Doctoral Thesis

Improved Configuration and Analysis Solutions for Time Sensitive Networks with Support for Legacy Systems

Daniel Bujosa Mateu
School of Innovation, Design and Engineering (IDT)
Mälardalen University

Main Supervisor: **Mohammad Ashjaei**
Co-supervisors: **Alessandro V. Papadopoulos**
Thomas Nolte
Julián Proenza

Abstract

The new generation networks and systems are increasingly demanded to present higher capacities such as flexibility, adaptability, or real-time capabilities. In this context, Time-Sensitive Networking (TSN) standards provide Ethernet with functionalities capable of meeting these new demands. However, the adoption of TSN by the industry has been limited due to the high complexity of configuring and analyzing its various mechanisms. This thesis aims to address this issue by developing tools to facilitate the adoption of TSN in the industry. To this end, we have created tools to enable efficient and semi-automatic TSN configuration and analysis. These tools can efficiently configure TSN networks based on traffic characteristics while supporting legacy devices in case of migration or integration of legacy systems to TSN. Additionally, the analysis tools enable guaranteeing the compliance of the TSN configuration with the network requirements with reduced waste of resources.

Specifically, we have devised a mapping tool called LETRA to distribute the Ethernet-based traffic among the three basic types of TSN traffic, i.e. Scheduled Traffic (ST), Audio-Video Bridging Traffic (AVB), and Best-Effort Traffic (BE). Since ST requires traffic scheduling, we developed HERMES, a heuristic scheduler that generates efficient schedules in a fast manner. For AVB traffic, which requires some previous schedulability analysis to guarantee it meets its time requirements, we propose a new Worst-Case Response Time Analysis (WCRTA) which reduces the pessimism of previous approaches. Finally, regarding the support for the integration of legacy devices in TSN systems, we have developed mechanisms to avoid adverse consequences resulting from the lack of synchronization between legacy devices and TSN.

Sammanfattning

Den nya generationens nätverk och system ställs inför allt högre krav på kapacitet, såsom flexibilitet, anpassningsförmåga och realtidskapabiliteter. I detta sammanhang erbjuder standarderna för Time-Sensitive Networking (TSN) Ethernet-funktioner som kan möta dessa nya behov. Trots detta har industrins adoption av TSN varit begränsad på grund av den höga komplexiteten i att konfigurera och analysera dess olika mekanismer. Denna avhandling syftar till att hantera detta problem genom att utveckla verktyg för att underlätta industrins användning av TSN. För detta ändamål har vi skapat verktyg som möjliggör effektiv och semi-automatisk konfiguration och analys av TSN. Dessa verktyg kan effektivt konfigurera TSN-nätverk baserat på trafikens egenskaper och samtidigt stödja äldre enheter vid migrering eller integration av äldre system till TSN. Dessutom möjliggör analysverktygen att säkerställa att TSN-konfigurationen uppfyller nätverkskraven med minskad resursförbrukning.

Vi har utvecklat ett verktyg, LETRA, för att fördela Ethernet-baserad trafik mellan de tre grundläggande typerna av TSN-trafik: Scheduled Traffic (ST), Audio-Video Bridging Traffic (AVB) och Best-Effort Traffic (BE). Eftersom ST kräver schemaläggning av trafik, har vi tagit fram HERMES, en heuristisk schemaläggare som genererar effektiva scheman på kort tid. För AVB-trafik, som kräver en förhandsanalys av schemaläggbarhet för att säkerställa att dess tidskrav uppfylls, föreslår vi en ny Worst-Case Response Time Analysis (WCRTA) som minskar pessimismen jämfört med tidigare metoder. Slutligen, när det gäller stöd för integration av äldre enheter i TSN-system, har vi utvecklat mekanismer för att undvika negativa konsekvenser som kan uppstå på grund av bristande synkronisering mellan äldre enheter och TSN.

Acknowledgment

To be written

Daniel Bujosa, Västerås, January 2025

List of Publications

Papers included in this thesis[1]

Paper A: Daniel Bujosa, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte. *LETRA: Mapping Legacy Ethernet-Based Traffic into TSN Traffic Classes.* In the proceedings of the 26^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), 2021.

Paper B: Daniel Bujosa, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte. *HERMES: Heuristic Multi-queue Scheduler for TSN Time-Triggered Traffic with Zero Reception Jitter Capabilities.* In the proceedings of the 30^{th} International Conference on Real-Time Networks and Systems (RTNS), 2022.

Paper C: Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei. *An Improved Worst-Case Response Time Analysis for AVB Traffic in Time-Sensitive Networks.* In the proceedings of the 45^{th} IEEE Real-Time Systems Symposium (RTSS), 2024.

Paper D: Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei. *Reducing Pessimism in Response Time Analysis of AVB Traffic in TSN.* MRTC Report, MDU 2024.

[1]The included papers have been reformatted to comply with the thesis layout.

Paper E: Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei. *TALESS: TSN with Legacy End-Stations Synchronization.* In the 5^{th} Open Journal of the Industrial Electronics Society (OJIES) volume, 2024.

Related publications, not included in this thesis

Daniel Bujosa, Ines Álvarez, Julián Proenza. *CSRP: An Enhanced Protocol for Consistent Reservation of Resources in AVB/TSN.* In the IEEE Transactions on Industrial Informatics (TII) 2020.

Daniel Bujosa, Daniel Hallmans, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte. *Clock Synchronization in Integrated TSN-Ethercat Networks.* In the proceedings of the 25^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) 2020.

Ines Alvarez, Luis Moutinho, Paulo Pedreiras, Daniel Bujosa, Julián Proenza, Luis Almeida. *Comparing admission control architectures for real-time Ethernet.* In the 8^{th} IEEE Access Volume, 2020.

Daniel Bujosa, Andreas Johansson, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte. *The Effects of Clock Synchronization in TSN Networks with Legacy End-Stations.* In the proceedings of the 27^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) 2022.

Ines Álvarez, Daniel Bujosa, Bjarne Johansson, Mohammad Ashjaei, Saad Mubeen. *Centralised Architecture for the Automatic Self-Configuration of Industrial Networks.* In the proceedings of the 28^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) 2023.

Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei. *Introducing Guard Frames to Ensure Schedulability of All TSN Traffic Classes.* In the proceedings of the 28^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) 2023.

Contents

I

Thesis

Chapter 1

Introduction

The rapid evolution of industrial networks and systems has led to an increasing demand for advanced capabilities such as flexibility, adaptability, and real-time performance. This demand is driven by the need to enhance operational efficiency, optimize resource utilization, and improve product capabilities. Emerging technologies frequently offer novel solutions that provide competitive advantages through cost reduction, product improvement, and sustainability. However, the adoption of these technologies often presents significant challenges related to their practicality and integrability within existing industrial frameworks.

One such emerging technology is Time-Sensitive Networking (TSN). TSN standards, developed under the auspices of the IEEE, extend Ethernet functionalities to meet stringent industrial requirements. These standards provide, among others, precise clock synchronization, low-latency and deterministic data transmission with zero jitter, support for varying types of real-time requirements within a single network, fault tolerance mechanisms, and dynamic network reconfiguration capabilities.

The journey of TSN began with the establishment of the IEEE Audio-Video Bridging (AVB) Task Group in 2005, which aimed to equip Ethernet with soft real-time capabilities tailored for audio and video streaming. As interest in AVB standards grew across various sectors, including automotive [1], automation [2], and energy distribution [3], the AVB Task Group evolved into the TSN

Task Group in 2012, broadening its scope to address diverse industrial needs. These initiatives resulted in the development of key standards such as IEEE Std 802.1AS [4] for clock synchronization, and IEEE Std 802.1Q [5] for the Time-Aware Shaper (TAS), the Credit-Based Shaper (CBS), and the Stream Reservation Protocol (SRP).

Despite the significant advantages offered by TSN, its adoption in the industry has been limited due to the complexity of configuring and analyzing its various mechanisms. Implementing TSN requires substantial investments in new devices, highly skilled personnel, and considerable time for installation, configuration, and deployment. Established companies face additional challenges in integrating legacy systems with TSN, further complicating the adoption process.

