Degree of Schedulability of Mixed-Criticality Real-time Systems with Probabilistic Sporadic Tasks

Boudjadar, Jalil; David, Alexandre; Kim, Jin Hyun; Larsen, Kim Guldstrand; Nyman, Ulrik; Skou, Arne; Mikučionis, Marius

Published in:
Theoretical Aspects of Software Engineering Conference (TASE), 2014

DOI (link to publication from Publisher):
10.1109/TASE.2014.27

Publication date:
2014

Document Version
Accepted author manuscript, peer reviewed version

Degree of Schedulability of Mixed-Criticality Real-time Systems with Probabilistic Sporadic Tasks

A.Jalil Boudjadar, Alexandre David, Jin Hyun Kim, Kim G. Larsen, Marius Mikučionis, Ulrik Nyman, Arne Skou
Institute of Computer Science, Aalborg University, Denmark

***Abstract*—We present the concept of degree of schedulability for mixed-criticality scheduling systems. This concept is given in terms of the two factors 1) Percentage of Missed Deadlines (PoMD); and 2) Degradation of the Quality of Service (DoQoS). The novel aspect is that we consider task arrival patterns that follow user-defined continuous probability distributions. We determine the degree of schedulability of a single scheduling component which can contain both periodic and sporadic tasks using statistical model checking in the form of UPPAAL SMC. We support uniform, exponential, Gaussian and any user-defined probability distribution.**

I. INTRODUCTION

When constructing an embedded software system engineers could be interested, not only, in whether or not the system always meets its requirements, but also how it behaves with insufficient resources. For mixed-criticality systems in can be of special importance to know how the quality of service is degraded for the soft-real time parts if they are provided with less resources than needed. A certain level of degradation may be acceptable in a given setting and we thus consider it important to answer questions regarding schedulability with estimates of the quality instead of just providing a yes/no answer.

This paper presents the degree of schedulability as a new concept to measure the degradation of the quality of service of a given system. The degree of schedulability is given in terms of two metrics; the *Percentage of Missed Deadlines* and the average delay per missed deadline, called *Degradation of Quality of Service*.

A hierarchical scheduling system [10] is a component-based system encompassing global resources shared between the system components. The system workload consists of a set of tasks declared with a set of timing attributes such as period, deadline and execution time.

The sporadic task model [3] [15] has received research attention [22] [21] over the years because of its usefulness in modeling recurring processes for hard-real-time systems.

It is obvious that it is not possible to guarantee the schedulability of real-time systems where sporadic tasks can occur arbitrarily frequently [3]. To alleviate this the sporadic task model operates with a minimal inter-arrival time. In many mixed-critical systems like tracking systems and automotive info-tainment systems, arrival times are adequately described by specific distribution functions. In this paper, we model real world events separately from the tasks. Events are modeled according to stochastic arrival patterns with an inter-arrival time that could be zero.

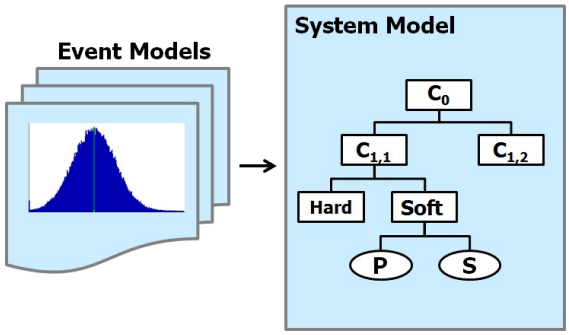


Fig. 1. Overview of framework setup.

We consider a framework where the triggering mechanism of sporadic tasks is located outside the hierarchical scheduling system as illustrated in Fig. 1. Each sporadic task has its individual arrival pattern, which is characterized by a continuous probability distribution. In this paper, we are focusing on modeling and analyzing a single component at a time, with both sporadic and periodic real-time tasks. This analysis method fits within a larger compositional analysis framework [4], which analyzes a complete hierarchical scheduling system. Our main contributions are:

- We introduce continuous probability distributions to model sporadic events that trigger the execution of sporadic tasks.
- We study the system schedulability and determine the *degree of schedulability* (Sched°) in terms of the *Percentage of Missed Deadlines* (PoMD); and average delay per missed deadline, called *Degradation of Quality of Service* (DoQoS).
- We provide a framework including explicit environment models, which allows us to model and analyze mixed-criticality hierarchical scheduling systems.

