A General Job Model for Real-Time Analysis

Iwan Feras Fattohi
feras.fattohi@uni-ulm.de
Ulm University, Institute of Embedded Systems/Real-Time Systems
Ulm, Germany

Christian Prehofer
c.prehofer@eu.denso.com
DENSO AUTOMOTIVE Deutschland
Munich, Germany

Frank Slomka
frank.slomka@uni-ulm.de
Ulm University, Institute of Embedded Systems/Real-Time Systems
Ulm, Germany

ABSTRACT

The complexity and restricted knowledge of modern autonomous systems requires real-time analyses based on execution traces. To this end, we present a general job model, the job algebra, where any trace is represented by a linear combination of jobs in a vector space. The reason is that any analysis based on this space is applicable to any sequence of jobs. Moreover, the algebra provides a systematic approach to real-time analysis by separating the modeling of events, workload and schedulers. We demonstrate the algebra by composing classical models such as hierarchical event streams and generalized multiframe tasks by linear combination.

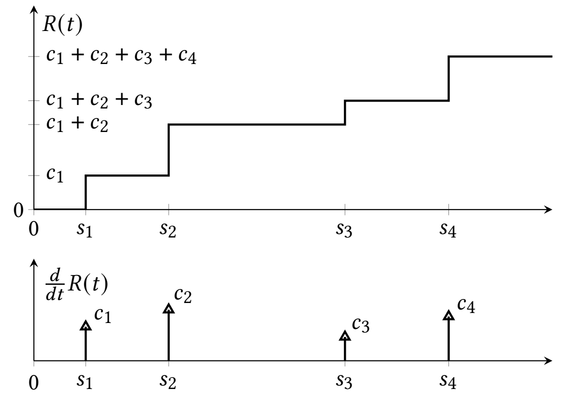

Figure 1: Cumulative workload $R(t)$ by the jobs $d/dtR(t)$.

1 INTRODUCTION

Consider an embedded system consisting of an object detection and an engine management system (EMS) that controls the acceleration of a car. The end-to-end latency from the perception of an object via the EMS to the reaction of the car depends on both of these components [4]. To ensure that the latency does not violate given timing or safety constraints, we have to analyze the EMS and the object detection with respect to their timing behavior.

Contrary to the assumptions in classical real-time analysis (RTA) [11], [19], [23], [25], [31], the task model of such an autonomous driving (AD) application may not be given for the following reasons: On the one hand, Amert et al. [5] report that the hardware model of a graphics processing unit (GPU) may not be publicly available due to proprietary reasons. This means execution times of tasks and the scheduling policy of the GPU may be unpredictable. On the other hand, it is common to apply well-known software libraries of image processing algorithms to implement AD applications. Those libraries can however be proprietary or optimized towards the throughput of GPUs rather than the predictability of the scheduling [4]. More precisely, AD applications, such as in adaptive AUTOSAR, allow the creation of threads during runtime. The latter currently imposes a challenge in the modeling of task sets that change over time [1]. As the hardware (GPUs) or software (libraries) may be unknown, we have to evaluate execution traces of jobs to gather further information about the task model.

Tracing is commonly applied to estimate unknown task parameters such as the execution time [4], [5]. In addition, the period can be determined if a control flow or call graph of the program is given [14] [41]. Furthermore, there exist different trace models to apply real-time calculus (RTC) [17] and network calculus (NC) [26].

However, the RTC already provides an implicit trace model. More precisely, consider a task with period p and worst-case execution time (WCET) c. Then, one can show for the well-known [25] cumulative workload functions $\lceil t/p \rceil \cdot c$ and $\lfloor t/p + 1 \rfloor \cdot c$ that

$$\frac{d}{dt}\left\lceil \frac{t}{p} \right\rceil \cdot c = \sum_{n=-\infty}^{\infty} c \cdot \delta(t - n \cdot p) = \frac{d}{dt}\left\lfloor \frac{t}{p} + 1 \right\rfloor \cdot c \qquad (1)$$

holds [33], [38] where δ represents the Dirac impulse [13]. Every change in the ceiling or floor function is described by a Dirac impulse. Moreover, a change describes the request of a task, so that a Dirac impulse $c \cdot \delta(t - n \cdot p)$ represents a job requested at $t = np$ with WCET c. This model can be extended to an arbitrary cumulative workload function $R(t)$ of the RTC, as depicted in Figure 1. We observe that the derivative $d/dt\, R(t)$ of the cumulative workload provides a description of jobs by a series of shifted and scaled Dirac impulses.

