Open Problems in Multi-Modal Scheduling Theory for Thermal-Resilient Multicore Systems

Nathan Fisher Masud Ahmed Pradeep M. Hettiarachchi

Department of Computer Science, Wayne State University, Detroit, Michigan, USA
{fishern, masud, pradeepmh}@wayne.edu

I. Introduction

Recent research on thermal-aware real-time systems has focused upon developing design frameworks that ensure predictable timing behavior for systems executing in dynamic thermal environments. For such dynamic environments, a *thermally-resilient real-time system* [4] must adapt both its timing requirements and thermal-dissipation rates in response to a changing environmental temperature. On the hardware side, thermal dissipation rates might potentially be reduced via dynamic voltage/frequency scaling (DVFS) and/or dynamic power management (DPM) techniques such as putting the CPU into low-power/idle states. On the software side, the adaptation of timing requirements can be achieved through *multi-modal tasks systems*, providing the means to change task timing parameters (e.g., execution requirement, period, deadline, etc.) and/or to add/drop tasks from the system. The ultimate goal for thermal-resilient real-time systems is the capability of a real-time system designer to accurately predict (at design time) what real-time mode can be supported for any given environmental temperature.

Our initial work on the development of a design framework for thermal-resilient real-time systems focused on single-core systems (see Hettiarachchi et al. [4]). To achieve both adaptivity and predictability in our design framework, we introduced the concept of a *real-time performance mode* which explicitly coupled hardware and software modes together. The hardware mode was characterized by the *periodic resource model* [12] which can quantify the execution supply of a non-continuously executing resource (e.g., a processor that switches from an active to an inactive state periodically to change the thermal dissipation of the CPU). Each software mode was characterized by a *sporadic task system* [7]. In complementary work, we developed single-core schedulability analysis for the real-time performance mode model [2] specified by this periodic-resource/sporadic-task-system combination; thus, the resulting framework is able to provide hard-real-time guarantees.

The above design framework has subsequently been extended to multicore platforms [5]. However, a restrictive limitation is that currently the framework supports only partitioned scheduling of tasks for each mode. The reason for this limitation is that ***there is a fundamental gap in multi-mode scheduling theory for multicore platforms with non-continuous execution patterns***. As our thermal-resiliency framework (and thermal-aware real-time systems, in general) requires schedulability analysis for dynamic processing platforms that utilize DVFS/DPM, it is important that new analysis techniques be developed that will permit these systems to implement a larger set of the global-scheduling algorithms. In this abstract, we will briefly describe the real-time performance mode (i.e., HW/SW-coupled multi-modal) model in the context of global scheduling, outline important scheduling theory questions for this model, and review relevant prior research.

II. Real-Time Performance Mode Model for Globally-Scheduled Systems

We consider a system with m number of identical CPU cores. The set of real-time performance modes is denoted $\mathcal{M}$. Each real-time performance mode $M^{(i)} \in \mathcal{M}$ is characterized by a three-tuple $(\tau^{(i)}, \Omega^{(i)}, \Delta^{(i)})$. The first parameter $\tau^{(i)}$ denotes a sporadic task system with n_i tasks. Each sporadic task $\tau_j^{(i)} \in \tau^{(i)}$ is characterized by a three-tuple $(e_j^{(i)}, d_j^{(i)}, p_j^{(i)})$ where $e_j^{(i)}$ is the *worst-case execution requirement*, $d_j^{(i)}$ is the *relative deadline*, and $p_j^{(i)}$ is the *minimum inter-arrival separation parameter*. The first job of $\tau_j^{(i)}$ may arrive at any time after mode $M^{(i)}$ is activated.

The second parameter of the real-time performance mode $\Omega^{(i)}$ denotes the resource parameters (i.e., the execution supply guaranteed by the underlying multicore platform). In our previous work, we used a periodic resource $(\Pi^{(i)}, \Theta^{(i)})$ to represent the periodic active/idle times of each core. The term $\Theta^{(i)}$ represents the minimum resource active durations that is guaranteed in periodic $\Pi^{(i)}$ length intervals when the system is operating in mode $M^{(i)}$ mode. However, this model is not expressive enough or flexible enough for global scheduling. Thus, for globally-scheduled systems, $\Omega^{(i)}$ may be expressed in a more appropriate compositional scheduling model. Potential candidate models for $\Omega^{(i)}$ include the *parallel-supply function* (PSF) abstraction [1] and the *multiprocessor periodic resource* (MPR) [11] model. As an example, in the MPR model, $\Omega^{(i)}$ can be characterized by a three-tuple $(\Pi^{(i)}, \Theta^{(i)}, m^{(i)})$ where the MPR guarantees a total of $\Theta^{(i)}$ units of execution in each $\Pi^{(i)}$-length period with maximum level of parallelism of $m^{(i)}$ (i.e., at any given time there are at most $m^{(i)}$ processors executing tasks of $\tau^{(i)}$).