This thesis aims to address these adoption difficulties by developing tools to facilitate the efficient configuration and analysis of TSN . The proposed tools will enable effective TSN network configuration based on traffic characteristics while preserving legacy devices during migration or integration. Additionally, the analysis tools will ensure the compliance of the TSN configuration with the network requirements with a reduced waste of resources. The focus will be on mapping, scheduling, and analyzing the feasibility of unscheduled traffic, as well as assessing the impact of legacy systems on these processes. By providing practical solutions to the challenges of TSN adoption, this research seeks to pave the way for the broader application of TSN in industrial networks.

1.1 TSN Adoption Challenges and Contributions

A key feature of TSN is its ability to handle diverse traffic types, enabling the management of traffic with varying priorities and real-time characteristics on the same network. To this end, TSN defines three basic traffic types: Scheduled Traffic (ST), AVB traffic, and Best-Effort (BE) traffic. The first challenge in TSN configuration is efficiently classifying Ethernet-based traffic, whether legacy or new, into these traffic types to increase schedulability and compliance with timing requirements. To address this, we developed the Legacy Ethernet-based Traffic Mapping Tool (LETRA), a methodology and tool designed to classify Ethernet-based traffic based on its characteristics.

The second challenge is related to ST traffic. Indeed this type of traffic operates on a fixed cyclic schedule, specifying the exact transmission time for each frame. This requires a fast and scalable scheduler to accommodate dynamic and large-scale systems. In response, we created the Heuristic Multi-Queue Scheduler (HERMES), which provides high schedulability with very low scheduling times and ensures zero jitter for ST traffic, even in large systems.

The third challenge is related to AVB traffic. Unlike ST, where timing requirements are guaranteed by construction, AVB traffic is scheduled online and relies on schedulability analysis, such as Worst-Case Response Time Analysis (WCRTA), to verify its real-time compliance. Minimizing pessimism in WCRTA is essential to improving resource efficiency. In this work, we significantly reduce the pessimism compared to existing analyses, enhancing the practical usability of AVB traffic in TSN systems.

Finally, integrating legacy systems into TSN networks presents additional challenges, particularly synchronization issues, as many legacy devices are not compatible with TSN synchronization protocols. This work addresses these challenges by developing a synchronization mechanism called TSN with Legacy End-Stations Synchronization (TALESS) .

These contributions are integrated into a comprehensive toolchain capable of automatically and efficiently migrating Ethernet-based traffic to TSN, with support for legacy systems. The toolchain, illustrated in Fig. 1.1, facilitates the practical and cost-efficient adoption of TSN in both new and legacy environments.

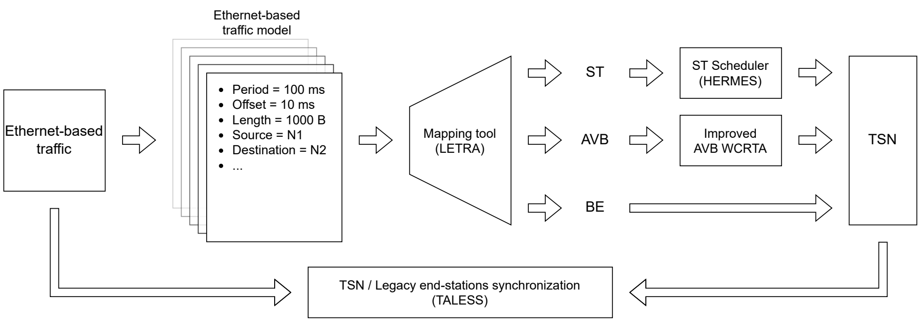


Fig. 1.1: Proposed toolchain solution.

1.2 Thesis outline

The outline of this thesis is as follows:

Sections 2 and 3 provide an overview of the main TSN mechanisms addressed in this work and the state of the art concerning various aspects of these mechanisms, respectively. Section 4 introduces the problem formulation, while Section 5 describes the employed research methodology. Finally, Sections 6 and 7 present the contributions of this thesis, along with the conclusions and potential directions for future research.

Chapter 2

Background

This section presents the mechanisms involved in TSN traffic shaping and synchronization. These mechanisms are primarily responsible for providing TSN with its significant flexibility and adaptability, which are highly relevant in industrial applications. They are crucial for the correct configuration and analysis of TSN traffic and are therefore essential for understanding this work.

2.1 TSN Traffic

TSN end-stations communicate by transmitting Ethernet frames through routes composed of links and TSN switches. The output ports of TSN end-stations and switches consist of up to eight First-In First-Out (FIFO) queues, each associated with a priority level. These queues can apply different traffic-shaping mechanisms. Depending on the priority and the mechanisms applied, the traffic associated with each priority is classified into one of the three available traffic classes: ST, AVB, and BE. ST has the highest priority among the traffic classes, while BE has the lowest priority. It is important to note that multiple queues can cover the same traffic class; for example, AVB can consist of classes A, B, and C, each associated with a different priority level. Fig. 2.1 shows an example of a TSN output port with four queues configured as follows: ST with the highest priority, AVB classes A and B with the second and third highest priorities, and BE traffic with the lowest priority. Next, we explain the operation and configuration of each TSN traffic type.

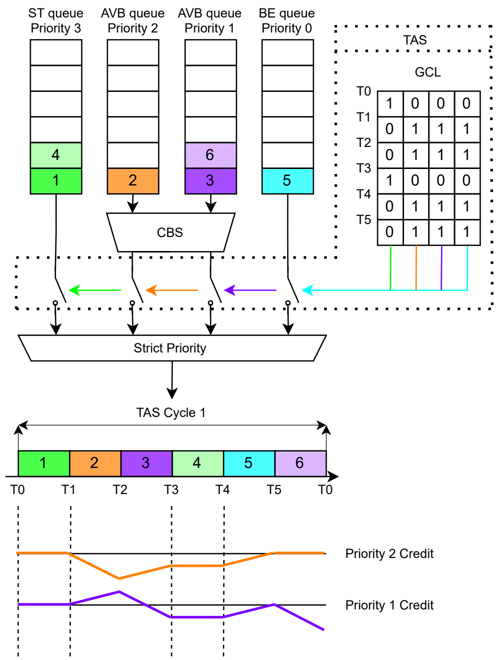


Fig. 2.1: A TSN output port with four FIFO queues: one ST queue, two AVB queues, and one BE queue.

2.1.1 Scheduled Traffic

ST frames are scheduled offline, allowing knowing the exact transmission time slot of each ST frame, thereby guaranteeing their timing requirements by construction. However, to ensure that ST frames are transmitted according to their schedule, it is necessary to prevent interference from other frames that could delay the transmission of ST frames. This is achieved through the Time-Aware Shaper (TAS) mechanism described in IEEE 802.1Qbv [6]. According to this mechanism, each queue has an associated gate that can be either open or closed (see Fig. 2.1). Frames in a queue can be transmitted when the gate is open; otherwise, the frames are blocked from transmission. The gates are controlled by the Gate Control List (GCL), which specifies when the gates should be open, and it is a cyclic list that repeats the schedule. The time that gates are open or closed can be specified at the nanosecond level for each entry of the GCL, and we refer to the interval during which a gate is open as a *transmission window*.

Fig. 2.1 shows an example of TAS operation. In this example, two ST frames (frames 1 and 4) with a period of 6 time units are transmitted through a switch port. In this figure, the vertical dashed lines indicate the ST window. According to this example, from time T0 to T1, the ST gate is open (shown as 1 in the GCL), while the lower-priority gates are closed (shown as 0 in the GCL), allowing the transmission of frame 1. Then, in the time interval between T1 and T3, the ST gate is closed, allowing the transmission of lower-priority traffic. Finally, in the time interval between T3 and T4, the ST gate opens again, allowing the transmission of frame 4 without interference from other frames. After this, the ST gate closes again, and the lower-priority queues reopen, allowing the transmission of the remaining lower-priority traffic. Only one cycle of frame transmissions is represented at the bottom of Fig. 2.1. Other cycles may show differences in the transmission of lower-priority traffic but would maintain the transmission of frames 1 and 4 in the intervals T0-T1 and T3-T4, respectively.