In this paper, we present the idea of a general job model. It is based on a linear algebra where any job sequence is represented by a linear combination of Dirac impulses. Any trace can be modeled due to the closure of a vector space of impulses. The main benefits of this algebraic approach are that any RTA modeled based on this space is implicitly applicable to any job trace. On top of that, any method of the RTC can be applied to this space by using the differential calculus of impulses [18] [38].

2 RELATED WORK

The literature on models for RTA can be divided into three different classes: event, workload and scheduler models. An overview is given in Figure 2. The seminal work by Liu & Layland [25] presents the first model for RTA where periodically activated tasks are assumed to execute with their WCET and have a deadline equal to their period. Based on the WCET as the workload model, the hardware

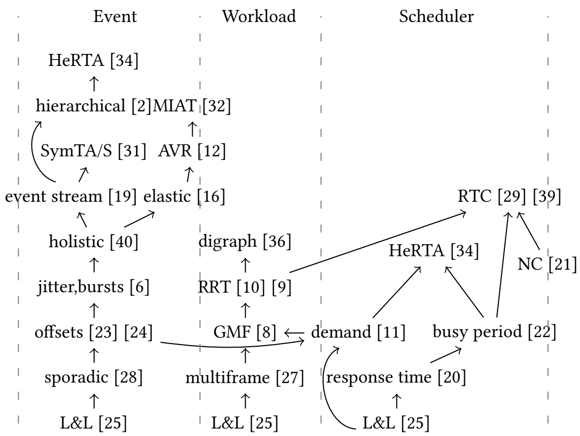

Figure 2: Relationships of models. An arc points from a more special to a more general model.

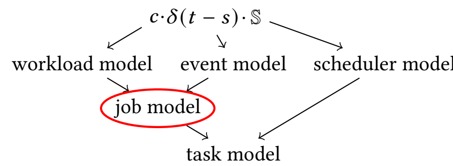

Figure 3: Relationships among the models with the job model being the focus of this work.

is completely abstracted which is also the case in the presented literature.

By an event model, we mean a formal description of time points where the execution of tasks is requested. The event model by [25] is extended by [23] and [24] to consider offsets, sporadic tasks [28], and generalized by [6] to describe jitter and bursty events. The work of [40] extends the latter approach to distributed systems. Event streams [19] generalize the jitter and bursty events models based on an event description in the interval domain. The compositional performance analysis [31] combines the distributed system and event stream model to a system-wide response time analysis based on local analysis and input-output arrival curves from the RTC [39] which originates from the NC [21]. Furthermore, event streams and the demand bound test [11] are combined to a faster test based on an approximation algorithm [3] that is generalized to hierarchical event streams [2]. These event models have in common that an event is implicitly defined by the proposed analysis functions. The Heaviside real-time analysis (HeRTA) [34] proposes an event definition that is independent from the workload and scheduler model and the analysis. Another research direction are elastic tasks which are introduced by [16] where the period is not a fixed but a varying value in a given interval. This model is generalized by [15] to describe rate-dependent tasks [12] [32].

A workload model abstracts from the code of a task in terms of execution time. The model by [25] where the whole workload of a task is described by its WCET is generalized to the multiframe model [27], such that a task cycles through a list of execution times with a fixed period. The generalized multiframe (GMF) model [8] relaxes the fixed period by allowing different time distances between jobs. Further generalizations are the real-time recurring (RRT) [9] [10] and digraph task model [36] that describe the workload dependent on the control flow of the task (branches and loops). A survey on workload models is given by [37].