The third parameter $\Delta^{(i)}$ denotes the minimum amount of time that the processor will remain in mode $M^{(i)}$. The motivation for this parameter comes from discrete control where changes in a system (plant) occur only at discrete intervals of time. In our thermal-resilient design framework, the mode changes occur only at these sampling intervals. In fact, it is convenient and often the case that $\Delta^{(i)}$ can be set as a multiple of the resource period $\Pi^{(i)}$.

This work has been supported in part by the US National Science Foundation (Grant Nos. CNS-0953585 and CNS-1205338).

§**Mode-Change Semantics.** A *mode-change request* $\mathsf{mcr}_k \equiv (M^{(i)}, M^{(j)}, t_k)$ consists of *transition time* (t_k), the *old mode* $M^{(i)}$ executing prior to t_k, and the new mode $M^{(j)}$ executing after t_k (where $i, j \in \{1, \ldots, q\}$). We assume that if $i < j$ then mcr_i occurs prior to mcr_j. For $\mathsf{mcr}_{k-1} \equiv (M^{(h)}, M^{(i)}, t_{k-1})$ and $\mathsf{mcr}_k \equiv (M^{(i)}, M^{(j)}, t_k)$, the difference $t_k - t_{k-1}$ must be at least $\Delta^{(i)}$ to ensure the minimum activation time for mode $M^{(i)}$. Figure 1 illustrates a mode-change-request sequence.

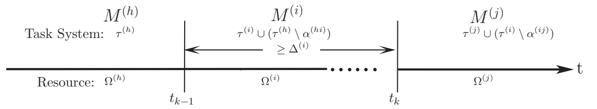
$M^{(h)}$
Task System: $\tau^{(h)}$
$M^{(i)}$
$\tau^{(i)} \cup (\tau^{(h)} \setminus \alpha^{(hi)})$
$\geq \Delta^{(i)}$
$M^{(j)}$
$\tau^{(j)} \cup (\tau^{(i)} \setminus \alpha^{(ij)})$
t
Resource: $\Omega^{(h)}$
$\Omega^{(i)}$
$\Omega^{(j)}$
t_{k-1}
t_k

Fig. 1. Mode-change sequence.

Tasks may be divided into groups based on their importance at the time of the mode change request $\mathsf{mcr}_k \equiv (M^{(i)}, M^{(j)}, t_k)$. (See Real and Crespo [10] for complete taxonomy). Some important tasks may need to continue to execute without being affected by the mode change request. We call these tasks *unchanged tasks* denoted by $\tau^{(ij)}$. Some less important tasks may be removed from the system immediately at the time of the mode change request. We call these tasks *aborted tasks* and denote them by $\alpha^{(ij)}$. For some tasks, immediate termination may leave the system in an inconsistent state; we call such tasks *finished tasks* and characterize them as being members of the set $\tau^{(i)} \setminus (\alpha^{(ij)} \cup \tau^{(ij)})$. We allow a job from a finished task at the time of a mode change request to complete its remaining execution. Given the above definitions, we may distinguish three phases with respect to a mode-change request $\mathsf{mcr}_k = (M^{(i)}, M^{(j)}, t_k)$ (from the previous request $\mathsf{mcr}_{k-1} = (M^{(h)}, M^{(i)}, t_{k-1})$):

1) $[t_{k-1}, t_k)$: jobs of $\tau^{(i)}$ are executed upon $\Omega^{(i)}$;
2) $[t_k, t_{k+1})$: incomplete jobs of $(\tau^{(i)} \setminus \alpha^{(ij)})$ at t_k and jobs of $\tau^{(j)}$ execute upon $\Omega^{(j)}$;
3) $[t_{k-1}, t_{k+1})$: unchanged tasks $(\tau^{(ij)})$ will act independent of mode change request.

