Formal Timing Analysis of Non-Scheduled Traffic in Automotive Scheduled TSN Networks

Fedor Smirnov*, Michael Glaß†, Felix Reimann‡, Jürgen Teich*
*Hardware/Software Co-Design, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany
†Institute of Embedded Systems/Real-Time Systems, Ulm University, Germany
‡Audi Electronics Venture GmbH, Germany
{fedor.smirnov, juergen.teich}@fau.de, michael.glass@uni-ulm.de, felix.reimann@audi.de

***Abstract*—To cope with requirements for low latency, the upcoming Ethernet standard Time-Sensitive Networking (TSN) provides enhancements for scheduled traffic, enabling mixed-criticality networks where critical messages are sent according to a system-wide schedule. While these networks provide a completely predictable behavior of the scheduled traffic by construction, timing analysis of the critical non-scheduled traffic with hard deadlines remains an unsolved issue. State-of-the-art analysis approaches consider the interference that unscheduled messages impose on each other, but there is currently no approach to determine the worst-case interference that can be imposed by scheduled traffic, the so-called *schedule interference* (SI), without relying on restrictions of the shape of the schedule. Considering all possible interference scenarios during each calculation of the SI is impractical, as it results in an explosion of the computation time. As a remedy, this paper proposes a) an approach to integrate the analysis of the worst-case SI into state-of-the-art timing analysis approaches and b) preprocessing techniques that reduce the computation time of the SI-calculation by several orders of magnitude without introducing any pessimism.**

I. Introduction

To enable real-time applications that require low latency or even deterministic message delivery, like X-By-Wire or advanced driver assistance systems, the upcoming TSN standard introduces, among other things, enhancements for *scheduled traffic* [4]. In a TSN network, the output ports of the switches are equipped with *Time-Aware Shapers* (TASs). These shapers provide time-triggered link access for scheduled messages by transmitting their frames according to a *port schedule*. In combination with the synchronization of the talkers sending scheduled messages, the TAS enables the implementation of networks with a system-wide transmission schedule. For each scheduled message, the schedule defines points in time when the message is sent by the talker and the so-called *port slots* (PSs) on the Ethernet ports, during which frames of the message have exclusive access to the transmission queue of the port and cannot encounter any interference from unscheduled frames. The PSs are defined statically and unscheduled traffic is never transmitted within these slots.

To guarantee that unscheduled critical messages are delivered within their deadlines, a formal timing analysis of the network is required. Existing analysis approaches like [1] or [8] can determine the interference that the unscheduled messages impose on each other, but have to be extended to account for the interference by the scheduled traffic.

Contributions. In this paper, we present a formal method for the analysis of the interference that scheduled traffic in a TSN network imposes on each unscheduled message. We assume that the scheduled traffic routed through the analyzed port is described by a port schedule. As the generation of a valid global schedule is already a difficult task, our approach does not make any assumptions about the shape of the port schedules resulting from the global schedule. Consequently, an exhaustive examination of all arrival points of the unscheduled workload that could possibly lead to the maximal SI is necessary to derive a safe bound. This results in long run times of the timing analysis and makes the approach impractical for an automated design space exploration (DSE). To cope with this issue, we present preprocessing techniques that, prior to the timing analysis, transform the given schedule into a data structure that contains only information relevant for the worst-case and can be used to quickly obtain the worst-case SI for an analyzed time interval of arbitrary length. With this, the SI is calculated faster by several orders of magnitude.

In the experiment section of this paper, we demonstrate the potential of TSN networks to reduce the delay of critical messages, the applicability of the presented analysis, and the speedup and the exactness of the results achieved by the proposed preprocessing.