Compared to treating the minimal inter-arrival time as a period, our method aims at providing both more realistic and optimistic resource estimates for sporadic tasks. The rest of the paper is structured as follows: Section II cites relevant related work. Section III introduces the compositional analysis framework. In Sections IV, V and VI we introduce respectively continuous sporadic tasks, the models used to analyze them and the actual analysis. Finally, we conclude in Section VII.

II. RELATED WORK

In this section we present related work with a specific focus on sporadic tasks. To the best of our knowledge, there is no

previous related work which uses continuous probabilities to characterize the arrival patterns of sporadic tasks. The sporadic task model [3], [15], which is an extension of an earlier task model known as the Liu and Layland (LL) [12] task model has received immense research attention over the years. In [22], the authors propose a framework for the schedulability analysis of real-time systems, where they define a generalized model for sporadic tasks to characterize more precisely the task arrival times. The authors characterize each task by two constraints: *higher instantaneous arrival rate* which bounds the maximum number of task arrivals during some small time interval; *lower average arrival rate* which is used to specify the maximum number of arrivals over some longer time interval. In [21], the authors propose a method to control the preemptive behavior of real-time sporadic task systems by the use of CPU frequency scaling. They introduced a new sporadic task model in which the task arrival may deviate, according to a *discrete* time probability distribution, from the minimum inter-arrival time. In fact, a task arrival T may deviate with a delay t if the probability of T to occur at instant t is greater than a certain threshold. Based on the probability of arrivals, the authors propose an on-line algorithm computing CPU frequencies that guarantee non-preemptiveness of task behavior while preserving system schedulable. In [3], the authors propose an exact schedulability analysis by providing some necessary and sufficient conditions for a sporadic task system to be schedulable. In fact, the authors consider sporadic tasks with minimum inter-arrival time as periodic tasks, then define the set of legal requests that a task may perform. Based on such a function, they analyze the system schedulability regardless of the schedulability policy. However, considering sporadic tasks with known minimum inter-arrival times as periodic tasks may lead the schedulability analysis to be pessimistic and seriously overestimates the number of task arrivals. Our work differs by modeling probabilistic inter-arrival times and quantifying the system schedulability according to hard and soft real-time requirements. A concept similar to PoMD as introduced in this paper is given in [14]. The term "degree of schedulability" was first introduced in [16] to characterize the sum of response time delays from the individual deadlines. We define the concept DoQoS in a similar way, but focus on the total amount of time by which deadlines are missed. We define our notion of *degree of schedulability* (Sched$^\circ$) by combining PoMD and DoQoS into one measure.

III. Compositional Analysis Framework for Sched$^\circ$

A hierarchical scheduling system [1] consists of a set of concurrent real-time components sharing a set of resources according to a scheduling policy. Each component can again be internally organized as a set of components, giving the organization of the system a tree like structure. The use of temporal partitioning [18] between the components is motivated by the fact that it provides reduction of complexity, separation of concerns, confinement of failure modes, and temporal isolation among system applications. One obvious partitioning of the components in a mixed criticality system is to group them according to their criticality. Such a grouping enables easier certification of the safety critical components when they have minimal communication with the non safety critical parts [17].

In this paper we focus on the schedulability analysis of one component inside a hierarchical scheduling system. Formally, a hierarchical scheduling system $S = (C, R, A)$ is given by a set of hierarchical components C, a set of resources R and a scheduling algorithm A. A component, in turn, can be either a hierarchical unit $(\{C_1, .., C_n\}, A)$ of other components C_i, or a basic composition (W, A) of a workload W, together with a scheduling policy A. The workload W is a set of real-time tasks having time constraints like deadline, execution time and next arrival. The real-time interface $\mathcal{I}$ [20] of a component $C(W, A)$ specifies the collective resource requirements that the workloads W performs under the scheduling policy A. $\mathcal{I}$ is simply given by a period p and a budget b in our framework.

