Jitter Propagation in Task Chains

Wang, Shumo; Bini, Enrico; Deng, Qingxu ; Maggio, Martina

Published in:
2025 IEEE Real-Time Systems Symposium (RTSS)

2025

Link to publication

Citation for published version (APA):
Wang, S., Bini, E., Deng, Q., & Maggio, M. (2025). Jitter Propagation in Task Chains. In *2025 IEEE Real-Time Systems Symposium (RTSS)*

Total number of authors:
4

Jitter Propagation in Task Chains

Shumo Wang*, Enrico Bini†, Qingxu Deng*, Martina Maggio‡
*Northeastern University (China), †University of Turin (Italy) ‡Saarland Univ. (Germany) & Lund University (Sweden)

***Abstract*—Chains of tasks are ubiquitous and used in a broad spectrum of applications. In these chains, tasks execute according to their timing. Then, they communicate by writing to and reading from shared memory. The schedule of tasks and the read/write instants are naturally subject to uncertainties (variability in the execution time, interference due to shared resources of higher priority tasks, etc.). Despite the impact of uncertainties, we believe that current analysis of task chains cannot handle them properly. In this paper, we borrow the notion of jitter to model uncertainties and we propose a novel event model that explicitly captures jitter in read and write operations, decoupled from task scheduling. We develop a (linear-time complexity) compositional analysis framework that tracks how this jitter propagates across chains and impacts metrics such as reaction time, data age, and end-to-end latency. Our model supports arbitrary communication paradigms (e.g., implicit, LET, mid-execution) and is applicable to the analysis of real-world frameworks such as ROS2 without requiring intrusive changes.**

I. Introduction

The split of a complex activity into an ordered sequence of tasks is a fundamental paradigm in distributed systems. This approach is crucial for large-scale computation because it enables parallel processing, workload distribution, and fault tolerance. By structuring software as *chains of tasks*, systems can leverage modularity and scalability, ensuring that each task incrementally contributes to the final outcome. A clear example of this approach is a computer vision pipeline, largely used in real-time systems such as in autonomous driving.

The existence of a task chain implies some form of communication between tasks. Tasks communicate using different mechanisms, broadly classified as *synchronous*, where tasks block on communication, or *asynchronous*, where tasks do not block. In synchronous communication, a reader (or consumer) task is activated only when new data produced by tasks that communicate with the consumer is available. This activation behavior corresponds to the task being blocked (i.e., waiting) until the required data arrives. In practice, the task may be scheduled, but will yield until the data is available. This means its timing depends on the tasks it reads from. In asynchronous communication, tasks run independently of data availability. This raises research challenges related to data propagation along the chain and the time required for this propagation. This delay has been extensively studied under various terms such as *reaction time*, *data age*, and *end-to-end latency*. These studies are particularly relevant to the automotive industry [1] and the ROS2 communication model [2], [3].

The execution of tasks (in chains or elsewhere) is inherently subject to variability. Execution times can fluctuate due to processor states (e.g., cache memory) or interference over shared resources. The uncertainty increases further when tasks

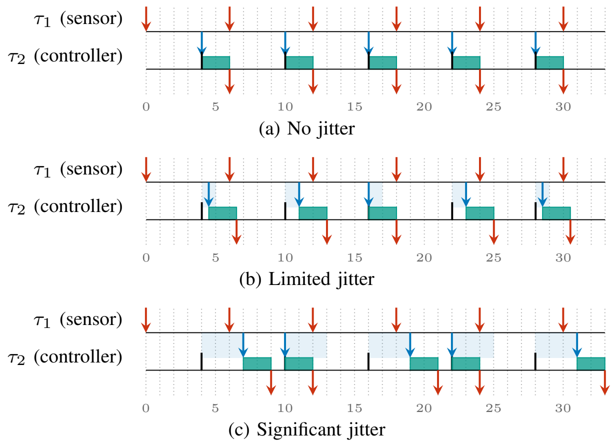


Fig. 1: Impact of jitter in data communication. The chain includes a sensor task τ_1 (period $T = 6$, start at 0) and a controller task τ_2 (period 6, start at 4, execution time $C = 2$). Without jitter (Fig. 1a), the sensor-to-actuator delay is a constant 6. With small jitter (Fig. 1b), the delay ranges from 6 to 7 due to τ_2's read jitter. With jitter 3 (Fig. 1c), some data may be overwritten (e.g., at times 6 or 18). The worst case occurs when τ_2's reads alternate between earliest and latest instants. Down arrows mark data acquisition/release; shaded areas show the possible read interval.

run on remote nodes, such as in cloud computing. We model such variability as *jitter*, i.e., bounded deviations from strictly periodic behavior. For example, release jitter models bounded uncertainty in the instants when tasks are activated [4]. Other examples include variability in read/write instants due to communication delays introduced by network transmission, or mid-execution data access in multi-threaded tasks.

We now discuss an example with a chain that connects a sensor task τ_1 with a controller task τ_2. Task τ_1 reads a physical quantity from the environment every period T, and writes the quantity in memory where the controller task can retrieve and use it. In the example we assume that the operation of τ_1 is instantaneous. Task τ_2 reads the data from memory with the same period T (but not synchronized with the period of τ_1) and then produces an actuation signal to be applied in the physical world. For simplicity, the example makes the assumption that the controller is executed in isolation (i.e., with no interference) and takes a constant processing time.

Fig. 1 shows the execution in three alternative scenarios: without any jitter (Fig. 1a), with small jitter (Fig. 1b), and with a larger jitter (Fig. 1c). In the ideal scenario with no jitter, the controller acquires all data sampled every 6 time units and the sensor-to-actuator delay is constant (and equal to 6 time units). When the jitter is small, the read jitter of τ_2 is transferred to the chain delay, with the longest possible delay occurring when τ_2 reads the latest. When the read jitter of τ_2 increases, a new behavior may occur: not all data sampled by τ_1 is actually used by τ_2. In the figure, the data reads that τ_1 performs at times $0, 12, \ldots$ may be overwritten by the following sample. In this scenario, the controller task τ_2 can use only half of the sampled data, which may lead to control problems. Also, we observe that the read instants of τ_2 leading to this particular scenario are alternating between the latest and the earliest possible within the jitter interval. This example helps to set precisely the subject of our investigation, which is the analysis of task chains with jitter in both the read and write instants.

A. *Contribution*

In this paper we propose a general event model for read/write instants, which accounts for jitter. We develop a compositional analysis for our events along a cause-effect chain. Specifically, the key contributions of the paper are: (i) an event model for periodic tasks that includes explicit jitter in read/write events, independent of the scheduling model, (ii) the definition of effective write/read sequence to isolate the subset of events that contribute to actual communication, (iii) the introduction of formal conditions and constructive algorithms to derive a reduced, jitter-aware event model for task chains, and (iv) a compositional method to combine per-task jitter effects across an entire chain.

Our model departs from established communication paradigms such as Logical Execution Time (LET) [5], [6]. LET enforces fixed read/write instants within each task, thereby removing jitter at the task level. This combines task scheduling with communication timing, and does not account for uncertainties. Furthermore, even if each task's events are perfectly periodic, differences in task periods or offsets can still cause variability at the chain level, a phenomenon that our model captures explicitly. Our approach retains the LET principle of decoupling communication timing from task execution, but generalizes it by allowing bounded jitter in both read and write events. This extension enables analysis of more realistic systems in which communication is not perfectly periodic, while still supporting the compositional reasoning benefits of LET.

B. *Related work*

The analysis of distributed real-time systems has been an active research area for a long time. Bettati and Liu [7] studied a flow-shop problem with end-to-end deadlines. Tindell and Clark introduced the *holistic analysis*, which evaluates both processor schedulability and message transmission over a bus [4]. Sun and Liu [8] proposed several synchronization protocols to regulate task releases while enforcing the precedence constraints of chains. Palencia and Gonzalez-Harbor [9] enhanced holistic analysis by incorporating per-processor priorities. With EDF-scheduled transactions, Pellizzoni and Lipari [10] demonstrated that assigning offsets, rather than allowing jitter to propagate, leads to performance advantages.

Beyond holistic analysis, several studies have focused on alternative approaches for analyzing real-time systems. Henia and Ernst [11] introduced event-stream analysis, which accounts for the co-scheduling of tasks within the same chain on a shared CPU. Schliecker and Ernst [12] proposed an iterative method for computing end-to-end latencies, considering both path forking and merging. Schlatow and Ernst [13] extended busy-window analysis to task chains in static-priority preemptive systems. Kurtin et al. [14] refined the estimation of buffer sizes and latencies by leveraging offsets, and Choi et al. [15] used response-time analysis to compute inter/intra-graph interference.

In event-triggered chains, tasks are activated by the output of preceding tasks, a form of synchronous communication in which the triggering condition is the arrival of a specific event (e.g., a data item or signal). For event-triggered processing chains with limited buffer sizes, [16] developed a framework to analyze non-skippable delays using real-time calculus. Furthermore, [17] studied the reaction time of event-triggered processing chains under finite buffer constraints.

Asynchronous communication, with non-blocking read/write over shared memory, has become increasingly relevant, particularly in automotive and robotic applications. In this case, tasks are normally activated by timers, forming time-triggered task chains. In [18], an upper bound on the maximum reaction time for periodic task chains was proposed. Durr et al. [19] derived upper bounds for reaction time and data age in sporadic task chains using forward and backward job analysis. In [20], an upper bound on end-to-end latency was analyzed by dividing cause-effect chains into sub-chains. Sinha and West used constraint programming to find execution times and periods of pipelined tasks that meet the end-to-end constraints and schedulability requirements [21].

A prominent application of time-triggered task chains is the Robot Operating System 2 (ROS2) [22]. In [23], upper bounds on end-to-end timing for systems on a single ECU were provided. In [24], the analysis was extended to multi-executor ROS2 systems, determining upper bounds for reaction time and data age. Tang et al. [25] leveraged data refresh semantics to optimize end-to-end latency and derived buffer size configurations that minimize latency. Most of the aforementioned studies focus on end-to-end latency while neglecting the effects of jitter, which is shown to be relevant in applications such as control systems [26].

One commonly used approach to mitigate jitter is the adoption of the LET paradigm [5], [6]. Several works have explored the impact of LET on real-time systems. Among these, Kordon and Tang [27] analyzed the maximum data age in LET-based periodic task communication, while [28] developed a bound for the reaction time of sporadic tasks under LET constraints. Bini et. al [29] proposed to eliminate the jitter in chains by purposely inserting a copier task. LET-based models, however, assume that read/write operations occur at predefined moments, such as task release times in LET mode or task start times in implicit communication mode.

A similar assumption exists in ROS2 research [30], where jitter mitigation relies on reserving server completion LET but requires modifications to ROS2 itself rather than analyzing the framework's existing capabilities.

Other work considers a mix of event-triggered, time-triggered, and data-fusion mechanisms [23], [31], [32]. They analyze task chains composed of multiple triggering methods. However, they often rely on fixed read/write timing assumptions or fail to fully account for jitter. In this paper, we aim at proposing a model and an analysis that overcomes this limitation and fully considers the read and write jitter of tasks.