2.1.2 Audio-Video Bridging Traffic

Unlike ST, AVB traffic is scheduled online. This provides greater flexibility, allowing for more efficient transmission of both periodic and aperiodic traffic at the cost of increased jitter, while still maintaining Real-Time (RT) capabilities due to its deterministic nature. To achieve this, the AVB Task Group introduced the Credit-Based Shaper (CBS) [7], which defines credits for AVB queues. The credit is consumed when a frame in that AVB queue is sent and replenished when frames are waiting to be transmitted or the credit is negative. AVB queues can only transmit when their credit is positive or zero and their gate is open according to the GCL. As a result, CBS limits the maximum bandwidth that an AVB queue can use and allows the transmission of lower-priority traffic even if higher-priority traffic is waiting to be transmitted when its credit is negative. This reduces buffering and improves the Quality of Service (QoS) for lower-priority traffic. Although the activation time of AVB traffic is unknown due to possible interference from other AVB classes or ST frames, its Worst-Case Response Time (WCRT) can be calculated through Worst-Case Response Time Analysis (WCRTA) such as in[8].

Fig. 2.1 shows an example of CBS operation for two AVB queues (Classes A and B) interacting with two higher-priority ST frames and one lower-priority BE queue. In this figure, in addition to the vertical dashed lines indicating the ST windows, we now also consider the slopes indicating the evolution of the credit. As shown in the figure, during the interval T0-T1 corresponding to the transmission of ST frame 1, the credit of both queues remains frozen at zero despite frames waiting to be transmitted. After this, since the credit of both AVB queues is zero, the AVB class A queue begins to transmit frame 2 since it has higher priority. While the frame is transmitted the AVB Class A credit is consumed and the credit of class B increases. After transmitting the frame, at T2, the credit of the AVB class A queue is negative; therefore, class B begins to transmit. After the transmission of the AVB class B frame, at T3, the transmission window for ST frame 4 begins, closing all other gates to ensure its uninterrupted transmission. After transmitting the ST frame at T3, the credit of the AVB traffic gates remains negative; therefore, the BE frame can be transmitted even though higher-priority frames are waiting to be transmitted. Finally, at T5, the credit of the AVB class B queue returns to zero, allowing frame 6 to be transmitted.

2.1.3 Best-Effort traffic

BE traffic does not have real-time guarantees, although it has guaranteed the bandwidth that is not reserved by AVB queues. A queue carrying BE traffic has the lowest priority and does not incorporate CBS. Therefore, it can only transmit frames if its gate is open and all other AVB queues have negative credit or there is no AVB traffic ready for transmission, as shown in Fig. 2.1. In the case of multiple BE queues, in addition to the aforementioned conditions, lower-priority BE queues can only transmit if no frames are waiting to be transmitted in higher-priority BE queues.

2.2 TSN Clock Synchronization

The Generalized Precision Time Protocol (gPTP) is the protocol that provides clock synchronization for TSN. It is described in the IEEE 802.1AS standard [4] and consists of three main components: the Best Master Clock Algorithm (BMCA), the Propagation Delay Measurement (PDM) mechanism, and the Transport of Time-synchronization Information (TTI). The BMCA is used to determine the grandmaster clock, which serves as the reference clock in the TSN network, as well as to establish the hierarchy among the different TSN devices. The PDM mechanism is employed once the hierarchy is established to measure the propagation delay between systems. Finally, the TTI mechanism is used to distribute the grandmaster clock time to synchronize the other TSN devices. Together, these three mechanisms can provide clock time precision among TSN devices down to tens of nanoseconds. Each of these mechanisms is detailed below.

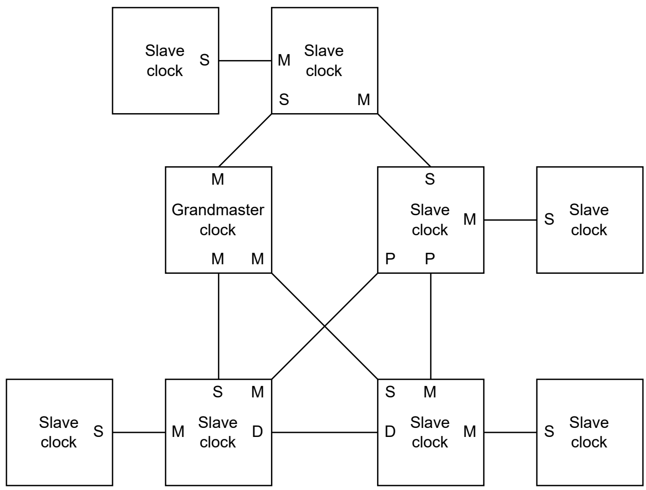


Fig. 2.2: Example of TSN time-synchronization spanning tree.

2.2.1 Best Master Clock Algorithm

The BMCA constructs a time-synchronization spanning tree with the grandmaster as the root. An example of this can be seen in Fig. 2.2. As shown in the figure, each TSN device can be either a grandmaster or a slave, while ports can be Master (M), Slave (S), Passive (P), or Disabled (D). To determine these roles, each system periodically sends a special broadcast message called an `announce message`. The `announce message` contains various parameters, but for simplicity, we focus on two: `systemIdentity` and `stepsRemoved`. The `systemIdentity` specifies the quality of the clock of the TSN device sending the message, while `stepsRemoved` indicates the distance from the transmitter to the receiver. Specifically, `stepsRemoved` is incremented each time the `announce message` is forwarded. This means that in a linear topology with three TSN devices, when the first device in the line transmits its `announce message`, the message reaches the second device with a `stepsRemoved` value of zero. However, this device will increment `stepsRemoved` before forwarding it to the next TSN device. Thus, the last device will receive the `announce message` sent by the first device with a `stepsRemoved` value of one.

As shown in Fig. 2.3, when a TSN device does not have an assigned role, it can become either a slave or the grandmaster clock. If the system receives an `announce message` from a better clock, i.e., a message with a better `systemIdentity` parameter, it becomes a slave. Conversely, if the time-aware system does not receive any `announce message` from a better clock after a defined period (determined by the periodicity of the `announce message`), it becomes the grandmaster clock. If a TSN device is already a grandmaster or a slave, a similar process occurs. If the grandmaster receives an `announce message` from a better clock, it becomes a slave. Finally, if a slave does not receive an `announce message` from a better clock after a defined period, it becomes the grandmaster. Additionally, depending on the proximity to the grandmaster, TSN ports have different roles, which can be determined by the `stepsRemoved` parameter. Specifically, the port closest to the grandmaster clock becomes the slave port, and only one port in the system can have this role. On a link, the port closest to the grandmaster clock becomes a master port. Finally, if a port is disabled, it becomes a disabled port, and if it is none of the master, slave, or disabled ports, it becomes a passive port.

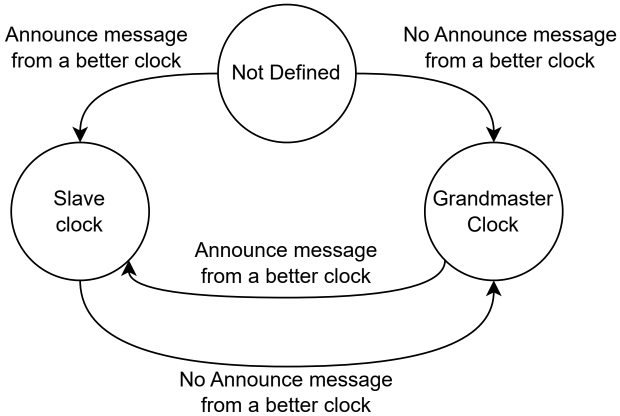


Fig. 2.3: Time-aware system BMCA evolution.

