Shape-Aware Analysis of End-to-End Latency Under LET

Mario Günzel
TU Dortmund University
Email: mario.guenzel@tu-dortmund.de

Matthias Becker
KTH Royal Institute of Technology Stockholm
Email: mabecker@kth.se

Daniel Casini
Scuola Superiore Sant'Anna
Email: daniel.casini@santannapisa.it

***Abstract*—Modern real-time cyber-physical systems often include cause-effect chains of interconnected tasks that require to satisfy timing constraints. While worst-case end-to-end latency analysis has received significant attention, real-world systems frequently require richer timing guarantees, including average latency, throughput, and tolerance to bounded violations.**

This paper introduces a novel shape-aware analysis technique for characterizing the reaction time of cause-effect chains under the Logical Execution Time (LET) model. By determining a set of anchor points, our method allows to fully describe the end-to-end latency curve and enables the computation of diverse metrics, such as minimum, maximum, and average latency, throughput, weakly-hard (m, k) constraints, and maximum exceedance intervals. The evaluation leverages real-world use-cases to show how our unified framework extends the end-to-end timing analysis capabilities, allowing to efficiently compute a large set of metrics.

***Index Terms*—End-to-End Latency, Cause-Effect Chain, Logical Execution Time (LET).**

I. Introduction

Modern embedded cyber-physical systems often consist of software pipelines, known as *cause-effect chains*, that span sensing, processing, and actuation stages. They have been present in embedded systems since the early beginning of the century, both in traditional applications (e.g., automotive or industrial systems) up to modern AI-based use cases like Advanced Driving Assistance Systems (e.g., in automated lane keeping). For cause-effect chains in safety-critical systems, the end-to-end timing behavior of the entire chain is of interest.

While worst-case latency bounds remain a key requirement in safety-critical systems, they are often insufficient to capture the full temporal behavior needed in practice. Designers of modern systems are often called to satisfy *mixed and heterogeneous timing constraints*, ranging from classical maximum to minimum [55], [57] and average end-to-end latency bounds [40], or they can frequently operate in environments where occasional timing violations are acceptable, provided that they remain bounded and infrequent [1], [9]. A case in point is the ECRTS 2022 Industrial Challenge by ARM [1]. It explicitly mentions that for their head pose estimation application "a small number of frames can be dropped without affecting the general functionality of the application" [1]: this is likely to be modeled as a cause-effect chain of tasks under weakly-hard constraints, which require at most m instances to exceed the timing constraint in every window of k instances [9]. Here, respecting the weakly-hard constraints is important to display to the driver data that is updated enough to avoid providing misleading information (e.g., stale navigation data or traffic signs) while, at the same time, not requiring additional hardware compute power that would instead be needed if the system was designed so that all deadlines must be respected. Even further, modern systems may also need to meet throughput constraints (e.g., camera pipelines with a minimum FPS), or to define new metrics, such as maximum exceedance intervals that can capture the longest duration over which a system continuously exceeds a given latency bound, offering insight into the worst-case clustering of timing violations of a chain.

Related work.[1] The literature on the end-to-end analysis of chains is divided into two main categories: time-triggered chains, where computational activities (tasks) execute according to their periodic behavior and communicate via shared-memory variables, and data-driven (or event-driven) chains [31], where tasks are activated by the termination of the predecessors or by a message coming from a middleware implementing the publish-subscribe paradigm [54], [59]. This paper refers to the former category. Here, the seminal work on the end-to-end latency was by Davare et al. [15], in which the end-to-end latency was considered from a design-time optimization perspective. Later, Feiertag et al. [17] proposed concrete definitions of data-age and reaction time, and many analytical solutions to derive their *maximum* values have been developed, as summarized in [25]. Typical approaches either fully unroll task instances in a safe analysis window [4], [5], [22], [23], [36], [56] or provide closed-form solution by abstraction of scheduling behavior [15], [16], [42], [43]. Furthermore, several optimizations of worst-case end-to-end latency through choice of system design parameters and scheduling modification have been proposed [10], [15], [27], [45], [46], [52], [60].

Limitation of existing approaches. Despite the need for general methods to satisfy rich and diverse timing constraints, *most of the aforementioned methods focus only on a subset of the problem*, typically, the *maximum end-to-end latency*. The reason for this gap is fairly simple to explain: the timing of

[1]For convenience of exposition, we restrict this discussion to papers related to end-to-end latency analysis only, while we discuss papers related to specific metrics in the respective subsections of Section VI. Enumeration-based analyses from the literature are discussed in Section VII.

cause-effect chains is characterized by complex interactions, which makes it not trivial to derive analysis methods even if only one metric and scenario at a time is considered, e.g., due to complex scheduling effects that affect the scheduling of individual tasks and, in turn, the timing of the chain. While the introduction of the *Logical Execution Time* (LET) paradigm [32], a well-established communication model in automotive [28] in which communicating tasks read and write data at fixed time instants, allows to abstract most scheduling effects, this determinism has been exploited only to a very limited extent yet in the analysis of alternative end-to-end latency metrics.

Contributions. This paper fills this gap by developing a *shape-aware analysis* for the reaction time end-to-end latency metric of cause-effect chains of communicating tasks under LET.

- We present a method to describe the shape of the end-to-end latency curve by a set of *anchor points* in Section IV.
- Afterwards, we propose an approach to efficiently compute anchor points using *partitioned job chains* [24] in Section V.
- Section VI demonstrates the usefulness of the description by anchor points by proposing concrete definitions and shape-aware analyses for a rich portfolio of timing metrics: *minimum*, *maximum*, and *average* reaction time, *throughput*, *longest consecutive exceedance*, and *chain-aware weakly-hard (m,k) constraints* on reaction time. For most metrics, we provide the first formal definitions for cause-effect chains, and the shape-aware approach allows us to conduct their exact analysis. For metrics related to the maximum reaction time, we provide a discussion of enumeration-based approaches in Section VII.
- In Section VIII, we evaluate our approach for multiple case studies and synthetic workloads, showing that our approach efficiently yields results.

II. System Model

In this work, we consider a set $\mathbb{T}$ of *arbitrary-deadline*, *periodic* tasks communicating using the *Logical Execution Time (LET)* paradigm. This section details the underlying task, communication, and scheduling model.

Task Model: Each task $\tau \in \mathbb{T}$ is specified by a tuple $(C_\tau, T_\tau, D_\tau, \phi_\tau) \in \mathbb{R}^4$, and recurrently releases jobs $\tau(j),\ j \in \mathbb{N}{=}\{0,1,2,\ldots\}$ according to its description. Specifically, $C_\tau \geq 0$ is the Worst-Case Execution Time (WCET), i.e., the maximum amount of time that each job of τ can be executed. The task period $T_\tau > 0$ and phase ϕ_τ determine the time of releases, i.e., the first job $\tau(0)$ is released at ϕ_τ, and subsequent jobs are released every T_τ time units. Specifically, job $\tau(j),\ j \in \mathbb{N}$ is released at time $\phi_\tau + j \cdot T_\tau$. The relative deadline $D_\tau > 0$ specifies the timing requirement of τ. That is, each job $\tau(j)$ must finish until its absolute deadline $\phi_\tau + j \cdot T_\tau + D_\tau$. We consider arbitrary-deadline tasks, where $D_\tau \leq T_\tau$ and $D_\tau > T_\tau$ is allowed.

Communication Model: Data transmission between tasks is achieved by communication, modeled via shared resources. That is, each task reads data from a shared resource and writes data to a shared resource. In this work, we consider the LET [32], [35] model, where a task reads data at each of its job releases and writes data at each job's absolute deadline. Specifically, the read-event of $\tau(m)$ is at time

$$\mathrm{re}(\tau(m)) := \phi_\tau + m \cdot T_\tau, \tag{1}$$

and the write-event of $\tau(m)$ is at time

$$\mathrm{we}(\tau(m)) := \phi_\tau + m \cdot T_\tau + D_\tau. \tag{2}$$

Please note that by the choice of phases and arbitrary deadlines, the communication model is as general as the *flexible LET (fLET)* model discussed in [60]. We assume that communication overheads are negligible and that data can be transmitted between two jobs even if the read- and write-events occur at the same time, which is realized by existing LET implementations [7], [8], [12].

Scheduling Model: The results of this work are not tied to any specific scheduling model. That is, typical schedulers such as partitioned or global versions of Fixed-Priority (FP), or Earliest-Deadline-First (EDF) can be applied. However, to ensure that the communication via LET can be realized, the schedulability of the task set needs to be verified (i.e., all jobs finish until their absolute deadline). This can be achieved by employing existing schedulability tests (see [2], [13]).

III. Cause-Effect Chains and End-to-End Latency

A sequence of communicating tasks, often achieving complex functionalities, is called a *cause-effect chain*, denoted by

$$E = (\tau_1 \rightarrow \cdots \rightarrow \tau_n) \tag{3}$$

with $n \in \mathbb{N}_{\geq 1}$ and $\tau_i \in \mathbb{T}$ for all $i = 1, \ldots, n$. Specifically, for each $i = 1, \ldots, n-1$, task τ_i writes data to a shared resource that is read by τ_{i+1}. We follow a typical assumption [25] that sampling of data occurs at the read-events of the source task τ_1. As noted in [25], different sampling rates can be modeled by inserting an additional sampling task at the beginning of the cause-effect chain. Furthermore, if data dependencies are described by a Directed Acyclic Graph (DAG), we can analyze each path in the DAG as an individual cause-effect chain. We denote by $H(E) \coloneqq \mathrm{LCM}(T_{\tau_1}, \ldots, T_{\tau_n})$ the hyperperiod of E.

To analyze the timing behavior of E, the *end-to-end latency* is of interest, i.e., the time that it takes to propagate data through the whole cause-effect chain E. For defining the end-to-end latency, there are two different perspectives:

- **Reaction time**: *How long does it take until an external activity is processed?* For that, the end-to-end latency is determined in a forward manner, i.e., starting from an external activity, the progression of data through the system is traced.
- **Data age**: *How old is data used in an actuation?* For that, the latency is determined in a backward manner, i.e., given an actuation that uses data from the cause-effect chain, the data is traced back to its origin.

Due to space constraints, **this work focuses on the perspective of the reaction time only**. While it has been shown that the maximum value of the reaction time and data age are the same [24], other metrics (e.g., minimum or average) potentially do not follow this equivalence. Hence, metrics and shape-aware analysis for the data age will be derived in future work.

To trace the progression of data and the origin of data, *job chains* can be utilized. Specifically, a job chain characterizes the data path of a specific data token through the task instances.

Definition 1 (Job chain)**.** Consider a cause-effect chain $E = (\tau_1 \to \cdots \to \tau_n)$. A job chain $(\tau_1(j_1), \ldots, \tau_n(j_n))$, with $j_1, \ldots, j_n \in \mathbb{N}$, is a sequence of jobs, such that each job reads the data at or after it was written by the previous job. That is, $\mathrm{we}(\tau_i(j_i)) \leq \mathrm{re}(\tau_{i+1}(j_{i+1}))$ for all $i = 1, \ldots, n{-}1$.

To trace the data propagation, we rely on a specific form of job chains, namely, immediate forward job chains [16].