The formalization of interference among tasks is described by a scheduler model. Liu & Layland [25] present the critical instant theorem and the utilization test to model and analyze interference in fixed priority (FP) and earliest deadline first (EDF) schedulings. In the Liu & Layland approach, the scheduler is directly modeled in the analysis of the task set. The utilization test of EDF is generalized to the demand bound test [11] as a necessary schedulability test for any scheduling policy and a necessary and sufficient test for EDF. The busy period approach [22] and the response time analysis [20] generalize the critical instant theorem and separate model and analysis of interference and can be specialized to e.g. FP [22] and EDF [30], [35]. The work of [34] extends the EDF response time analysis of [35] and presents a scheduler model that is independent from the event and workload model.

We observe that the three models have different concepts and research directions. Especially in the beginning of real-time scheduling theory, mixed models are presented. Later on, the three models are more separated to specialize RTA to complex event and workload patterns as shown in the left and middle column of Figure 2. Another approach is the RTC that does not assume a certain event or workload pattern. Our work extends the approach of [34] and presents a model to describe any complex job pattern.

3 THE JOB ALGEBRA

If we want to analyze a computing system for an arbitrarily given digital input, i.e. a sequence of events, then we should be able to model any such signal. The HeRTA framework [34] proposes to model an event by a Dirac impulse. This event model is independent from the workload and scheduler model, as depicted in Figure 3. Furthermore, HeRTA also presents an independent scheduler model. This work extends HeRTA and shows that job sequences can be independently described from schedulers based on the idea of a vector space of job sequences. This means any job sequence can be constructed by linear combination.

3.1 Event model

We apply the event model of [34] to derive a vector space in which any job (or event) sequence can be described. A trivial signal that jumps from zero to one at some time point $t \in \mathbb{R}$ can be described by the Heaviside function [13]

$$\mathbb{H}(t) = \begin{cases} 0 & , t < 0 \\ 1 & , t > 0 \\ \mathbb{H}(0) & , t = 0 \end{cases} \tag{2}$$

where $\mathbb{H}(0) \in [0, 1]$. Consider an event that occurs at time point $t_\epsilon \in \mathbb{R}$ and let t describe the time in our system. If the event has not yet occurred, i.e. $t < t_\epsilon$, then $\mathbb{H}(t - t_\epsilon) = 0$. If the event has already occurred, i.e. $t > t_\epsilon$, then $\mathbb{H}(t - t_\epsilon) > 0$. Therefore, we model with the Heaviside function whether an event has already occurred. However, if $t = t_\epsilon$, then the Heaviside function does not provide an exact value. Distribution theory [13] proposes to describe the discontinuous change of the Heaviside function by a functional known as the Dirac impulse

$$\delta(t) = \frac{d}{dt}\mathbb{H}(t) \tag{3}$$

which implies $\delta(t) = 0$ if $t \neq t_\epsilon$. This means we can exactly model the occurrence of an event at time point t_ϵ by a Dirac impulse. A sequence of events occurring at time points $s_1, s_2, s_3 \in \mathbb{R}$ can be described by a series of shifted Dirac impulses

$$\delta(t-s_1) + \delta(t-s_2) + \delta(t-s_3) = \sum_{n=1}^{3} \delta(t-s_n) \tag{4}$$

3.2 Workload model

The workload of a job is modeled by execution time which is in turn described by a positive real number $c \in \mathbb{R}_{>0}$. To describe workload in an interval, we apply a definite integral of the Dirac impulse

$$\int_a^b c\,\delta(t-t_\epsilon)\,dt = \begin{cases} c & ,t_\epsilon \in (a,b) \\ 0 & ,t_\epsilon \notin [a,b] \\ c\,\mathbb{H}(0) & ,t_\epsilon \in \{a,b\} \end{cases} \tag{5}$$

that returns the workload c, if the event $\delta(t - t_\epsilon)$ occurs in the interval (a, b) and that lets us decide at the interval boundaries a and b whether to count the event, which is a fundamental concept for RTA[34]. To account for the boundary cases, the upper and lower Heaviside function are defined

$$\overline{\mathbb{H}}(t) = \begin{cases} 0 & ,t < 0 \\ 1 & ,t \geq 0 \end{cases} \quad \underline{\mathbb{H}}(t) = \begin{cases} 0 & ,t \leq 0 \\ 1 & ,t > 0 \end{cases} \tag{6}$$