II. System model

In this paper, we use $\mathbb{N}$ to refer to the set of natural numbers, with $0 \in \mathbb{N}$, and $\mathbb{R}$ for the reals. We use

$$\lfloor x \rceil_m = x - \left\lfloor \frac{x}{m} \right\rfloor m. \tag{1}$$

to denote the modulo-m operator for any real valued $x \in \mathbb{R}$ and $m \in \mathbb{R}$, with $m \neq 0$. We will be using the relation between the modulo and the ceiling operators [29], i.e.,

$$\lfloor -x \rceil_m = \left\lceil \frac{x}{m} \right\rceil m - x. \tag{2}$$

A. Events, readings, and writings

Following established examples [5], [6], [20], we decouple the execution of tasks from the instants when they read and write some data. Hence, we use the following terms

- a *read event* $r \in \mathcal{R}$, associated to the recurrent instants when a task reads data written by other tasks. $\mathcal{R}$ denotes the set of all read events,
- a *write event* $w \in \mathcal{W}$, associated to the recurrent instants when a task writes data for other tasks, with $\mathcal{W}$ denoting the set of all write events.
- Also, we use the generic term *event* $e \in \mathcal{E}$, with $\mathcal{E} = \mathcal{R} \cup \mathcal{W}$, to denote both readings and writings.

Also, this model seamlessly allows communication to happen in the middle of job execution, as industry demands [33].

The instants associated to an event are generically defined below as an ordered set of infinite instants.

Definition 1 (Event sequence t_e)**.** *For any event $e \in \mathcal{E}$, we call* event sequence *any set*

$$t_e = \{t_e(j) \in \mathbb{R} : \forall j \in \mathbb{N},\ t_e(j) < t_e(j+1)\}. \tag{3}$$

$t_e(j)$ is the absolute time of the j-th instant of the sequence t_e associated to the event e, and we may call any event sequence t_e, more simply a sequence.

The presence of jitter in events introduces non-determinism: many event sequences are possible for the read and write events, as shown in the example of Fig. 1. Hence, we introduce below the *event series* to parametrize the allowed uncertainties, and the set of *feasible sequences* which contains all sequences t_e compatible with a given specification.

Definition 2 (Event series e)**.** *An event series $e \in \mathcal{E}$ is the triple (T_e, Φ_e, J_e), where*

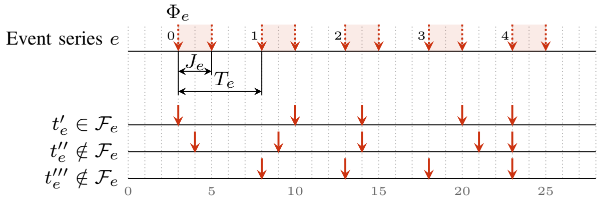


Fig. 2: Illustrating the definitions of event series and feasible sequences. On top, a shaded area bounded by dotted arrows represents the time intervals in which events may occur. The first interval starts at Φ_e, with $\Phi_e = 3$ in the example. The width of each interval corresponds to the jitter J_e ($= 2$ in the example). The intervals repeat every $T_e = 5$. Three sequences t'_e, t''_e, and t'''_e are represented. The sequence t'_e is feasible. The sequence t''_e is not feasible because $t''_e(3) = 21 \notin [18, 20]$. Finally, $t'''_e \notin \mathcal{F}_e$ as well. In fact, despite each $t'''_e(j)$ belongs to some interval, they belong to a "wrong" interval. In fact, $t'''_e(0) = 8 \notin [3, 5]$, as required by Eq. (5).

- *$T_e \in \mathbb{N}$ is the* period *of the event series;*
- *$\Phi_e \in \mathbb{R}$ is the* offset *of the event series; and*
- *$J_e \in \mathbb{R}$ is the* jitter *of the event series.*

We may call an event series e more simply series. Any event series $e \in \mathcal{E}$ may generate many different sequences of instants, compatibly with the specification of period T_e, offset Φ_e and jitter J_e. This is encoded by the next definition.

Definition 3 (Feasible sequences $\mathcal{F}_e$)**.** *Given an event series $e \in \mathcal{E}$, we define the* set of feasible sequences *$\mathcal{F}_e$ as follows. An event sequence t_e belongs to $\mathcal{F}_e$ if and only if*

$$t_e(j) = j\,T_e + \Phi_e + \delta_e(j), \tag{4}$$

with $\delta_e(j)$ representing the variability *of the sequence t_e and constrained by the jitter with*

$$\forall j \in \mathbb{N}, \qquad 0 \le \delta_e(j) \le J_e. \tag{5}$$

The jitter J_e limits the amount of possible variability in the occurrence of the events through Eq. (5). For example, the event series r_2 associated with the read event of task τ_2 in Fig. 1c has period $T_{r_2} = 6$, offset $\Phi_{r_2} = 4$, and jitter $J_{r_2} = 3$. The actual event sequence is $t_{r_2} = \{7, 10, 19, 22, 31, \ldots\}$ with $\delta_{r_2} = \{3, 0, 3, 0, 3, \ldots\}$. In the special case with no jitter ($J_e = 0$), Eq. (5) gives that the only feasible sequence in $\mathcal{F}_e$ is $t_e = \{\Phi_e, T_e + \Phi_e, 2T_e + \Phi_e, \ldots\}$. As the jitter grows, a larger variability is allowed. Fig. 2 illustrates how the parameters of an event series concur to define the feasible sequences.

B. Tasks and chains

We analyze a chain of n periodic tasks $\tau_1 \to \cdots \to \tau_n$:

- task τ_1 reads from the environment (i.e. a sensor task),
- $\forall i = 2, \ldots, n$, task τ_i reads the data written by τ_{i-1}, and
- task τ_n writes to the environment (i.e. an actuator task).

For each task τ_i, we only need to model the read and write instants which participate to the chain under analysis. This yields the following definition.

Definition 4. *A* task *τ_i with period P_i is characterized by a pair (r_i, w_i), with*

- *the read series $r_i \in \mathcal{R}$ of the task*
- *the write series $w_i \in \mathcal{W}$ of the task,*

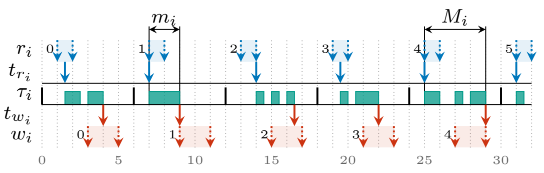


Fig. 3: Relation between the task schedule and the read/write event series, in case of implicit communication. Solid arrows in blue denote the read sequence, arrows in red the write sequence. The read/write events r_i/w_i are also drawn. Also, the minimum and maximum task latencies, m_i and M_i, are represented.

and with $T_{r_i} = T_{w_i} = P_i$*,* $\Phi_{r_i} \le \Phi_{w_i}$*.*

The last conditions on the read/write series of τ_i enforce that read and write series of a task have the same periodicity as the task itself, and that writes follow reads. For all $j \in \mathbb{N}$, the data fetched at time $t_{r_i}(j)$ is processed by the j-th job of task τ_i, and the processed data is then written at time $t_{w_i}(j)$.

In the literature, there are many ways to define the characteristics of $r_i \in \mathcal{R}$ and $w_i \in \mathcal{W}$ from task execution parameters, scheduling policy, and available processing/communication resources. For example, if τ_i is a LET task with period P_i, offset O_i, and relative deadline D_i, then its read/write events are characterized by:

- read series: $T_{r_i} = P_i$, $\Phi_{r_i} = O_i$, $J_{r_i} = 0$, and
- write series: $T_{w_i} = P_i$, $\Phi_{w_i} = O_i + D_i$, $J_{w_i} = 0$.

If instead a task is employing implicit communication, by reading and writing at the beginning and the end of jobs respectively, then the events would be characterized by

- read series: $T_{r_i} = P_i$, $\Phi_{r_i} = S_i^-$, $J_{r_i} = S_i^+ - S_i^-$, and
- write series: $T_{w_i} = P_i$, $\Phi_{w_i} = R_i^-$, $J_{w_i} = R_i^+ - R_i^-$,

with S_i^-, S_i^+ being the earliest/latest start time among all τ_i jobs, and R_i^-, R_i^+ being the earliest/latest response time among all jobs, all measured from job releases. Fig. 3 illustrates such a definition. Also, some measurement-based approach may be used to infer the event series characteristics, by examining execution traces under different scenarios [34].

Also, we need to define the minimum/maximum *task latency* introduced by τ_i, which we define by

$$m_i = \inf_j \{t_{w_i}(j) - t_{r_i}(j)\}, M_i = \sup_j \{t_{w_i}(j) - t_{r_i}(j)\}. \quad (6)$$

If τ_i is LET or τ_i executes non-preemptively with a constant execution time, then $m_i = M_i$. If τ_i uses implicit communication, then m_i and M_i are bounded by $\max\{0, R_i^- - S_i^+\}$ and $R_i^+ - S_i^-$, respectively. If instead, we have no additional information on the task execution, the next corollary exploits the parameters of series r_i and w_i to bound m_i and M_i.

Corollary 1. *Given a task* τ_i *defined as in Def. 4, its latency is bounded by*

$$m_i \ge \max\{0, \Phi_{w_i} - \Phi_{r_i} - J_{r_i}\}, \quad M_i \le \Phi_{w_i} - \Phi_{r_i} + J_{w_i}. \quad (7)$$

Proof. Eq. (7) follows by replacing the definition of sequences of (4) in (6), and exploiting the bound of (5) and that $\forall i, j,\ t_{r_i}(j) \le t_{w_i}(j)$. □

The maximum reaction time, or First-to-First latency to use Feiertag's terminology [35], of τ_i is then equal to

$$D_i^{\mathsf{FF}} = \sup_j \{t_{w_i}(j) - t_{r_i}(j-1)\} \le P_i + M_i \le P_i + \Phi_{w_i} - \Phi_{r_i} + J_{w_i}. \quad (8)$$

Other types of latency can also be found using a similar logic [35], [36].

Given this model, the data propagation along the chain is

1) the concatenation of some event $r_i \in \mathcal{R}$ and $w_i \in \mathcal{W}$, occurring within task τ_i, and
2) the concatenation of some $w_i \in \mathcal{W}$ with $r_{i+1} \in \mathcal{R}$, which implements the communication from τ_i to τ_{i+1}.

In Section III, we start by analyzing the interaction between deterministic write and read sequences (Def. 1). Then, Section IV extends the analysis to the (non-deterministic) write-to-read series (Def. 2). Finally, in Section V we resume the task model and we illustrate how to compose tasks jitter into a chain-level jitter.

III. Pair of write-to-read event sequences

We investigate here the interaction between the write series $w_i \in \mathcal{W}$ of some task τ_i with the read series $r_{i+1} \in \mathcal{R}$ of the next task τ_{i+1} in the chain. To lighten the notation, both in Sections III and IV, we drop the task index because the communication always happens between a write series and the read series of the next task on the chain.

Let t_w and t_r be the sequences of write and read instants, respectively. As observed in the literature [29], [37], [38], some instants in t_w and t_r are redundant as they do not contribute to the data communication. In fact:

- If $[t_w(j), t_w(j+1)) \cap t_r$ is empty for some $j \in \mathbb{N}$, then the data written at time $t_w(j)$ is overwritten at time $t_w(j+1)$, without having been read. Hence, $t_w(j)$ can be removed from t_w without affecting the communication.
- Similarly, if $t_w \cap [t_r(j), t_r(j+1))$ is empty for some $j \in \mathbb{N}$, at instants $t_r(j)$ and $t_r(j+1)$ the same data is read. Hence the second read instant $t_r(j+1)$ can be removed without altering the communication pattern.