Definition 2 (Immediate forward job chain)**.** A job chain $(\tau_1(j_1), \ldots, \tau_n(j_n))$ is immediate forward if for all $i \in \{1, \ldots, n{-}1\}$ the job $\tau_{i+1}(j_{i+1})$ is the *earliest* job of τ_{i+1} with read-event no earlier than the write-event of $\tau_i(j_i)$, i.e., $j_{i+1} = \min\{\xi \in \mathbb{N} \,|\, \mathrm{re}(\tau_{i+1}(\xi)) \geq \mathrm{we}(\tau_i(j_i))\}$. We denote this immediate forward job chain by $fc(E, j_1)$.

To include the time points for external sensing/trigger activity and for the completion of data processing, we equip job chains with additional timing parameters at the front and at the end, following [22], [23].

Definition 3 (Immediate forward augmented job chain)**.** Let $E = (\tau_1 \to \cdots \to \tau_n)$ be a cause-effect chain and $t \in \mathbb{R}$ a time point representing an external activity. We define the immediate forward augmented job chain at time t as

$$fac(E, t) \coloneqq (t, fc(E, j_1), p) \tag{4}$$

where $\tau_1(j_1)$ is the first job with read-event after time t (i.e., $j_1 = \min\{\xi \in \mathbb{N} \,|\, \mathrm{re}(\tau_1(\xi)) > t\}$) and $p = \mathrm{we}(\tau_n(j_n))$ is the time when the data is fully processed.

Determining an immediate forward augmented job chain $fac(E, t)$ is equivalent to tracing the forward propagation of data for an external activity at time t. Hence, the reaction time for an external activity at time t is determined by the length $p - t$ of the immediate forward augmented job chain $fac(E, t) = (t, fc(E, j_1), p)$.

Definition 4 (Reaction time)**.** Given a cause-effect chain E and a time point t representing the time of an external activity, the reaction time is defined as

$$RT(E, t) \coloneqq \ell(fac(E, t)), \tag{5}$$

where $\ell(fac(E, t)) \coloneqq p - t$ is the length of $fac(E, t) = (t, fc(E, j_1), p)$.

To account for the fact that most applications are only concerned with the system behavior when the system is properly warmed up, we follow the definition of the *warm-up* as established in [24] It leverages immediate backward job chains (typically used to study the data age) to determine the first time that data is available to jobs.

Definition 5 (Immediate backward job chain)**.** A job chain $(\tau_1(j_1), \ldots, \tau_n(j_n))$ is immediate backward if for all $i \in \{1, \ldots, n{-}1\}$ the job $\tau_i(j_i)$ is the *latest* job of τ_i with write-event no later than the read-event of $\tau_{i+1}(j_{i+1})$, i.e., $j_i = \max\{\xi \in \mathbb{N} \,|\, \mathrm{we}(\tau_i(\xi)) \leq \mathrm{re}(\tau_{i+1}(j_{i+1}))\}$. We denote this immediate backward job chain by $bc(E, j_n)$.

Definition 6 (Warm-up)**.** Given a cause-effect chain $E = (\tau_1 \to \cdots \to \tau_n)$, let $W_1, \ldots, W_n \in \mathbb{N}$ such that $bc(E, W_n) = (\tau_1(W_1), \ldots, \tau_n(W_n))$ is the *first* immediate backward job chain that exists. We say that τ_i is warmed up at job $\tau_i(W_i)$.

Following the concept of warm-up, the reaction time is only considered when $t > \mathrm{re}(\tau_1(W_1))$. Specifically, $RT(E, \bullet)$ can be considered as function over the time domain $\mathbb{R}_{>\mathrm{re}(\tau_1(W_1))}$:

$$RT(E, \bullet)\colon \quad \mathbb{R}_{>\mathrm{re}(\tau_1(W_1))} \to \mathbb{R}_{\geq 0} \tag{6}$$

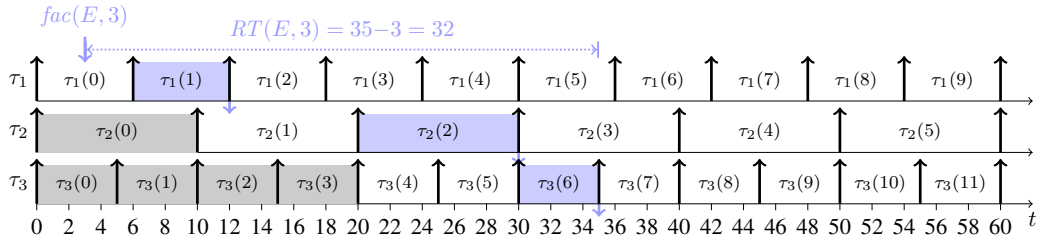
In the following, we examine the shape of this function in Section IV, discuss how to efficiently determine this shape in Section V, and provide a shape-aware analysis of different metrics in Sections VI.

IV. Shape of End-to-End Latency

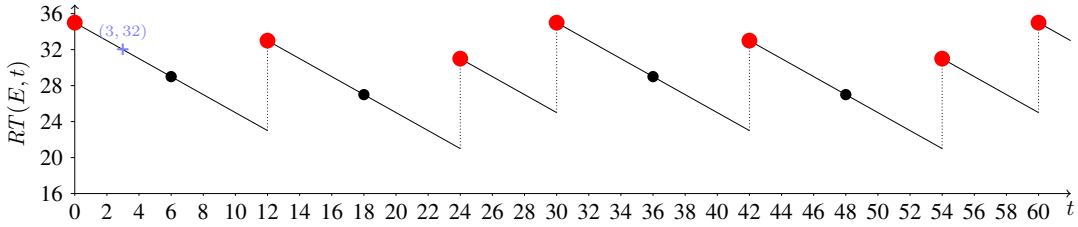
In this section, we analyze the shape of the function $RT(E, \bullet)$. Specifically, this section shows that $RT(E, \bullet)$ has a *piece-wise linear* structure and can therefore be fully specified by a set of *anchor points*. In the following, we always consider a single chain E, but our methods and consideration generalizes to systems of multiple chains (with possibly shared tasks) by analyzing each chain independently. Before we dive into the discussion and formal proofs, we consider an example system, depicted in Figure 1, with release pattern illustrated in Figure 1a and reaction time function depicted in Figure 1b. We note that Figure 1c is not discussed in this section but only relevant for the efficient shape determination in Section V.

Example 7 (Example System)*.* We consider an example system of three tasks $\mathbb{T} = \{\tau_1, \tau_2, \tau_3\}$. The tasks have periods $T_{\tau_1} = 6$, $T_{\tau_2} = 10$, and $T_{\tau_3} = 5$. For simplicity, the deadlines are chosen implicit and all tasks are released synchronously, i.e., $D_{\tau_i} = T_{\tau_i}$ and $\phi_{\tau_i} = 0$ for all $i = 1, 2, 3$. As noted in Section II, the WCET is irrelevant for communication under LET, as long as the schedulability of the task set is guaranteed.

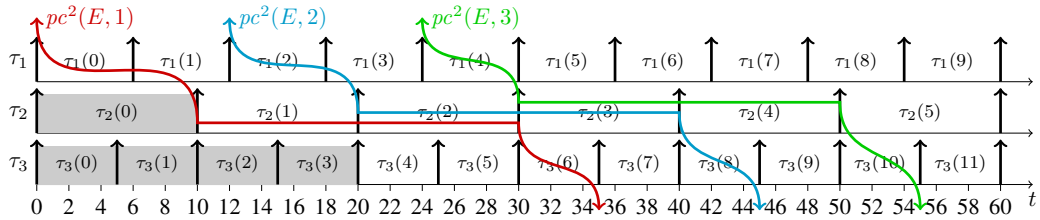
Figure 1a depicts the release pattern of the example system, which completely determines the communication behavior of the system due to the LET paradigm of the implicit-deadline tasks, i.e., each job reads at its release and writes at its deadline which is at the subsequent job release. Since there is no immediate backward job chain with last job $\tau_3(3)$, the first immediate backward job chain is $bc(E, 4) = (\tau_1(0), \tau_2(1), \tau_3(4))$. Hence, by Definition 6, $W_1 = 0$, $W_2 = 1$, and $W_3 = 4$. Therefore, the tasks are warmed up at jobs $\tau_1(0)$, $\tau_2(1)$, and $\tau_3(4)$, and the jobs before the warm-up are marked gray.



(a) Release pattern of the example system. Job releases are depicted by upward facing job arrows. Jobs before the warm-up (i.e., those before the first immediate backward job chain) are marked gray.



(b) Reaction time $RT(E,t)$ over time t after the warm-up $\mathrm{re}(\tau_1(W_1)) = 0$. The red points (i.e., the minimal anchor points) describe the whole curve.



(c) The 2-partitioned job chains (as introduced in Section V) over one hyperperiod correspond to the minimal anchor points with start time and length of the partitioned job chain as x-entry and y-entry of the anchor point, respectively.

Figure 1: An example system of three tasks with periods $T_{\tau_1} = 6$, $T_{\tau_2} = 10$, and $T_{\tau_3} = 5$, all with implicit deadlines. The x-axis shows the time t, and the cause-effect chain $E = (\tau_1 \to \tau_2 \to \tau_3)$ is analyzed.

The reaction time function $RT(E, \bullet)$ over time is depicted in Figure 1b. Exemplary, the immediate forward augmented job chain $fac(E, 3)$ of E at time $t = 3$ is depicted in blue in Figure 1a. We observe that it has a length of $\ell(fac(E, 3)) = 35 - 3 = 32$. Hence, $RT(E, 3) = 32$, and the corresponding point $(3, 32)$ is marked in the plot in Figure 1b. We observe that $RT(E, \bullet)$ is a *piece-wise linearly decreasing* function and increases non-continuously in certain points.

In this example, due to the piece-wise linear structure of $RT(E, \bullet)$, we can see that it could be fully described by the large red points in Figures 1b. The remainder of this section concerns with the formal description of this behavior. We start by considering the reaction time $RT(E, \bullet)$.

By Definition 4, the reaction time $RT(E, t)$ is determined as the length of the immediate forward augmented job chain $fac(E, t) = (t, fc(E, j_1), p)$ for E at time t. As long as the immediate forward job chain $fc(E, j_1)$ used inside the immediate forward augmented job chain does not change (i.e., j_1 remains the same), increasing the time t linearly decreases the length of $fac(E, t)$ and consequently the value of $RT(E, t)$. Only when t reaches the value $\mathrm{re}(\tau_1(j_1))$, the chain $fc(E, j_1)$ inside $fac(E, t)$ is replaced by $fc(E, j_1 + 1)$ and the length can potentially increase.

In the system from Example 7 with schedule in Figure 1a, $fac(E, 4)$ and $fac(E, 5)$ use the same immediate forward job chain as $fac(E, 3)$, namely $fc(E, 1)$. Hence, they still end at 35, and we observe a linear decrease in Figure 1b from $t = 3$ to $t = 5$. Since there is a read-event of τ_1 at time $t = 6$, the chain $fac(E, 6)$ uses another immediate forward job chain $fc(E, 2)$, which potentially leads to increased length. However, since $fc(E, 2)$ still ends in the same job, which is $\tau_3(6)$, the length continues decreasing linearly. Only when t reaches 12, we observe an increase because $fac(E, 12)$ uses $fc(E, 3) = (\tau_1(3), \tau_2(3), \tau_3(8))$, which ends with a job $\neq \tau_3(6)$. The points $t = \mathrm{re}(\tau_1(j))$ for $j \geq W_1$ where the reaction time can potentially increase are depicted in Figure 1b. In-between, the function decreases linearly. The points where an actual increase occurs are marked red. We generalize this observation of the shape of $RT(E, \bullet)$ formally using the following lemma.