In a compositional schedulability analysis framework [7], [4], a hierarchical system is said to be schedulable if each component is schedulable.

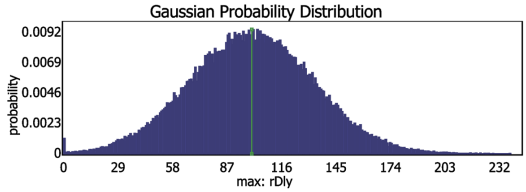
The analytical analysis approaches [11], [3], [13] compute whether or not a system is schedulable, according to a scheduling policy, by giving a firm response to the following question: is the demand bound function $\mathtt{dbf}$ of each component workload W, over a time interval t, lower or equal to the supply bound function $\mathtt{sbf}$ of a resource according to interface $\mathcal{I}$, over the same time interval, i.e. $\forall t > 0 \ \mathtt{dbf}_A(W, t) \leq \mathtt{sbf}_{\mathcal{I}}(t)$. If such an equation is satisfied, the component is said to be schedulable. In the same way, in a model-based setting [21], [2], [9], [4] a system is said to be schedulable if the error locations, stating the deadline violation, are unreachable.

In contrast to the mentioned techniques, we do not only consider if a system is schedulable or not, but we provide the *degree of schedulability* (Sched$^\circ$) as a way to measure how schedulable a system is. We define the Sched$^\circ$ of an entity (system, component or task) by the two concepts: *Percentage of Missed Deadlines* (PoMD) and *Degradation of Quality of Service* (DoQoS). Each of these concepts can be computed for either a task, a component or a complete embedded system. They should be measured or simulated over a sufficiently large time bounded run and a sufficiently large number of runs in order to obtain usable values.

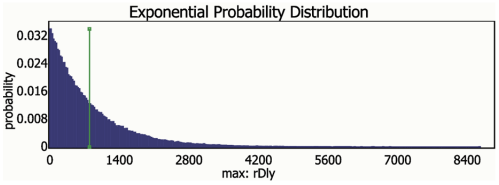
The PoMD is computed over a set of traces as the number of missed deadlines divided by the number of triggerings of the associated tasks. The DoQoS is defined as 0 if no deadline is missed, otherwise it is the accumulation of overruns divided by the number of missed deadlines. Thus, representing the average overrun when a deadline is missed.

IV. Continuous Probability Based Sporadic Tasks

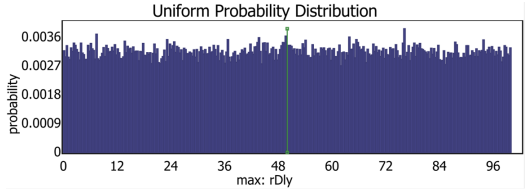
In this section, we introduce the characteristics of the probability-based sporadic tasks. Our framework models both a fixed inter-arrival time and a probability distribution. Obviously, a task cannot arrive before the inter-arrival time, and the inter-arrival time can potentially be set to zero. After the inter-arrival time, the arrival of a given task delays with δ according to a continuous probability distribution, such as Gaussian $\mathcal{N} = (\mu, \sigma^2)$ with a mean value μ and a variance σ^2 (Fig. 2(a)). Fig. 2 shows the three specific probability



(a) Gaussian (100,100)



(b) Exponential (1/800)



(c) Uniform (1,100)

Fig. 2. Probabilistic arrival patterns.

distributions we consider in our setting: Gaussian, exponential and uniform. As the probability distribution is a parameter of the sporadic tasks in our framework, any user defined probability distribution can be used.

A. Probability Distributions

We have implemented the continuous probability distributions we consider via a set of UPPAAL embedded functions over the time domain. An example of a Gaussian normalized curve, generated by UPPAAL SMC, is depicted in Fig. 2(a) where the x axis represents continuous time from 0 to 100, μ=50 , and σ =50. Fig. 2(b) shows an exponential probability distribution, with the rate of exponential λ being $\frac{1}{800}$. The smaller λ is, the more spread out the distribution is. In contrast to the two previous probability distributions, the uniform distribution (Fig. 2(c)) has a equal probability for all time instances up to a maximum time where the probability drops to zero.