2.2.2 Propagation Delay Measurement

Once the synchronization tree with the grandmaster as the root is created, the slaves can perform PDM to measure the propagation delay. This process is illustrated in Fig. 2.4. PDM begins with a slave device sending a `Pdelay_request` message through its slave port to another device, which can be the grandmaster or another slave, and recording the time when the message was transmitted ($T1$). The responder receives the message through its master port, records the time when the message was received ($T2$), sends $T2$ back to the initiator, and records the time when the message was transmitted ($T3$). The initiator receives $T2$ and records the time when the message was received ($T4$). Finally, the responder sends $T3$ to the initiator, allowing it to calculate the delay using $Delay = \frac{(T4-T1)-(T3-T2)}{2}$.

2.2.3 Transport of Time-synchronization Information

Once the spanning tree with the grandmaster as the root has been built and the delays between slave ports and master ports have been measured, TTI is

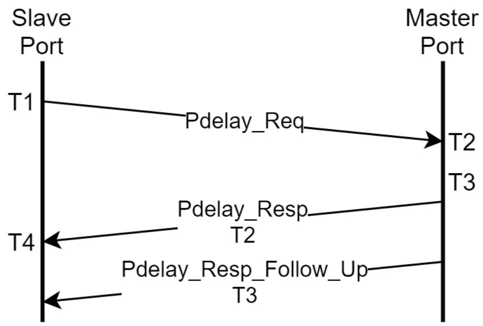


Fig. 2.4: PDM diagram.

executed. TTI consists of the transmission of the local time of the TSN devices through their corresponding master ports to the slave devices connected to them. Once the TTI message is received through the slave port, the slave device updates its local time by adding the delay measured through PDM to the time received.

Chapter 3

State of the Art Review

The significant contributions of the TSN TG since 2012 have spurred extensive research within the community, focusing on the study, application, and enhancement of TSN. For instance, Ashjaei et al. in [9] propose a comprehensive and up-to-date survey of TSN-related research. The effects of the time-aware shaper were examined in [10]. Regarding fault tolerance, Kehrer et al. [11] addressed fault tolerance issues and Álvarez et al. [12] proposed time redundancy to handle temporary faults. Scheduling policies were explored by Umadevi et al. in [13], while load balancing was investigated in [14].

In the context of dynamic system networks, several studies have explored reconfiguration and integration. For example, the works by Gutierrez et al. in [15] and [16] introduced a Configuration Agent for TSN, which continuously monitors the network for changes and automatically updates the configuration to adapt while maintaining the desired quality of service. This element significantly enhances TSN's adaptability by providing online auto-configuration. On the other hand, Said et al. in [17] analyzed the ability of SDN to accelerate the Time-to-Integrate process in evolving topologies from a synchronization perspective.

Regarding the integration of legacy networks into TSN, Nsaibi et al. in [18] integrated several TSN standards into Sercos III, a closed system that allows standard Ethernet devices to be connected, enhancing its performance. Similarly, Seijo et al. in [19] proposed a methodology for integrating wireless TSN

(802.11). Additionally, Szancer et al. in [20] suggested a migration method for SERCOS III to TSN, adopting TSN's gPTP due to the incompatibility of synchronization mechanisms between the two protocols. For TSN and PROFINET integration, Schriegel et al. in [21] proposed a new type of switch to map PROFINET traffic onto TSN, though it does not prevent clock drift between TSN and PROFINET end-stations, which could have adverse effects. Lastly, Barzegaran et al. in [22] and Zhao et al. in [23] introduced TSN schedulers for unscheduled and unsynchronized legacy traffic with high jitter, but they do not guarantee zero-jitter reception and overlook clock drift. Both studies assume a constant period for legacy frames, which may not hold true due to drift from the TSN perspective. As demonstrated by Chouksey et al. in [24], there are still numerous challenges in achieving proper integration between TSN and non-TSN devices.

This work focuses on the configuration and analysis of TSN, as well as the migration and integration of legacy systems to facilitate its adoption by the industry. Specifically, we will study the mapping of traffic into the three types of TSN traffic: ST, AVB, and BE. Additionally, traffic classified as ST must be scheduled to ensure its timing requirements are guaranteed by construction; therefore, we will also study ST schedulers. Moreover, AVB traffic needs to be analyzed to guarantee it meets its timing requirements, therefore we will also examine the AVB WCRTA. Finally, a critical aspect of migrating and integrating legacy systems is ensuring synchronization between different devices in the network. Consequently, we will also investigate the synchronization of TSN in heterogeneous networks, which combine both legacy and TSN devices.

3.1 TSN Mapping

In the context of traffic mapping, Gavriluţ et al. in [25] proposed a metaheuristic method to map mixed-criticality applications into TSN traffic classes. While this method shares similar objectives with our work, it does not encompass all scenarios found in industrial applications. Specifically, it is more suitable for situations with limited mixed-criticality levels in legacy systems with limited timing information.

Other studies, such as the one by Docquier et al. in [26], focus on mapping specific types of traffic, whereas the proposal by the Industrial Internet Consortium in [27] examines the characteristics of various TSN traffic types and offers guidance on mapping procedures. Despite these contributions, to the best of our knowledge, there are no automated tools available that map Ethernet-based traffic into TSN based on its characteristics.

3.2 ST Scheduling

In the field of ST scheduling within TSN networks, various approaches have been proposed, many of which are compiled, analyzed, and experimentally compared in a recent survey by Xue et al. in [28]. For instance, Craciunas et al. in [29] presented an SMT-based scheduler capable of managing networks with multiple ST queues. The work by Gavriluţ et al. in [30] proposed a GCL synthesis approach using the Greedy Randomized Adaptive Search Procedure (GRASP) meta-heuristic [31], which considers AVB traffic. Additionally, Zhou et al. in [32] present an ST scheduling approach based on Satisfiability Modulo Theories (SMT) for TSN networks that incorporates frame preemption to enhance schedulability.

Other works combine routing and scheduling. For instance, Gavriluţ et al. in [33] also suggested a joint routing and scheduling strategy for both ST and AVB traffic, employing an integrated heuristic and meta-heuristic approach. This strategy utilizes the K-Shortest Path (KSP) method [34] for routing and GRASP for scheduling both ST and AVB traffic simultaneously. Hellmanns et al. in [35] and Schweissguth et al. in [36] present an optimization framework for joint routing and scheduling in TSN networks using Integer Linear Programming (ILP), achieving efficient and feasible configurations for complex TSN setups. Moreover, Pahlevan et al. in [37] introduced joint routing and scheduling algorithms, formalized as a meta-heuristic scheduling approach based on a Genetic Algorithm (GA). Furthermore, Atallah et al. in [38] proposed a greedy heuristic algorithm for joint topology, routing, and scheduling synthesis to support seamless redundant transmission for ST.

Most of these solutions rely on ILP, CP or SMT solvers, with some leveraging meta-heuristics like GA. However, these methods often exhibit high time complexity, limiting their scalability. To address this, Pahlevan et al. in [39] proposed a heuristic routing and scheduling algorithm called Heuristic List Scheduler (HLS), which is restricted to a single ST queue. Additionally, Syed et al. in [40] compared four heuristic algorithms that combine routing and scheduling: Modified Most Loaded Heuristic (MML), Bottleneck Heuristic (BN), Coefficient of Variation Heuristic (CV) [41][42], and Modified Dot Product Heuristic (MDP) [43]. Finally, Zhang et al. in [44] propose a scalable and flow-aware scheduling approach for TSN networks, leveraging divisibility theory to partition the no-wait scheduling problem into smaller subproblems, which are solved using heuristics. Their method achieves efficient scheduling while ensuring scalability, supporting multiple traffic flows and queues, and maintaining compatibility with practical TSN constraints.

A recent survey by Xue et al. in [45] identified various features and characteristics of existing ST schedulers. Despite the extensive research on TSN scheduling, many solutions are not inherently compatible with the specific characteristics of legacy traffic, such as offsets, drifts, or specific reception jitters. Moreover, these solutions are often either highly time-complex or very limited in terms of schedulability.

3.3 AVB Worst-Case Response Time Analysis

Since the AVB standard was introduced in 2011, numerous studies have tackled the challenge of analyzing its WCRT. These analyses can be broadly categorized into three approaches: busy period analysis, Network Calculus, and eligible interval analysis. These analyses are summarized in a recent survey [46], where Wang et al. conducted a systematic review to classify existing schedulability analysis methods, evaluate their strengths and weaknesses, and highlight the challenges and future research directions that remain in TSN schedulability analysis.