Lemma 8. *The function $RT(E, \bullet)$ restricted to the interval $[\mathrm{re}(\tau_1(m)), \mathrm{re}(\tau_1(m+1)))$ for any $m \in \mathbb{N}$ is continuous and linearly decreasing. That is,*

$$RT(E, t) = RT(E, \mathrm{re}(\tau_1(m))) - (t - \mathrm{re}(\tau_1(m))) \tag{7}$$

for all $t \in [\mathrm{re}(\tau_1(m)), \mathrm{re}(\tau_1(m+1)))$ and $m \in \mathbb{N}$.

Proof. By considering Definitions 3 and 4, we know that $RT(E,t) = \ell(fac(E,t))$, with $fac(E,t) = (t, fc(E,j_1), p)$ being the immediate forward augmented job chain with index $j_1 = \min\{\xi \in \mathbb{N} \mid \mathrm{re}(\tau_1(\xi)) > t\}$, and with p being the time of the write-event of the last job of $fc(E,j_1)$. Since $t \in [\mathrm{re}(\tau_1(m)), \mathrm{re}(\tau_1(m{+}1)))$ by the lemma assumption, we know that j_1 must be $j_1 = \mathrm{re}(\tau_1(m{+}1))$. Hence, $fac(E,t) = (t, fc(E,m{+}1), p)$ holds.

When comparing $fac(E,t)$ with $fac(E,\mathrm{re}(\tau_1(m))) = (\mathrm{re}(\tau_1(m)), fc(E,m{+}1), p)$, we notice that they have both the same middle entry $fc(E,m{+}1)$ and therefore the same last entry p. Hence, by definition, the reaction time at $\mathrm{re}(\tau_1(m))$ is $RT(E,\mathrm{re}(\tau_1(m))) = \ell(fac(E,\mathrm{re}(\tau_1(m)))) = p - \mathrm{re}(\tau_1(m))$, and the reaction time at t is $RT(E,t) = p - t$. We obtain:

$$\begin{aligned} RT(E,t) - RT(E,\mathrm{re}(\tau_1(m))) &= (p-t) - (p - \mathrm{re}(\tau_1(m))) \\ &= -(t - \mathrm{re}(\tau_1(m))) \end{aligned}$$

By adding $RT(E,\mathrm{re}(\tau_1(m)))$ on both sides, we prove (7). □

Based on this lemma, the function $RT(E,\bullet)$ can be fully described by its function values $RT(E,t)$ in the points $t \in \{\mathrm{re}(\tau_1(m)) \mid m \in \mathbb{N}\}$. Such a set of points that fully describes the behavior of the reaction time we call set of *anchor points*.

Definition 9 (Anchor points for RT)**.** Given a cause-effect chain E, a set $\mathcal{A}(E) \subseteq \mathbb{R}_{\geq \mathrm{re}(\tau_1(W_1))} \times \mathbb{R}_{\geq 0}$ is called a set of anchor points for the reaction time of E, if for all $t > \mathrm{re}(\tau_1(W_1))$ there exists a *unique* anchor point $(x^t, y^t) \in \mathcal{A}(E)$ with $x^t = \max\{x \mid (x,y) \in \mathcal{A}(E), x \leq t\}$ (i.e., the *latest* anchor point no later than t), and

$$RT(E,t) = y^t - (t - x^t) \tag{8}$$

holds for all $t > \mathrm{re}(\tau_1(W_1))$.

There are several possible sets of anchor points, and one is already indicated by Lemma 8. That is, we can define a set of anchor points $\mathcal{A}^1(E)$ by considering all the time points where a read-event of τ_1 occurs. Formally, we define

$$\mathcal{A}^1(E) \coloneqq \left\{ \left(x, \ell(fac(E,x))\right) \,\middle|\, x = \mathrm{re}(\tau_1(m)), m \geq W_1 \right\}. \tag{9}$$

In Figure 1b, this set of anchor points for the example system is depicted by the black and red points, namely $\mathcal{A}^1(E) = \{(0{,}35), (6{,}29), (12{,}33), (18{,}27), (24{,}31), \ldots\}$.

While this set of anchor points can be used to describe the behavior of $RT(E,\bullet)$, it actually includes a lot of *redundant* anchor points (denoted by the small black points in Figure 1b) which can safely be removed. We can derive a minimal set of anchor points by removing the redundant anchor points from $\mathcal{A}^1(E)$. Formally, we define this minimal set of anchor points as the subset

$$\mathcal{A}^*(E) \subseteq \mathcal{A}^1(E) \tag{10}$$

which fulfills the conditions from Definition 9, such that for all $(x,y) \in \mathcal{A}^*(E)$ the set $\mathcal{A}^*(E) \setminus \{(x,y)\}$ is not a set of anchor points anymore. In Figure 1b, this set of minimal anchor points for the example system is depicted by the red points, namely $\mathcal{A}^*(E) = \{(0{,}35), (12{,}33), (24{,}31), \ldots\}$. In the following section, we demonstrate how to determine this set of anchor points efficiently by building a set of anchor points based on *partitioned job chains* proposed in [24] and by exploiting that the reaction time repeats after one hyperperiod.

V. Efficient Shape-Determination

After describing the shape of $RT(E,\bullet)$ by anchor points in the previous section, in this section we discuss the efficient determination of such sets of anchor points of an arbitrary chain. Again, while the focus here is on a single chain, the efficient shape-determination and the shape-aware analysis presented in the next section can be applied to a system of multiple chains by analyzing each chain independently.

We first start with the observation that while in general the number of anchor points of a set of recurrent tasks is infinite, in praxis, the reaction time function repeats every hyperperiod, and a finite set of anchor points is sufficient to describe the whole function.

Lemma 10. *Given a cause-effect chain E with hyperperiod $H(E)$, then $RT(E,t) = RT(E,t+H(E))$ holds for all time points $t \geq \mathrm{re}(\tau_1(W_1))$.*

Proof. Consider a time $t \geq \mathrm{re}(\tau_1(W_1))$ in which the external activity takes place. Let j_1 be the first read-event of τ_1 after t. Analogously, $j_1 + \frac{H(E)}{T_{\tau_1}}$ is the first read-event of τ_1 after $t + H(E)$. Since the pattern of read-events and write-events of each task τ_i repeats every hyperperiod $H(E)$ starting from $\tau_i(W_i)$, the immediate forward job chain $fc(E, j_1 + \frac{H}{T_{\tau_1}})$ is the same as $fc(E,j_1)$ shifted by one hyperperiod $H(E)$. Hence, if $fac(E,t) = (t, fc(j_1), p)$, then $fac(E, t+H(E)) = (t + H(E), fc(j_1 + \frac{H(E)}{T_{\tau_1}}), p + H(E))$, and $RT(E, t+H(E)) = (p + H(E)) - (t + H(E)) = p - t = RT(E,t)$. □

As a result of Lemma 10, it is sufficient to derive anchor points with first entry in the half-opened interval

$$I(E) \coloneqq \left[\mathrm{re}(\tau_1(W_1)),\, \mathrm{re}(\tau_1(W_1)) + H(E)\right), \tag{11}$$

and we denote such anchor points using the notation $\mathcal{A}(E)|_{I(E)}$. In particular, since the anchor points in $\mathcal{A}^1(E)$ correspond to the read-events of τ_1, the number of anchor points inside the intervals of interest is bounded by $\left|\mathcal{A}^*(E)|_{I(E)}\right| \leq \left|\mathcal{A}^1(E)\right|_{I(E)}\Big| = \frac{H(E)}{T_{\tau_1}}$, and we can determine the anchor points by constructing $\frac{H(E)}{T_{\tau_1}}$ many immediate forward augmented job chains.

In the remainder of this section, we show that it is actually sufficient to traverse only $\frac{H(E)}{\max_{i=1}^{n} T_{\tau_i}}$ many times through the task set to determine all relevant anchor points for the reaction time. Since real-world applications are often times based on (semi-)harmonic periods [38], this significantly reduces the runtime of determining anchor points. We achieve this by leveraging partitioned job chains, introduced in [24]. While it is already shown in [24, Theorem 14] that p-partitioned job chains can be utilized to describe the *maximum* reaction time, independent of the choice of p, in this work we show that they

can even be utilized to describe the *exact shape* of reaction time.

Definition 11 (Partitioned job chains)**.** Let $E = (\tau_1 \to \cdots \to \tau_n)$ be a cause-effect chain, $p \in \{1, \dots, n\}$, and $m \geq W_p$. We define the m-th p-partitioned job chain for E as

$$pc^p(E, m) \coloneqq \big(bc(E_p^-, m), fc(E_p^+, m+1)\big), \tag{12}$$

where $bc(E_p^-, m) = (\tau_1(j_1), \dots, \tau_p(j_p))$ is the immediate backward job chain for the cause-effect chain $E_p^- = (\tau_1 \to \cdots \to \tau_p)$ with $j_p = m$, and $fc(E_p^+, m+1) = (\tau_p(j'_p), \dots, \tau_n(j'_n))$ is the immediate forward job chain for $E_p^+ = (\tau_p \to \cdots \to \tau_n)$ with $j'_p = m+1$.

The length $\ell(pc^p(E,m))$ of the partitioned job chain is $\ell(pc^p(E,m)) \coloneqq end(pc^p(E,m)) - start(pc^p(E,m))$, with start and end defined by $start(pc^p(E,m)) \coloneqq \mathrm{re}(\tau_1(j_1))$ and $end(pc^p(E,m)) \coloneqq \mathrm{we}(\tau_n(j'_n))$, respectively.

Examples of partitioned job chains can be found in Figure 1c. The partitioned job chain $pc^2(E,1)$ consists of the immediate backward job chain $(\tau_1(0), \tau_2(1))$ and the immediate forward job chain $(\tau_2(2), \tau_3(6))$. From the image, we observe that the start time of partitioned job chains (i.e., $0, 12, 24, \dots$) corresponds to the x-value of the minimal anchor points. Similarly, the length (i.e., $35, 33, 31, \dots$) corresponds to the y-value of the minimal anchor points.

Building upon this observation, given an arbitrary cause-effect chain E and $p \in \{1, \dots, n\}$, we define a set

$$\mathcal{A}^p(E) \coloneqq \{(start(pc^p(E,m)), \ell(pc^p(E,m))) \,|\, m \in M^p\} \tag{13}$$

where M^p is the set of integers $m \geq W_p$ for which $start(pc^p(E, m+1)) \neq start(pc^p(E,m))$. In Theorem 12, we prove that this is a set of anchor points.

Please note that for $p = 1$ the definition from Equation (13) overloads the notation $\mathcal{A}^1(E)$ from Equation (9) on purpose, to reflect the fact that both definitions are equivalent. That is, $start(pc^1(E,m)) = \mathrm{re}(\tau_1(m))$ and

$$\ell(pc^1(E,m)) = \ell(fac(E, \mathrm{re}(\tau_1(m)))) \tag{14}$$

by [24, Corollary 10, 1.]. This means that the definitions $\mathcal{A}^1(E)$ from (13) and (9) can be used interchangeably.