B. Conceptual Sporadic Task Model

Our conceptual event model is shown in Fig. 3(a). When the delay has elapsed, the event triggers the corresponding task and moves to the location InterArrivalWait waiting for one inter-arrival time I before starting a new round. The conceptual task model (Fig. 3(a)) starts in location Wait waiting for the triggering event (trigger?) and then moves to the location Run. Depending on whether or not the deadline of the task is missed, the task moves either directly to location Wait or to location MissedDeadline. If the task finishes its current execution after missing a deadline, the overrun will be measured (used for estimating the DoQoS) before moving to the location Wait. The conceptual models shown in Fig. 3 are

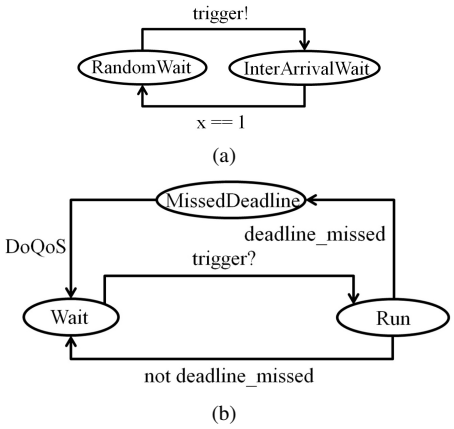


Fig. 3. Conceptual model of a sporadic task and its triggering event.

instantiated as UPPAAL models. Because of space limitations the UPPAAL models are not presented in this paper.

V. Analysis Models for the Degree of Schedulability

For our compositional analysis framework, the hierarchical scheduling systems and their analysis elements consist of environment models, scheduling models, resource model, and task models.

We are using UPPAAL SMC to perform a formalized statistical simulation of our models, known as Statistical Model Checking (SMC). SMC enables quantitative performance measurements instead of the Boolean (true, false) evaluation that symbolic model checking techniques provide. We can summarize the main features of UPPAAL SMC in the following:

- Stopwatches [8] are clocks that can be stopped and resumed without a reset. They are very practical to measure the execution time of preemptive tasks.
- Simulation and estimation of the expected minimum or maximum value of expressions over a set of runs, `E[bound](min:expr)` and `E[bound](max:expr)`, for a given simulation time and/or number of runs specified by `bound`.
- Probability evaluation `Pr[bound] (P)` for a property `P` to be satisfied within a given simulation time and/or number of runs specified by `bound`. `P` is specified using either LTL or tMITL logic.

The disadvantage of using statistical model checking is that it will not provide complete certainty that a property is satisfied, but only verify it up to a specific confidence level, given as an analysis parameter [6].

A. Resource model

The resource model of this paper is a periodic one, which provides a specific amount of resources to a set of tasks or components [19]. The resource model is given by a stochastic supplier, which supplies a resource allocation non-deterministically over supplier's period.

Fig. 4 shows supplier's supplying and tasks' running of T_3^p and T_4^s. In this setting, T_4^s has priority over T_3^p, and

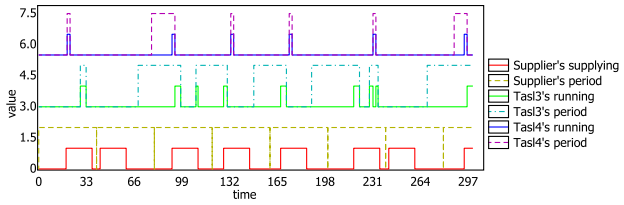


Fig. 4. Supplier and Task Execution.

