau_L} u_i^L$,
- $U_H^L \overset{\text{def}}{=} \sum_{\tau_i \in \tau_H} u_i^L$ and
- $U_H^H \overset{\text{def}}{=} \sum_{\tau_i \in \tau_H} u_i^H$

where, $u_i^L = C_i^L / T_i$ and $u_i^H = C_i^H / T_i$
Outline

1. Motivation
2. Open Problem
3. Possible Solution
4. Design Methodology
Challenge: Determining the worst-case mode switch pattern
**Challenge**: Determining the worst-case mode switch pattern

**Solution**: Fluid model

- Execution rate ($\theta_i$) determines the mode switch instant ($C_i^L/\theta_i^L$)
- Non-MC systems: Most fluid algorithms are optimal
Design Methodology

Design of optimal scheduling algorithm involves

1. *In LO mode*: Schedule LO-criticality tasks as late as possible
2. *In LO mode*: Schedule HI-criticality tasks with their virtual deadline \( \left( \frac{C_i^L}{\theta_i^L} \right) \)
3. *In HI mode*: Optimal scheduling of HI-criticality tasks inclusive of carry-over demand of HI-criticality tasks.
HI-mode schedulability: In the absence of LO-tasks, fluid scheduling can optimally schedule HI-tasks in HI-mode.
**Design Methodology**

**HI-mode schedulability**: In the absence of LO-tasks, fluid scheduling can optimally schedule HI-tasks in HI-mode.

**LO-mode schedulability**:  
1. Use DP-Fair to schedule HI-tasks in LO mode  
   - Virtual deadline \( \frac{C^L_i}{\theta^L_i} \) and actual deadline \( T_i \)  
2. Schedule LO-tasks as late as possible
Thank you..!

Questions..?