The elimination of the redundant instants from both t_w and t_r enables capturing the core of the timing of communication. For this purpose, we formulate the following definition.

Definition 5 (Effective write/read sequence)**.** *Given any write/read sequence* (t_w, t_r)*, the* effective write/read sequence (t_w^*, t_r^*) *obtained from* (t_w, t_r) *is a pair of event sequences that satisfy the following conditions:*

$$\forall j \in \mathbb{N}, \quad t_w^*(j) = \max\{x \in t_w : x \le t_r^*(j)\}, \quad (9)$$

$$\forall j \in \mathbb{N}, \quad t_r^*(j) = \min\{y \in t_r : t_w^*(j) \le y\}, \quad (10)$$

$$\nexists x \in t_w, \nexists y \in t_r : \quad x \le y < t_w^*(0), \quad (11)$$

$$\nexists j \in \mathbb{N}, \nexists x \in t_w, \nexists y \in t_r : \ t_r^*(j) < x \le y < t_w^*(j+1). \quad (12)$$

Rephrasing the constraints above in words. The condition of Eq. (9) ensures that data written at any $t_w^*(j)$ is never overwritten before being read. Eq. (10) ensures that at $t_r^*(j)$ we are reading fresh data, not previously read. Eq. (11) says that no previous communication occurs before $t_w^*(0)$. Finally, Eq. (12) ensures that no communication of fresh data happens between consecutive pairs $(t_w^*(j), t_r^*(j))$. Such a notion is closely related to the one of publishing/reading points [39].

The determination of the effective sequence is also illustrated constructively in Algorithm 1. Such an algorithm only serves the purpose of offering a procedural view for Def. 5. Hence, it is not problematic if, for example, its "stopping condition" (Line 11) is left unspecified.

Algorithm 1 Construction of an effective write/read sequence.

```
1:  function EffectiveSequence($t_w, t_r$)
2:      wrCurSet ← $t_w$, rdCurSet ← $t_r$                    ▷ Initialization
3:      $j \leftarrow 0$
4:      repeat
5:          wrNext ← min wrCurSet
6:          $t_r^*(j) \leftarrow \min\{y \in t_r : \text{wrNext} \leq y\}$
7:          $t_w^*(j) \leftarrow \max\{x \in t_w : x \leq t_r^*(j)\}$   ▷ from Eq. (9)
8:          wrCurSet ← wrCurSet $\cap\, (t_w^*(j), +\infty)$
9:          rdCurSet ← rdCurSet $\cap\, (t_r^*(j), +\infty)$
10:         $j \leftarrow j + 1$
11:     until (stopping condition)
12:     return $(t_w^*, t_r^*)$
```

We now proceed with the analysis of effective sequences. The following lemma proves that the effective sequence (t_w^*, t_r^*) produced from any generic pair (t_w, t_r), is always interleaving write and read instants. Thus, it avoids data overwrite and guarantees no double read of the same value.

Lemma 1. *The effective write/read sequence (t_w^*, t_r^*) for any (t_w, t_r) guarantees that*

$$\forall j \in \mathbb{N}, \quad t_w^*(j) \leq t_r^*(j) < t_w^*(j+1) \tag{13}$$

Proof. From (9) the first inequality of Eq. (13) follows. We prove $t_r^*(j) < t_w^*(j+1)$ by contradiction. We assume $\exists \hat{\jmath} \in \mathbb{N}$ with $t_w^*(\hat{\jmath}+1) \leq t_r^*(\hat{\jmath})$, which implies $t_r^*(\hat{\jmath}) \in \{y \in t_r : t_w^*(\hat{\jmath}+1) \leq y\}$. From Eq. (10), it means that

$$t_r^*(\hat{\jmath}+1) = \min\{y \in t_r : t_w^*(\hat{\jmath}+1) \leq y\} \leq t_r^*(\hat{\jmath}),$$

which contradicts the strictly increasing ordering in t_r^* that follows from (3) and the fact that $t_r^* \subseteq t_r$. Hence, Eq. (13) and the lemma are proved. □

In words, Lemma 1 states that between every pair of consecutive write instants of t_w^* (resp. read instants of t_r^*) there is *one and only one* read instant of t_r^* (resp. write instant of t_w^*). Hence, the removal of any instant from them would prevent the communication of fresh data.

An example is shown in Fig. 4.

IV. Pair of write-to-read event series

In Section III, we determine the effective write/read sequence from any pair (t_w, t_r). Lifting this definition from (deterministic) event sequences to (non-deterministic) event series poses some challenges. The main issue lies in the

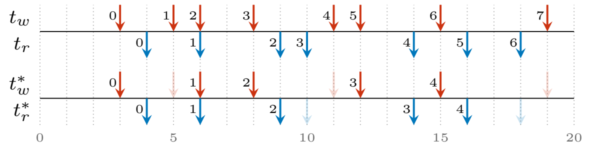


Fig. 4: From the write/read sequence (t_w, t_r), its effective write/read sequence (t_w^*, t_r^*) is shown. It can be observed that the following instants (in semi-transparent color) $t_w(1) = 5$, $t_w(4) = 11$, and $t_w(7) = 19$ do not belong to t_w^* because overwritten by some later write instant in t_w. Similarly, instants $t_r(3) = 10$ and $t_r(6) = 18$ do not belong to t_r^* because they read the same value read earlier at previous instants $t_r(2) = t_r^*(2) = 9$ and $t_r(5) = t_r^*(4) = 16$, respectively.

impossibility to propose a constructive definition of effective series. In fact, as series w and r may generate any sequence $t_w \in \mathcal{F}_w$, $t_r \in \mathcal{F}_r$, it is not clear which sequence one should use, nor how to combine them.

To deal with this issue, we build on Def. 5 and establish the *effective* property for a pair of event series (w, r), defining the effective write/read series (w^*, r^*).

Definition 6 (Effective write/read series)**.** *Given a pair of write/read event series $(w, r) \in \mathcal{W} \times \mathcal{R}$, we say that $(w^*, r^*) \in \mathcal{W} \times \mathcal{R}$ is an* effective write/read event series *when:*

$$\forall t_w \in \mathcal{F}_w,\ \forall t_r \in \mathcal{F}_r, \quad t_w^* \in \mathcal{F}_{w^*},\ t_r^* \in \mathcal{F}_{r^*}. \tag{14}$$

with (t_w^*, t_r^*) being the effective sequence of (t_w, t_r).

In words, we require that the effective write/read series (w^*, r^*) can represent all behaviors of the original (w, r). The definition does so by:

- allowing the choice of **any** feasible sequence t_w and t_r,
- requiring that its effective sequence (t_w^*, t_r^*), which are the subsequences of (t_w, t_r) actually realizing the communication, belongs to the feasible sequences of w^* and r^*, respectively.

Def. 6 is not constructive: it is not instructing one on how to determine the pair of events (w^*, r^*). Rather, it is a "wishful statement" of the expected property (w^*, r^*). Also, nothing can be asserted about existence and uniqueness: an effective write/read event series may not exist (see the example of Fig. 5 described in next paragraphs). If exists, it is not unique.

We start with a preliminary result that sets a property of any (w^*, r^*).

Theorem 1. *Let (w^*, r^*) be an effective write/read series of any $(w, r) \in \mathcal{W} \times \mathcal{R}$. Then*

$$T_{w^*} = T_{r^*}. \tag{15}$$

Proof. We apply Def. 6: for any choice of $t_w \in \mathcal{F}_w$ and $t_r \in \mathcal{F}_r$, let (t_w^*, t_r^*) be its effective sequence. From (14), we have that $t_w^* \in \mathcal{F}_{w^*}$ and $t_r^* \in \mathcal{F}_{r^*}$. This means that we can write the two sequences t_w^* and t_r^* explicitly as

$$t_w^*(j) = j\,T_{w^*} + \Phi_{w^*} + \delta_{w^*}(j),$$
$$t_r^*(j) = j\,T_{r^*} + \Phi_{r^*} + \delta_{r^*}(j)$$

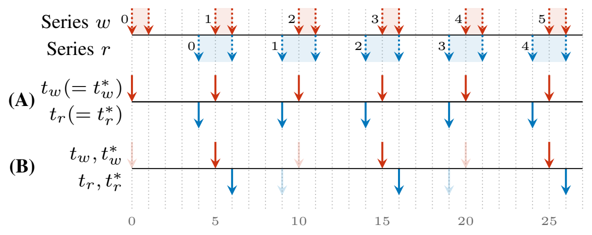


Fig. 5: Example of a pair (w, r) of series. We show two cases of $t_w \in \mathcal{F}_w$ and $t_r \in \mathcal{F}_r$ and their corresponding effective write/read sequence (t_w^*, t_r^*). In case **(A)**, the effective sequence (t_w^*, t_r^*) is the same as the original pair (t_w, t_r). In case **(B)**, the sequence t_r has some variability. Consequently, in the effective write/read sequence (t_w^*, t_r^*), half of the instants are discarded (represented in semi-transparent color).

with $0 \le \delta_{w^*}(j) \le J_{w^*}$ and $0 \le \delta_{r^*}(j) \le J_{r^*}$, for some J_{w^*}, J_{r^*}. From Lemma 1, it follows that

$$\begin{aligned} t_w^*(j) \le t_r^*(j) &< t_w^*(j+1), \\ j\,T_{w^*} + \Phi_{w^*} + \delta_{w^*}(j) &\le j\,T_{r^*} + \Phi_{r^*} + \delta_{r^*}(j) \\ &< (j+1)\,T_{w^*} + \Phi_{w^*} + \delta_{w^*}(j+1), \\ j \ge 1, \quad T_{w^*} + \tfrac{\Phi_{w^*}+\delta_{w^*}(j)}{j} &\le T_{r^*} + \tfrac{\Phi_{r^*}+\delta_{r^*}(j)}{j} \\ &< T_{w^*} + \tfrac{T_{w^*}+\Phi_{w^*}+\delta_{w^*}(j+1)}{j}. \end{aligned}$$

Making now the limit for j to infinity, the fractions of the above inequalities tend to zero as the numerators are bounded. From the squeeze theorem, we find $T_{r^*} = T_{w^*}$, as needed. ☐

The proved property that $T_{w^*} = T_{r^*}$ means that if effective the write/read series (w^*, r^*) exists, then the events w^* and r^* must advance with the same pace. The common period for the pair (w^*, r^*) supports the following definition and a more compact notation.