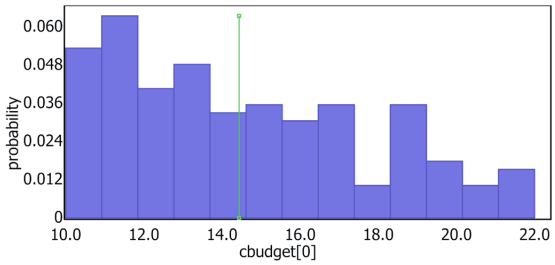


Fig. 5. Budgets causing deadline miss

executes sporadically over a uniform distribution. Thus, the execution period of T_4^s is irregular. The supplier at the bottom is supplying non-deterministically so the supplying is also irregular within the period. The detailed explanation can be found in our previous paper [4]. In order to estimate the sufficient budget of a supplier that makes the workload of a component schedulable, we present another stochastic supplier as shown in Fig. 5.(a). It starts supplying by selecting a random amount of budget using gbudget[supid] and cbudget[supid]. UPPAAL SMC checks whether any task misses deadline and generates a probability distribution of budgets leading to a deadline miss of a component. Fig. 5.(b) shows the estimated budget numbers that makes the component of T_3^p and T_4^s non-schedulable, and it can be concluded that 23 is the minimum budget for the component.

B. PoMD Calculator Model

For our framework, we provide 4 different task templates: hard real-time and soft real-time templates for periodic and sporadic tasks. The hard real-time task stops running immediately when it misses a deadline. Meanwhile, the soft real-time task continues to run until the end of simulation time while measuring PoMD and DoQoS. The variables cntExecution[tid], cntMissDline[tid], and DoQoS[tid] are used to calculate PoMD and DoQoS with PoMD calculator in Fig. 6 and the following queries:

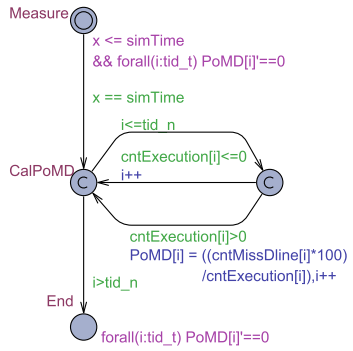


Fig. 6. PoMD calculator

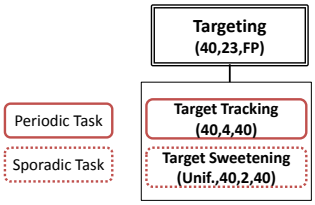


Fig. 7. Targeting component

$$E[gClock<=simTime;simNum] (max: PoMD[tid])$$
$$E[gClock<=simTime;simNum] (max: DoQos[tid])$$

Where tid is the id of the task to be analyzed.

For every simulation (simNum) of which time is up to simTime, one PoMD[tid] is obtained by calculating the percentage of the accumulated number of missed deadlines of cntMissDline[tid], and the count of task executions of cntExecution[tid]. Finally, PoMD is calculated from the average of PoMD[tid]s of all traces. DoQoS[tid] measures delay after a task misses a deadline, and the maximum DoQoS[tid] is selected from one trace. Finally, DoQoS is determined by calculating the average of the maximum DoQoS[tid]s of each individual simulation trace.

VI. ANALYSIS OF THE DEGREE OF SCHEDULABILITY

We use as a running example in this section the Targeting component of Fig. 7. The workload is characterized by a periodic task $T_3^p(40, 4, 40)$ and a sporadic task $T_4^s(Unif., 40, 2, 40)$. In our setting, T_4^s has priority over T_3^p. Both tasks are scheduled according to the fixed priority scheduling (FPS) policy. The sporadic task T_4^s follows the uniform probability distribution between 0 and 20 time units. The analysis is performed in the following steps:

1) Estimate a budget for the component as described in Section V.
2) Analyze the Sched$^\circ$ for the estimated and lower budgets.

To estimate the budget of a component, we use the budget estimation technique described in [4]. As a result, we found that 23 time units every 40 time units is a good candidate as a sufficient budget for both tasks. In order to have valid results, in the next analysis section we perform experiments where we analyze the same system with a varying amount of traces and simulation time. When reaching more than 1,000 traces and a simulation time of more than 100,000 time units we see that the results stabilize.

A. Analysis Results

In Table I, we show that T_3 and T_4 are schedulable under the budget (40, 23) even if T_4 is treated as a periodic task with a period equal to the minimal inter-arrival time. This is classical worst-case budget estimation, and our analysis also

TABLE I
THE DEGREE OF SCHEDULABILITY OF TASKS UNDER PERIODIC EVENTS

Component ((40, 23), FPS)	PoMD	DoQoS
T_3^p(40, 4), T_4^s(40, 2),	0 0	0 0

TABLE II
THE DEGREE OF SCHEDULABILITY OF TASKS UNDER EXPONENTIAL DISTRIBUTION.

