ECRTS 7th-10th July 2015 Lund, Sweden

Static CRPD-Aware Real-Time Scheduling

ECRTS-WiP

Guillaume PHAVORIN, Pascal RICHARD
LIAS/Université de Poitiers
{guillaume.phavorin,pascal.richard}@univ-poitiers.fr

Claire MAIZA
Verimag/INP Grenoble
claire.maiza@imag.fr

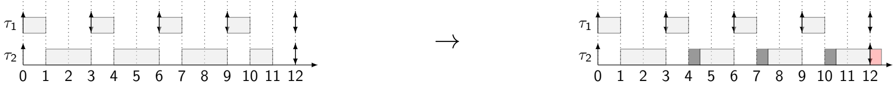
July 8th, 2015

Context

Actual platforms: CPUs + caches

$\Rightarrow$ **Cache-Related Preemption Delays (CRPD)**

$\rightarrow$ cannot be neglected



$\rightarrow$ WCET depends on tasks running concurrently (inter-task interference in the cache)

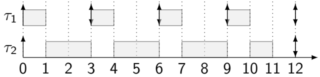
$\Rightarrow$ WCETs are not independent

Context

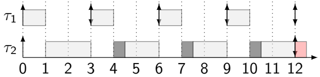
Actual platforms: CPUs + caches

⇒ **Cache-Related Preemption Delays (CRPD)**

→ cannot be neglected



→



→ WCET depends on tasks running concurrently (inter-task interference in the cache)

⇒ WCETs are not independent

- *preemption-aware scheduling*: preemption thresholds, deferred preemption, preemption points...
- CRPD-*aware scheduling* → scheduling decisions using CRPD values

CRPD-aware scheduling problem

$\tau_i(C_i, T_i, D_i, s_i)$

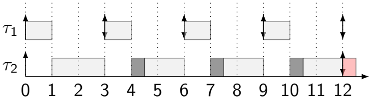
- additional task parameter $\rightarrow$ s_i: worst-case CRPD for τ_i
- C_i: WCET without CRPD

CRPD-aware scheduling problem

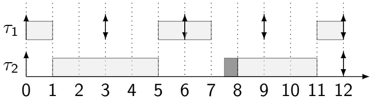
$\tau_i(C_i, T_i, D_i, s_i)$

- additional task parameter $\rightarrow$ s_i: worst-case CRPD for τ_i
- C_i: WCET without CRPD

$\rightarrow$ RM and EDF **not optimal**



$\rightarrow$

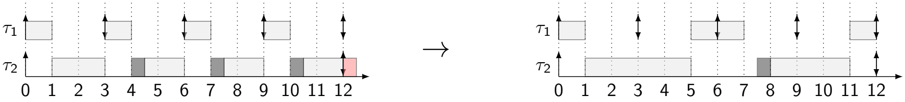


CRPD-aware scheduling problem

$\tau_i(C_i, T_i, D_i, s_i)$

- additional task parameter $\rightarrow$ s_i: worst-case CRPD for τ_i
- C_i: WCET without CRPD

$\rightarrow$ RM and EDF **not optimal**



$\rightarrow$ **NP-hard** in the strong sense

A solution using static scheduling

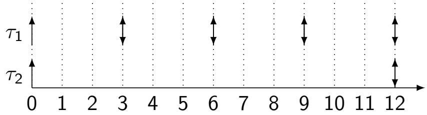
- static scheduling $\rightarrow$ schedule computed offline

A solution using static scheduling

- static scheduling → schedule computed offline
- mathematical problem → Mixed-Integer Linear Program (MILP)
  - schedule → set of slices delimited by releases and deadlines

$\tau_1(1, 3, 3, 0.25)$	$J_1(0, 1, 3, 0.25)$	$J_{1,1} \in [0, 3)$
	$J_2(3, 1, 6, 0.25)$	$J_{2,2} \in [3, 6)$
	$J_3(6, 1, 9, 0.25)$	$J_{3,3} \in [6, 9)$
	$J_4(9, 1, 12, 0.25)$	$J_{4,4} \in [9, 12)$
$\tau_2(7, 12, 12, 0.5)$	$J_5(0, 7, 12, 0.5)$	$J_{5,1} \in [0, 3)$
		$J_{5,2} \in [3, 6)$
		$J_{5,3} \in [6, 9)$
		$J_{5,4} \in [9, 12)$

→



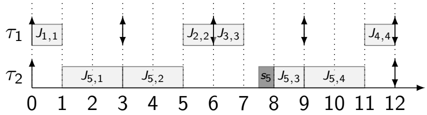
A solution using static scheduling

- static scheduling $\rightarrow$ schedule computed offline
- mathematical problem $\rightarrow$ Mixed-Integer Linear Program (MILP)
  - schedule $\rightarrow$ set of slices delimited by releases and deadlines

    $\rightarrow$ jobs scheduled in slices $\rightarrow$ job-pieces

$\tau_1(1,3,3,0.25)$	$J_1(0,1,3,0.25)$ $J_2(3,1,6,0.25)$ $J_3(6,1,9,0.25)$ $J_4(9,1,12,0.25)$	$J_{1,1} \in [0,3)$ $J_{2,2} \in [3,6)$ $J_{3,3} \in [6,9)$ $J_{4,4} \in [9,12)$
$\tau_2(7,12,12,0.5)$	$J_5(0,7,12,0.5)$	$J_{5,1} \in [0,3)$ $J_{5,2} \in [3,6)$ $J_{5,3} \in [6,9)$ $J_{5,4} \in [9,12)$

$\rightarrow$



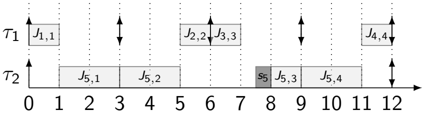
A solution using static scheduling

- static scheduling → schedule computed offline
- mathematical problem → Mixed-Integer Linear Program (MILP)
  - schedule → set of slices delimited by releases and deadlines

    → jobs scheduled in slices → job-pieces

$\tau_1(1,3,3,0.25)$	$J_1(0,1,3,0.25)$	$J_{1,1} \in [0,3)$
	$J_2(3,1,6,0.25)$	$J_{2,2} \in [3,6)$
	$J_3(6,1,9,0.25)$	$J_{3,3} \in [6,9)$
	$J_4(9,1,12,0.25)$	$J_{4,4} \in [9,12)$
$\tau_2(7,12,12,0.5)$	$J_5(0,7,12,0.5)$	$J_{5,1} \in [0,3)$
		$J_{5,2} \in [3,6)$
		$J_{5,3} \in [6,9)$
		$J_{5,4} \in [9,12)$

→



  - objective function → minimize the overall CRPD
  - under several constraints:
    - every job-piece executed in its slice
    - only one job-piece executed at a time
    - CRPD when another job-piece scheduled between 2 job-pieces of a same job

→ full MILP in the paper

Future works

- evaluate on-line scheduling policies (RM,EDF...)
  - loss of schedulability
  - number of preemptions
- extend the mathematical problem to take into account improved cache analysis (preempted + preempting tasks)
- devise online heuristics scheduling algorithms

Future works

- evaluate on-line scheduling policies (RM,EDF...)
  - loss of schedulability
  - number of preemptions
- extend the mathematical problem to take into account improved cache analysis (preempted + preempting tasks)
- devise online heuristics scheduling algorithms

More information around the poster