Definition 7 (Period of the effective write/read series)**.** *Let (w^*, r^*) be an effective write/read series of any $(w, r) \in \mathcal{W} \times \mathcal{R}$. We define* period of the series *T_* as the value T_{w^*} (also equal to T_{r^*} from Theorem 1).*

To offer more insights on effective write/read series (w^*, r^*), we propose an example of (w, r) with no effective series. Consider the pair (w, r) with periods $T_w = T_r = 5$, offsets $\Phi_w = 0$, $\Phi_r = 4$, and jitters $J_w = 1$, $J_r = 2$, illustrated in Fig. 5. Since Eq. (14) has the $\forall$ operator, we select two cases for (t_w, t_r) (labeled "**(A)**" and "**(B)**" in the figure) and show that they would require two incompatible effective event series. For case **(A)**, we pick the pair of feasible sequences $(t_w, t_r) = (\{0, 5, 10, 15, 20, 25, \ldots\}, \{4, 9, 14, 19, 24, \ldots\})$. The effective write/read sequence of this one is itself $(t_w^*, t_r^*) = (t_w, t_r)$ because every read in t_r follows a distinct write, and every write is followed by a distinct read (see Def. 5 for the formal properties). From Def. 2, any effective series (w^*, r^*) with $t_w^* \in \mathcal{F}_{w^*}$, $t_r^* \in \mathcal{F}_{r^*}$ must then necessarily have period $T_* = 5$. Case **(B)** is built by delaying the $t_r(j)$ by 2 for any even j, that is we choose $t_r = \{6, 9, 16, 19, 26, \ldots\}$. Such a modification is indeed compatible with $J_r = 2$. The effective write/read sequence (t_w^*, t_r^*) of this case **(B)**, however, does change drastically as we get $(t_w^*, t_r^*) = (\{5, 15, 25, \ldots\}, \{6, 16, 26, \ldots\}) = (\{10j + 5\}, \{10j + 6\})$ (refer to Fig. 5 for an illustration). In fact, every other write and read instants are dropped because they are overwritten or they read the same value read previously. Hence, the sequence (t_w^*, t_r^*) can only belong to some effective series with period $T_* = 10$, which is incompatible with $T_* = 5$ of case **(A)**. We conclude that an effective series (w^*, r^*), as defined in Def. 6, cannot exists for (w, r). This example demonstrates that the determination of effective series (w^*, r^*) is only possible under some given hypothesis.

We now investigate how to determine an effective write/read series (w^*, r^*) from any pair $(w, r) \in \mathcal{W} \times \mathcal{R}$, and the hypotheses that make it possible. Ideally, we aim for:

- the smallest possible period T_*, to have the events in the series w^* and r^* as frequent as possible,
- the earliest possible offsets Φ_{w^*} and Φ_{r^*}, to start the events as soon as possible,
- the shortest possible jitters J_{w^*} and J_{r^*}, to minimize the event uncertainty, and
- the most permissive possible existence hypotheses.

Next lemma establishes a first boundary on the possible period T^* of the effective write/read series.

Lemma 2. *Let (w^*, r^*) be the effective write/read series of any (w, r), and let T_* be the period of w^* and r^*. Then it is*

$$T_* \ge \max\{T_w, T_r\}. \tag{16}$$

Proof. First we show that $T_w \le T_*$. For all $j \in \mathbb{N}$, we have $t_w(j) \le t_w^*(j)$ because $t_w \subseteq t_w^*$. Also, $t_w^* \in \mathcal{F}_{w^*}$ from (14) of Def. 6. Then we can write

$$j\,T_w + \Phi_w + \delta_w(j) = t_w(j) \le t_w^*(j) \le j\,T_* + \Phi_{w^*} + J_{w^*}.$$

Dividing both sides by j and making the limit for $j \to \infty$, we find $T_w \le T_*$. By replacing "w" with "r" above, we also find $T_r \le T_*$. Hence, Eq. (16) follows. ☐

Lemma 2 is setting a hard lower bound to how small we can hope to find the common period T_* of the effective series (w^*, r^*). For this reason, in the remainder of this paper, we will seek effective write/read series with the minimal period $T_* = \max\{T_w, T_r\}$. The offset Φ_{w^*} and Φ_{r^*} and the jitter J_{w^*} and J_{r^*} for the series w^* and r^* are still to be found and are the core contributions of the remaining part of this section.

To encode a possible misalignment between the start of the two sequences t_w and t_r, we now introduce

$$\Delta = \Phi_r - \Phi_w, \tag{17}$$

If the write sequence t_w starts after $t_r(0)$, we have $\Delta < 0$.

We divide the analysis in three cases, depending on the periods T_w and T_r. Theorem 2 gives the result for $T_w = T_r$, Theorem 3 in the case of $T_w > T_r$, and finally Theorem 4 explores the case $T_w < T_r$. The general structure of the following results (Theorems 2, 3, and 4) is as follows: (i) the theorem states some possible parameters for the events, and (ii) the proof verifies that these parameters satisfy Def. 6.

Theorem 2. *Let (w, r) be two event series with $T_w = T_r$. Then if*

$$J_w \le \lfloor \Delta \rfloor_{T_*} < T_* - J_r, \tag{18}$$

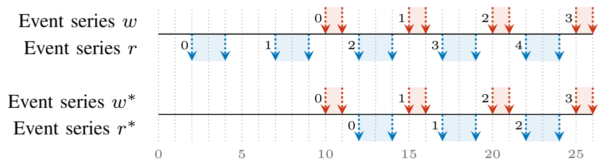


Fig. 6: Example of determination of (w^*, r^*) from a given (w, r) through Theorem 2. In the example, $\Phi_w = 10$ and $\Phi_r = 2$, then $\Delta = -8$. The hypothesis of (18) holds because $\lfloor\Delta\rfloor_{T_*} = \lfloor -8\rfloor_5 = 2$, which belongs to $[J_w, T_* - J_r)$ equal to $[1, 3)$ for this example. The event w^* is the same as w. The event r^* is instead postponed, with an offset $\Phi_{r^*} = \Phi_w + \lfloor\Delta\rfloor_{T_*} = 12$.

an effective write/read series (w^*, r^*) exists, with period $T_* = T_w = T_r$ and parameters

$$\begin{cases} \Phi_{w^*} = \Phi_w, \quad \Phi_{r^*} = \Phi_w + \lfloor\Delta\rfloor_{T_*}, & \text{if } \Delta < 0 \\ \Phi_{w^*} = \Phi_r - \lfloor\Delta\rfloor_{T_*}, \quad \Phi_{r^*} = \Phi_r, & \text{if } \Delta \geq 0 \end{cases}$$
$$J_{w^*} = J_w, \quad J_{r^*} = J_r. \tag{19}$$

Proof. First, we remind that Δ, defined in Eq. (17) is the difference $\Phi_r - \Phi_w$ between the start of the two sequences t_r and t_w. We split the proof in two cases, depending on Δ. If $\Delta < 0$ and Eq. (18) holds, data written at $t_w(0)$ will certainly not be overwritten, as shown in Fig. 6. The initial part of t_r, instead, may read no data, hence t_r^* must start later than t_r. We aim at proving that the pair t_w^* and t_r^* defined by

$$\forall j \in \mathbb{N}, \quad t_w^*(j) = t_w(j), \quad t_r^*(j) = t_r\left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) \tag{20}$$

is effective write/read sequence for t_w and t_r and feasible for the events w^* and r^* specified by the parameters of (19).

To show that the properties of Def. 5 hold for t_w^* and t_r^* of Eq. (20), we need to rework $t_r^*(j)$ as follows

$$\begin{aligned} t_r^*(j) &= t_r\left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) \\ &= \left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) T_* + \Phi_r + \delta_r\left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) \\ &= jT_* - \left\lfloor\frac{\Delta}{T_*}\right\rfloor T_* + \Phi_w + \Delta + \delta_r\left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) \\ &= jT_* + \Phi_w + \lfloor\Delta\rfloor_{T_*} + \delta_r\left(j - \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right). \end{aligned} \tag{21}$$

Then we have that $\forall j \in \mathbb{N}$,

$$t_w^*(j) = t_w(j) \leq jT_* + \Phi_w + J_w \leq jT_* + \Phi_w + \lfloor\Delta\rfloor_{T_*} \leq t_r^*(j)$$

with the last two inequalities holding respectively for hypothesis of (18) and the above expression of $t_r^*(j)$. Also, from the hypothesis $\delta_r(j) \leq J_r$ of (18), and from $\delta_w(j) \geq 0$, we have

$$t_r^*(j) \leq jT_* + \Phi_w + \lfloor\Delta\rfloor_{T_*} + J_r < jT_* + \Phi_w + T_* \leq t_w^*(j+1).$$

From the found $t_w^*(j) \leq t_r^*(j) < t_w^*(j+1)$ Eq. (9) and (10) follow because no element of t_w and t_r exists in the interval $\big(t_w^*(j), t_r^*(j)\big)$. Eq. (11) follows because from (20) no other point in t_w is earlier than $t_w^*(0)$. Eq. (12) holds because no element of t_w belongs to the open interval $\big(t_r^*(j), t_w^*(j+1)\big)$.

To prove $t_w^* \in \mathcal{F}_{w^*}$, as required by Def. 6, we simply observe that $t_w^* = t_w$ from Eq. (20), and $\mathcal{F}_w = \mathcal{F}_{w^*}$ from the parameters of (19). From the expression of t_r^* of Eq. (21) and the parameters of r^* of Eq. (19), also $t_r^* \in \mathcal{F}_{r^*}$ follows. This concludes the case $\Delta < 0$.

The case with $\Delta \geq 0$ follows the same logic. In this case

$$\forall j \in \mathbb{N}, \quad t_w^*(j) = t_w\left(j + \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right), \quad t_r^*(j) = t_r(j)$$

and rewriting $t_w^*(j)$ gives

$$\begin{aligned} t_w^*(j) &= \left(j + \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) T_* + \Phi_w + \delta_w\left(j + \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right) \\ &= jT_* + \Phi_r - \lfloor\Delta\rfloor_{T_*} + \delta_w\left(j + \left\lfloor\frac{\Delta}{T_*}\right\rfloor\right). \end{aligned}$$

Following similar steps as in the case $\Delta < 0$, we find

$$t_w^*(j) \leq jT_* + \Phi_r - \lfloor\Delta\rfloor_{T_*} + J_w \leq jT_* + \Phi_r \leq t_r^*(j)$$
$$t_r^*(j) \leq jT_* + \Phi_r + J_r < jT_* + \Phi_r + T_* - \lfloor\Delta\rfloor_{T_*} \leq t_w^*(j+1),$$

which implies again that $\forall j \in \mathbb{N}$, $t_w^*(j) \leq t_r^*(j) < t_w^*(j+1)$. With the same arguments as the case $\Delta < 0$, from the inequalities above Eq. (9), (10), (11), and (12) follow. From the expression of the parameters of (w^*, r^*) of (19), it holds that $t_w^* \in \mathcal{F}_{w^*}$ and $t_r^* \in \mathcal{F}_{r^*}$. This concludes the proof of the case $\Delta \geq 0$ and then the whole proof. □

We underline that Theorem 2 is precisely stating the hypothesis that avoids the non-existence of an effective series of the example of Fig. 5. When condition (18) is false, no effective write/read series (w^*, r^*) can exist. In fact, depending on the variability $\delta_w(j)$ and $\delta_r(j)$, the effective sequences can yield different periods T_* (in Fig. 5, $T_* = 5$ in case **(A)** and $T_* = 10$ in case **(B)**), which is incompatible with any (w^*, r^*).

If the offset of events w or r may be assigned, then from Theorem 2 a simple test enables to check the existence of an offset assignment fulfilling the condition of (18).

Corollary 2. *If*

$$J_w + J_r < T_* \tag{22}$$

then an assignment for Φ_w or Φ_r that fulfills the condition of Eq. (18) of Theorem 2 exists.