Therefore, workload associated with events is modeled by multiplying positive real numbers to the events

$$c_1\,\delta(t-s_1) + c_2\,\delta(t-s_2) + c_3\,\delta(t-s_3) = \sum_{n=1}^{3} c_n\delta(t-s_n) \tag{7}$$

3.3 Job model

The term $c_n\,\delta(t - s_n)$ describes a job that is requested at time point s_n with workload c_n. We can easily observe that a sequence of jobs is a linear combination of scaled Dirac impulses. Furthermore, we have separately modeled events and workload. The factor c_n is the workload model and the Dirac impulse $\delta(t - s_n)$ the event model. We generalize this idea and present the job space.

Definition 3.1 (Job space). The set

$$\Delta = \left\{ \sum_{n=1}^{N} c_n\,\delta(t-s_n) \,\middle|\, N \in \mathbb{N}, \forall n\colon c_n \in \mathbb{R}_{>0}, s_n \in \mathbb{R} \right\} \tag{8}$$

is called **job space** where c_n, s_n and N are respectively the **workload**, **shift** and **degree**, and $c_n\,\delta(t - s_n)$ is called a **job** and

$$c^n s_n := \sum_{n=1}^{N} c_n\,\delta(t-s_n) \tag{9}$$

is called a **job train**.

The following theorem states that the job space is a vector space which implies that Δ includes any sequence of jobs that can be constructed by a linear combination.

Theorem 3.2 (Job vector space [18]). *Let*

$$+_\Delta\colon \Delta\times\Delta \to \Delta, ((c^n s_n)_1, (c^n s_n)_2) \mapsto (c^n s_n)_1 +_\mathbb{R} (c^n s_n)_2 \tag{10}$$

$$\cdot_\Delta\colon \mathbb{R}\times\Delta \to \Delta, \qquad (\lambda, c^n s_n) \mapsto \lambda \cdot_\mathbb{R} c^n s_n \tag{11}$$

be two operations defined on Δ. Then $V_\Delta = (\Delta, +_\Delta, \cdot_\Delta)$ is a vector space.

Proof. The proof is shown in [18]. □

Note that we explicitly denoted addition and multiplication of real numbers by $+_\mathbb{R}$ and $\cdot_\mathbb{R}$ to avoid ambiguity. Based on the vector space property, we can model any trace of jobs in the job space. The main benefit of this property is that any RTA that is defined for the job space is applicable to any trace and therefore to any sequence of jobs. In other words, we can construct a job model in the job space independent from any analysis, so that we can separate these two development processes completely from each other. We demonstrate this in Section 3.4 after presenting some of the well-known job models of the literature in the job space.

Let us first mention that the period from classical task models is a short-form notation for job trains that exhibit periodic request times. To see this, consider the job train from Equation (7) and assume that $c_1 = c_2 = c_3 = 2$ and $s_1 = 0, s_2 = 3$ and $s_3 = 6$. Then,

$$2\,\delta(t-0) + 2\,\delta(t-3) + 2\,\delta(t-6) = \sum_{n=0}^{2} 2\,\delta(t-3n) = \sum_{n=0}^{2} c\,\delta(t-np)$$

represents the job train of three jobs of a Liu & Layland task [25] with a WCET and period of $c = 2$ and $p = 3$. If we change the workloads of the jobs to $c_1 = 1, c_2 = 2$ and $c_3 = 3$, then

$$1\,\delta(t-0) + 2\,\delta(t-3) + 3\,\delta(t-6) = \sum_{n=0}^{2} c_n\,\delta(t-3n) = \sum_{n=0}^{2} c_n\,\delta(t-np)$$