II. Related Work

A rich work on the timing analysis of switched Ethernet exists in literature. Formal analysis approaches like real-time calculus (RTC) [8] or compositional performance analysis (CPA) [1] have been proven to provide safe worst-case latency bounds for Ethernet. Analysis for IEEE 802.1Q can be found, e.g., in [2]. To our knowledge, there are currently two works presenting approaches for the formal analysis of an Ethernet TSN network: In [6], the authors assume that the schedule on the output ports of TSN switches consists of just one periodic slot, while the authors of [7] assume a schedule that consists of one periodic slot per critical traffic class. During the design of realistic networks with many messages with different application-specific periods, it is very hard to generate overlap-free schedules following these restrictions. In contrast to both works, the applicability of the analysis approach presented in this work does not depend on any restrictions regarding the shape of the schedules on the output ports and can, consequently, be used with any schedule.

III. Timing Analysis

In this section, we use the busy-period analysis for strict priority non-preemptive Ethernet (IEEE 802.1 Q) presented in [1] to show how our analysis of the SI can be integrated into state-of-the-art analysis frameworks.

Timing description. The timing of a message transmission in an Ethernet network is characterized by two values: the *delay* and the *jitter*. The delay is the transmission time of the message in the best case. The jitter is the difference between the worst- and the best-case transmission times of

978-3-9815370-8-6/17/$31.00

the message. In Ethernet networks, jitter is mainly caused by the interference of other messages on the switch output ports.

Delay analysis. The message delay is analyzed by summing up the constant delay contributions of all resources passed by the frames of the message on the way to the listener.

Jitter analysis. The main focus of this paper is the analysis of the jitter experienced by unscheduled messages in a TSN network. Without loss of generality, we assume a network configuration as described in Section I, where the frames of the scheduled messages do not experience any jitter.

Assume that we want to analyze the interference imposed on a frame of the unscheduled message i on the output port processing the messages of the set F. Based on their traffic class, all messages on this port are separated in sets relatively to the analyzed message i: the low-priority message set L, the same-priority message set S, and the high-priority message set H. The maximal interference t_i on the n-th frame of the analyzed message, arriving at the time point t_a, is given by a sum of the maximal interference caused by low-priority traffic t_L, the transmission time of previous frames of the analyzed message t_O, the interference caused by same-priority traffic t_S, and the interference of high-priority traffic t_H.

Note: In the network configuration considered in this paper, all unscheduled messages are in the preemptable, while all scheduled messages are in the express preemption class. The preemption mechanisms are therefore only relevant when calculating the interference caused by scheduled traffic.

Low-Priority Interference The worst-case low-priority interference t_L is given by the transmission time of the biggest low-priority frame on the analyzed resource.

Transmission Time of Previous Frames When calculating the length of the n-level busy period (defined in [1]), the transmission time of previous frames of the analyzed message t_O is given by $(n-1) * C_a^+$, where C_a^+ is the maximal transmission time of a frame of the analyzed message.

Same-Priority Interference In the worst case, the analyzed frame is slowed down by the maximal amount of same-priority traffic that can arrive before t_a, the arrival point of the analyzed frame. The same-priority interference is given by:

$$t_S(t_a) = \sum_{s \in S} \left(\eta_s^+(t_a) * C_s^+ \right) \tag{1}$$

where $\eta_j^+(t)$ is the so-called *event function* that provides the maximal number of frames of the message j that can arrive within the time interval t. This value can be derived, e.g., by applying the event model presented in [5] or the arrival curves used in RTC [8]. The sum of t_L, t_O, and t_S is denoted as the *fix interference* $t_f(n, t_a)$.

High-Priority Interference The high-priority message set H is a union of the set of the unscheduled high-priority messages H_u and the set of the scheduled high-priority messages H_s. All high-priority frames that arrive before t_d, the point in time when the analyzed frame is selected for transmission, contribute to t_H. As t_d itself depends on the amount of high-priority traffic transmitted before the analyzed frame, t_H is calculated with a fixed-point iteration:

$$t_H(t_d) = \min_{t_d \in T_d} \Big(\sum_{h \in H_u} \left(\eta_h^+(t_d) * C_h^+ \right) + \upsilon^+(t_d + C_a^+) \Big) \tag{2}$$