Theorem 12 (Anchor points from partitioned job chains)**.** *Let $E = (\tau_1 \to \cdots \to \tau_n)$ be a cause-effect chain and $p \in \{1, \dots, n\}$. The set $\mathcal{A}^p(E)$ is a set of anchor points for the reaction time according to Definition 9.*

To prove Theorem 12, we follow a constructive argument. To show that $\mathcal{A}^p$ is a set of anchor points, we show that for an external activity at time t and m being the largest integer such that $start(pc^p(E,m)) \leq t$, $end(pc^p(E,m))$ is the time at which the data from t is fully processed. Hence, $RT(E,t) = \ell(pc^p(E,m)) - (t - start(pc^p(E,m)))$. We achieve this by comparing the forward job chain from $fac(E,t)$ with the backward job chains used in $pc^p(E,m)$ and $pc^p(E, m+1)$, using the following helper lemma. In particular, we leverage an observation from [24]: given an immediate forward chain and an immediate backward chain, if at some job index the forward is not later than the backward, this remains true for all larger indices; dually, if at some job index the backward is not earlier than the forward, this remains true for all smaller indices.

Lemma 13. *Given an immediate forward job chain $fc(E, j_1) = (\tau_1(j_1), \dots, \tau_n(j_n))$ and an immediate backward job chain $bc(E, j'_n) = (\tau_1(j'_1), \dots, \tau_n(j'_n))$ for a cause-effect chain E. If there exists an $i \in \{1, \dots, n\}$ such that $j_i \leq j'_i$, then $j_i \leq j'_i$ for all $i \in \{1, \dots, n\}$.*

Proof of Lemma 13. This follows from [24, Lemma 13]. Only for completeness, we repeat the lemma [24, Lemma 13]:

"*Let c be a job chain for E, let $\vec{c}$ be an immediate forward job chain for E, and let $\overleftarrow{c}$ be an immediate backward job chain for E.*

1) *If there exists $i \in \{1, \dots, |E|\}$ such that $\vec{c}(i) \preccurlyeq c(i)$, then $\vec{c}(j) \preccurlyeq c(j)$ for all $j \in \{i, \dots, |E|\}$.*
2) *If there exists $i \in \{1, \dots, |E|\}$ such that $c(i) \preccurlyeq \overleftarrow{c}(i)$, then $c(j) \preccurlyeq \overleftarrow{c}(j)$ for all $j \in \{1, \dots, i\}$."*[2]

By choosing $\vec{c} = fc(E, j_1)$ and $c = bc(E, j'_n)$ for the first result [24, Lemma 13, 1.], we obtain that $\vec{c}(\xi) \preccurlyeq c(\xi)$ for all $\xi \in \{i, \dots, n\}$ (in the notation of [24]), which translates to $j_\xi \leq j'_\xi$ for all $\xi \in \{i, \dots, n\}$ (in our notation).

Furthermore, by choosing $\overleftarrow{c} = bc(E, j'_n)$ and $c = fc(E, j_1)$ for the second result [24, Lemma 13, 2.] we obtain $j_\xi \leq j'_\xi$ for all $\xi \in \{1, \dots, i\}$ as well. □

With Lemma 13 we are equipped to prove Theorem 12.

Proof of Theorem 12. To prove the theorem, we need to show that $\mathcal{A}^p$ satisfies the conditions from Definition 9. Since the definition of M^p only allows partitioned job chains with different starting times, the uniqueness of (x^t, y^t) is ensured. Hence, it remains to prove Equation (8), i.e., we need to show that $RT(E,t) = y^t - (t - x^t)$ for all $t > \mathrm{re}(\tau_1(W_1))$.

To that end, let $t \geq \mathrm{re}(\tau_1(W_1))$ be the time of an external activity and (x^t, y^t) the corresponding anchor point. By definition, $y^t = \ell(pc^p(E,m))$ for an $m \in M^p$ such that $start(pc^p(E,m)) = x^t \leq t$ and $start(pc^p(E, m+1)) > t$. We use the following notation to denote the job indices of the different chains which are to be considered for the proof:

$$\begin{aligned} pc^p(E,m) &= ((\tau_1(b_1^1), \dots, \tau_p(b_p^1)), (\tau_p(f_p^1), \dots, \tau_n(f_n^1))) \\ pc^p(E,m{+}1) &= ((\tau_1(b_1^2), \dots, \tau_p(b_p^2)), (\tau_p(f_p^2), \dots, \tau_n(f_n^2))) \\ fac(E,t) &= (t, (\tau_1(j_1), \dots, \tau_n(j_n)), p) \end{aligned} \tag{15}$$

with $b_p^1 = m$ and $b_p^2 = m+1 = f_p^1$, by definition.

If we could show that $j_n = f_n^1$, then this would imply that $p = end(pc^p(E,m))$ and we would have proven (8) since:

$$\begin{aligned} y^t &= \ell(pc^p(E,m)) = p - start(pc^p(E,m)) && (16)\\ &= (t - start(pc^p(E,m))) + \ell(fac(E,t)) && (17)\\ &= (t - x^t) + RT(E,t) && (18) \end{aligned}$$

[2]We note that the notation $\vec{c}(j) \preccurlyeq c(j)$ and $c(j) \preccurlyeq \overleftarrow{c}(j)$ is used in [24] to specify that the j-th job of the job chain on the left-hand side occurs no later than the j-th job of the job chain on the right-hand side.

Hence, *it suffices to prove* $j_n = f_n^1$.

To show that $j_n = f_n^1$, we consider the following: Since $start(pc^p(E, m{+}1)) > t$, we know that $j_1 \leq b_1^2$. Therefore, by Lemma 13, also $j_p \leq b_p^2 = m+1 = f_p^1$. Furthermore, if $j_p < f_p^1$ would hold, then $j_p \leq b_p^1$ would hold, and by Lemma 13, $j_1 \leq b_1^1$, which contradicts $x^t \leq t < \mathrm{re}(\tau_1(j_1))$. Hence, $j_p = f_p^1$. This implies $j_n = f_n^1$ because both $(\tau_p(f_p^1), \ldots, \tau_n(f_n^1))$ and $(\tau_p(j_p), \ldots, \tau_n(j_n))$ are immediate forward job chains for $E_p^+ = (\tau_p \to \cdots \to \tau_n)$. Since, as argued above, $j_n = f_n^1$ is sufficient, this concludes the proof of Theorem 12 for the reaction time. □

Since the reaction time repeats every hyperperiod (according to Lemma 10), it is sufficient to only compute the anchor points in $\mathcal{A}^p(E)$ during the interval I. Since further the partitioned job chains repeat every hyperperiod, we know that $pc^p(E, W_p + \frac{H(E)}{T_{\tau_p}})$ starts at time $start(pc^p(E, W_p + \frac{H(E)}{T_{\tau_p}})) = start(pc^p(E, W_p)) + H(E)$. Therefore, only the first $\frac{H(E)}{T_{\tau_p}}$ partitioned job chains contribute to the anchor points in I, and the number of anchor points is bounded by $\left|\mathcal{A}^*(E)|_{I(E)}\right| \leq \left|\mathcal{A}^p(E)|_{I(E)}\right| = \frac{H(E)}{T_{\tau_p}}$. The determination of anchor points based on p-partitioned job chains is summarized in Algorithm 1.

Algorithm 1 Determination of anchor points.

Input: Cause-effect chain E, index $p \in \{1, \ldots, n\}$
Output: Set of anchor points $\mathcal{A}^p(E)|_{I(E)}$

1: Initialize $\mathcal{A}^p(E)|_{I(E)} \leftarrow \emptyset$
2: Determine the warm up W_p ▷ Definition 6
3: **for all** $m = W_p, \ldots, W_p + \frac{H(E)}{T_{\tau_p}} - 1$ **do**
4: Determine $pc^p(E, m)$ ▷ Definition 11
5: Add $(start(pc^p(E, m)), \ell(pc^p(E, m)))$ to $\mathcal{A}^p(E)|_{I(E)}$
6: **return** $\mathcal{A}^p(E)|_{I(E)}$

According to that algorithm, determining the shape of end-to-end latency can be done with the following complexity.

Theorem 14 (Complexity)**.** *Let E be a cause-effect chain of n tasks. The computational complexity for determining $\mathcal{A}^p(E)|_{I(E)}$ is $\mathcal{O}(n \cdot \frac{H(E)}{T_{\tau_p}})$. Specifically, by choosing p as the task with the maximal period, i.e., $T_p = \max_{i=1}^n T_{\tau_i}$, we can determine both sets of anchor points with a computational complexity of $\mathcal{O}(n \cdot \frac{H(E)}{\max_{i=1}^n T_{\tau_i}})$.*

Proof. The set of anchor points $\mathcal{A}^p(E)|_{I(E)}$ can be computed by constructing the partitioned job chains $pc^p(E, W_p), \ldots, pc^p(E, W_p + \frac{H(E)}{T_{\tau_p}} - 1)$. Each partitioned job chain is constructed by iterating through the task set once, as shown in [24]. Hence, the determination of anchor points can be done with computational complexity $\mathcal{O}(n \cdot \frac{H(E)}{T_{\tau_p}})$. □

In practice, tasks are often harmonic, max-harmonic or $(2, k)$-max-harmonic [27]. In such cases, the term $\frac{H(E)}{\max_{i=1}^n T_{\tau_i}}$ is bounded by a constant, and the computational complexity for determining the anchor points is linear.

VI. Shape-Aware Analysis

Assuming that the set of anchor points $\mathcal{A}$ is determined (e.g., using the results from the previous section), we discuss their application to derive diverse timing metrics of practical interest. Specifically, the metrics range from maximum, minimum, and average reaction time (Sections VI-B and VI-C), to throughput (Section VI-D), as well as weakly-hard and exceedance-based metrics (Sections VI-E and VI-F). While many of these metrics cover new ground in the sense that we are the first to provide a formal definition and an exact analysis, main purpose of this section is to showcase the potential of our approach in handling new types of metrics.

A. Preparation

The underlying premise of this section is that anchor points for the reaction time $\mathcal{A}$ are determined. This can be achieved by deriving $\mathcal{A}^p(E)|_{I(E)}$ using partitioned job chains, as in Section V, and by exploiting the repetitive nature of the end-to-end latency shape (formalized by Lemma 10). After removing redundant anchor points from $\mathcal{A}$, this defines the minimal set of anchor points $\mathcal{A}^*$.