Proof. If (22) then the interval $[J_w, T_* - J_r)$ is not empty. Also, from $J_w \geq 0$ and $T_* - J_r \leq T_*$ it follows that

$$[J_w, T_* - J_r) \subseteq [0, T_*).$$

We can then choose Φ_w or Φ_w such that

$$\lfloor\Phi_r - \Phi_w\rfloor_{T_*} = \lfloor\Delta\rfloor_{T_*} \in [J_w, T_* - J_r),$$

which makes (18) true. □

The result above will be exploited in the last part of the evaluation of Section VI.

Let us now address the case with $T_w > T_r$, seeking for an effective write/read series with the minimal period $T_* = \max\{T_w, T_r\}$.

Theorem 3. *Let (w, r) be two event series with $T_w > T_r$. Then if*

$$T_r + J_r \leq T_w - J_w, \tag{23}$$

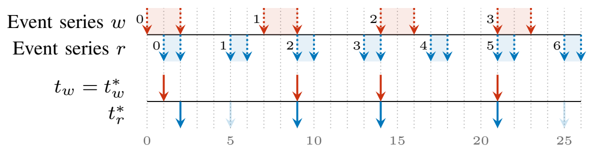


Fig. 7: The limit case of condition (23) that is $T_r + J_r = T_w - J_w$. In the sequences $t_w \in \mathcal{F}_w$ and $t_r \in \mathcal{F}_r$, a small increase in T_r, J_r or J_w, or a small decrease in T_w, may lead $t_w(2) = 14$ to overwrite $t_w(1) = 9$.

an effective write/read series (w^*, r^*) *exists with period* $T_* = T_w$ *and parameters*

$$\Phi_{w^*} = \Phi_{r^*} = \Phi_w + \max\left\{0, \left\lfloor \frac{\Delta + J_r - T_r}{T_w} \right\rfloor + 1\right\} T_w,$$
$$J_{w^*} = J_w, \quad J_{r^*} = T_r + J_w. \tag{24}$$

Proof. From (4) and (5), we find

$$\max_j\{t_r(j+1) - t_r(j)\} \le T_r + J_r$$
$$T_w - J_w \le \min_j\{t_w(j+1) - t_w(j)\},$$

and then the hypothesis of Eq. (23) implies

$$\max_j\{t_r(j+1) - t_r(j)\} \le \min_j\{t_w(j+1) - t_w(j)\}. \tag{25}$$

Eq. (25) guarantees that there is always at least a read instant $y \in t_r$ in $\big[t_w(j), t_w(j+1)\big)$ for any j such that $t_w(j) \ge t_r(0)$. Fig. 7 shows the limit case of Eq. (23), in which Eq. (23) holds with the equal sign and all written data is read at some $y \in t_r$.

With Eq. (23) holding, we know that an effective write/read series exists with period $T_* = T_w$. We are now seeking the other parameters of w^* and r^*. When Eq. (23) holds, write instants in t_w cannot be recurrently overwritten. Then, the write sequence t_w^* of the effective write/read sequence has the same period T_w of t_w and is

$$t_w^*(j) = t_w(j + k_w) = j\,T_w + \Phi_w + k_w T_w + \delta_w(j), \tag{26}$$

for some $k_w \ge 0$, with $k_w T_w$ representing some additional offset of t_w^* w.r.t. t_w. From Eq. (10), $t_r^* \subseteq t_r$. Then we write $t_r^*(j) = t_r(j_r)$ for some $j_r \in \mathbb{N}$ to be found. Eq. (13) enforces $t_r^*(j) = t_r(j_r) = j_r T_r + \Phi_r + \delta_r(j_r) \ge t_w^*(j) = j\,T_w + \Phi_w + k_w T_w + \delta_w(j)$. Eq. (10) ensures that j_r must be the smallest integer satisfying the inequality, hence

$$j_r = \left\lceil \frac{j\,T_w + k_w T_w - \Delta + \delta_w(j) - \delta_r(j_r)}{T_r} \right\rceil. \tag{27}$$

Such index j_r must necessarily be non-negative. It is necessary to enforce it because a very large Δ (recalling that $\Delta = \Phi_r - \Phi_w$) may give a negative j_r. The additional offset k_w of t_w^* w.r.t. t_w can enforce $j_r \ge 0$, that is

$$\left\lceil \frac{k_w T_w - \Delta + \delta_w(j) - \delta_r(j_r)}{T_r} \right\rceil \ge 0 \quad \Leftrightarrow$$
$$\frac{k_w T_w - \Delta + \delta_w(j) - \delta_r(j_r)}{T_r} > -1$$

since $\delta_w(j) \ge 0$ and $\delta_r(j) \le J_r$, the inequality above is implied by

$$\frac{k_w T_w - \Delta - J_r}{T_r} > -1 \quad \Leftrightarrow \quad k_w T_w > \Delta + J_r - T_r$$
$$\Leftrightarrow \quad k_w \ge \left\lfloor \frac{\Delta + J_r - T_r}{T_w} \right\rfloor + 1.$$

This enables the determination of the parameters of the event w^*, because t_w^* in Eq. (26) belongs to $\mathcal{F}_{w^*}$ when

$$T_* = T_w, \quad \Phi_{w^*} = \Phi_w + \overbrace{\max\left\{0, \left\lfloor \frac{\Delta + J_r - T_r}{T_w} \right\rfloor + 1\right\}}^{k_w} T_w,$$
$$J_{w^*} = J_w.$$

We are now going to introduce in $t_r(j_r)$ the value of j_r obtained from Eq. (27). Before doing so, we simplify the term $k_w T_w - \Delta$ according to

$$k_w T_w - \Delta = \Phi_{w^*} - \Phi_w - \Delta = \Phi_{w^*} - \Phi_r.$$

Introducing the expression of j_r from Eq. (27) in $t_r(j_r)$ gives

$$\begin{aligned}
t_r^*(j) &= \left\lceil \frac{j\,T_w + \Phi_{w^*} - \Phi_r + \delta_w(j) - \delta_r(j_r)}{T_r} \right\rceil T_r \\
&\quad + \Phi_r + \delta_r(j_r) \\
&= \left\lceil \frac{j\,T_w + \Phi_{w^*} - \Phi_r + \delta_w(j) - \delta_r(j_r)}{T_r} \right\rceil T_r \\
&\quad - (j\,T_w + \Phi_{w^*} - \Phi_r + \delta_w(j) - \delta_r(j_r)) \\
&\quad + \Phi_{w^*} + \delta_w(j) + j\,T_w \\
&= j\,T_w + \Phi_{w^*} \\
&\quad + \underbrace{\lfloor \Phi_r - \Phi_{w^*} - j\,T_w - \delta_w(j) + \delta_r(j_r) \rfloor_{T_r} + \delta_w(j)}_{\delta_{r^*}(j)}.
\end{aligned} \tag{28}$$

The last step exploits the property specified in Eq. (2).

We conclude by observing that $\delta_{r^*}(j)$ in (28) can vary between 0 and $T_r + J_w$, which yields the final parameters of the event r^* as follows

$$\Phi_{r^*} = \Phi_{w^*}, \qquad J_{r^*} = T_r + J_w.$$

as in Eq. (24), concluding the proof. □

Theorem 3 states a powerful result as it finds an effective write/read series (w^*, r^*) with the smallest possible period. Such a result, however, requires to satisfy the hypothesis of Eq. (23). If this hypothesis does not hold, the existence of any period $T_* \ge \max\{T_w, T_r\}$ for the effective write/read series is an open problem.

We now complete the analysis with the case with $T_w < T_r$.

Theorem 4. *Let* (w, r) *be two event series with* $T_w < T_r$. *Then if*

$$T_w + J_w \le T_r - J_r, \tag{29}$$

an effective write/read series (w^*, r^*) exists with period $T_* = T_r$ and parameters

$$\Phi_{r^*} = \Phi_r + \max\left\{0, \left\lceil \frac{J_w - \Delta}{T_r} \right\rceil\right\} T_r, \qquad J_{r^*} = J_r,$$
$$\Phi_{w^*} = \Phi_{r^*} - T_w, \quad J_{w^*} = T_w + J_r. \tag{30}$$

As this proof mimics the one of Theorem 3, some steps may be commented less verbosely.

Proof. We aim for an effective series period $T_* = \max\{T_w, T_r\}$ equal to T_r in this case. To guarantee that every read event actually reads new data, we need

$$\max_j\{t_w(j+1) - t_w(j)\} \leq \min_j\{t_r(j+1) - t_r(j)\}$$
$$\Leftrightarrow \quad T_w + J_w \leq T_r - J_r,$$

which is our hypothesis of (29). Then, the read sequence t_r^* of the effective write/read sequence has the same period T_r of t_r, that is

$$t_r^*(j) = j\,T_r + \Phi_r + k_r T_r + \delta_r(j), \qquad k_r \in \mathbb{N} \tag{31}$$

with k_r representing some possibly later start of t_r^* w.r.t. t_r.

The instant $t_w^*(j)$, equal to $t_w(j_w)$ for some j_w, is the last write event before $t_r^*(j)$, then it needs to satisfy $t_w(j_w) = j_w T_w + \Phi_w + \delta_w(j_w) \leq t_r^*(j) = j\,T_r + \Phi_r + k_r T_r + \delta_r(j)$. From (9), j_w must be the biggest integer satisfying the inequality, that is

$$j_w = \left\lfloor \frac{j\,T_r + k_r T_r + \Delta + \delta_r(j) - \delta_w(j_w)}{T_w} \right\rfloor. \tag{32}$$

Such index j_w must necessarily be non-negative. If j_w is negative for $k_r = 0$, we can increase k_r to make it such,

$$\left\lfloor \frac{k_r T_r + \Delta + \delta_r(j) - \delta_w(j_w)}{T_w} \right\rfloor \geq 0 \qquad \Leftrightarrow$$
$$\frac{k_r T_r + \Delta + \delta_r(j) - \delta_w(j_w)}{T_w} \geq 0$$

which is guaranteed by

$$\frac{k_r T_r + \Delta - J_w}{T_w} \geq 0 \quad \Leftrightarrow \quad k_r \geq \left\lceil \frac{J_w - \Delta}{T_r} \right\rceil.$$

Hence, by setting $k_r = \max\left\{0, \left\lceil \frac{J_w - \Delta}{T_r} \right\rceil\right\}$, we have that $t_r^* \in \mathcal{F}_{r^*}$, as required by Def. 6, when

$$T_* = T_r, \qquad \Phi_{r^*} = \Phi_r + k_r\,T_r, \qquad J_{r^*} = J_r.$$

Before replacing the expression of j_w of (32) in $t_w(j_w)$ to find $t_w^*(j)$, it is convenient to manipulate $k_r T_r + \Delta$ as follows

$$k_r T_r + \Delta = \overbrace{\Phi_{r^*} - \Phi_r}^{k_r T_r} + \overbrace{\Phi_r - \Phi_w}^{\Delta} = \Phi_{r^*} - \Phi_w.$$

Replacing now j_w of (32) in $t_w(j_w)$, we can find $t_w^*(j)$

$$\begin{aligned} t_w^*(j) &= \left\lfloor \frac{j\,T_r + \Phi_{r^*} - \Phi_w + \delta_r(j) - \delta_w(j_w)}{T_w} \right\rfloor T_w + \Phi_w \\ &\qquad + \delta_w(j_w) \\ &= \left\lfloor \frac{j\,T_r + \Phi_{r^*} - \Phi_w + \delta_r(j) - \delta_w(j_w)}{T_w} \right\rfloor T_w \\ &\quad - (j\,T_r + \Phi_{r^*} - \Phi_w + \delta_r(j) - \delta_w(j_w)) \\ &\quad + \Phi_{r^*} + \delta_r(j) + j\,T_r \\ &= j\,T_r + \Phi_{r^*} + \delta_r(j) \\ &\quad - \lfloor j\,T_r + \Phi_r^* - \Phi_w + \delta_r(j) - \delta_w(j_w) \rfloor_{T_w} \end{aligned} \tag{33}$$

with the exploitation of property of (1). From such expression for $t_w^*(j)$, we find that $j\,T_r + \Phi_{r^*} - T_w < t_w^*(j) \leq j\,T_r + \Phi_{r^*} + J_r$. Hence, from the expression of event sequences of (4) and (5), the parameters of w^* that guarantees $t_w^* \in \mathcal{F}_{w^*}$ are

$$\Phi_{w^*} = \Phi_{r^*} - T_w, \qquad J_{w^*} = T_w + J_r.$$

as in Eq. (30), concluding the proof. □

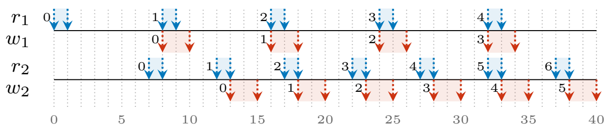
Now, the analysis of the write-to-read series is complete. Next, we combine them into a whole chain analysis.