The first term in the sum yields the transmission time of the maximal amount of unscheduled high-priority frames that arrive before the analyzed frame is selected for transmission. $\upsilon^+(t_d + C_a^+)$ is the cumulative interference of all scheduled messages on the analyzed port before the transmission of the analyzed frame is completed. Hereby $\upsilon^+(t)$ is a function that provides the maximal SI that can be imposed during a time interval with the length t. We denote the time interval used as input for $\upsilon^+(t)$ as the *analyzed time interval* (ATI). In Sec. IV, we provide an efficient way to calculate $\upsilon^+(t)$. The time interval t_d is given by the minimal time needed to transmit both, the fix interference, arriving before the analyzed frame, and the high-priority interference:

$$T_d = \{ t \in \mathbb{R}_0^+ \mid \left(t_f(n, t_a) + t_H(t) \right) \leq t \} \tag{3}$$

IV. Calculation of the SI

In this section, we introduce an efficient way to calculate the SI for a given ATI. We assume that the scheduled traffic on the analyzed resource is described by a set of so-called *interference slots* (ISs). Each IS is characterized by its start point s and its end point e. A port schedule is transformed into a set of ISs by by appending the guard band time interval at the start of each PS and a time interval for the preemption header at the end of each PS. Both intervals have a fix length. The ISs describe time intervals when the transmission of unscheduled frames is impossible, either because scheduled frames are being transmitted or as a result of the preemption and/or the guard band mechanisms. For details concerning these mechanisms, please refer to [3] and [4].

A. Exhaustive Analysis

Only an arrival at a start point of an IS can lead to the worst-case interference as in this case, the whole IS contributes to the interference before any unscheduled workload is transmitted. The worst-case SI for a given ATI can therefore be found by examining the SI resulting from an arrival at each of the start points of the ISs. The interference during an ATI with the length t resulting from the arrival at the point t_a is given by:

$$i(t_a, t) = \max_{b \in B(t)} \Big(\sum_{1 \leq i \leq b} (e_i - s_i) \Big) \tag{4}$$

with

$$B(t) = \{ b \in \mathbb{N}^+ \mid t_a \leq s_b \leq t_a + t \} \tag{5}$$

where s_1 and e_1 are the start and the end points of the first IS after the arrival point, s_2 and e_2 are the start and the end points of the second IS after the arrival point and so on. The worst-case SI is the biggest value that is found by applying Eq. (4) at the start point of each IS. The time needed to calculate the SI in this way grows quadratically with the number of ISs which can get very big in realistic networks. This is critical as during the timing analysis, the SI is calculated in the innermost loop of multiple fixed-point iterations. We overcome this problem by preprocessing the system-wide schedule.

B. Static Interference

The maximal SI that can occur during one complete HP with the length h, $I_s(h)$, is calculated before the start of the timing analysis by summing up the lengths of all ISs. The so-called *static interference* i_s for the ATI with the length t is then given by:

$$i_s(t) = \lfloor t/h \rfloor * I_s(h) \tag{6}$$

Where $\lfloor t/h \rfloor$ is the number of complete HPs contained within the ATI while $I_s(h)$ is the maximal SI per complete HP. During the run time of the timing analysis, $i_s(t)$ is calculated in a short constant time.

C. Dynamic Interference

Besides the static interference, the so-called *dynamic interference* i_d that is imposed during the remaining time $t_r(t) = t - \lfloor t/h \rfloor * h$ has to be calculated. Depending on the size of the HP, i_d may constitute a large part of, or even the entire SI. Moreover, i_d for a given ATI can only be determined during the timing analysis of the network, as for different ATIs, different arrival points of the unscheduled traffic may lead to the maximal interference. We now present three techniques for the calculation of i_d during the timing analysis, differing in computation time and pessimism.

Naive Approach. In the *naive approach* (NA), we conservatively assume that during the incomplete HP, the scheduled traffic imposes the same amount of interference as during a complete HP, i.e., $i_d(t_r) = I_s(h)$. As the NA is based on the precalculated value for $I_s(h)$, i_d is calculated in constant time at the cost of a significant pessimism.

