Science of Computer Programming 113 (2015) 236–260

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

A reconfigurable framework for compositional schedulability and power analysis of hierarchical scheduling systems with frequency scaling ☆

Abdeldjalil Boudjadar [a], Alexandre David [b], Jin Hyun Kim [c], Kim G. Larsen [b], Marius Mikučionis [b], Ulrik Nyman [b,*], Arne Skou [b]

[a] Linköping University, Sweden
[b] Computer Science, Aalborg University, Denmark
[c] INRIA/IRISA, Rennes, France

ARTICLE INFO

Article history:
Received 30 June 2014
Received in revised form 25 September 2015
Accepted 7 October 2015
Available online 22 October 2015

Keywords:
Hierarchical scheduling systems
Schedulability analysis
Power consumption
Voltage/frequency scaling
UPPAAL

ABSTRACT

This paper presents a compositional framework for the modeling and analysis of hierarchical scheduling systems. We consider both schedulability and energy consumption of individual components, while analyzing a single core setting with a voltage frequency scaling CPU. According to the CPU frequency scaling, each task has a set of different execution times. Thus, the energy consumption of the whole system varies from one execution to another.

We analyze each component individually by checking the feasibility of its workload against both the CPU availability and energy consumption constraints of such a component. Our periodic task model considers both static and dynamic priorities together with preemptive and non-preemptive behaviors. The models are realized using different forms of Hybrid Automata, all of which are analyzed using variants of UPPAAL. The CPU frequencies, task behavior and scheduling policies used in each component are some of the reconfigurable parameters of the system. Finally, we demonstrate the applicability and scalability of our framework by analyzing the schedulability and power consumption of an avionics system.

1. Introduction

In the design of modern automotive systems, in order to reduce the system cost a manufacturer devotes strong efforts in minimizing the resource requirements of individual components, provided by different suppliers, in order to maximize the number of components to be integrated on a given platform while ensuring the whole system to be continually feasible. Such concurrent components might share platform resources (e.g. processors, battery).

Resource utilization represents a common challenge in embedded systems, and thus it is important to have, for each individual part of the system, a scheduling policy which is both efficient and reliable. Scheduling is a widely used mechanism for guaranteeing that the different components of a system will be provided with the correct amount of resources. Many classical approaches have been developed to analyze different types of scheduling systems. The working hypothesis behind

☆ The research presented in this paper has been partially supported by EU Artemis Projects CRAFTERS and MBAT.

* Corresponding author.
E-mail addresses: abdeldjalil.boudjadar@liu.se (A. Boudjadar), adavid@cs.aau.dk (A. David), jin-hyun.kim@inria.fr (J.H. Kim), kgl@cs.aau.dk (K.G. Larsen), marius@cs.aau.dk (M. Mikučionis), ulrik@cs.aau.dk (U. Nyman), ask@cs.aau.dk (A. Skou).

http://dx.doi.org/10.1016/j.scico.2015.10.003
0167-6423/

the research presented in this paper is that a model-based approach to schedulability and power analysis has advantages of being more flexible than the classical analytical approaches in terms of expressiveness and precision.

In this paper, we propose a compositional model-based framework for analyzing the schedulability and power consumption of hierarchical scheduling systems. Our framework is implemented using different forms of stopwatch timed automata which are analyzed using variants of Uppaal. In order to capture the effect of preemption in hierarchical scheduling systems, we are using the concept of stopwatches. Stopwatches generally present a problem for the exact system analysis as they belong to the undecidable fragment of hybrid automata. However, since only expressions depending on the discrete part of the state can be assigned to a clock or stopwatch in Uppaal, this ensures the decidability of the system (for further details see Section 3.5).

We have not found any alternative way to model preemptive hierarchical scheduling systems, with a dense time semantics, using standard timed automata. We believe that it is indeed impossible, but have not tried to prove this fact. According to [30], the schedulability checking problem for non-preemptive scheduling policies is decidable because the schedulability question can be translated to a reachability question. Moreover, such a problem is also decidable in case of preemptive algorithms if the computation times are constant single values. Otherwise, the schedulability checking is undecidable if the following holds:

- the execution times of tasks are intervals given in terms of best and worst cases.
- the finishing time of a task execution (instance) influences the release of new instances.
- a ready task is allowed to preempt a running task.

A proof of the decidability of FPS (Fixed Priority Scheduling) and EDF (Earliest Deadline First) is presented in [30].

Model checking in general suffers from state space explosion, where the state space that needs to be explored grows exponentially in the size of the parallel composition. In order to combat the state space explosion and be able to use model checking on larger problems, we have chosen to decompose the models. This has, in many other settings, proven to be fruitful way of applying model checking to real problems. The size of the final system is unimportant, but rather the maximum number of tasks that needs to be analyzed together. Thus our framework can be scaled to larger systems if the system tasks are grouped into components in a hierarchy.

A hierarchical scheduling system consists of a finite set of components, a scheduling policy and resources (energy and processor time). Each component, in turn, is the parallel composition of a finite set of entities which are either tasks or other components together with a scheduling policy to manage the component workload. Tasks are instances of the same timed automaton template with different (input) parameters. Thanks to the parameterization, the framework can easily be instantiated for a specific hierarchical scheduling application. Similarly, each scheduling policy (e.g. EDF: Earliest Deadline First, FP: Fixed Priority, RM: Rate Monotonic) is modeled separately and can be instantiated for any component. We extend our framework with power consumption by adding a new constraint attribute to the component interface. For energy awareness, we analyze the energy consumed by a component workload against the power consumption constraint of that component. In this way a fixed frequency for each task is found on a voltage/frequency scaling platform, for which the component is both schedulable and satisfies the maximum power constraint. For the sake of simplicity, we focus on single core systems but the framework is also applicable for the case of multi-core systems.

Compositional analysis was first introduced [18,33] as a key model-checking technology, to deal with state space explosion caused by the parallel composition of components. We are applying compositional verification to the domain of schedulability analysis. By schedulability analysis, we mean checking whether a set of real-time tasks can be scheduled without missing any deadline.

We analyze the model in a compositional manner by layers/levels; the schedulability of each level is analyzed together with the interface specifications of the level directly below it. In this analysis, we non-deterministically supply the required resources of each component. This fact is viewed by the component entities as a contract by which the component must supply the required resources, provided by the parent level component, to its sub-entities for each period. The main contribution of the paper is combining:

- *a compositional analysis approach* where the schedulability of a system relies on the recursive schedulability analysis of its individual subsystems.
- *energy consumption analysis* of components on a voltage/frequency scaling platform where the energy consumed by the workload is analyzed against the component energy constraint.
- *an adaptable schedulability framework* where a system structure can be instantiated in different configurations to fit different applications.
- *error traces*: we demonstrate the usefulness of model-based approaches by showing that error traces can be used to diagnose why a system is non-schedulable.
- *formal basis*: we describe the specification and semantics of our framework in terms of transition systems.

This paper is an extended version of [12] containing detailed background and extended related work. We also give a thorough introduction and overview of the formal basis of the modeling formalisms we use.