V. From write/read series to chain analysis

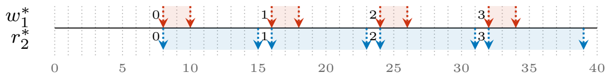
In this section, we exploit the properties of effective series to analyze task chains. For this purpose, we resume the task indices, with τ_1 being the first task of the chain and τ_n the last one, as introduced in Section II-B. Our proposed logic composes a pair of communicating tasks into an aggregate task, which fully represents all possible effective write/read sequences of the pair. Such a composition can be applied recursively to enable the analysis of the whole chain, as made by others [29], [36]. Since we focus on the aggregation of two tasks, we start assuming a simple two tasks chain $\tau_1 \to \tau_2$, with the goal of determining the aggregate task denoted by $\tau_{1,2}$ and its read and write events $r_{1,2}$, $w_{1,2}$ representing the behavior of the chain $\tau_1 \to \tau_2$. To support the explanation, we are using the example of Fig. 8. As in the example, we illustrate in detail the case with $P_1 > P_2$, giving the other cases less space.

Fig. 8a shows the original series: read/write series (r_1, w_1) of τ_1, and read/write series (r_2, w_2) of task τ_2. At the interaction between w_1 and r_2, some read instants are redundant. We apply then the results of Section IV to find an effective write/read series (w_1^*, r_2^*) from (w_1, r_2). When the write series period $T_{w_1} = P_1$ is longer than the period $T_{r_2} = P_2$, we apply Theorem 3 to determine an effective write/read series. First, we need to check if the hypothesis of Eq. (23) holds. For the example of Fig. 8a, it holds since $T_{r_2} + J_{r_2} = 5 + 1 \leq T_{w_1} - J_{w_1} = 8 - 2$.

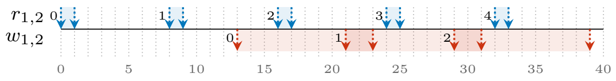
As the hypothesis of Eq. (23) holds, by applying Theorem 3 we can find an effective write/read series (w_1^*, r_2^*) from (w_1, r_2). In the case of the example, the common period T_*



(a) The original read and write event series of the two tasks..



(b) The effective write/read event series (w_1^*, r_2^*) found from (w_1, r_2). In this case, Theorem 3 is applied because $T_{w_1} > T_{r_2}$.



(c) The read and write events $(r_{1,2}, w_{1,2})$ of the chain $\tau_{1,2}$ are found. The read event $r_{1,2}$ is the same as r_1 because $w_1^* = w_1$. The write event series $w_{1,2}$, instead, is found from Eq. (36).

Fig. 8: Determining the read and write event series of the chain $\tau_{1,2}$ from the tasks τ_1 and τ_2. The two tasks have period $P_1 = 8$ and $P_2 = 5$. Their read and write events are $r_1 = (8, 0, 1)$, $w_1 = (8, 8, 2)$, $r_2 = (5, 7, 1)$, and $w_2 = (5, 13, 2)$, in the format (T_e, Φ_e, J_e) for any event e.

of w_1^* and r_2^* is the largest among the two periods, which is $T_{w_1} = 8$. The other parameters of (w_1^*, r_2^*) are found from (24)

$$\Delta = \Phi_{r_2} - \Phi_{w_1} = 7 - 8 = -1$$

$$\Phi_{w_1^*} = \Phi_{r_2^*} = \Phi_{w_1} + \max\left\{0, \left\lfloor \frac{\Delta + J_{r_2} - T_{r_2}}{T_{w_1}} \right\rfloor + 1\right\} T_{w_1} = 8,$$

$$J_{w_1^*} = J_{w_1} = 2, \quad J_{r_2^*} = T_{r_2} + J_{w_1} = 7.$$

This found effective write/read series (w_1^*, r_2^*) is drawn in Fig. 8b.

The effective series (w_1^*, r_2^*) models the combination of the effective write sequences of τ_1 and the effective read sequence of τ_2. To complete the chain model $\tau_1 \to \tau_2$, we need to find the two series $r_{1,2}$ and $w_{1,2}$. These two series respectively model the read of the head task of the chain and the write of the tail task of the chain $\tau_1 \to \tau_2$, respectively. The feasible sequences in $\mathcal{F}_{r_{1,2}}$ of $r_{1,2}$ are the read sequences of the jobs of τ_1 that write at time instants $t_{w_1^*} \in \mathcal{F}_{w_1^*}$. The feasible sequences in $\mathcal{F}_{w_{1,2}}$ of $w_{1,2}$ are the sequences of write instants of τ_2 jobs that read at time instant $t_{r_2^*} \in \mathcal{F}_{r_2^*}$. Hence, $r_{1,2}$ follows from the parameters of τ_1 and w_1^*. In a similar way, $w_{1,2}$ is found from the parameters of τ_2 and r_2^*.

When $P_1 = T_{w_1} > T_{r_2} = P_2$, as in the example of Fig. 8, the series w_1^* is the same as w_1, except a possibly later offset $\Phi_{w_1^*}$ in accordance to (24). Consequently, the read series $r_{1,2}$ has parameters

$$T_{r_{1,2}} = T_*, \; J_{r_{1,2}} = J_{r_1}, \; \Phi_{r_{1,2}} = \Phi_{r_1} + \Phi_{w_1^*} - \Phi_{w_1}. \quad (34)$$

In the example of Fig. 8, since $\Phi_{w_1^*} = \Phi_{w_1} = 8$, we simply have $r_{1,2} = r_1$ because all parameters coincide. Finding $w_{1,2}$ from r_2^* requires the determination of bounds on t_{w_2} from bounds on t_{r_2} and bounds on the separation between the read and write instants of τ_2. From (6), we have

$$m_i \le t_{w_i}(j) - t_{r_i}(j) \le M_i, \quad (35)$$

Algorithm 2 Determination of the read and write event series of the chain $\tau_{1,2}$ of the tasks τ_1 and τ_2 from the read and write series of each task.

```
Require: $(r_1, w_1), (r_2, w_2)$            ▷ read and write series of $\tau_1$, $\tau_2$
 1: $T_* = \max\{T_{w_1}, T_{r_2}\}$          ▷ also equal to $\max\{P_1, P_2\}$
 2: $T_{r_{1,2}} = T_{w_{1,2}} = T_*$  ▷ Period of $r_{1,2}$ and $w_{1,2}$ is the maximum
 3: if $P_1 > P_2$ then                                    ▷ Th. 3 applies
 4:     Get $\Phi_{w_1^*}$, $J_{w_1^*}$, $\Phi_{w_2^*}$, $J_{w_2^*}$ from (24)
 5:     $\Phi_{r_{1,2}} \leftarrow \Phi_{r_1} + (\Phi_{w_1^*} - \Phi_{w_1})$, $J_{r_{1,2}} \leftarrow J_{r_1}$      ▷ Eq. (34)
 6:     $\Phi_{w_{1,2}} \leftarrow \Phi_{r_2^*} + m_2, J_{w_{1,2}} \leftarrow J_{r_2^*} + M_2 - m_2$   ▷ Eq. (36)
 7: else if $P_1 < P_2$ then                               ▷ Th. 4 applies
 8:     Get $\Phi_{w_1^*}$, $J_{w_1^*}$, $\Phi_{w_2^*}$, $J_{w_2^*}$ from (30)
 9:     $\Phi_{w_{1,2}} \leftarrow \Phi_{w_2} + (\Phi_{r_2^*} - \Phi_{r_2})$, $J_{w_{1,2}} \leftarrow J_{w_2}$      ▷ Eq. (37)
10:     $\Phi_{r_{1,2}} \leftarrow \Phi_{w_1}^* - M_1, J_{r_{1,2}} \leftarrow J_{w_1^*} + M_1 - m_1$   ▷ Eq. (38)
11: else                 ▷ Th. 2 applies, only adjustment of offset if needed
12:     Get $\Phi_{w_1^*}$, $J_{w_1^*}$, $\Phi_{w_2^*}$, $J_{w_2^*}$ from (19)
13:     $\Phi_{r_{1,2}} \leftarrow \Phi_{r_1} + (\Phi_{w_1^*} - \Phi_{w_1})$, $J_{r_{1,2}} \leftarrow J_{r_1}$      ▷ Eq. (34)
14:     $\Phi_{w_{1,2}} \leftarrow \Phi_{w_2} + (\Phi_{r_2^*} - \Phi_{r_2})$, $J_{w_{1,2}} \leftarrow J_{w_2}$      ▷ Eq. (37)
15: $r_{1,2} \leftarrow (T_{r_{1,2}}, \Phi_{r_{1,2}}, J_{r_{1,2}})$
16: $w_{1,2} \leftarrow (T_{w_{1,2}}, \Phi_{w_{1,2}}, J_{w_{1,2}})$
17: return $(r_{1,2}, w_{1,2})$        ▷ read and write series of the chain $\tau_{1,2}$
```

with m_i and M_i, lower and upper bound to the separation between read and write instants of τ_i. Then, for τ_2 we can establish the following bounds on the write instants

$$t_{r_2}(j) + m_2 \le t_{w_2}(j) \le t_{r_2}(j) + M_2,$$
$$jP_2 + \Phi_{r_2} + m_2 \le t_{w_2}(j) \le jP_2 + \Phi_{r_2} + J_{r_2} + M_2$$

which yields the following parameters of $w_{1,2}$ from r_2^*

$$T_{w_{1,2}} = T_*, \; \Phi_{w_{1,2}} = \Phi_{r_2^*} + m_2, \; J_{w_{1,2}} = J_{r_2^*} + M_2 - m_2. \quad (36)$$

For the example of Fig. 8, we first find $m_2 = \Phi_{w_2} - \Phi_{r_2} - J_{r_2} = 5$ and $M_2 = \Phi_{w_2} - \Phi_{r_2} + J_{w_2} = 8$ from Corollary 1. Then the parameters of the event $w_{1,2}$ are $T_{w_{1,2}} = 8$, $\Phi_{w_{1,2}} = 13$, and $J_{w_{1,2}} = 10$.