describes the job train of a multiframe task [27]. If we also change the shifts between the jobs in a common period, then we get a GMF task [8]

$$\begin{aligned} & 1\,\delta(t-0) + 2\,\delta(t-2) + 3\,\delta(t-3) \\ & + 1\,\delta(t-7) + 2\,\delta(t-9) + 3\,\delta(t-10) \\ & + 1\,\delta(t-14) + 2\,\delta(t-16) + 3\,\delta(t-17) \\ & = \sum_{n=0}^{2} c_1\,\delta(t-7n) + \sum_{n=0}^{2} c_2\,\delta(t-2-7n) + \sum_{n=0}^{2} c_3\,\delta(t-3-7n) \\ & = \sum_{n=0}^{2}\sum_{m=1}^{3} c_m\,\delta(t-\phi_m-np) \end{aligned}$$

where $p = 7$ is the period and $\phi_1 = 0, \phi_2 = 2$ and $\phi_3 = 3$ are the shifts (offsets) of the frames relative to the period. If the shifts ϕ_m have equal differences among each other, e.g. if we set $s_3 = 4$, then we get a periodic task with burst [34]

$$\begin{aligned} & \sum_{n=0}^{2} c_1\,\delta(t-7n) + \sum_{n=0}^{2} c_2\,\delta(t-2-7n) + \sum_{n=0}^{2} c_3\,\delta(t-4-7n) \\ & = \sum_{n=0}^{2}\sum_{m=0}^{2} c_m\,\delta(t-mq-np) \end{aligned}$$

where each job of a burst has its own WCET c_m and $q = 2$ is the inner period of the burst. This means we have a multiframe task with bursts. We can generalize those examples to a GMF task and a multiframe task with bursts generating N periods of M frames and respectively N bursts of M jobs:

$$\sum_{n=0}^{N-1}\sum_{m=1}^{M} c_m\,\delta(t-\phi_m-np) \tag{12}$$

$$\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} c_m\,\delta(t-mq-np) \tag{13}$$

We observe for this example that a multiframe task with bursts is a special case of a GMF task since ϕ_m may be factorized to mq. The multiframe task with bursts can be described by a convolution of its outer and inner period [34]:

$$\sum_{n=0}^{N-1}\delta(t-np)*\sum_{m=0}^{M-1} c_m\,\delta(t-mq) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} c_m\,\delta(t-mq-np) \tag{14}$$

Further inner or outer periods can be also added, i.e. a burst inside a burst (or hierarchical event stream [2]), by applying further convolutions [18]:

$$\sum_{n=0}^{N-1}\delta(t-np)*\sum_{m=0}^{M-1} c_m\,\delta(t-mq)*\sum_{k=0}^{K-1}\delta(t-kr) \tag{15}$$

$$=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\sum_{k=0}^{K-1} c_m\,\delta(t-kr-mq-np) \tag{16}$$

with $r \in \mathbb{R}$. We observe that convolution represents any nested periodic behavior of jobs. On top of that, we can now add the GMF task to the hierarchical event stream and still get a valid job train:

$$\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} c_m\,\delta(t-\phi_m-np) + \sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\sum_{k=0}^{K-1} c_m\,\delta(t-kr-mq-np)$$

By valid we mean that the addition of the GMF task and the hierarchical event stream is again a vector of the job space which follows from the closure of the vector space operations. For all those presented job models, we know that their job trains are elements of the job space. This means we can linearly combine any of the job models of the literature or construct new ones and always know that the output is again a valid job train. Therefore, any analysis that is derived for an arbitrary vector of the job space is applicable to any job model.

3.4 Scheduler model

This section applies the scheduler model of [34] to demonstrate the independent modeling of jobs and task schedulers. To begin with, we assign job trains to tasks, so that we can define interference among job trains by assigning a priority to tasks. Note that this approach represents the classic AUTOSAR concept of runnables and tasks [7]. Furthermore, we assign to each job of the train a relative deadline for the purpose of RTA.

