Two Protocols to Reduce the Criticality Level of Multiprocessor Mixed-Criticality Systems

François Santy  Gurulingesh Raravi  Geoffrey Nelissen  Vincent Nélis

Pratyush Kumar  Joël Goossens  Eduardo Tovar
System Model

- Constrained-deadline sporadic tasks
  \[ \tau_1, \tau_2, \tau_3, \ldots, \tau_n \]

- Identical multiprocessors
  \[ P_1, P_2, P_3, P_4, \ldots, P_m \]

- Multi-level mixed-criticality system
Task characteristics: $<L_i, C_i, T_i, D_i>$

- $L_i$: criticality level of the task
- $C_i$: vector of WCETs $<C_i(1), C_i(2), \ldots, C_i(L)>$
Motivation

• An example
  
  - \( L = \{ \text{LO, HI} \} \)
  
  - \( \tau = \{ \tau_1, \tau_2, \tau_3 \} \)
  
  - \( \pi = \{ \pi_1, \pi_2 \} \)
  
  \( \tau_1 = <\text{HI, [6, 9], 10, 10}> \)
  
  \( \tau_2 = <\text{HI, [6, 9], 10, 10}> \)
  
  \( \tau_3 = <\text{LO, [4, 4], 10, 10}> \)
Motivation

• An example
  
  – \( L = \{ \text{LO, HI} \} \)
  
  – \( \tau = \{ \tau_1, \tau_2, \tau_3 \} \)
  
  – \( \pi = \{ \pi_1, \pi_2 \} \)

\[
\begin{align*}
\tau_1 &= <\text{HI}, [6, 9], 10, 10> \\
\tau_2 &= <\text{HI}, [6, 9], 10, 10> \\
\tau_3 &= <\text{LO}, [4, 4], 10, 10> 
\end{align*}
\]
• Safe Criticality Reduction (SCR) Problem
  – When to reduce the criticality of the system so that:
    • the lower criticality tasks can be (re-)activated
    • without jeopardizing the schedulability
A trivial solution to the SCR problem?

- **Uniprocessor:**
  - Reduce when the processor is idle

- **Multiprocessor extension**
  - Reduce when all the processors are idle

![Diagram showing process execution and overruns](image)
Contributions of this work

• Two protocols for the SCR problem:
  1. FTP protocol
     • for Fixed Task Priority schedulers
  2. FJP protocol
     • for Fixed Job Priority schedulers
FTP protocol: Illustration

• An example
  – \( L = \{\text{LO, HI}\} \)
  – \( \tau = \{\tau_1, \tau_2, \tau_3, \tau_4\} \)
  – \( \pi = \{\pi_1, \pi_2\} \)
  – Overrun

\[
\begin{align*}
\tau_1 &= <\text{HI}, [3, 4], 5, 5> \quad f_1 = 6 \\
\tau_2 &= <\text{HI}, [3, 4], 9, 9> \quad f_2 = 6 \\
\tau_3 &= <\text{LO}, [1, 1], 5, 5> \quad f_3 = \text{NA} \\
\tau_4 &= <\text{HI}, [4, 9], 11, 11> \quad f_4 = 9
\end{align*}
\]
FTP protocol: Properties

• Upper bound on the time to reduce CL from HI to LO
  – \(\sum R(i)\) of all the tasks with \(L_i \geq HI\)

• The protocol is correct
  – does not jeopardize the schedulability

• The protocol is applicable to
  – any fixed task priority scheduler

• Less run-time overhead
FJP protocol: The Main Idea

- Maintain the *reference schedule* at each criticality level

![Diagram of FJP protocol]
FJP protocol: The Main Idea

• Compare the following schedules
  – *Actual schedule (act-sched)* at *current level* CL
  – *Reference schedule (ref-sched)* at *lower level* LL<CL
FJP protocol: The Main Idea

• Compare act-sched and ref-sched
  – For an active job of task \( \tau_i \) at time \( t \), let
    • \( CE_i(t) \): duration of execution at CL upto time \( t \) (act-sched)
    • \( act\text{-rem}_i(t, LL) \): actual worst-case remaining time at LL
      – \( act\text{-rem}_i(t, LL) = WCET_i(LL) - CE_i(t) \)
    • \( LE_i(t) \): duration of execution at LL upto time \( t \) (ref-sched)
    • \( ref\text{-rem}_i(t, LL) \): reference worst-case remaining time at LL
      – \( ref\text{-rem}_i(t, LL) = WCET_i(LL) - LE_i(t) \)

  – SCR condition: for all \( \tau_i \in \tau \) with \( L_i \geq CL \)
    1. \( act\text{-rem}_i(t, LL) \geq 0 \)
    2. \( act\text{-rem}_i(t, LL) \leq ref\text{-rem}_i(t, LL) \)
FTP protocol: Properties

- The protocol is correct
  - does not jeopardize the schedulability

- The protocol is applicable to
  - any fixed job priority scheduler

- Incurs higher run-time overhead!
Conclusions and Discussions

• Two new protocols for SCR problem
  – FTP protocol
  – FJP protocol

• Generic solutions
  – Multi-criticality level systems
  – Multiprocessor systems
  – Constrained-deadline sporadic tasks
  – Global and partitioned scheduling
Thank You!!