Component ((40, 18), FP)	Rate of Exp. of T_4^s	PoMD	DoQoS
T_3^p(40, 4)	1/100,000	0.350 ± 0.435	3.630 ± 0.672
	1/1000	0.363 ± 0.434	4.950 ± 0.840
	1/10	0.488 ± 0.049	8.404 ± 1.186
T_4^s(Exp., 40, 2, 40)	1/100,000	0.233 ± 0.284	0.022 ± 0.040
	1/1000	0.267 ± 0.035	2.187 ± 0.435
	1/10	0.072 ± 0.016	6.766 ± 0.636

confirms that tasks miss exactly 0% of their deadlines and have a DoQoS of 0. Throughout the running example, we use FPS scheduling but our framework supports other scheduling policies. For a deficit budget (40, 18), Table. II shows the degree of schedulability when the sporadic task T_4^s is assumed to follow an exponential probability distribution with different rates of exponential. Table III shows the Sched$^\circ$ for the two tasks given a uniform probability distribution for triggering the sporadic task T_4^s. Table IV shows the results of our analysis when using different Gaussian distributions, all with a mean value μ of 10 and different deviations σ.

VII. CONCLUSION

We have presented a compositional method for analyzing the degree of schedulability of hierarchical real-time systems. The system is modeled in terms of components containing periodic and sporadic tasks. In order to characterize more accurately the arrival time of sporadic tasks, we introduced continuous probability distributions. Given hard and soft real-time requirements, our approach provides probabilistic guarantees on the system schedulability. The Sched$^\circ$ is defined by the two factors: 1) PoMD and 2) DoQoS. These concepts are helpful when analyzing systems or components with insufficient budgets to meet all deadlines. UPPAAL SMC is used to perform statistical model checking, in order to compute the DoQoS and PoMD. A future work could be the application of the current framework to analyze the energy efficiency [5] of hierarchical scheduling systems.

TABLE III
THE DEGREE OF SCHEDULABILITY OF TASKS UNDER UNIFORM DISTRIBUTION.

Component ((40, 18), FP)	PoMD	DoQoS
T_3^p(40, 4), T_4^s(Unif., 40, 2, 40)	0.497 ± 0.051 0.052 ± 0.015	7.564 ± 1.126 6.515 ± 0.688

TABLE IV
THE DEGREE OF SCHEDULABILITY WITH GAUSSIAN DISTRIBUTION

Component ((40, 18), FP)	(μ, σ) of T_4^s	PoMD	DoQoS
T_3^p(40, 4)	(10,10)	2.000 ± 0.653	3.677 ± 2.411
	(10, 8)	1.440 ± 0.559	3.024 ± 1.914
	(10, 5)	1.640 ± 0.608	3.260 ± 1.966
	(10, 1)	0.400 ± 0.579	3.943 ± 2.611
T_4^s(Gauss., 40, 2, 40)	(10,10)	0.350 ± 0.255	1.687 ± 1.339
	(10, 8)	0.490 ± 0.326	1.135 ± 1.076
	(10, 5)	0.390 ± 0.249	0.551 ± 0.672
	(10, 1)	0.600 ± 0.324	0.704 ± 0.768

REFERENCES

[1] M. Anand, S. Fischmeister, and I. Lee. A comparison of compositional schedulability analysis techniques for hierarchical real-time systems. *ACM Trans. Embed. Comput. Syst.*, 13(1):2:1–2:37, Sept. 2013.

[2] M. Åsberg, T. Nolte, and P. Pettersson. Prototyping and code synthesis of hierarchically scheduled systems using times. *Journal of Convergence (Consumer Electronics)*, 1(1):77–86, December 2010.

[3] S. K. Baruah, A. K. Mok, and L. E. Rosier. Preemptively scheduling hard-real-time sporadic tasks on one processor. In *In Proceedings of the 11th Real-Time Systems Symposium*, pages 182–190. IEEE Computer Society Press, 1990.