Formally, a task $\tau = ((c^n s_n)_\tau, \boldsymbol{d}_\tau, \Pi_\tau)$ is defined by a job train $c^n s_n \in \Delta$ denoted by $(c^n s_n)_\tau$, a deadline vector $\boldsymbol{d}_\tau \in \mathbb{R}^N_{>0}$ with N being the degree of $c^n s_n$ and $\Pi_\tau \in \mathbb{N}$ being the priority. The relative deadline of job $c_n\,\delta(t-s_n)$ is denoted by d^n_τ and its absolute deadline is $D^n_\tau = s_n + d^n_\tau$. Then, a task set is denoted by $\Gamma = \{\tau\}$. For example, we can now encapsulate the job trains of the GMF task and the hierarchical event stream into two tasks of a task set $\Gamma = \{\tau_1, \tau_2\}$:

$$\underbrace{\sum_{n=0}^{N-1}\sum_{m=1}^{M} c_m\,\delta(t-\phi_m-np)}_{(c^n s_n)_{\tau_1}}$$

$$+\underbrace{\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\sum_{k=0}^{K-1} c_m\,\delta(t-kr-mq-np)}_{(c^n s_n)_{\tau_2}} = \sum_{i=1}^{2}(c^n s_n)_{\tau_i} = \sum_{\tau\in\Gamma}(c^n s_n)_\tau$$

After the encapsulation of jobs in tasks, we can define the interference by using the scheduler model of [34]. In particular, we demonstrate here an EDF scheduler with tie breaking by priorities. The task scheduler

$$\mathbb{S}_{D^n_\tau, D^n_{\tau'}} = \underline{\mathbb{H}}(D^n_\tau - D^n_{\tau'}) + \delta_{D^n_\tau, D^n_{\tau'}} \cdot \overline{\mathbb{H}}(\Pi_\tau - \Pi'_\tau) \tag{17}$$

$$\delta_{D^n_\tau, D^n_{\tau'}} = \begin{cases} 1 & , D^n_\tau = D^n_{\tau'} \\ 0 & , D^n_\tau \neq D^n_{\tau'} \end{cases} \tag{18}$$

describes for the job under consideration $c_n\,\delta(t-s_n)$ with absolute deadline D^n_τ the interference by another job of the task τ' according to EDF with tie-breaking. Based on the scheduler $\mathbb{S}_{D^n_\tau, D^n_{\tau'}}$ and the job train of the task set Γ, we can now compute the interference on the job under consideration $c_n\,\delta(t-s_n)$ in the interval $[a,b)$ by

$$R_{\Gamma,\tau}(t, \Delta\frac{b}{a}) = \int_{-\infty}^{\infty}\sum_{\tau'\in\Gamma}(c^n s_n)_{\tau'} \cdot \mathbb{S}_{D^n_\tau, D^n_{\tau'}} \cdot \overline{\mathbb{H}}(t-a)\,\underline{\mathbb{H}}(b-t)\,dt \tag{19}$$

which is known as the interference request bound function from [34]. At this point, we can apply any of the analyses from [34] as for example the response time analysis or the demand bound test for any job train. For detailed examples on how to compute Equation (19), see [34].

4 CONCLUSIONS

This short paper introduced a vector space of job sequences, called the job algebra, to model any trace of jobs for real-time analysis. We have exemplary shown that classical real-time task models can be easily described in the vector space based on the idea that every task generates a sequence of jobs. The main benefit of the algebraic approach is that any trace of jobs can be modeled in the vector space which means any analysis on this space is applicable to any modeled trace. As a result, a systematic and general approach of modeling and analysis of real-time systems is possible by the separation of these processes. In future work, we want to present our recent results on the generalization of real-time analysis based on the job algebra.

ACKNOWLEDGEMENT

This work was partially supported by the Federal Ministry for education and research (BMBF) under the grant agreement number 01IS21031B and partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SL 47/20-1 P513671169.