Exact Approach. The *exact approach* (EA) calculates i_d with minimal pessimism by applying Eq. (4) to find the maximal dynamic interference during the time t_r. It has a complexity that is quadratic in the number of ISs, so that its computation time becomes impractical for bigger IS numbers.

Dominance-based Approach. The *dominance-based approach* (DBA) significantly reduces the computation time of the EA and still provides exact results by preprocessing the port schedule and transforming it into a data structure that **a)** only contains information regarding the worst-case interference and **b)** enables a quick derivation of the worst-case interference for arbitrary ATIs.

DBA: Interference Table. In the first step of the DBA, the set of the ISs is transformed into the so-called *interference table* (IT). The n ISs are numbered in ascending order, where the x-th IS r_x is characterized by its start $s(r_x)$ and its end $e(r_x)$. The IT has n rows and n columns. The entry a_{ij} in the row i and the column j represents the interference that is imposed on unscheduled traffic that arrives at the start of the i-th IS by the first j ISs that are encountered. The entry is characterized by the values $d(a_{ij})$ and $f(a_{ij})$. $f(a_{ij})$ is the overall interference by all ISs between the start of the i-th and the end of the j-th IS. $d(a_{ij})$ is the distance between the start point of the j-th IS and the start of the i-th IS. $d(a_{ij})$ therefore provides the minimal time after which an unscheduled workload arriving at s_i experiences the SI $f(a_{ij})$. The IT fully describes the interference that is imposed on arbitrary amounts of unscheduled traffic arriving at one of the n candidate points. In the next steps, all information that is irrelevant for the worst-case is eliminated from the IT.

DBA: Entry dominance. Only the so-called *dominant entries* are relevant for the calculation of the SI. Entry a is dominated by entry b if $d_b \leq d_a \wedge f_b \geq f_a$, as in this case, entry b describes a situation where a bigger or equal SI is imposed on the unscheduled workload during a shorter or equal ATI than in the situation described by the entry a. An entry is dominant if it is not dominated by any other entry in the IT.

DBA: Reduction. After the creation of the IT, all non-dominant entries are iteratively eliminated. Each iteration starts by finding the set of entries relevant for the shortest time interval (the entries with the smallest d-value). This set always contains one dominant entry. All entries dominated by this entry are eliminated from the IT, while the dominant entry is added to the so-called *interference list* (IL), ordered in ascending order according to the d-value of its entries.

$i\backslash j$	1		2		3	
1	I.	~~0 | 3~~	II.	4 | 5	V.	11 | 9
2	I.	~~0 | 2~~	III.	7 | 6	V.	~~16 | 9~~
3	I.	0 | 4	IV.	9 | 7	V.	~~13 | 9~~

TABLE I: Example for the DBA

DBA: Interference List. The IL contains only information that is relevant for the worst case. It is used to quickly derive i_d for the given time interval t_r with:

$$i_d(t_r) = f(p) + \max\big(0, t - d(p+1) + f(p+1) - f(p)\big) \quad (7)$$

Where p is the position of the IL-entry with the biggest d value that is smaller than t. The first term of the sum is the interference from the complete ISs that contribute to the entry at position p, while the second term provides the worst-case interference that is imposed by one incomplete IS.

During the timing analysis, the maximal dynamic interference is computed by **a)** finding the right list position p and **b)** applying Eq. (7). As the IL is ordered, step **a)** is performed using logarithmic search, while step **b)** is performed in constant time. The complexity of finding i_d is consequently $O(2 \log n)$ with the number of ISs as problem size. Thus, applying the DBA reduces the complexity of the calculation of the SI from quadratic to logarithmic.

Example We demonstrate the DBA using an IS set with three ISs and a HP of 20 *time units* (TUs). The ISs start at 3, 7, and 14 and have a length of 3, 2, and 4 TUs. The generation process of the corresponding IL is illustrated in Tab. I. During the first iteration (I.) of the reduction process, the entries relevant for the time interval of 0 TUs are examined. Hereby, the dominant entry $0 \mid 4$ is found, while the other two entries are eliminated. After the found entry has been added to the IL, the second iteration (II.) of the reduction process is started and the dominant entry $4 \mid 5$ is found. After the fifth iteration (V.), the reduction process is finished.