The busy period analysis focuses on identifying the critical instant that produces the WCRT. Diemer et al. in [47] introduced one of the first AVB WCRTA based on this method, though it was limited to a single output port and a single AVB queue, excluding ST interference. This work was later extended in [48] to include two AVB queues (Class A and Class B), but still did not consider ST interference and remained confined to one output port. Finally, Lo Bello et al.

in [49] further advanced this approach by incorporating interference from other AVB classes and ST traffic, both with and without preemption of non-ST traffic, and included multi-hop calculations. However, this analysis was still limited to two AVB classes and exhibited optimism in ST interference calculations.

The second approach utilizes Network Calculus, a theoretical framework for performance analysis in communication networks, to calculate the maximum delay each frame may experience. Zhao et al. in [50] and [51] implemented an AVB WCRTA using Network Calculus, enabling WCRT calculations from transmitter to receiver for multiple AVB queues, considering ST interference with and without preemption. However, Network Calculus faces limitations with loop networks involving circular dependencies and allocates the same bandwidth for an AVB class on all links, reducing configuration flexibility. This method, along with the one proposed in [49], uniquely provides WCRT from transmitter to receiver, including ST interference and preemption. Other Network Calculus-based WCRTAs, such as the one proposed by Seliem et al. in [52], are designed for specific setups and impose significant constraints, such as preventing simultaneous opening of AVB Class A and B gates, limiting CBS arbitration capabilities.

The third approach defines the eligible interval, the time frame during which each frame can be transmitted. Bordoloi et al. in [53] introduced an early version of this WCRTA. Later, this work was extended by Cao et al. in [54, 55, 56] coining the term eligible interval and demonstrating that the WCRT depends on the maximum credit attainable by the AVB class, which is bounded and calculable. This method results in less pessimistic WCRTs compared to busy period analysis or Network Calculus and can be applied to any number of AVB classes. However, it does not account for ST interference and is limited to a single output port. Additionally, Maxim et al. in [57] extended this analysis to include ST interference, though without preemption, and it presents optimism in the ST interference calculations.

As reviewed, existing analysis approaches either have limitations or exhibit optimism. This work aims to address these issues by proposing a new AVB WCRTA for multi-hop networks, with significantly less pessimism compared to current state-of-the-art solutions.

3.4 Synchronization

Clock synchronization is a critical aspect of TSN technology. However, to our knowledge, no existing work has addressed the adverse effects caused by the lack of synchronization in heterogeneous TSN networks that integrate one or more Ethernet-based legacy systems through a TSN communications subsystem. Most studies have focused on integrating TSN with wireless and 5G networks. For instance, Haxhibeqiri et al. in [58] implemented a low-overhead beacon-based time synchronization method to achieve precise synchronization in wireless networks within highly deterministic TSN environments. Additionally, Baniabdelghany et al. in [59] and Romanov et al. in [60] explored extending IEEE 802.1AS and IEEE 802.11 to facilitate TSN integration with wireless networks.

The challenges of integrating wired TSN and WLAN technologies, along with a potential solution in a hybrid TSN device architecture, were discussed by Seijo et al. in [61]. Furthermore, Gundall et al. in [62] presented TSN clock synchronization aligned with 5G specifications, while Chai et al. in [63] proposed a data packet relay method to address cross-domain clock synchronization issues in 5G-TSN networks. The performance of 5G-TSN networks in terms of clock synchronization was also evaluated in several studies, including [64], [65], and [66].

Conversely, limited research has explored synchronization in heterogeneous TSN networks, which incorporate both TSN and non-TSN devices. For example, Xue et al. in [67] presented a method for maintaining synchronization across TSN sub-networks connected through non-TSN switches. This approach involves estimating the delays experienced by synchronization messages passing through these devices and configuring TSN networks to minimize these delays. In contrast, our work focuses on integrating legacy systems into a single TSN network. Notably, there are even fewer studies addressing synchronization between legacy end-stations and TSN. For instance, in [68] we proposed a methodology for integrating EtherCAT and TSN in terms of clock synchronization, and the work in [20], as introduced before, proposed a method for SERCOS III consisting in adopting gPTP to enable its synchronization with TSN. However, this type of integration requires customized solutions for each protocol, posing challenges for broader TSN adoption due to the significant time and resources needed for design and implementation, as well as compatibility issues.

Our approach differs from these solutions. In our proposed TSN heterogeneous networks, legacy systems are not synchronized with TSN, as they operate on distinct synchronization protocols. Instead, TSN adjusts its schedule to the clock timing of legacy systems to mitigate the negative impacts of the lack of synchronization.

Chapter 4

Problem Formulation

In recent years, there has been a growing interest in TSN within the industry. This interest is driven by TSN's features, which include high bandwidth combined with real-time capabilities, traffic flexibility, fault tolerance mechanisms, and more. These characteristics are crucial for the industry, both for the integration and interoperability of different operational levels and for the development of increasingly large and complex products and solutions. However, the costs associated with acquiring TSN devices, as well as the time and resources required for their installation and configuration, along with the expertise needed to perform these tasks, can impact the cost-effectiveness of the investment. This is particularly relevant for established and operational factories, as adopting TSN would require replacing all existing devices, networks, and solutions with a new system implementing TSN. Therefore, in this work, we analyze the shortcomings of TSN in the configuration process, including traffic mapping, scheduling, and analysis. Additionally, we examine the limitations of TSN in managing the migration and integration of legacy systems and propose solutions.

Given a TSN network with specific communication requirements or a legacy system with its legacy end-stations, communication protocols, and traffic with different timing requirements, TSN must be configured as easily and efficiently as possible to minimize resource waste and reduce costs. In the case of TSN adoption by legacy networks, this also includes the ability to change the communication subsystem to a single TSN network without affecting the correctness of the service provided by the system, allowing the system to continue operating better or the same as before by leveraging the benefits provided by TSN.

To facilitate the industry's adoption of TSN solutions, a proper TSN network traffic configuration methodology must be designed. This poses several challenges, which are addressed in this work. Firstly, the traffic must be classified into the three previously mentioned TSN traffic classes to ensure the system is schedulable and meets its timing requirements. Secondly, traffic classified as ST must be scheduled according to its requirements. Traffic scheduling must be scalable to allow for the configuration of large systems and must be fast to support new or legacy reconfiguration mechanisms. Thirdly, the WCRT of traffic classified as AVB must be analyzed to ensure it meets its timing requirements. Finally, one of the main challenges of heterogeneous TSN networks is the lack of synchronization. In the case of legacy network migration, the ST schedule must align with the transmission of legacy frames even when synchronization between legacy end-stations and the TSN network is not possible.

Next, we take a closer look at the limitations identified through the literature review.

4.1 Ethernet-based Traffic Mapping into TSN

The mapping process involves grouping network traffic based on its requirements and characteristics into the three types of TSN traffic: ST, AVB, and BE. One of the main features of TSN, and the reason behind the industry's interest in adopting it, is its traffic flexibility, i.e., its ability to combine various types of traffic with different timing requirements on the same network. This capability allows TSN to integrate different systems with varying characteristics into a single network, which would typically be managed by different communication protocols, thereby limiting connectivity and increasing system complexity. Different types of traffic exhibit characteristics that are not directly linked to the various TSN traffic types; hence, a proper mapping methodology is crucial for the correct configuration of TSN. To achieve this, we must identify the traffic characteristics that are relevant to defining its behavior in the TSN network and, based on these characteristics, distribute the traffic among the different TSN traffic classes. This must be done in the most efficient, fast, and automated way possible. On the one hand, efficient mapping reduces the waste of network resources resulting from incompatibilities between traffic characteristics and the assigned TSN traffic class. On the other hand, modern industrial networks can consist of hundreds or thousands of frames with different characteristics, making fast and automated traffic classification indispensable.