REFERENCES

[1] Inchron AG. 2023. *Modeling Threads and Thread Pools on AUTOSAR Adaptive Platform Applications.* https://www.inchron.com/parallelism-and-worker-threads/

[2] Karsten Albers, Frank Bodmann, and Frank Slomka. 2006. Hierarchical event streams and event dependency graphs: A new computational model for embedded real-time systems. In *18th Euromicro Conference on Real-Time Systems (ECRTS'06).* IEEE, 10–pp.

[3] Karsten Albers and Frank Slomka. 2004. An event stream driven approximation for the analysis of real-time systems. In *Proceedings. 16th Euromicro Conference on Real-Time Systems, 2004. ECRTS 2004.* IEEE, 187–195.

[4] Tanya Amert, Michael Balszun, Martin Geier, F Donelson Smith, James H Anderson, and Samarjit Chakraborty. 2021. Timing-predictable vision processing for autonomous systems. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 1739–1744.

[5] Tanya Amert, Nathan Otterness, Ming Yang, James H Anderson, and F Donelson Smith. 2017. GPU scheduling on the NVIDIA TX2: Hidden details revealed. In *2017 IEEE Real-Time Systems Symposium (RTSS).* IEEE, 104–115.

[6] Neil Audsley, Alan Burns, Mike Richardson, Ken Tindell, and Andy J Wellings. 1993. Applying new scheduling theory to static priority pre-emptive scheduling. *Software engineering journal* 8, 5 (1993), 284–292.

[7] AUTOSAR. 2023. *AUTOSAR Classic Platform.* https://www.autosar.org/standards/classic-platform

[8] Sanjoy Baruah, Deji Chen, Sergey Gorinsky, and Aloysius Mok. 1999. Generalized multiframe tasks. *Real-Time Systems* 17, 1 (1999), 5–22.

[9] Sanjoy K Baruah. 1998. A general model for recurring real-time tasks. In *Proceedings 19th IEEE Real-Time Systems Symposium (Cat. No. 98CB36279).* IEEE, 114–122.

[10] Sanjoy K Baruah. 2003. Dynamic-and static-priority scheduling of recurring real-time tasks. *Real-Time Systems* 24, 1 (2003), 93–128.

[11] Sanjoy K Baruah, Louis E Rosier, and Rodney R Howell. 1990. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *Real-time systems* 2, 4 (1990), 301–324.

[12] Alessandro Biondi, Alessandra Melani, Mauro Marinoni, Marco Di Natale, and Giorgio Buttazzo. 2014. Exact interference of adaptive variable-rate tasks under fixed-priority scheduling. In *2014 26th euromicro conference on real-time systems.* IEEE, 165–174.

[13] R.N. Bracewell. 2000. *The Fourier Transform and Its Applications.* McGraw Hill. https://books.google.de/books?id=ZNQQAQAAIAAJ

[14] Max Brand, Albrecht Mayer, and Frank Slomka. 2020. NITRO: Non-Intrusive Task Detection and Monitoring in Hard Real-Time Systems. In *Proceedings of the 28th International Conference on Real-Time Networks and Systems.* 78–88.

[15] Giorgio C Buttazzo, Enrico Bini, and Darren Buttle. 2014. Rate-adaptive tasks: Model, analysis, and design issues. In *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 1–6.

[16] Giorgio C Buttazzo, Giuseppe Lipari, and Luca Abeni. 1998. Elastic task model for adaptive rate control. In *Proceedings 19th IEEE Real-Time Systems Symposium (Cat. No. 98CB36279).* IEEE, 286–295.

[17] Gonzalo Carvajal, Mahmoud Salem, Nirmal Benann, and Sebastian Fischmeister. 2018. Enabling rapid construction of arrival curves from execution traces. *IEEE Design and Test* 35, 4 (2018), 23–30.

[18] Iwan Feras Fattohi, Frank Slomka, and Christian Prehofer. 2023. The Shirac Algebra. *To be published* (2023).

[19] Klaus Gresser. 1993. Echtzeitnachweis ereignisgesteuerter realzeitsysteme. (1993).