We now, more concisely, complete the analysis with the remaining cases. When $P_1 < P_2$, then it is now r_2^* which is the same as r_2 with a possibly later offset. Hence, $w_{1,2}$ is parametrized by

$$T_{w_{1,2}} = T_*, \; J_{w_{1,2}} = J_{w_2}, \; \Phi_{w_{1,2}} = \Phi_{w_2} + \Phi_{r_2^*} - \Phi_{r_2}. \quad (37)$$

The derivation of $r_{1,2}$ from w_1^* and m_1, M_1 mimics the steps of the previous case. From (35), we find

$$t_{w_1}(j) - M_1 \le t_{r_1}(j) \le t_{w_1}(j) - m_1$$

which yields

$$T_{r_{1,2}} = T_*, \; \Phi_{r_{1,2}} = \Phi_{w_1^*} - M_1, \; J_{r_{1,2}} = J_{w_1^*} + M_1 - m_1. \quad (38)$$

Finally, the case with $P_1 = P_2$ is the easiest, because no jitter is changed, only the offsets may increase to synchronize. Hence, in this case, the series $r_{1,2}$ and $w_{1,2}$ are simply given by Eq. (34) and (37), respectively. In all cases, the period $P_{1,2}$ of the chain $\tau_{1,2}$ is always $P_{1,2} = T_{r_{1,2}} = T_{w_{1,2}} = \max\{P_1, P_2\}$.

Algorithm 2 illustrates in an algorithmic form all the above steps. Indeed the procedure can be applied iteratively, so that the whole chain $\tau_{1,n}$ can be summarized by the read/write series $(r_{1,n}, w_{1,n})$ and their corresponding parameters.

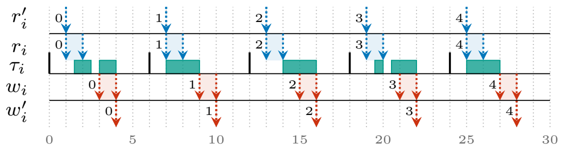


Fig. 9: Example of jitter elimination. For any task τ_i, its read/write event series are (r_i, w_i) with $T_{r_i} = T_{w_i} = P_i = 6$, $\Phi_{r_i} = 1$, $J_{r_i} = 1$, $\Phi_{w_i} = 3$, $J_{w_i} = 1$. To eliminate jitter, we insert an auxiliary event sequence r_i' before the read and w_i' after the write, such that $T_{r_i'} = T_{w_i'} = P_i = 6$, $\Phi_{r_i'} = 1$, $\Phi_{w_i'} = 4$, and $J_{r_i'} = J_{w_i'} = 0$, thereby making he execution of tasks resemble the Logical-Execution-Time (LET) mechanism.

From the parameters of the aggregate task $\tau_{1,n}$, bounds on the chain latencies [35], [36] can be found. For example, from (8) the maximum reaction time, or First-to-First latency in Feiertag's terminology [35], is equal to

$$D^{\mathsf{FF}}_{\mathsf{bound}} = T_{1,n} + \Phi_{w_{1,n}} - \Phi_{r_{1,n}} + J_{w_{1,n}}. \tag{39}$$

On the one hand, we provide a linear-time algorithm that computes an upper bound on chain latencies when no other bound is known. On the other hand, our results (Theorems 2, 3, 4) rely on hypotheses. If they do not hold, Algorithm 2 cannot be applied. Next, we present an *active* analysis, which reduces the jitter by some LET mechanism when needed.

A. Active analysis

In Logical Execution Time (LET), the execution of tasks is wrapped by a LET mechanism which reads and writes data at jitter-free instants. Fig. 9 illustrates how the jitter of the read and write event series of any task τ_i can be eliminated with a "prefix" event series r_i' and a "postfix" series w_i' with parameters

$$\begin{aligned} T_{r_i'} &= P_i, & \Phi_{r_i'} &= \Phi_{r_i}, & J_{r_i'} &= 0, \\ T_{w_i'} &= P_i, & \Phi_{w_i'} &= \Phi_{w_i} + J_{w_i}, & J_{w_i'} &= 0. \end{aligned} \tag{40}$$

The same mechanism can be implemented in our logic. By observing that a jitter reduction makes the condition of Eq. (18), (23), and (29) always looser, we proceed as follows in the aggregation of tasks:

- We aggregate tasks following the procedure of Algorithm 2.
- When the condition of the hypothesis of any of the theorems does not hold, we can properly add the events r_i' and w_{i+1}' computed from (40), with parameters of Eq. (40).

By doing so, the jitter between τ_i and τ_{i+1} is eliminated and the hypotheses of the theorems are more likely to hold.

VI. Evaluation

Our methods are evaluated on randomly generated task chains. The length n of the chain is randomly picked in $\{3, 5, 8, 10\}$. The maximum chain length of 10 borrowed from the Autoware application[1], widely used by the community to simulate real-world scenarios. The task periods are drawn from the set $T = \{1, 2, 5, 10, 20, 50, 100, 200, 1000\}$ as in the

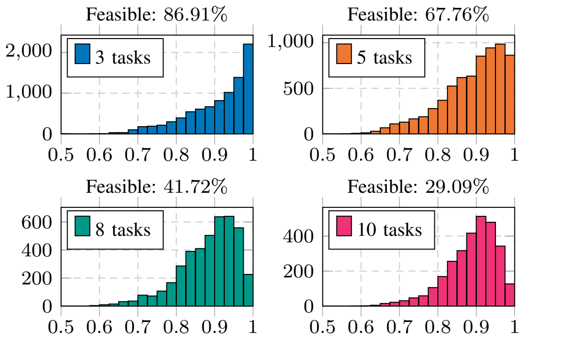


Fig. 10: Distribution of the ratio R of the maximum reaction time over the bound of Eq. (39), with the passive analysis when jitter is based on task scheduling (the task parameters are not changed). 10000 random samples were generated. On top of each plot, beside the "Feasible" label, we report the percentage of sample task sets for which our analysis was applicable.

automotive benchmark proposed by Kramer et al. [40]. By this choice of periods, the minimun ratio $\min_{P_i > P_j} \frac{P_i}{P_j}$ is equal to 2, making our passive analysis applicable in a significant number of cases.

We performed two families of experiments: the *passive* analysis (Figures 10, 11, and 12), in which the chain is analyzed as generated, and the *active* analysis (Figures 13 and 14), in which the jitter is possibly eliminated by LET or the offset adjusted through Corollary 2 when Algorithm 2 fails because of an excessive jitter, as described in Section V-A. For each family of experiments, we evaluated both the jitter as it emerges from task schedules, and a generic jitter as fraction of the period. The Python code implementing both passive and active analyses is publicly available[2].

Schedule-based jitter: To simulate a task schedule, we generate the execution time of tasks with UUniFast [41] set to a total utilization of 50%. All tasks are scheduled onto the same CPU with Rate Monotonic priority ordering. We compute the maximum reaction time [20] and denote its value by $D^{\mathsf{FF}}_{\mathsf{base}}$. Then, to apply our method, we use the same schedule to determine the read and write series r_i and w_i of any task τ_i, as described in Section II-B. We apply Algorithm 2 to find the event series $(r_{1,n}, w_{1,n})$ of the whole chain $\tau_{1,n}$ and then the bound $D^{\mathsf{FF}}_{\mathsf{bound}}$ on the First-to-First latency from (39). We evaluate the pessimism of the bound by plotting the distribution of the ratio $R = D^{\mathsf{FF}}_{\mathsf{base}}/D^{\mathsf{FF}}_{\mathsf{bound}}$ over all random chains for which Algorithm 2 completes. Values of R closer to 1 indicate a tighter bound.

Jitter as period percentage: One of the key issues with our approach is the possible inapplicability of Algorithm 2 due to the hypotheses of the composition theorems. We evaluate this weakness by spanning jitter values independently of schedules. We do so by setting:

- Random read offset $\Phi_{r_i} \in [0, T_{r_i} - 1]$,
- Write offset set to the read offset plus one period, $\Phi_{w_i} = \Phi_{r_i} + P_i$,
- Read/write jitter J_{r_i} and J_{w_i} equal to a percentage $x\,\%$ of the period P_i, with $x \in \{0, 2, 5, 10, 20, 30, 40, 50\}$. De-

[1]urlhttps://github.com/ros-realtime/reference-system/

[2]https://github.com/wsm6636/jitter_event_code/

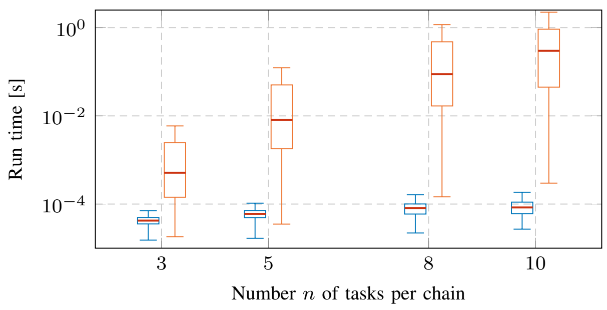


Fig. 11: Comparison of runtime between classic max reaction time computation [20] (shown in orange) and our composition of Algorithm 2 (shown in blue), under passive analysis when jitter is based on task scheduling

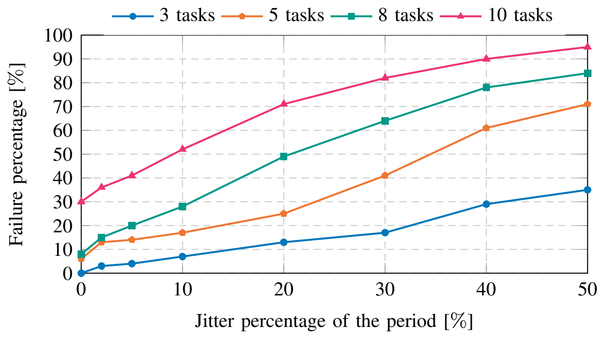


Fig. 12: The failure percentage of Algorithm 2 increases as jitter as percentage of period increases, under passive analysis.

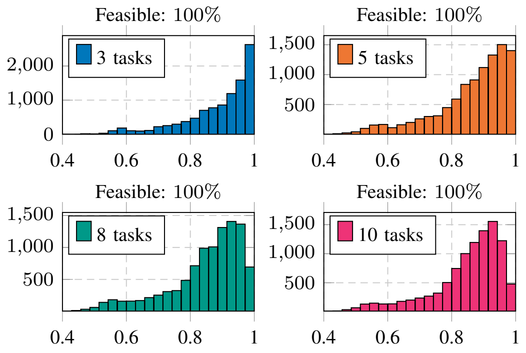


Fig. 13: Distribution of the ratio R of the global maximum reaction time over the bound of Eq. (39), with the active analysis when jitter is based on task scheduling. The jitter percentage is set to 20%.