[4] A. Boudjadar, A. David, J. H. Kim, K. G. Larsen, M. Mikučionis, U. Nyman, and A. Skou. Hierarchical scheduling framework based on compositional analysis using uppaal. In *Formal Aspect of Component Software, FACS 2013*, LNCS Volume 8348. Springer, 2013.

[5] A. Boudjadar, A. David, J. H. Kim, K. G. Larsen, M. Mikučionis, U. Nyman, and A. Skou. Schedulability and energy efficiency for multi-core hierarchical scheduling systems. In *International Congress on Embedded Real-Time Software and Systems, ERTS 2014*, 2014.

[6] P. E. Bulychev, A. David, K. G. Larsen, M. Mikucionis, D. B. Poulsen, A. Legay, and Z. Wang. Uppaal-smc: Statistical model checking for priced timed automata. In H. Wiklicky and M. Massink, editors, *QAPL*, volume 85 of *EPTCS*, pages 1–16, 2012.

[7] L. Carnevali, A. Pinzuti, and E. Vicario. Compositional verification for hierarchical scheduling of real-time systems. *IEEE Transactions on Software Engineering*, 39(5):638–657, 2013.

[8] F. Cassez and K. G. Larsen. The impressive power of stopwatches. In C. Palamidessi, editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 138–152. Springer, 2000.

[9] A. David, K. G. Larsen, A. Legay, and M. Mikučionis. Schedulability of herschel-planck revisited using statistical model checking. In *ISoLA (2)*, volume 7610 of *LNCS*, pages 293–307. Springer, 2012.

[10] Z. Deng and J. W. s. Liu. Scheduling real-time applications in an open environment. In *in Proceedings of the 18th IEEE Real-Time Systems Symposium, IEEE Computer*, pages 308–319. Society Press, 1997.

[11] J. Lee, L. T. X. Phan, S. Chen, O. Sokolsky, and I. Lee. Improving resource utilization for compositional scheduling using dprm interfaces. *SIGBED Rev.*, 8(1):38–45, Mar. 2011.

[12] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, Jan. 1973.

[13] J. Lorente, G. Lipari, and E. Bini. A hierarchical scheduling model for component-based real-time systems. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 8 pp.–, 2006.

[14] S. Manolache, P. Eles, and Z. Peng. Optimization of soft real-time systems with deadline miss ratio constraints. In *Real-Time and Embedded Technology and Applications Symposium, Proceedings. RTAS 2004. 10th IEEE*, pages 562–570, 2004.

[15] A. K. Mok. Fundamental design problems of distributed systems for the hard-real-time environment. Technical report, Cambridge, MA, USA, 1983.

[16] P. Pop, P. Eles, and Z. Peng. Schedulability-driven communication synthesis for time triggered embedded systems. *Real-Time Systems*, 26(3):297–325, 2004.

[17] P. Pop, L. Tsiopoulos, S. Voss, O. Slotosch, C. Ficek, U. Nyman, and A. Lopez. Methods and tools for reducing certification costs of mixed-criticality applications on multi-core platforms: the recomp approach. In *WICERT 2013 proceedings*, 2013.

[18] K. Purna and D. Bhatia. Temporal partitioning and scheduling data flow graphs for reconfigurable computers. *Computers, IEEE Transactions on*, 48(6):579–590, Jun 1999.

[19] I. Shin and I. Lee. Periodic resource model for compositional real-time guarantees. In *RTSS*, pages 2–13. IEEE Computer Society, 2003.

[20] I. Shin and I. Lee. Compositional real-time scheduling framework with periodic model. *ACM Trans. Embedded Comput. Syst.*, 7(3), 2008.

[21] A. Thekkilakattil, R. Dobrin, and S. Punnekkat. Probabilistic preemption control using frequency scaling for sporadic real-time tasks. In *The 7th IEEE International Symposium on Industrial Embedded Systems*, June 2012.

[22] Y. Zhang, D. K. Krecker, C. Gill, C. Lu, and G. H. Thaker. Practical schedulability analysis for generalized sporadic tasks in distributed real-time systems. In *Proceedings of ECRTS '08*, pages 223–232, Washington, DC, USA, 2008. IEEE Computer Society.