For the feasibility of our analysis, we limit our attention to the anchor points within one hyperperiod. This is sufficient since the reaction time repeats after one hyperperiod (cf. Lemma 10). A possible candidate $I(E)$ for a hyperperiod interval is already defined by Eq. (11). However, for the convenience of our analysis, we would like to ensure that there are anchor points on the left and right border of the analysis interval, which is not inherently true for $I(E)$ since there might not be an anchor point at $\mathrm{re}(\tau_1(W_1)) + H(E)$ in $\mathcal{A}^*$. To mitigate this problem, we use the analysis interval

$$I'(E) \coloneqq \big[x_0,\, x_0{+}H(E)\big) \tag{19}$$

where x_0 is the minimal first entry of anchor point in $\mathcal{A}^*$ with $x_0 > \mathrm{re}(\tau_1(W_1))$. Since anchor points $\mathcal{A}^*$ correspond to the local maxima of the function $RT(E, \bullet)$ and the local maxima repeat every hyperperiod, this ensures that there are anchor points on the borders of $I'(E)$. Please note that we use $\overline{I'}(E) = \big[x_0,\, x_0{+}H(E)\big]$ to denote the closure of $I'(E)$. Specifically, $\mathcal{A}^*(E)|_{\overline{I'(E)}}$ contains the anchor points on the borders. We denote the anchor points by

$$\mathcal{A}^*(E)|_{\overline{I'(E)}} = \{(x_0, y_0), \ldots, (x_g, y_g)\} \tag{20}$$

ordered by the size of their x-entry, i.e., $x_{i-1} < x_i$ for all $i = 1, \ldots, g$.

B. Maximum and Minimum Reaction Time

Maximum and minimum latency of cause-effect chains are important parameters for cyber-physical systems, e.g., to ensure the stability of control systems. For example, Sun and Chen [57] analyze discrete-time systems with time-varying delay, which require both maximum and minimum latency guarantees. Sinnema and Maggio [55] analyze control performance under sensor data misalignment, for which both the maximum and minimum latency values for each sensor chain are relevant. While existing analyses only focus on the

computation of maximum end-to-end latency (e.g., [6], [11], [24], [37], [42]), here we demonstrate how our shape-aware approach can be used to also determine the *minimum* reaction time. *To the best of our knowledge, this is the first analysis for minimum reaction time of cause-effect chains under LET.* We start by defining the maximum and minimum reaction time.

Definition 15 (Maximum reaction time)**.** Given a cause-effect chain E the maximum reaction time is defined as $MaxRT(E) \coloneqq \sup_{t>\mathrm{re}(\tau_1(W_1))} RT(E,t)$.

Definition 16 (Minimum reaction time)**.** Given a cause-effect chain E the minimum reaction time is defined as $MinRT(E) \coloneqq \inf_{t>\mathrm{re}(\tau_1(W_1))} RT(E,t)$.

Since the anchor points pose the local maxima for the reaction time function and the reaction time repeats after one hyperperiod, the globally maximal values are determined by:

$$MaxRT(E) = \max_{i=0,\dots,g-1} y_i \tag{21}$$

The minimal reaction time is achieved at the end of linear extension starting from an anchor point (e.g., see Fig.1b), i.e.,

$$MinRT(E) = \min_{i=0,\dots,g-1} (y_i - (x_{i+1} - x_i)) \tag{22}$$

While the maximum reaction time presented here follows the classical definition by Davare et al. [15] and by Feiertag et al. [17], similar metrics with subtle distinctions have also been introduced. One example is the *maximum reduced reaction time* ($MaxRedRT$) [25], which does not count the time from the external activity but only starting from the data sampling:

$$MaxRedRT(E) = MaxRT(E) - T_1 \tag{23}$$

Another example is the *reactive time* ($Reac$) introduced by Sun et al. [58]. This metric focuses on the reaction time of those events that are actually propagated without being overwritten (cf. Definition 3 and Equation (4) in [58]). Hence, this metric corresponds to the maximal reaction time that occurs within one period T_1 before the actual data propagation. In terms of shape-aware analysis, the actual data propagation occurs at the end of the linear extension starting from an anchor point. Therefore, the reactive time is determined by:

$$Reac(E) = \max_{i=0,\dots,g-1} (y_i - (x_{i+1} - x_i)) + T_1 \tag{24}$$

C. Average Latency

Besides the maximum and minimum latency, average-case latency can be particularly relevant when the consistent responsiveness is important for user experience, e.g., in modern automotive infotainment systems, in which responsiveness to user inputs such as gestures or voice commands depend heavily on average-case behavior. While average-case latency has been formally defined in other contexts, e.g., network queuing theory [33], [40], *to the best of our knowledge, we are the first to formally define and analyze them in the context of cause-effect chains under LET.*

Definition 17 (Average latency)**.** Given a cause-effect chain E, the average reaction time (under a uniform distribution) is defined as:

$$AvRT(E) \coloneqq \frac{1}{H(E)} \int_{I'(E)} RT(E,t)\,dt \tag{25}$$

To calculate the average latency, we divide the full integral into subintervals $[x_i, x_{i+1})$. For each of these subintervals, the area below the function (recalling Figure 1b) is composed of a rectangle with base $x_{i+1} - x_i$ and with height $\hat{y}_i \coloneqq (y_i - (x_{i+1} - x_i))$, and a triangle on top of that with base $x_{i+1} - x_i$ and height $y_i - \hat{y}_i$. This results in a total area of $(x_{i+1} - x_i) \cdot (\hat{y}_i + \frac{y_i - \hat{y}_i}{2}) = (x_{i+1} - x_i) \cdot (\frac{y_i + \hat{y}_i}{2})$. We obtain:

$$\begin{aligned} AvRT(E) &= \frac{1}{H(E)} \sum_{i=0}^{g-1} \int_{[x_i, x_{i+1})} RT(E,t)\,dt \\ &= \frac{1}{H(E)} \sum_{i=0}^{g-1} (x_{i+1} - x_i) \cdot (\frac{y_i + \hat{y}_i}{2}) \\ &= \frac{1}{2H(E)} \sum_{i=0}^{g-1} (x_{i+1} - x_i) \cdot (y_i + \hat{y}_i) \end{aligned} \tag{26}$$

D. Throughput

While typically not interpreted as a real-time metric, the throughput plays a crucial role in the design of real-time systems. For example, vision-based pipelines capture images continuously, process them via neural networks, and forward the results to decision-making components. A recent example of a similar processing pipeline has been considered by Schowitz et al. [53] in the context of the NVIDIA Holoscan system, a novel embedded software stack for image streams, mainly targeting medical device applications. Similar pipelines also appear in autonomous driving, where video input is used for image segmentation and real-time control. In this context, a metric of interest is the *FPS* (frames per second), i.e., guaranteeing that the throughput is at least a certain number f of FPS, which corresponds to guaranteeing that at least f distinct data frames are processed every second. This translates well to the following throughput definition where we are interested in the rate of data samples that are processed without being overwritten.

Definition 18 (Throughput)**.** For a cause-effect chain E, the throughput is defined as the number of immediate forward job chains $fc(E, j_1) = (\tau_1(j_1), \dots, \tau_n(j_n))$ that end at different jobs $\tau_n(j_n)$ of τ_n, divided by the hyperperiod $H(E)$. Formally, we aggregate different job indices j_n that are reachable by immediate forward job chains (i.e., $\exists j_1 \colon fc(E, j_1) = (\tau_1(j_1), \dots, \tau_n(j_n))$) into a set, and divide by the hyperperiod:

$$Thr(E) \coloneqq \frac{1}{H(E)} \left| \left\{ j_n \;\middle|\; \begin{matrix} \exists j_1 \colon \mathrm{re}(\tau_1(j_1)) \in I'(E), \\ fc(E, j_1) = (\tau_1(j_1), \dots, \tau_n(j_n)), \end{matrix} \right\} \right| \tag{27}$$

The throughput can easily be computed leveraging the shape of the reaction time function by considering the cardinality of $\mathcal{A}^*$ in the interval $I'(E)$, i.e.,

$$Thr(E) = \frac{1}{H(E)} \cdot \left| \mathcal{A}^* \big|_{I'(E)} \right|, \tag{28}$$

since each minimal anchor point corresponds to one immediate forward job chain where no data is overwritten. While some works provide a specialized analysis [14], [21] to guarantee a certain throughput, e.g., based on weighted marked graphs Petri Nets [21], *they do not consider cause-effect chains under the LET paradigm.*

E. Weakly-Hard Chain-Level (m,k) Constraints

Weakly-hard (m, k) constraints [9] traditionally describe the number of deadline misses m that can be tolerated in any window of k subsequent jobs. While (m, k) constraints are practically useful for real-world systems, which in some cases do not require meeting hard deadlines [1], [47], *to the best of our knowledge, such metrics have not been formally defined in the context of cause-effect chains*, likely because weakly-hard real-time systems are hard to analyze even in classical conditions. For example, the state-of-the-art analysis for tasks under fixed-priority scheduling in [49] requires an optimization problem with more than 40 constraints. Similarly, the work in [48], which considers the impact of deadline misses of individual tasks on time-triggered chains (i.e., it does not define end-to-end weakly-hard latency metrics), requires a complex optimization problem. Differently, our shape-aware analysis provides a very convenient way to compute weakly-hard (m, k) reaction time.[3]

Intuitively, we say that E satisfies (m, k) reaction time constraint with latency bound B if among any k consecutive samples the system fully processes at least m of them within the end-to-end latency constraint B. Formally, we make the following definition.

Definition 19 (Weakly-hard (m, k) reaction time constraint)**.** A chain E satisfies (m, k) reaction time constraint with latency bound B if

$$\sup_{i>W_1} \sum_{\xi=i}^{i+k-1} 1 \cdot [\ell(fc(E, \xi)) > B] \leq m \tag{29}$$

where $[\ell(fc(E, \xi)) > B]$ denotes the Iverson bracket, which returns 1 if $\ell(fc(E, \xi)) > B$ and 0 otherwise.

Please note that our definition of weakly-hard reaction-time constraint does not involve modifying the system behavior (e.g., different from some previous work assuming the job-kill policy when dealing with (m, k) constraints at the level of individual tasks instances [46])), since the constraint is applied only at the chain level for the reaction time, with the task system assumed to be schedulable, i.e., individual task deadlines are satisfied. Hence, data propagation evolves independently of meeting or missing the bound B.

[3]While this section focuses on the typical definition of (m, k) constraints [9], we note that variants of the (m, k) constraint model, such as those that consider only consecutive deadlines misses [41], could also easily be supported by shape-aware analysis when treated in a similar manner.

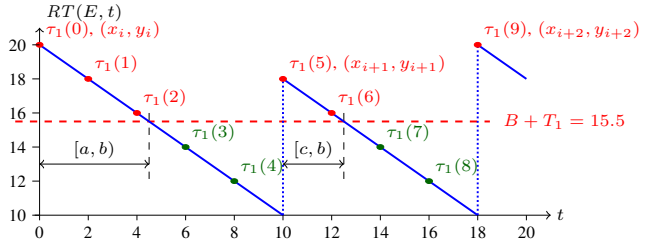


Figure 2: Illustration of reaction time between three consecutive anchor points (x_i, y_i) to (x_{i+2}, y_{i+2}). A threshold of 15.5 is highlighted in red. Red circles indicate jobs that exceed the threshold (i.e., violate the constraint), while green circles indicate jobs that meet it. The exceedance intervals $[a, b)$ and $[c, d)$ are additionally highlighted.

To check whether the (m, k) reaction-time constraint is fulfilled, first, we relate the definition to immediate forward *augmented* job chains, which are described by shape-aware analysis, through $\ell(fc(E, \xi)) = \ell(fac(E, \mathrm{re}(\tau_1(\xi)) - T_1)) - T_1 = \ell(fac(E, \mathrm{re}(\tau_1(\xi - 1)))) - T_1$. This leads us to an equivalent formulation of Equation (29), namely $\sup_{i \geq W_1} \sum_{\xi=i}^{i+k-1} 1 \cdot [\ell(fac(E, \mathrm{re}(\tau_1(\xi)))) > B + T_1] \leq m$.