[20] Mathai Joseph and Paritosh Pandya. 1986. Finding response times in a real-time system. *Comput. J.* 229, 5 (1986), 390–395.

[21] Jean-Yves Le Boudec and Patrick Thiran. 2001. *Network calculus: a theory of deterministic queuing systems for the internet.* Springer.

[22] John P Lehoczky. 1990. Fixed priority scheduling of periodic task sets with arbitrary deadlines. In *[1990] Proceedings 11th Real-Time Systems Symposium.* IEEE, 201–209.

[23] Joseph Y-T Leung and ML Merrill. 1980. A note on preemptive scheduling of periodic, real-time tasks. *Information processing letters* 11, 3 (1980), 115–118.

[24] Joseph Y-T Leung and Jennifer Whitehead. 1982. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance evaluation* 2, 4 (1982), 237–250.

[25] Chung Laung Liu and James W Layland. 1973. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)* 20, 1 (1973), 46–61.

[26] Natchanon Luangsomboon, Robert Hesse, and Jörg Liebeherr. 2017. Fast min-plus convolution and deconvolution on GPUs. In *Proceedings of the 11th EAI International Conference on Performance Evaluation Methodologies and Tools.* 126–131.

[27] Aloysius K Mok and Deji Chen. 1997. A multiframe model for real-time tasks. *IEEE transactions on Software Engineering* 23, 10 (1997), 635–645.

[28] Aloysius Ka-Lau Mok. 1983. *Fundamental design problems of distributed systems for the hard-real-time environment.* Ph. D. Dissertation. Massachusetts Institute of Technology.

[29] Martin Naedele, Lothar Thiele, and Michael Eisenring. 1999. Characterizing variable task releases and processor capacities. *IFAC Proceedings Volumes* 32, 2 (1999), 8521–8526.

[30] José C Palencia and Michael González Harbour. 2003. Offset-based response time analysis of distributed systems scheduled under EDF. In *15th Euromicro Conference on Real-Time Systems, 2003. Proceedings.* IEEE, 3–12.

[31] Kai Richter. 2004. *Compositional scheduling analysis using standard event models.* Ph. D. Dissertation.

[32] Mohammadreza Sadeghi, Marco Philippi, Amir Mahdian, and Frank Slomka. 2022. MIAT Efficient analysis of adaptive variable-rate tasks. *Journal of Systems Architecture* 127 (2022), 102472.

[33] Amol Sasane. 2017. *A friendly approach to functional analysis.* World Scientific.

[34] Frank Slomka and Mohammadreza Sadeghi. 2021. Beyond the limitations of real-time scheduling theory: a unified scheduling theory for the analysis of real-time systems. *SICS Software-Intensive Cyber-Physical Systems* 35, 3 (2021), 201–236.

[35] Marco Spuri and Giorgio Buttazzo. 1996. Scheduling aperiodic tasks in dynamic priority systems. *Real-Time Systems* 10, 2 (1996), 179–210.

[36] Martin Stigge, Pontus Ekberg, Nan Guan, and Wang Yi. 2011. The digraph real-time task model. In *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium.* IEEE, 71–80.

[37] Martin Stigge and Wang Yi. 2015. Graph-based models for real-time workload: a survey. *Real-time systems* 51, 5 (2015), 602–636.

[38] Robert S. Strichartz. 2003. *A guide to distribution theory and Fourier transforms.* World Scientific Publishing Company.

[39] Lothar Thiele, Samarjit Chakraborty, and Martin Naedele. 2000. Real-time calculus for scheduling hard real-time systems. In *2000 IEEE International Symposium on Circuits and Systems (ISCAS)*, Vol. 4. IEEE, 101–104.

[40] Ken Tindell and John Clark. 1994. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and microprogramming* 40, 2-3 (1994), 117–134.

[41] Haibo Zeng and Marco Di Natale. 2013. Outstanding paper award: Using max-plus algebra to improve the analysis of non-cyclic task models. In *2013 25th Euromicro Conference on Real-Time Systems.* IEEE, 205–214.