Semi-Clairvoyance in Mixed-Criticality Scheduling

Kunal Agrawal & Sanjoy Baruah
Washington University in St. Louis

Alan Burns
The University of York

BACKGROUND

In **mixed-criticality scheduling theory**

- each job is characterized as being of either **high criticality** or **low criticality**
- it is assumed that each job is characterized by **two WCET parameters**: a low-criticality one and a high-criticality one

A **run-time algorithm** is defined to be **correct** if it

- meets **all deadlines** if each job completes execution within its low-criticality WCET, and
- meets **all high-criticality deadlines** if each job completes execution within its high-criticality WCET

Two different kinds of run-time algorithms have been considered:

- **Clairvoyant** algorithms know **beforehand** whether any job is going to exceed its low-criticality WCET or not
- **On-line** algorithms are non-clairvoyant: they only learn whether any job will exceed its low-criticality WCET upon some job actually doing so

Clairvoyance is an **abstract idealization** that cannot be realized in practice; it provides a basis against which the performance of (actual) on-line algorithms may be compared

IN THIS PAPER…

We **introduced the notion** of **semi-clairvoyance** for run-time algorithms

- A semi-clairvoyant run-time algorithm knows whether a job will exceed its low-criticality WCET **at the instant that the job arrives** at the scheduler
- Whereas clairvoyance is an abstraction, we identified conditions under which **semi-clairvoyance is realizable in practice**

We **derived an algorithm**, **LPSC***, for scheduling collections of dual-criticality **jobs**

- LPSC has **polynomial run-time** and a **speedup factor of 1.5** when compared to an idealized clairvoyant scheduler (We have also shown that this is the smallest possible speedup factor)
  - No non-clairvoyant on-line algorithm for jobs can have a speedup factor smaller than 1.61
  - Hence semi-clairvoyance **strictly dominates** non-clairvoyance

We **derived an algorithm** for scheduling collections of dual-criticality **tasks**

- This algorithm is **optimal**, and has **polynomial running time**
  - No non-clairvoyant on-line algorithm for tasks with speedup factor smaller than 4/3 is known

---

*LPSC stands for ***L****inear-****P****rogramming based ****S****emi-****C****lairvoyant scheduler*

Semi-Clairvoyance in Mixed-Criticality Scheduling