Since the reaction time repeats every hyperperiod, it is sufficient to check only the immediate forward augmented job chains in the interval $I'(E)$. Let $\psi \in \mathbb{N}$ such that $x_0 = \mathrm{re}(\tau_1(\psi))$. Then we know that the interval $I'(E)$ contains $\mathrm{re}(\tau_1(\psi)), \mathrm{re}(\tau_1(\psi + 1)) \ldots, \mathrm{re}(\tau_1(\psi + \frac{H(E)}{T_1} - 1))$. Hence, we can write Equation (29) as:

$$\max_{i=0,\ldots,\frac{H(E)}{T_1}-1} \sum_{\xi=\psi+i}^{\psi+i+k-1} 1 \cdot [\ell(fac(E, \mathrm{re}(\tau_1(\xi)))) > B + T_1] \leq m \tag{30}$$

For this formula, ξ can only take values between $\psi + 0$ and $\psi + (\frac{H(E)}{T_1} - 1) + k - 1 = \psi + \frac{H(E)}{T_1} + k - 2$. Hence, only $[\ell(fac(E, \mathrm{re}(\tau_1(\xi)))) > B + T_1]$ with $\xi = \psi, \ldots, \psi + \frac{H(E)}{T_1} + k - 2$ are relevant. We store all of them in a list L using F in the i-th entry to denote a fail (i.e., $[\ell(fac(E, \mathrm{re}(\tau_1(\psi + i)))) > B + T_1] = 1$) and using P in the i-th entry to denote a pass (i.e., $[\ell(fac(E, \mathrm{re}(\tau_1(\psi + i)))) > B + T_1] = 1$). This list can exemplary look like $L = (F, F, P, P, F, \ldots, P)$ or equivalently represented using a sequence of interleaved pairs (a, P), (b, F) with $a, b \in \mathbb{N}$, representing the number of consecutive pass or fail events, as

$$L = ((2, F), (3, P), (2, F), \ldots, (1, P)). \tag{31}$$

To determine this list L, we exploit the property that between any two consecutive anchor points (x_i, y_i) and (x_{i+1}, y_{i+1}) in $\mathcal{A}^*(E)|_{\overline{I'(E)}}$ the reaction time linearly decreases according to Definition 9. As illustrated in Figure 2, the total number of read-events between two anchor points (x_i, y_i) and (x_{i+1}, y_{i+1}) is $N_i^{anc} := (x_{i+1} - x_i)/T_1$. Among these, the first $N_i^{fail} := \min\left(\left\lceil \frac{y_i - (B+T_1)}{T_1} \right\rceil, N_i^{anc}\right)$ lead to a fail entry

in L and the remaining $N_i^{pass} := N_i^{anc} - N_i^{fail}$ lead to a pass entry in L. Hence, we can derive L by iterating over the set of anchor points $\mathcal{A}^*(E)|_{\overline{I'(E)}}$ as summarized in Algorithm 2. Please note that starting from line 9, the repetitiveness of

Algorithm 2 Construct F/P list.

```
1:  L ← empty                                   ▷ Initialize empty list
2:  ℓ ← 0                                       ▷ Length 0
3:  for all i = 0, ..., g−1 do                  ▷ Fill using anchor points
4:      N_i^anc ← (x_{i+1} − x_i) / T_1
5:      N_i^fail ← min( ⌈(y_i − (B + T_1)) / T_1⌉ , N_i^anc )
6:      L.add((N_i^fail, F))
7:      L.add((N_i^anc − N_i^fail, P))
8:      ℓ ← ℓ + N_i^anc
9:  for all i = 0, 1, ... do                    ▷ Repeat until required length
10:     if ℓ ≥ H(E)/T_1 + k then
11:         break
12:     else
13:         L.add(L[i])
14:         ℓ ← ℓ + L[i][0]
```

the reaction time allows repeating the entries of L until the required length (i.e., $\frac{H(E)}{T_1} + k - 1$ fail and pass entries) is reached. Once the list L is created, it can be iterated[4] to check whether the (m, k) weakly-hard constraint is satisfied.

F. Longest Consecutive Exceedance

Similar to weakly-hard constraints, other forms of non-hard real-time metrics are often useful to be defined, as for example control systems can be tolerant to a certain degree of exceedance with respect to the latency bound B. Also in areas that involve humans, such as multimedia playback, temporary exceedance can be acceptable. For example, the study in [19] evaluates the impact of sensor to display delay in avionic scenarios, where the impact of the latency on pilot handling characteristics is evaluated. As an outcome, *desired* latency values for different operating conditions are identified. Kumar and Thiele [39] investigate transient overload and its effect on task response times. In this context, they propose a technique to compute the longest interval in which task deadlines are missed, as *settling time*. *To the best of our knowledge, a similar metric for cause-effect chains has so far not been captured by the literature.* Here, we define the metric of longest consecutive exceedance and show how it can be efficiently computed from the shape of the end-to-end latency.

Definition 20 (Longest consecutive exceedance)**.** Given a cause-effect chain E constraint with latency bound B. The longest consecutive exceedance of the reaction time, denoted $LE(E, B)$, is defined as the maximum duration $a \in \mathbb{R}_{\geq 0}$ such that there exists a time point $b \geq \mathrm{re}(\tau_1(W_1))$ for which the reaction time exceeds the constraint B continuously over the interval $(b, b + a)$, i.e., $RT(E, t) > B$ for all $t \in (b, b + a)$.

To compute the longest consecutive exceedance of the reaction time, we follow a procedure of four steps: First, we determine the exceedance intervals $EI(E, B)$ within $I'(E)$ by iterating over the anchor points $\mathcal{A}^*(E)|_{I'(E)}$:

$$EI(E,B) = \{[x_i, x_i + \min(y_i - B, x_{i+1} - x_i)) \\ | i = 0, \ldots, g-1, \quad y_i > B\} \quad (32)$$

Figure 2 depicts the first two exceedance intervals $[0, 4.5)$ and $[10, 12.5)$ of the reaction time with $B + T_1 = 15.5$. Second, since the reaction time repeats every hyperperiod[5], we can determine the exceedance intervals over two hyperperiods by shifting $EI(E, B)$:

$$EI^2(E,B) = EI(E,B) \\ \cup \{[c + H(E), d + H(E)) \,|\, [c, d) \in EI(E,B)\} \quad (33)$$

Third, we merge subsequent intervals in $EI^2(E, B)$ if their endpoints coincide, i.e., $[c_1, d_1)$ and $[c_2, d_2)$ become $[c_1, d_2)$ if $d_1 = c_2$. Fourth, we determine the longest exceedance as

$$LE(E,B) = \max\left\{d - c \,\middle|\, [c, d) \in EI^2(E,B)\right\}. \quad (34)$$

The computational complexity of all the analyzed metrics is polynomial in the number of anchor points, provided that the anchor points and hyperperiod are precomputed.

VII. Discussion on Enumeration-Based Approaches

Several enumeration-based approaches are proposed in literature to compute different maximum end-to-end latency metrics of cause-effect chains under LET communication. While early work in 2017–2020 analyzed the maximum reduced data age (MaxRedDA) [6], [37], [42], following the terminology of [25], we limit our discussion metrics for reaction time. Approaches by Kordon et al. [37] and by Günzel et al. [26] analyze the maximum reaction time (MaxRT) by enumerating forward job chains, i.e., constructing chains starting from the first task in the chain. Sun et al. [58] calculate the reactive time (Reac), as in Equation (24), based on backward job chains (called P-C chains in their work). The work by Günzel et al. [24] introduces an analysis based on partitioned job chains, i.e., starting from any intermediate job. While the approach presented in this work is also based on partitioned job chains, as anchors are determined by constructing partitioned job chains, the analysis covers previously analyzed metrics and offers numerous additional ones. A summary of the enumeration-based approaches analyzing metrics for reaction time is presented in Table I. Since each analysis determines the exact result of the corresponding metrics, we only conduct a runtime comparison in Section VIII-B with the most recent result of each approach, i.e., forward [26], backward [58], and partitioned [24].

[4]To iterate the list more efficiently, we can start iterating L only at entries of the form (b, F), i.e., entries corresponding to failures. This is safe because the maximal value in Equation (30) is obtained for i such that $\ell(fac(E, \mathrm{re}(\tau_1(\psi{+}i)))) > L + T_1$.

[5]Considering two hyperperiods for the longest consecutive exceedance is necessary because the longest exceedance interval can spread over the bound of one hyperperiod.

Table I: Comparison of approaches for enumeration-based analysis of reaction time metrics for LET chains.

Year	Paper	Approach	Metrics
2020	Kordon et al. [37]	Forward	MaxRT
2023	Günzel et al. [26]	Forward	MaxRT
2023	Sun et al. [58]	Backward	Reac
2023	Günzel et al. [24]	Partitioned	MaxRT, MaxRedRT
2026	**This paper**	Partitioned	MaxRT, MinRT, MaxRedRT, Reac, AvRT, Thr, (m,k), LE

VIII. Evaluation

In this section, we evaluate the different metrics using a Python3 implementation of shape-aware analysis[6]. Specifically, partitioned job chains are used to determine anchor points in $I(E)$ (cf. Section V), the anchor points are translated into anchors in $I'(E)$ (cf. Section VI-A), and the metrics presented in this paper are determined (cf. Sections VI-B–VI-F). The experiments are conducted on a dual-socket AMD EPYC 7742 system (128 cores, 256 threads, 251 GiB RAM, up to 2.25 GHz) running Debian GNU/Linux 12. Each analysis uses a single core, and only for the runtime experiments, multiple runs are parallelized. This section includes the evaluation using multiple case studies in Section VIII-A, and using synthetically generated cause-effect chains with two benchmarks in Sections VIII-B and VIII-C.

A. Case Studies

We now evaluate the different metrics for a large set of benchmarks from the literature, representing industrial case studies and applications to show real-world applicability of the methods of this paper. They are briefly described below:

- **Wat17:** The two chains with periodic tasks of the WATERS 2017 Challenge automotive case study by Bosch [29] (**Wat17-C1**–**Wat17-C2**).
- **Wat19:** Six chains from the WATERS 2019 Challenge prototype of a next-generation advanced driving assistance system by Bosch, described in detail in [51] (**Wat19-C1**–**Wat19-C6**).
- **RTSS21:** The five chains from the autonomous driving system presented by PerceptIn [50] in the RTSS 2021 Challenge (**RTSS21-C1**–**RTSS21-C5**).
- **APD:** The chain of the AUTOSAR Adaptive Platform brake assistant Demonstrator application, from [8].
- **Bec24:** The chain from the Brake-By-Wire application of a Swedish automotive Original Equipment Manufacturer (OEM), from the paper by Becker and Casini [3, Fig. 2].
- **Gem21:** A powertrain application from the paper by Gemlau et al. [20, Fig. 7] consisting of two chains, **Gem21-UP** (upper-path) and **Gem21-LP** (lower-path).
- **Iye20:** An Autonomous Emergency Braking System case study presented in [34, Fig. 7].