4.2 ST Scheduling

As previously mentioned, in TSN, a GCL is defined for each queue in each output port, specifying when the gate of each queue will be open. Scheduling ST and synthesizing it into GCLs has been proven to be an NP-complete problem [69]. Various solutions have been proposed in the literature to schedule ST in TSN networks, primarily based on Integer Linear Programming (ILP) and Constraint Programming (CP) [9]. Most of these schedulers aim to optimize one or more parameters, such as latency or jitter. However, these solutions are known to have high time complexity, meaning they require a significant amount of time to schedule large networks, making them generally not scalable. Additionally, these solutions are not suitable for systems that require dynamic and real-time reconfigurations, as the new configuration must be created within a bounded time. Some heuristic schedulers have also been proposed, such as [39], but their performance has not been adequately compared with ILP and CP solutions. In this context, this work aims to develop a heuristic scheduler capable of synthesizing the GCLs for TSN traffic mapped as ST with zero jitter, as this is its main advantage, while achieving acceptable performance and reduced scheduling times, thus enabling the configuration of large and dynamic networks.

4.3 AVB Worst-Case Response Time Analysis

The transmission of AVB traffic operates without a fixed schedule, providing flexibility unbounded by the constraints of ST. This feature is particularly advantageous for event-triggered traffic scenarios. Additionally, reducing the allocation of frames to ST increases the likelihood of achieving a feasible schedule. However, to meet the real-time demands of AVB traffic, a WCRTA is required. WCRTAs can be optimistic, exact, or pessimistic. An optimistic analysis is unacceptable, as it may fail to meet timing requirements, potentially leading to catastrophic consequences in critical systems. Conversely, a pessimistic WCRTA, while safe, results in wasted resources. An exact WCRTA is preferred for the analysis of any real-time systems; however, it is challenging to achieve and often impossible due to the consideration of worst-case scenarios. To the best of our knowledge, there is currently no exact WCRTA for TSN's AVB traffic. Therefore, providing a WCRTA with minimal pessimism is essential when an exact analysis is unavailable.

4.4 Legacy Systems Synchronization

ST requires the network to be fully synchronized. Otherwise, different devices might experience clock drift, causing the transmission and reception of frames to misalign with the scheduled windows. For instance, Fig. 4.1 illustrates the scenario where two unsynchronized end-stations exchange ST frames over a TSN network. As shown, the transmitter sends frames at a slower rate than the TSN network retransmits them, resulting in a negative drift at the receiver. Additionally, depending on the drift, frames periodically miss the transmission window, leading to intervals without frames at the receiver.

Despite the critical importance of synchronization, many legacy systems may not be capable of implementing TSN's synchronization protocols. Therefore, it is crucial to analyze the effects of the lack of synchronization in heterogeneous networks, i.e., networks combining TSN and legacy devices, and to develop mechanisms to mitigate potential adverse effects.

4.5 Research Challenges

Building on the identified issues, this thesis outlines four key research challenges that must be addressed to facilitate the seamless, efficient, and rapid

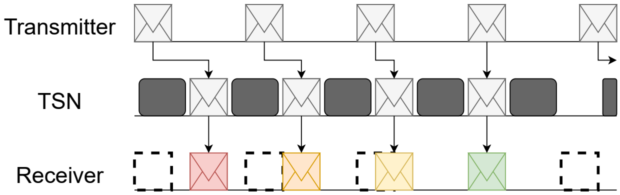


Fig. 4.1: Example of negative drift synchronization issue.

migration of Ethernet-based legacy traffic to TSN networks. These challenges are key to overcoming the technical and practical barriers associated with configuring and analyzing TSN for both new and legacy systems.

1. **Efficient and automated traffic mapping:** Classifying diverse Ethernet-based traffic, including legacy traffic, into the three TSN traffic classes (ST, AVB, and BE) presents a significant challenge. The mapping must maximize schedulability while ensuring compliance with timing requirements, even in large-scale networks with hundreds or thousands of frames. The complexity lies in automating this process to be both fast and resource-efficient, minimizing mismatches between traffic characteristics and TSN classes.

2. **Scalable zero jitter Scheduling of ST frames:** Scheduling ST in large and dynamic TSN networks is a computationally intensive problem. Existing scheduling methods struggle with scalability and real-time reconfiguration needs due to using ILP-based or CP-based methods. Developing a heuristic scheduler that balances scalability, speed, and schedulability without sacrificing performance is therefore essential.

3. **Minimizing Pessimism in AVB WCRTA:** In order to ensure that AVB traffic meets its timing requirements a WCRTA is required. Most existing approaches are either overly optimistic, risking deadlines, or excessively pessimistic, wasting resources. Achieving a safe WCRTA with minimal pessimism is a complex challenge.

4. **Synchronization in Heterogeneous TSN Networks:** Legacy systems often lack support for TSN's synchronization protocols, leading to mis-

alignments between the transmission schedules of legacy devices and TSN systems. Addressing this requires developing innovative mechanisms to mitigate synchronization issues, ensuring that the migrated legacy traffic operates seamlessly alongside TSN traffic, even in the absence of full synchronization capabilities.

These challenges collectively address the technical and practical barriers to the efficient configuration, analysis, and migration of Ethernet-based traffic to TSN, forming the foundation of this thesis.

Chapter 5

Research Methodology

This thesis aims to develop configuration and analysis tools that facilitate the adoption of TSN by the industry. To achieve this, we followed the hypothetico-deductive research method [70]. Fig. 5.1 illustrates the research process.

- **Start**: This project aims to expand the use of TSN in the industry by providing configuration and analysis tools with support for legacy systems.
- **Literature Review**: Once the objective was defined, we conducted an extensive state-of-the-art review on TSN traffic mapping, ST scheduling, AVB WCRTA, and TSN synchronization. This review included papers and standards to deeply understand the technology and identify potential improvements and points of conflict for the integration of legacy systems.
- **Identification of Limitations**: With the knowledge gained from the literature review, we identified the limitations of TSN in terms of mapping, ST scheduling, AVB analysis, and integration of legacy systems. When the review alone is not sufficient to fully identify the limitations, the possible limitations were formulated as hypotheses that we verified through experiments and/or models.
- **Problem Formulation**: At this stage, we identified the problems to be solved and the objectives to address them, based on the results obtained during the literature review and the identification of limitations.

- **Solution Proposal**: Once the problem has been identified and defined, novel solutions were discussed and proposed.

- **Implementation and Validation**: The solutions developed in the previous stage were implemented as tools or models. The performance of these implementations was evaluated through experiments in both real and synthetic networks. Real network experiments were complemented with simulations to assess aspects not easily evaluated through experiments, such as long-term effects. Additionally, the solutions were tested in specific use cases to examine their functionality in concrete contexts. When previous solutions existed, their performance was compared with the proposed solutions to identify and quantify the improvements achieved. In the case of problem-solving solutions, the adequacy of the solution or mitigation of the problem was verified.

- **Discussion and Results**: The effectiveness of the solution was inspected through a feedback loop. If the evaluation results were deemed inadequate, we revisited the proposed solution to refine the current solution or define a new one. Conversely, if the evaluation results were considered adequate, meaning the limitation identified through the literature review or models and/or experiments in the identification of limitations phase had been successfully resolved with a sufficient degree of certainty, the process concluded with the publication of the proposed solution.

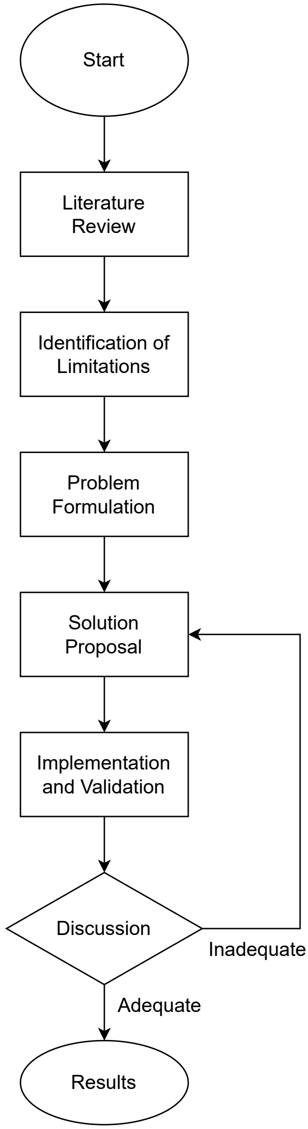


Fig. 5.1: Research Methodology

Chapter 6

Thesis Goals and Contributions

This thesis aims to enable and facilitate the industry's adoption of TSN. To achieve this, it is crucial to develop tools for the easy and efficient configuration and analysis of TSN, as well as its integration with legacy systems. Otherwise, the high costs associated with TSN adoption could render it non-viable, especially in the case of legacy systems migration, due to the significant time and material resources required for acquisition, migration, and integration.