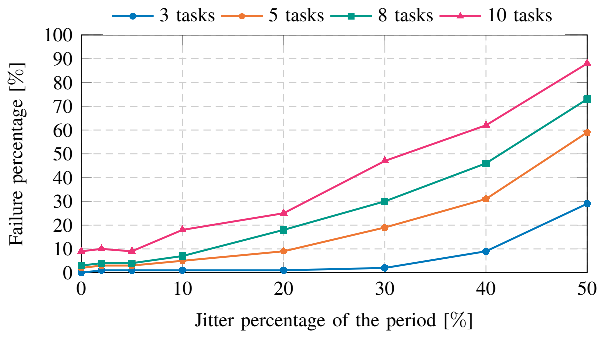


Fig. 14: Percentage of failure of Algorithm 2 increases as jitter as percentage of period increases with the active analysis: the jitter may be reduced with the introduction of LET, the offset may be changed through Corollary 2.

spite representing the jitter interval by a filled rectangle, no jitter distribution is assumed. Selecting 0 % for both J_{r_i} and J_{w_i} corresponds to LET communication.

Passive analysis: Fig. 10 shows the distribution of the ratio $R = D^{\mathsf{FF}}_{\mathsf{base}}/D^{\mathsf{FF}}_{\mathsf{bound}}$. We remark that the bound of Eq. (39) is always a safe upper bound ($R \leq 1$). As the number n of tasks in the chain grows, the pessimism of our bound increases (lower values of R) together with the inapplicability of our algorithm (decreasing "Feasible" values). This is expected as jitter grows with n. From Fig. 11, we see that the runtime of our method is orders of magnitude smaller than the baseline for maximum reaction time [20]. This is not surprising given the linear time complexity of our composition. Still, both methods complete in less than a second for $n \leq 10$.

In Fig. 12, we simulate the jitter as percentage of the task periods. The failure rate increases with both jitter percentage and chain length. Even when jitter is zero (equivalent to LET), some chains still fail due to conditions in Eq. (24) and (30).

Active analysis: As illustrated in Fig. 12, the failure of Algorithm 2 increases with the number of tasks and the jitter percentage. Figures 13 and 14 show the distribution of R and the failure rate when the chain is modified as follows. Firstly, if the condition of Eq. (23) or (29) are not true for some write task τ_i and reader τ_{i+1}, then two events w_i and r_{i+1} are transformed in w'_i, r'_{i+1} of Eq. (40), thus eliminating the jitter between τ_i and τ_{i+1}. Secondly, for event pairs that cannot satisfy Eq. (18), we adjust the offset of the read and write events based on Corollary 2 to make Eq. (18) true. Unlike Eq. (23) and (29), which are only related to period and jitter, the influence of offset in Eq. (18) is crucial.

From the histogram plots of Fig. 13, we observe that the jitter elimination now allows our algorithm to always complete successfully. Also, by comparing Fig. 14 with Fig. 12, we see that the active analysis reduces the failure rate of Algorithm 2.

VII. Conclusion and Future Works

In this paper, we proposed an event model for read and write events, in the context of cause-effect chains. Our model decouples the events from the scheduling of tasks. The most distinguishing feature of our work is the explicit account of the jitter in the event model. We then propose our jitter-aware analysis of the concatenation of write-to-read events.

The spectrum of future works is very broad. It spans from the combination of our analysis with schedulability analysis of tasks, to a more detailed model of uncertainties which enables a tighter analysis.

Acknowledgments

The authors would like to thank the reviewers of this paper, as well as the reviewers of an earlier version submitted to ECRTS'25. We are especially grateful to the shepherd for dedicating additional time to evaluate our manuscript. Receiving such high-quality feedback from the research community is truly rewarding.

This work is partially supported by the project "Trustworthy Cyber-Physical Pipelines", funded by the MAECI Italy-Sweden co-operation id. PGR02086, and by VR grant number 2023-06836. This work is also partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, via the NEST projects Cloud-Robotics: Intelligent Cloud Robotics for Real-Time Manipulation at Scale and by the National Natural Science Foundation of China "Analysis, Modeling and Algorithm Research on Real-time Scheduling Problems of Hybrid Business Flows in Industrial Heterogeneous Networks Based on TSN" (No.62072085), and the China Scholarship Council.

References

[1] S. Ravi, L. Beermann, O. Kotte, P. Pazzaglia, M. Vinnakota, D. Ziegenbein, and A. Hamann, "Timing-aware software-in-the-loop simulation of automotive applications with FMI 3.0," in *26th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS)*, 2023, pp. 62–72.

[2] T. Kronauer, J. Pohlmann, M. Matthé, T. Smejkal, and G. Fettweis, "Latency analysis of ros2 multi-node systems," in *IEEE MFI*, 2021, pp. 1–7.

[3] M. Günzel, H. Teper, K.-H. Chen, G. von der Brüggen, and J.-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)*, vol. 262, 2023, pp. 10:1–10:22.

[4] K. Tindell and J. Clark, "Holistic schedulability analysis for distributed hard real-time systems," *Microprocessing and microprogramming*, vol. 40, no. 2-3, pp. 117–134, 1994.

[5] T. A. Henzinger, B. Horowitz, and C. M. Kirsch, "Giotto: A time-triggered language for embedded programming," *Proceedings of the IEEE*, vol. 91, no. 1, 2003.

[6] C. M. Kirsch and A. Sokolova, "The logical execution time paradigm," *Advances in Real-Time Systems*, pp. 103–120, 2012.

[7] R. Bettati and J. W. Liu, "Algorithms for end-to-end scheduling to meet deadlines," in *Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing 1990*. IEEE, 1990, pp. 62–67.

[8] J. Sun and J. Liu, "Synchronization protocols in distributed real-time systems," in *Proceedings of 16th International Conference on Distributed Computing Systems*. IEEE, 1996, pp. 38–45.

[9] J. C. Palencia and M. Gonzalez Harbour, "Exploiting precedence relations in the schedulability analysis of distributed real-time systems," in *20th IEEE Real-Time Systems Symposium*, 1999, pp. 328–339.

[10] R. Pellizzoni and G. Lipari, "Holistic analysis of asynchronous real-time transactions with earliest deadline scheduling," *Journal of Computer and System Sciences*, vol. 73, no. 2, pp. 186–206, 2007.

[11] R. Henia and R. Ernst, "Improved offset-analysis using multiple timing-references," in *Proceedings of the Design Automation & Test in Europe Conference*, vol. 1. IEEE, 2006, pp. 6–pp.

[12] S. Schliecker and R. Ernst, "A recursive approach to end-to-end path latency computation in heterogeneous multiprocessor systems," in *Proceedings of the 7th IEEE/ACM international conference on Hardware/software codesign and system synthesis*, 2009, pp. 433–442.

[13] J. Schlatow and R. Ernst, "Response-time analysis for task chains in communicating threads," in *2016 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2016, pp. 1–10.

[14] P. S. Kurtin, J. P. Hausmans, and M. J. Bekooij, "Combining offsets with precedence constraints to improve temporal analysis of cyclic real-time streaming applications," in *IEEE RTAS*, 2016, pp. 1–12.

[15] J. Choi, H. Oh, and S. Ha, "A hybrid performance analysis technique for distributed real-time embedded systems," *Real-Time Systems*, vol. 54, pp. 562–604, 2018.

[16] L. T. X. Phan, R. Schneider, S. Chakraborty, and I. Lee, "Modeling buffers with data refresh semantics in automotive architectures," in *10th ACM EMSOFT*, 2010, pp. 119–128.

[17] Y. Tang, N. Guan, X. Jiang, Z. Dong, and W. Yi, "Reaction time analysis of event-triggered processing chains with data refreshing," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.

[18] A. Davare, Q. Zhu, M. Di Natale, C. Pinello, S. Kanajan, and A. Sangiovanni-Vincentelli, "Period optimization for hard real-time distributed automotive systems," in *DAC*, 2007.

[19] M. Dürr, G. V. D. Brüggen, K.-H. Chen, and J.-J. Chen, "End-to-end timing analysis of sporadic cause-effect chains in distributed systems," *Transactions on Embedded Computing Systems*, vol. 18, no. 5, 2019.

[20] M. Günzel, K.-H. Chen, N. Ueter, G. von der Brüggen, M. Dürr, and J.-J. Chen, "Timing analysis of asynchronized distributed cause-effect chains," in *Real Time and Embedded Technology and Applications Symposium (RTAS)*, 2021.

[21] S. Sinha and R. West, "End-to-end scheduling of real-time task pipelines on multiprocessors," *Journal of Systems Research*, vol. 2, no. 1, 2022.

[22] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, 2022.

[23] 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)*. IEEE, 2022, pp. 53–65.

[24] H. Teper, T. Betz, M. Günzel, D. Ebner, G. Von Der Brüggen, J. Betz, and J.-J. Chen, "End-to-end timing analysis and optimization of multi-executor ROS2 systems," in *30th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2024, pp. 212–224.

[25] Y. Tang, X. Jiang, N. Guan, X. Luo, M. Yang, and W. Yi, "Timing analysis of processing chains with data refreshing in ROS2," *Journal of Systems Architecture*, vol. 155, p. 103259, 2024.

[26] A. Cervin, "Stability and worst-case performance analysis of sampled-data control systems with input and output jitter," in *IEEE American Control Conference, ACC*, Jun. 2012, pp. 3760–3765.

[27] A. Kordon and N. Tang, "Evaluation of the age latency of a real-time communicating system using the let paradigm," in *Euromicro Conference on Real-Time Systems (ECRTS)*, 2020.

[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)*, 2017.

[29] E. Bini, P. Pazzaglia, and M. Maggio, "Zero-jitter chains of periodic let tasks via algebraic rings," *IEEE Transactions on Computers*, vol. 72, no. 11, pp. 3057–3071, 2023.

[30] H. Abaza, D. Roy, B. Trach, W. Chang, S. Saidi, A. Motakis, W. Ren, and Y. Liu, "Managing end-to-end timing jitters in ROS2 computation chains," in *Proceedings of the 32nd International Conference on Real-Time Networks and Systems*, 2024, pp. 229–241.

[31] A. Yano and T. Azumi, "Deadline miss early detection method for mixed timer-driven and event-driven dag tasks," *IEEE Access*, vol. 11, pp. 22 187–22 200, 2023.

[32] D. Yamazaki and T. Azumi, "Chain-aware scheduling for mixed timer-driven and event-driven dag tasks," in *Proc. of APRIS*, 2022, pp. 42–49.

[33] S. Kato, "The art of open source and real-time for autonomous driving," Keynote talk at RTSS'23, Dec. 2023, taipei, Taiwan.

[34] B. B. Brandenburg, C. Courtaud, F. Marković, and B. Ye, "LiME: The Linux real-time task model extractor," in *Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2025, pp. 255–269.

[35] 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 (CRTS)*, 2008.

[36] 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, pp. 6:1–6:25.

[37] 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*, vol. 80, 2017.

[38] P. Pazzaglia and M. Maggio, "Characterising the effect of deadline misses on time-triggered task chains," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.

[39] 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*, vol. 37, no. 11, pp. 2244–2254, 2018.

[40] S. Kramer, D. Ziegenbein, and A. Hamann, "Real world automotive benchmarks for free," in *6th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, vol. 130, 2015, p. 43.

[41] E. Bini and G. C. Buttazzo, "Measuring the performance of schedulability tests," *Real-time systems*, vol. 30, no. 1, pp. 129–154, 2005.