The dominant entries found through the reduction of the IT are used to generate the IL: $[(0 \mid 4), (4 \mid 5), (7 \mid 6), (9 \mid 7), (11 \mid 9)]$. This IL completely describes the worst-case SI that can be imposed during ATIs with a length ranging from 0 to 20 TUs. The exact worst-case SI for a specific ATI can be found by applying Eq. (7).

V. Case Study

In our first experiment, we compare the computation time and the pessimism of the DBA to the two other approaches presented in Section IV by analyzing the interference caused by schedules with a HP of 100 ms with different numbers of ISs distributed regularly over the HP. Each IS represents a scheduled message with a frame size of 1,500 Bytes in a 100-MBit-Ethernet network. The results of the first experiment are illustrated in Figs. 1 and 2. Figure 1 shows how the computation time of the calculation of the SI scales for 5 (left), 15 (middle), and 100 (right) ISs depending on the length of the ATI. The plots show the relation between the computation time of the EA (dashed) and the DBA compared to the NA, which always requires constant time. The computation times of the DBA and the EA show a similar behavior in that they both grow for a bigger number of ISs and have local minima at the points where the preprocessing-based calculation of the static interference described in Section IV-B reduces the

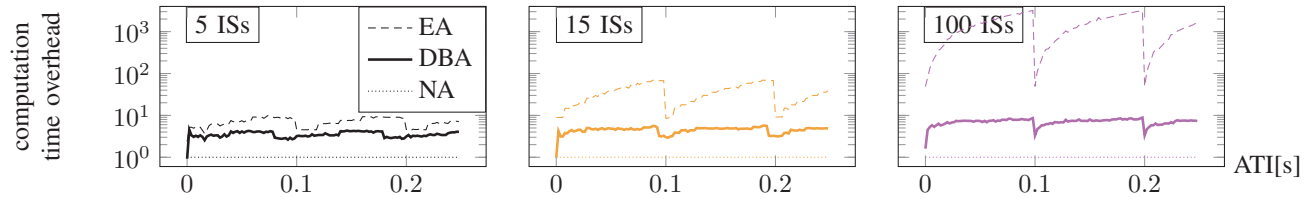

Fig. 1: **Computation times of the three SI approaches relatively to the NA (log-scale):** While the DBA never takes more than 10 times longer than the NA, the computation time of the EA is up to 3000 times longer.

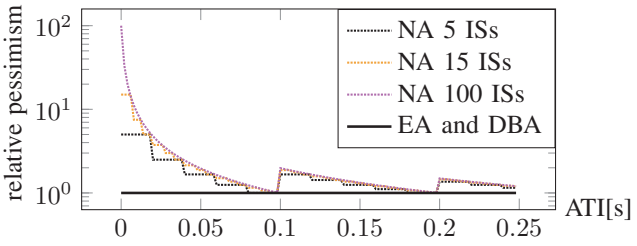

Fig. 2: **Pessimism of the three SI approaches relatively to the EA (log-scale):** While the NA is pessimistic by a factor of up to 100, the DBA always provides exact results.

computation time of the SI calculation (the points around 0.1 and 0.2). However, while the DBA, even for very big IS-numbers, takes at most around 10 times longer than the NA, the EA is up to 3000 times slower. Figure 2 illustrates the pessimism of the NA (dotted) and the DBA compared to the EA when analyzing the SI of 5 (black), 15 (orange/light gray) and 100 (violet/dark gray) ISs. While the DBA provides exact results, the NA is pessimistic by a factor of up to 100. The pessimism of the NA gets smaller when more ISs contribute to the worst-case interference and when the ATI contains full HPs. The results of this experiment give evidence that the DBA calculates the SI with the same precision as the EA. At the same time, the DBA is approximately 400 times faster than the EA for a port slot number of 100, which is a realistic value for complex networks with different application-specific message periods. The usage of the DBA, hence, provides exact results while offering a tremendous speedup for the timing analysis.