6.1 Research Goals

The main goal of this thesis, as outlined in Section 12.1 and presented in Fig. 1.1, is **to develop an integrated toolchain that enables the near-automatic migration of Ethernet-based traffic, either legacy or not, to TSN networks in a fast and resource-efficient manner**. Achieving this goal introduces several research challenges related to configuration and analysis, as detailed in Section 4.5, which are addressed through the following research goals:

- **RG**$_1$: Develop a mapping methodology to classify Ethernet-based traffic into the three types of TSN traffic, namely ST, AVB, and BE, reducing the waste of resources in order to maximize the schedulability of the traffic.

- **RG**$_2$: Propose a scalable and fast heuristic scheduler for ST traffic with support for multiple ST queues with two modes of zero or relaxed reception jitter. The scheduler must achieve schedulability comparable to that of CP-based or ILP-based schedulers while significantly reduces the time to obtain the solution ensuring scalability.

- **RG**$_3$: Propose a safe and less pessimistic AVB Worst-Case Response Time Analysis compared to the existing analysis solutions in the literature. This improvement must enhance the efficiency of AVB in critical real-time systems, thereby increasing the overall practicality of TSN and increasing its appeal for industry adoption.

- **RG**$_4$: Design and implement a synchronization mechanism to enable ST communications for legacy end-stations within TSN networks. This solution must be fully integrated into TSN devices, operating transparently to legacy end-stations and to existing legacy synchronization protocols.

6.2 Research Contributions

The research contributions in the thesis to address the research goals are as follows:

- RC$_1$: We designed an Ethernet-based traffic model capable of describing the traffic of any Ethernet-based communication protocol in terms of parameters relevant for the configuration of TSN, including traffic characteristics such as period and traffic requirements such as deadline. This contribution partially fulfills our objective defined in RG$_1$ by providing a model that defines the behavior of new or legacy traffic through parameters that will later be used for the mapping.

- RC$_2$: We developed a mapping methodology that can distribute Ethernet-based frames characterized by the model proposed in RC$_1$ into different types of TSN traffic. We implemented the mapping method as a tool called Legacy Ethernet-based Traffic Mapping Tool (LETRA), along with a TSN traffic scheduler and an AVB WCRTA, and conducted a series of evaluations on various synthetic networks. The results showed that the proposed mapping method achieves up to a 90% improvement in traffic schedulability compared to an intuitive mapping method in a multi-switch network architecture. This contribution completes the second part of our goal defined in RG_1.

- RC_3: We proposed a heuristic scheduler for ST traffic in TSN networks, called Heuristic Multi-queue Scheduler (HERMES), which leverages multiple queues for ST traffic to provide high schedulability with very low scheduling times. Frames in HERMES can be configured to be scheduled in two modes: zero or relaxed reception jitter, offering better control for users. Through a series of experiments, we demonstrated that HERMES can outperform CP-based solutions, resulting in more schedulable networks by utilizing multiple queues, and providing results up to 800 times faster. This contribution fulfills our goal defined in RG_2.

- RC_4: We proposed a new AVB WCRTA method that addresses the optimism issues identified in previous proposals. Additionally, our new WCRTA demonstrates a reduction in pessimism of up to 90% compared to previous approaches. This contribution achieves our goal defined in RG_3.

- RC_5: We analyzed the consequences of the lack of synchronization and designed a mechanism called TSN With Legacy End-Stations Synchronization (TALESS) at the traffic scheduling level to provide a solution to the problems resulting from the lack of synchronization. TALESS operates solely within the TSN communication subsystem, offering a transparent solution for legacy end-stations and independent of the synchronization protocol of legacy devices. This contribution provides a complete solution to our goal defined in RG_4.

6.3 Included Papers

The research contributions are proposed in the form of published papers in conferences and journals. The order of the papers is in accordance with the contributions. Five papers are included in the PhD thesis: Papers A, B, C, D, and E.

6.3.1 Mapping the Included Papers with the Research Goals and Contributions

The mapping of the aforementioned research goals and thesis contribution into published publications is shown in Fig. 6.1.

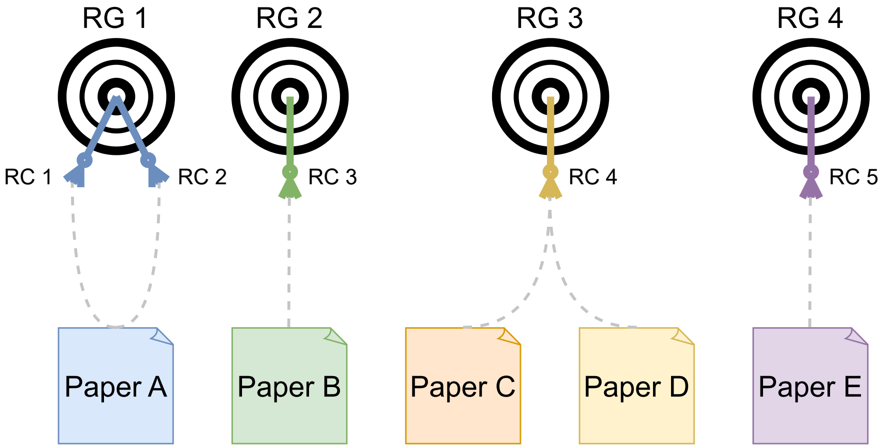


Fig. 6.1: Mapping the included papers with the Research Goals and Thesis Contributions

6.3.2 Paper A

Title:

LETRA: Mapping Legacy Ethernet-Based Traffic into TSN Traffic Classes

Authors:

Daniel Bujosa, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte.

Status:

Published in the 26^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), 2021.

Abstract:

This paper proposes a method to efficiently map the legacy Ethernet-based traffic into Time Sensitive Networking (TSN) traffic classes considering different traffic characteristics. Traffic mapping is one of the essential steps for industries to gradually move towards TSN, which in turn significantly mitigates the management complexity of industrial communication systems. In this paper, we first identify the legacy Ethernet traffic characteristics and properties. Based on the legacy traffic characteristics we presented a mapping methodology to map them into different TSN traffic classes. We implemented the mapping method as a tool, named Legacy Ethernet-based Traffic Mapping Tool or LETRA, together with a TSN traffic scheduling and performed a set of evaluations on different synthetic networks. The results show that the proposed mapping method obtains up to 90% improvement in the schedulability ratio of the traffic compared to an intuitive mapping method on a multi-switch network architecture.

Authors' Contributions:

I was the main driver of the work under the supervision of the co-authors. The plan for the paper was formed in joint discussions with the co-authors. I performed the tool implementation and evaluations, and wrote the draft of the paper. The co-authors have reviewed the paper, after which I have improved it.

6.3.3 Paper B

Title:

HERMES: Heuristic Multi-queue Scheduler for TSN Time-Triggered Traffic with Zero Reception Jitter Capabilities

Authors:

Daniel Bujosa, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, Thomas Nolte.

Status:

Published in the 30^{th} International Conference on Real-Time Networks and Systems (RTNS), 2022.

Abstract:

The Time-Sensitive Networking (TSN) standards provide a toolbox of features to be utilized in various application domains.The core TSN features include deterministic zero-jitter and low-latency data transmission and transmitting traffic with various levels of time-criticality on the same network. To achieve a deterministic transmission, the TSN standards define a time-aware shaper that coordinates transmission of Time-Triggered (TT) traffic. In this paper, we tackle the challenge of scheduling the TT traffic and we propose a heuristic algorithm, called HERMES. Unlike the existing scheduling solutions, HERMES results in a significantly faster algorithm run-time and a high number of schedulable networks. HERMES can be configured in two modes of zero or relaxed reception jitter while using multiple TT queues to improve the schedulability. We compare HERMES with a constraint programming (CP)-based solution and we show that HERMES performs better than the CP-based solution if multiple TT queues are used, both with respect to algorithm run-time and schedulability of the networks.