III. Open Problems & Related Work

Informally, our main open problem statement is simple to state: *given an m-core platform and multi-modal system $\mathcal{M}$ determine whether all tasks of every modes will always meet all deadlines under any possible legal mode-change sequence*. However, there are many dimensions to this problem that make it possible to define a “family” of related subproblems. In particular, we may consider the following different aspects of the system:

D1 *Global Scheduling Algorithm*: schedulability analysis is needed for both global earliest-deadline-first and fixed-priority scheduling algorithms to be applicable for real systems that implement the thermal-resiliency framework.

D2 *Multiprocessor Resource Model*: As mentioned in the previous section, we can consider different models such as PSF or MPR for expressing the $\Omega^{(i)}$ parameter.

In addition to the schedulability analysis problem, an important variant problem is determining the minimum allocation for each mode to maintain schedulability. Some open questions for the allocation problem are: 1) What is the most appropriate definition of “minimum allocation” for thermal-resilient systems? (e.g., if $\Omega^{(i)}$ is an MPR, is minimizing the sum of the capacities a good objective to decrease total thermal dissipation?); and 2) How do we efficiently deal with the combinatorial nature of the problem? (I.e., there might be numerous combinations of capacities that are schedulable).

§**Relevant Prior Work.** Extensive research effort has been focused on developing multi-modal schedulability analysis for multicore platforms (e.g., see [3], [6], [8], [9]). To the best of our knowledge, each of these prior works assumes the underlying processing cores execute at a fixed speed/rate. However, we hope that these multi-modal results for fixed-rate multicore platforms might be extendable to platforms with DVFS/DPM capabilities to solve the open problems posed in this abstract. A starting point might be trying to extend the multi-modal analysis for uniform multiprocessors [13] to real-time performance modes.

References

[1] E. Bini, M. Bertogna, and S. Baruah. Virtual multiprocessor platforms: Specification and use. In *Proceedings of the 30th IEEE Real-Time Systems Symposium*, pages 437–446, Dec 2009.

[2] N. Fisher and M. Ahmed. Tractable real-time schedulability analysis for mode changes under temporal isolation. In *Proceedings of the 9th IEEE Symposium on Embedded Systems for Real-Time Multimedia (ESTImedia)*. IEEE Computer Society, October 2011.

[3] J. Goossens and P. Richard. Partitioned scheduling of multimode multiprocessor real-time systems with temporal isolation. In *Proceedings of the 21st International Conference on Real-Time Networks and Systems*, pages 297–305, 2013.

[4] P. M. Hettiarachchi, N. Fisher, M. Ahmed, L. Y. Wang, S. Wang, and W. Shi. The design and analysis of thermal-resilient hard-real-time systems. In *IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 67–76, 2012.

[5] P. M. Hettiarachchi, N. Fisher, and L. Y. Wang. Achieving thermal-resiliency for multicore hard-real-time systems. In *Proceedings of the 25th Euromicro Conference on Real-Time Systems*, pages 37–46, July 2013.

[6] J. Lee and K. G. Shin. Schedulability analysis for a mode transition in real-time multi-core systems. In *Proceedings of the 34th IEEE Real-Time Systems Symposium*, pages 11–20, Dec 2013.

[7] A. K. Mok. *Fundamental Design Problems of Distributed Systems for The Hard-Real-Time Environment*. PhD thesis, Laboratory for Computer Science, Massachusetts Institute of Technology, 1983. Available as Technical Report No. MIT/LCS/TR-297.

[8] V. Nelis, B. Andersson, J. Marinho, and S. M. Petters. Global-edf scheduling of multimode real-time systems considering mode independent tasks. pages 205–214, 2011.

[9] V. Nelis, J. Goossens, and B. Andersson. Two protocols for scheduling multi-mode real-time systems upon identical multiprocessor platforms. In *Proceedings of the 21st Euromicro Conference on Real-Time Systems*, July 2009.

[10] J. Real and A. Crespo. Mode change protocols for real-time systems: A survey and a new proposal. *Real-Time Syst.*, 26:161–197, March 2004.

[11] I. Shin, A. Easwaran, and I. Lee. Hierarchical scheduling framework for virtual clustering of multiprocessors. In *Proceedings of the Euromicro Conference on Real-Time Systems*, pages 181–190, July 2008.

[12] I. Shin and I. Lee. Compositional real-time scheduling framework with periodic model. *ACM Transactions on Embedded Computing Systems*, 7(3), April 2008.

[13] P. M. Yomsi., V. Nelis, and J. Goossens. Scheduling multi-mode real-time systems upon uniform multiprocessor platforms. In *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation*, pages 1–8, Sept 2010.