In the second experiment, we analyze the timing of an automotive communication network to examine the timing differences between a scheduled TSN network and a network with strict priority non-preemptive switched Ethernet (IEEE 802.1Q). The automotive network consists of 32 nodes exchanging 152 messages, with both single- and multi-cast messages. The first row of Tab. II contains the average jitter experienced by the messages of this network when using strict priority non-preemptive switched Ethernet (IEEE 802.1Q).

We use an evolutionary algorithm to find overlap-free schedules for the messages of the fourth traffic class. The timing of each overlap-free schedule that was found during the exploration was analyzed to obtain the values in Tab. II. While the analysis of the unscheduled network took 0.20 seconds, the analysis of a scheduled network took 0.23 seconds on average. As expected, the scheduled messages experience no jitter at all, while the jitter values of the other traffic classes substantially differ from those in the unscheduled network. The SI differs from the unscheduled high-priority interference in two aspects. On the one hand, the arrival points of the scheduled messages have constant offsets to each other so that no critical instant can occur and the number of messages interfering in the worst-case is smaller. Unscheduled messages with the highest priority (traffic class 3 in our case study) benefit from this circumstance. On the other hand, the GB-mechanisms in combination with the preemption of the unscheduled traffic lead to a bigger interference per interfering scheduled message. For low-priority unscheduled messages, which, in the worst-case, also experience the critical instants of the unscheduled messages with higher priority, this leads to jitter values bigger than those in unscheduled networks.

TABLE II: Average jitter per traffic class in seconds

Traffic class	1	2	3	4
IEEE802.1Q	3.0598	0.0049	0.0061	$1.2912 * 10^{-4}$
TSN min	3.0601	0.0048	0.0039	0
TSN average	3.0601	0.0048	0.0044	0
TSN max	3.0601	0.0048	0.0053	0

VI. Conclusions

In this paper, we present a formal analysis approach that provides a safe upper bound for the interference that scheduled messages in TSN networks impose on unscheduled traffic. We also propose preprocessing techniques that reduce the computation time of this approach by several orders of magnitude without introducing any additional pessimism.

Experimental results show that **a)** scheduled TSN networks can be used to provide interference-free delivery for critical messages and **b)** the resulting interference for the unscheduled messages massively depends on the shape of the used schedule and should be analyzed during the design of these networks.

References

[1] J. Diemer, J. Rox, et al. Modeling of Ethernet AVB networks for worst-case timing analysis. *MATHMOD*, 2012.

[2] F. Reimann, S. Graf, et al. Timing analysis of Ethernet AVB-based automotive E/E architectures. In *Proceedings of the Emerging Technologies & Factory Automation (ETFA)*, pp. 1–8. 2013.

[3] Time Sensitive Networking Task Group of IEEE 802.1. IEEE P802.1Qbu/D3.0 Draft Standard for Local and Metropolitan Area Networks Bridges and Bridged Networks Amendment: Frame Preemption. 2015.

[4] —. IEEE P802.1Qbv/D3.1 Draft Standard for Local and Metropolitan Area Networks Bridges and Bridged Networks Amendment: Enhancements for Scheduled Traffic. 2015.

[5] K. Richter. *Compositional Scheduling Analysis Using Standard Event Models*. Dissertation, University of Braunschweig, 2005.

[6] S. Thangamuthu, N. Concer, et al. Analysis of ethernet-switch traffic shapers for in-vehicle networking applications. In *Proceedings of the Design, Automation & Test in Europe (DATE)*, pp. 55–60. 2015.

[7] D. Thiele, R. Ernst, et al. Formal worst-case timing analysis of ethernet TSN's time-aware and peristaltic shapers. In *Proceedings of the Vehicular Networking Conference (VNC)*, pp. 251–258. 2015.

[8] L. Thiele, S. Chakraborty, et al. Real-time calculus for scheduling hard real-time systems. In *Proceedings of the International Symposium on Circuits and Systems (ISCAS)*, pp. 101–104. 2000.