[6]The full implementation and data is available on GitHub https://github.com/marioguenzel/Shape-Aware-E2E and Zenodo https://doi.org/10.5281/zenodo.18753595

Table II: Metrics computed for each case study. The metrics **Max**(RT), **Min**(RT), **Av**(RT), and **LE** are in ms. **(m,k)** reports the lowest m that is fulfilled for $k = 10$. **(m,k)** and **LE** use the latency constraint $B = 0.95 \cdot \textbf{Max}$.

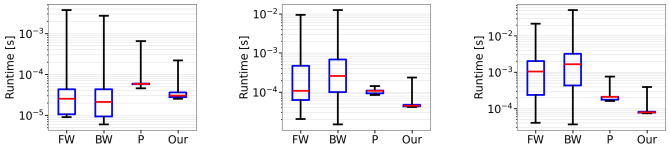
Case Study	Max	Min	Av	Thr	(m,k)	LE
Wat17-C1 [29]	50	40	45.0	0.100	(0,10)	2.50
Wat17-C2 [29]	212	112	162.0	0.010	(0,10)	10.60
Wat19-C1 [30]	908	470	689.0	0.003	(1,10)	45.40
Wat19-C2 [30]	855	445	650.0	0.003	(4,10)	42.75
Wat19-C3 [30]	65	45	55.0	0.067	(0,10)	3.25
Wat19-C4 [30]	98	53	75.5	0.030	(0,10)	4.90
Wat19-C5 [30]	164	86	125.0	0.015	(0,10)	8.20
Wat19-C6 [30]	430	220	325.0	0.005	(0,10)	21.50
RTSS-C1 [50]	610	510	560.0	0.010	(0,10)	30.50
RTSS-C2 [50]	608	476	542.0	0.010	(0,10)	30.40
RTSS-C3 [50]	710	610	660.0	0.010	(0,10)	35.50
RTSS-C4 [50]	410	310	360.0	0.010	(0,10)	20.50
RTSS-C5 [50]	320	220	270.0	0.010	(1,10)	16.00
APD [8]	275	225	250.0	0.020	(0,10)	13.75
Bec24 [3]	360	240	282.0	0.017	(0,10)	18.00
Gem21-UP [20]	19	13	16.0	0.200	(0,10)	0.95
Gem21-LP [20]	31	21	26.0	0.100	(0,10)	1.55
Iye20 [34]	360	310	335.0	0.020	(2,10)	18.00
Fre10-C1 [18]	45	35	40.0	0.100	(0,10)	2.25
Fre10-C2 [18]	35	25	30.0	0.100	(0,10)	1.75
Fre10-C3 [18]	55	45	50.0	0.100	(0,10)	2.75
Fre10-C4 [18]	45	35	40.0	0.100	(0,10)	2.25
Pag14-C1 [44]	70	50	60.0	0.050	(0,10)	3.50
Pag14-C2 [44]	50	30	40.0	0.050	(0,10)	2.50

- **Fre10:** Four chains from the engine control application case study from Frey [18]. (**Fre10-C1**–**Fre10-C4**).
- **Pag14:** The two chains of the ROSACE avionics case study of a flight controller [44](**Pag14-C1**–**Pag14-C2**).

Whenever a case study reported multiple chains with the same periods, deadlines and phases, we included only one in the evaluation since all of them would incur the same metric values under LET. If no phases or deadlines are reported, we assume synchronous releases and implicit deadlines.

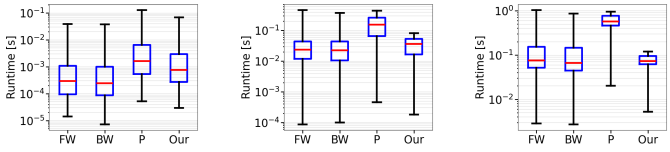
The results are summarized in Table II. Note that for the longest exceedance (LE) and the weakly-hard metric (m,k), we consider the latency constraint B set to 95% of the determined maximum reaction time. Furthermore, the weakly-hard metric reports the largest m for $k = 10$ such that the constraint is still fulfilled. MaxRedRT and Reac are evaluated as well but not reported due to space constraints. We observe that the distance between maximum and minimum reaction time differs quite a lot. Furthermore, there is a wide variety of (m, k) constraints that can be guaranteed to be fulfilled by our analysis, spanning from $m = 0$ to $m = 4$.[7] We note that the sum of the analyses of all the case studies took less than 0.062 seconds, and for 19 out of the 24 case studies $\mathcal{A}^*|_{I'(E)} = 3$ or less anchor points are sufficient to describe the full shape of maximum reaction time, demonstrating the usefulness of our approach.

[7]$m = 0$ is possible since (m,k) analyzes the exceedance of immediate forward job chains, which may be smaller than the maximum reaction time.



(a) $N = 5$. (b) $N = 20$. (c) $N = 50$.

Figure 3: Runtime comparison under the WATERS benchmark with different N of tasks per set.



(a) $N = 5$. (b) $N = 20$. (c) $N = 50$.

Figure 4: Runtime comparison under the UNIFORM benchmark with different N of tasks per set.

A more dedicated inspection of Table II reveals that there are interesting connections between the metrics:

- The average reaction time (**Av**) is the average of **Max** and **Min**.
- The longest exceedance (**LE**) is 5% of **Max**, while our latency constraint B is set to $100\% - 5\%$ of **Max**.

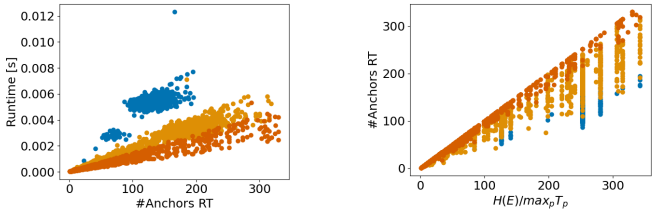
While first intuition tells us that these relations might be generally correct, further investigation is needed to argue about the generality of such statements. For example, when the latency constraint L is set to 80% of **Max**, we might jump to the conclusion that **LE** is 20% of **Max**. While this is true for many of the evaluated cases, it does not hold for all the case studies: With $B = 0.2 \cdot$ **Max**, we obtain **LE** $= 0.26 \cdot$ **Max** for Bec24, and even **LE** $= 3.78 \cdot$ **Max** for RTSS-C2. *We call for action to further investigate the connections drawn above and find requirements to generalize them.*

B. Runtime Comparison with Enumeration-Based Approaches

To compare the runtime with other enumeration-based approaches, we use synthetically generated cause-effect chains E with a given length N based on two benchmarks: **(i)** WATERS: Periods T_i are drawn according to the [38, Table III][8]; **(ii)** Uniform: Periods T_i are drawn uniformly at random from the set $\{10, 20, \ldots, 200\}$. In both cases, tasks are chosen to have implicit deadlines and phases drawn uniformly at random from $\{0, 1, \ldots, T_i - 1\}$. Since enumeration-based approaches are quite sensitive to the size of the hyperperiod, we only allow cases with $H(E) \leq 10^6$. We compare our analysis (**Our**) with approaches using forward enumeration (**FW**) [26], backward enumeration (**BW**) [58] and enumeration of partitioned job chains (**P**) [24], as discussed in Section VII. We measure the median runtime over 50 runs. The results are depicted in Figures 3 and 4, using boxplots to illustrate median, quartiles, and all values. We observe that our approach is quite efficient, especially given that it calculates all the metrics reported in Section VI while the enumeration-based approaches only determine one metric. We note that all literature results coincide with the corresponding metric from our analysis.

C. Controlling the Number of Anchor Points

To showcase how to control the runtime by the number of anchor points, we additionally evaluate the relation with the parameter $\frac{H(E)}{\max_p T_p}$ up to a value of 350. We report the median runtime over 50 runs using 1000 samples for each $N = 5, 20, 50$. Figure 5a illustrates the runtime over the number of anchor points, and Figure 5b illustrates the number of anchor points over the value $\frac{H(E)}{\max_p T_p}$. The colors red, orange, and blue show the results for $N = 5, 20, 50$, respectively. We observe that the runtime increases with the number of anchor points. Furthermore, by controlling the size of $\frac{H(E)}{\max_p T_p}$, the number of anchor points and hence the runtime can be controlled. We only present the plots for the Uniform benchmark, because under the WATERS benchmark the plots are less insightful: all chains only have one or two anchor points and the runtime never exceeds $4 \cdot 10^{-4}$ seconds.



(a) Runtime over number of anchor points.

(b) Number of anchor points over $\frac{H(E)}{\max_p T_p}$.

Figure 5: Runtime evaluation of the shape-aware analysis. The runtime behaves roughly linear with the number of anchor points. The number of anchor bounds is bounded by $\frac{H(E)}{\max_p T_p}$.

IX. Conclusion

This paper presented a shape-aware analysis which uses a description by anchor points to determine the shape of reaction time in LET-based cause-effect chains, laying the foundation for analyzing a rich portfolio of timing metrics of a modern real-time system. While this paper proposes concrete definitions and efficient analyses of *minimum*, *maximum*, and *average* reaction time, *throughput*, *longest consecutive exceedance*, and *chain-aware weakly-hard (m,k) constraints* on reaction time, it also opens the door for several further advancements, such as the definition of other timing metrics or the exploitation of shape information to simplify design choices. The shape-aware analysis of data-age-related metrics (following similar principles) and the exploration of relations between different metrics will be conducted in future work due to space constraints.

[8]Please note that the probabilities only sum up to 85% because 15% are reserved for angle-synchronous tasks which we do not consider. Therefore, in the generation all values are divided by 0.85.

ACKNOWLEDGMENT

This result is part of a project (PropRT) that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 865170). This work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project No. 569077889 (PEACH). This work was partially funded by the Swedish Research Council (VR) under the project nr. 2023-04773 and by Sweden's Innovation Agency via the NFFP8 project 2024-01267: PARTI. This work has been partially supported by the project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union – NextGenerationEU.

REFERENCES

[1] M. Andreozzi, G. Gabrielli, B. Venu, and G. Travaglini. Industrial challenge 2022: A high-performance real-time case study on arm. In M. Maggio, editor, *34th Euromicro Conference on Real-Time Systems (ECRTS 2022)*, volume 231 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

[2] S. Baruah, M. Bertogna, and G. Buttazzo. *Multiprocessor scheduling for real-time systems*. Springer, 2015.

[3] M. Becker and D. Casini. The material framework: Modeling and automatic code generation of edge real-time applications under the qnx rtos. *Journal of Systems Architecture*, 154:103219, 2024.

[4] M. Becker, D. Dasari, S. Mubeen, M. Behnam, and T. Nolte. Synthesizing job-level dependencies for automotive multi-rate effect chains. In *Proceedings of the 22nd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 159–169. IEEE, 2016.

[5] M. Becker, D. Dasari, S. Mubeen, M. Behnam, and T. Nolte. End-to-end timing analysis of cause-effect chains in automotive embedded systems. *Journal of Systems Architecture*, 80:104–113, 2017.

[6] M. Becker, D. Dasari, S. Mubeen, M. Behnam, and T. Nolte. End-to-end timing analysis of cause-effect chains in automotive embedded systems. *Journal of Systems Architecture - Embedded Systems Design*, 80:104–113, 2017.

[7] D. Bellassai, A. Biondi, A. Biasci, and B. Morelli. Supporting logical execution time in multi-core POSIX systems. *Journal of Systems Architecture*, 144:102987, 2023.

[8] D. Bellassai, C. Scordino, D. Casini, and A. Biondi. AP-LET: Enabling deterministic pub/sub communication in autosar adaptive. *Journal of Systems Architecture*, 162:103390, 2025.

[9] G. Bernat, A. Burns, and A. Liamosi. Weakly hard real-time systems. *IEEE Transactions on Computers*, 50(4):308–321, 2001.

[10] E. Bini, P. Pazzaglia, and M. Maggio. Zero-jitter chains of periodic let tasks via algebraic rings. *IEEE Transactions on Computers*, 72(11):3057–3071, 2023.

[11] E. Bini, P. Pazzaglia, and M. Maggio. Zero-jitter chains of periodic let tasks via algebraic rings. *IEEE Transactions on Computers*, 72(11):3057–3071, 2023.

[12] A. Biondi and M. Di Natale. Achieving predictable multicore execution of automotive applications using the LET paradigm. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 240–250, 2018.

[13] G. C. Buttazzo. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Third Edition*. Springer, 2011.

[14] A. Dasdan and R. Gupta. Faster maximum and minimum mean cycle algorithms for system-performance analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(10):889–899, 1998.

[15] A. Davare, Q. Zhu, M. D. Natale, C. Pinello, S. Kanajan, and A. L. Sangiovanni-Vincentelli. Period optimization for hard real-time distributed automotive systems. In *Design Automation Conference, DAC*, pages 278–283, 2007.

[16] M. Dürr, G. von der Brüggen, K.-H. Chen, and J.-J. Chen. End-to-end timing analysis of sporadic cause-effect chains in distributed systems. *ACM Trans. Embedded Comput. Syst. (Special Issue for CASES)*, 18(5s):58:1–58:24, 2019.

[17] N. Feiertag, K. Richter, J. Nordlander, and J. Jonsson. A compositional framework for end-to-end path delay calculation of automotive systems under different path semantics. In *Workshop on Compositional Theory and Technology for Real-Time Embedded Systems*, 2009.

[18] P. Frey. Ulmer Informatik Berichte Nr 2010-03 - Case Study: Engine Control Application. Technical report, University Ulm, 04 2010.

[19] J. D. Funk Jr, C. P. Beck, and J. B. Johns. Primary display latency criteria based on flying qualities and performance data. In *NASA. Ames Research Center, Piloting Vertical Flight Aircraft: A Conference on Flying Qualities and Human Factors*, 1993.

[20] K.-B. Gemlau, L. KÖHLER, R. Ernst, and S. Quinton. System-level logical execution time: Augmenting the logical execution time paradigm for distributed real-time automotive software. *ACM Trans. Cyber-Phys. Syst.*, 5(2), Jan. 2021.

[21] A. Ghamarian, M. Geilen, S. Stuijk, T. Basten, B. Theelen, M. Mousavi, A. Moonen, and M. Bekooij. Throughput analysis of synchronous data flow graphs. In *Sixth International Conference on Application of Concurrency to System Design (ACSD'06)*, pages 25–36, 2006.

[22] M. Günzel, K. Chen, N. Ueter, G. von der Brüggen, M. Dürr, and J. Chen. Timing analysis of asynchronized distributed cause-effect chains. In *RTAS*, pages 40–52. IEEE, 2021.

[23] M. Günzel, K. Chen, N. Ueter, G. von der Brüggen, M. Dürr, and J. Chen. Compositional timing analysis of asynchronized distributed cause-effect chains. *ACM Trans. Embed. Comput. Syst.*, 22(4):63:1–63:34, 2023.

[24] M. Günzel, H. Teper, K. Chen, G. von der Brüggen, and J. Chen. On the equivalence of maximum reaction time and maximum data age for cause-effect chains. In *35th Euromicro Conference on Real-Time Systems (ECRTS)*, 2023.

[25] M. Günzel, H. Teper, G. von der Brüggen, and J. Chen. End-to-end latency of cause-effect chains: A tutorial. *ACM Trans. Embed. Comput. Syst.*, 24(1):22:1–22:18, 2025.

[26] M. Günzel, N. Ueter, K. Chen, and J. Chen. Timing analysis of cause-effect chains with heterogeneous communication mechanisms. In *Proceedings of the 31st International Conference on Real-Time Networks and Systems, RTNS 2023, Dortmund, Germany, June 7-8, 2023*, pages 224–234. ACM, 2023.

[27] M. Günzel and M. Becker. Optimal task phasing for end-to-end latency in harmonic and semi-harmonic automotive systems. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2025.

[28] A. Hamann, D. Dasari, S. Kramer, M. Pressler, and F. Wurst. Communication centric design in complex automotive embedded systems. In *Euromicro Conference on Real-Time Systems, ECRTS*, pages 10:1–10:20, 2017.

[29] A. Hamann, D. Dasari, S. Kramer, M. Pressler, F. Wurst, and D. Ziegenbein. Waters industrial challenge 2017. In *International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, 2017.

[30] A. Hamann, D. Dasari, F. Wurst, I. Sañudo, N. Capodieci, P. Burgio, and M. Bertogna. Waters industrial challenge 2019. In *Proceedings of the 10th International Workshop on Analysis Tools and Methodologies for Embedded and Real-Time Systems (WATERS)*, 2019.

[31] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst. System level performance analysis - the SymTA/S approach. *IEEE Proceedings - Computers and Digital Techniques*, March 2005.

[32] T. A. Henzinger, B. Horowitz, and C. M. Kirsch. Embedded control systems development with giotto. In *Proceedings of the ACM SIGPLAN workshop on Languages, compilers and tools for embedded systems*, pages 64–72, 2001.

[33] N. Hohn, D. Veitch, K. Papagiannaki, and C. Diot. Bridging router performance and queuing theory. SIGMETRICS '04/Performance '04, page 355–366, New York, NY, USA, 2004. Association for Computing Machinery.

[34] P. Iyenghar, L. Huning, and E. Pulvermueller. Automated end-to-end timing analysis of autosar-based causal event chains. In *ENASE*, pages 477–489, 2020.

[35] C. M. Kirsch and A. Sokolova. The logical execution time paradigm. In *Advances in Real-Time Systems*, pages 103–120. Springer, 2012.

[36] T. Kloda, A. Bertout, and Y. Sorel. Latency analysis for data chains of real-time periodic tasks. In *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, pages 360–367, 2018.

[37] A. M. Kordon and N. Tang. Evaluation of the age latency of a real-time communicating system using the let paradigm. In *ECRTS 2020*, volume 165. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2020.

[38] S. Kramer, D. Ziegenbein, and A. Hamann. Real world automotive benchmarks for free. In *International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, 2015.

[39] P. Kumar and L. Thiele. Quantifying the effect of rare timing events with settling-time and overshoot. In *2012 IEEE 33rd Real-Time Systems Symposium*, pages 149–160. IEEE, 2012.

[40] J. P. Lehoczky. Real-time queueing network theory. In *Proceedings Real-Time Systems Symposium*, pages 58–67. IEEE, 1997.

[41] M. Maggio, A. Hamann, E. Mayer-John, and D. Ziegenbein. Control-system stability under consecutive deadline misses constraints. In *32nd Euromicro Conference on Real-Time Systems (ECRTS 2020)*, volume 165 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

[42] J. Martinez, I. Sañudo, and M. Bertogna. Analytical characterization of end-to-end communication delays with logical execution time. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2244–2254, 2018.

[43] J. Martinez, I. Sañudo, and M. Bertogna. End-to-end latency characterization of task communication models for automotive systems. *Real-Time Systems*, 56:315–347, 2020.

[44] C. Pagetti, D. Saussié, R. Gratia, E. Noulard, and P. Siron. The ROSACE case study: From simulink specification to multi/many-core execution. In *19th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 309–318, 2014.

[45] F. Paladino, A. Biondi, E. Bini, and P. Pazzaglia. Optimizing per-core priorities to minimize end-to-end latencies. In *36th Euromicro Conference on Real-Time Systems (ECRTS 2024)*.

[46] P. Pazzaglia, A. Biondi, and M. Di Natale. Optimizing the functional deployment on multicore platforms with logical execution time. In *2019 IEEE Real-Time Systems Symposium (RTSS)*, pages 207–219, 2019.

[47] P. Pazzaglia, A. Hamann, D. Ziegenbein, and M. Maggio. Adaptive design of real-time control systems subject to sporadic overruns. In *2021 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 1887–1892, 2021.

[48] P. Pazzaglia and M. Maggio. Characterizing the effect of deadline misses on time-triggered task chains. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(11):3957–3968, 2022.

[49] P. Pazzaglia, Y. Sun, and M. D. Natale. Generalized weakly hard schedulability analysis for real-time periodic tasks. *ACM Transactions on Embedded Computing Systems (TECS)*, 20(1):1–26, 2020.

[50] PerceptIn. Rtss 2021 industry challenge. In *Proceedings of the 42nd IEEE Real-Time Systems Symposium (RTSS)*, 2021. https://2021.rtss.org/wp-content/uploads/2021/06/RTSS2021-Industry-Challenge-v2.pdf.

[51] F. Rehm, D. Dasari, A. Hamann, M. Pressler, D. Ziegenbein, J. Seitter, I. Sañudo, N. Capodieci, P. Burgio, and M. Bertogna. Performance modeling of heterogeneous HW platforms. *Microprocessors and Microsystems*, 87:104336, 2021.

[52] J. Schlatow, M. Möstl, S. Tobuschat, T. Ishigooka, and R. Ernst. Data-age analysis and optimisation for cause-effect chains in automotive control systems. In *IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 1–9, 2018.

[53] P. Schowitz, S. Sinha, and A. Gujarati. Response-time analysis of a soft real-time nvidia holoscan application. In *2024 IEEE Real-Time Systems Symposium (RTSS)*, pages 57–69, 2024.

[54] G. Sciangula, D. Casini, A. Biondi, and C. Scordino. End-to-end latency optimization of thread chains under the dds publish/subscribe middleware. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2024.

[55] Y. Sinnema and M. Maggio. Analysis of control systems under sensor timing misalignments. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2025.

[56] C. Sofronis, S. Tripakis, and P. Caspi. A memory-optimal buffering protocol for preservation of synchronous semantics under preemptive scheduling. In *Proceedings of the 6th ACM & IEEE International conference on Embedded software*, pages 21–33, 2006.

[57] J. Sun and J. Chen. A note on stability and stabilization of discrete-time systems with time-varying delay. In *Proceedings of the 31st Chinese Control Conference*, pages 1430–1434, 2012.

[58] J. Sun, K. Duan, X. Li, N. Guan, Z. Guo, Q. Deng, and G. Tan. Real-time scheduling of autonomous driving system with guaranteed timing correctness. In *29th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2023, San Antonio, TX, USA, May 9-12, 2023*, pages 185–197. IEEE, 2023.

[59] H. Teper, M. Günzel, N. Ueter, G. von der Brüggen, and J.-J. Chen. End-to-end timing analysis in ros2. In *2022 IEEE Real-Time Systems Symposium (RTSS)*, pages 53–65, 2022.

[60] S. Wang, D. Li, A. H. Sifat, S.-Y. Huang, X. Deng, C. Jung, R. Williams, and H. Zeng. Optimizing logical execution time model for both determinism and low latency. In *2024 IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 135–148, 2024.