Authors' Contributions:

I was the main driver of the work under the supervision of the co-authors. The plan for the paper was formed in joint discussions with the co-authors. I performed the tool implementation and evaluations, and wrote the draft of the paper. The co-authors have reviewed the paper, after which I have improved it.

6.3.4 Paper C

Title:

An Improved Worst-Case Response Time Analysis for AVB Traffic in Time-Sensitive Networks

Authors:

Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei.

Status:

Published in the 45^{th} IEEE Real-Time Systems Symposium (RTSS), 2024.

Abstract:

Time-Sensitive Networking (TSN) has become one of the most relevant communication networks in many application areas. Among several traffic classes supported by TSN networks, Audio-Video Bridging (AVB) traffic requires a Worst-Case Response Time Analysis (WCRTA) to ensure that AVB frames meet their time requirements. In this paper, we evaluate the existing WCRTAs that cover various features of TSN, including Scheduled Traffic (ST) interference and preemption. We detect optimism problems in two of the existing WCRTAs, the analysis based on the busy period calculation and the analysis based on the eligible interval, when considering the effect of ST interference. Then, we propose a new analysis by developing a new ST interference calculation that can extend the analysis based on the eligible interval approach. The analysis covers the effect of the ST interference, the preemption by the ST traffic, and the multi-hop architecture. The resulting WCRTA, while safe, shows a significant improvement in terms of pessimism level compared to the existing analysis approaches based on the busy period and the Network Calculus model.

Authors' Contributions:

I was the main driver of the work under the supervision of the co-authors. The plan for the paper was formed in joint discussions with the co-authors. I performed the WCRTA implementation and evaluations, and wrote the draft of the paper. The co-authors have reviewed the paper, after which I have improved it.

6.3.5 Paper D

Title:

Reducing Pessimism in Response Time Analysis of AVB Traffic in TSN

Authors:

Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei.

Status:

MRTC technical report. Pending submission to a conference/journal.

Abstract:

Time-Sensitive Networking (TSN) is a set of standards with significant industrial relevance. Its primary advantage lies in its ability to simultaneously manage various types of traffic with distinct requirements, offering exceptional network flexibility. Among the traffic types, Audio-Video Bridging (AVB) is particularly notable for its real-time guarantees and adaptable online scheduling. To ensure that AVB frames meet their timing requirements, a Worst-Case Response Time Analysis (WCRTA) is essential. However, current WCRTAs are excessively pessimistic, often failing to guarantee schedulability for TSN networks operating at loads above 25% of the bandwidth, which limits their practical utility. In this study, we identified that the primary source of pessimism is the overly conservative calculation of interference from frames with the same priority as the frame under analysis. By analyzing and optimizing the Same-Priority Interference (SPI) calculation, we significantly enhance the schedulability of WCRTAs, thereby improving the overall efficiency of TSN networks.

Authors' Contributions:

I am the main driver of the work under the supervision of the co-authors. The plan for the paper was formed in joint discussions with the co-authors. I am performing the WCRTA implementation and evaluations, and writing the draft of the paper. The co-authors will review the paper, after which I will improve it.

6.3.6 Paper E

Title:

TALESS: TSN with Legacy End-Stations Synchronization

Authors:

Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, Mohammad Ashjaei.

Status:

Published in the Open Journal of the Industrial Electronics Society (OJIES), 2024

Abstract:

In order to facilitate the adoption of Time Sensitive Networking (TSN) by the industry, it is necessary to develop tools to integrate legacy systems with TSN. In this paper, we propose a solution for the coexistence of different time domains from different legacy systems, each with its corresponding synchronization protocol, in a single TSN network. To this end, we experimentally identified the effects of replacing the communications subsystem of a legacy Ethernet-based network with TSN in terms of synchronization. Based on the results, we propose a solution called TALESS (TSN with Legacy End-Stations Synchronization). TALESS can identify the drift between the TSN communications subsystem and the integrated legacy devices (end-stations) and then modify the TSN schedule to adapt to the different time domains to avoid the effects of the lack of synchronization between them. We validate TALESS through both simulations and experiments on a prototype. We demonstrate that thanks to TALESS, legacy systems can synchronize through TSN and even improve features such as their reception jitter or their integrability with other legacy systems.

Authors' Contributions:

I am the main driver of the work under the supervision of the co-authors. The plan for the paper was formed in joint discussions with the co-authors. I performed the tool implementation and evaluations, and wrote the draft of the paper. The co-authors have reviewed the paper, after which I have improved it.

Chapter 7

Conclusion and Future Directions

7.1 Conclusion

In this thesis, we aim to facilitate the industry's adoption of TSN through the development of a series of configuration and analysis tools, culminating in a comprehensive toolchain capable of migrating Ethernet-based traffic to TSN while supporting legacy devices. Fig. 1.1 presents a diagram of the toolchain. Specifically, we developed an Ethernet-based traffic model, which, in conjunction with a mapping tool called LETRA, classifies traffic into the three types of TSN traffic: ST, AVB, and BE. Subsequently, HERMES efficiently schedules the ST traffic and enables potential online reconfigurations. Additionally, AVB traffic is analyzed using the improved WCRTA, ensuring network schedulability with efficient use of resources. Finally, TALESS guarantees synchronization between the transmission of legacy end stations and the TSN schedule, preventing any negative consequences arising from the lack of synchronization between legacy end stations and TSN.

7.2 Future Directions

In future work, we aim to continue expanding the toolchain developed in this thesis. Firstly, in [71], we present the initial progress in integrating the ST scheduler with the AVB WCRTA. The current TSN scheduling and analysis process typically involves independently scheduling ST and then using the resulting ST schedule as input for the AVB WCRTA to assess whether AVB traffic meets its timing requirements. This approach is inefficient, as it often requires multiple iterations to identify an ST schedule that enables AVB schedulability. To address this, some ST schedulers aim to maximize the dispersion of ST transmission windows, enhancing AVB schedulability. However, these methods can be computationally intensive and may inadvertently compromise ST performance beyond what is necessary. To improve efficiency, we propose a novel approach that employs a modified AVB WCRTA to derive constraints for the ST scheduler. By leveraging the flexibility of ST scheduling, this method ensures AVB traffic schedulability from the outset, optimizing overall network performance in a more computationally efficient manner.

Secondly, we will seek to create input and output interfaces for the toolchain to automate the TSN adoption process. On one hand, we intend to develop automatic mechanisms for identifying and defining Ethernet-based traffic based on the application behavior, the network traffic report, or from sampling a legacy Ethernet-based network. On the other hand, we plan to utilize TSN configuration mechanisms to automatically deploy the configuration once the traffic migration is complete.

Finally, many aspects related to the support of legacy systems still need to be addressed. These include modifying the legacy Ethernet frames dynamically by incorporating elements, such as the VLAN tag, such that the TSN devices can correctly manage them. Moreover, potential incompatibilities with other mechanisms were not covered in this thesis, such as fault tolerance mechanisms and online resource reservation, which can be addressed in the future works.

Bibliography

[1] S. Samii and H. Zinner. Level 5 by Layer 2: Time-Sensitive Networking for Autonomous Vehicles. *IEEE Communications Standards Magazine*, 2(2):62–68, 2018.

[2] M. Wollschlaeger, T. Sauter, and J. Jasperneite. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Industrial Electronics Magazine*, 11(1):17–27, 2017.

[3] R. Salazar, T. Godfrey, L. Winkel, N. Finn, C. Powell, B. Rolfe, and M. Seewald. Utility Applications of Time Sensitive Networking White Paper (D3). Technical report, IEEE, 2018.

[4] IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.

[5] IEEE 802.1Q-2022: IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks, 2022.

[6] IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic. *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)*, pages 1–57, 2016.

[7] IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams. *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pages C1–72, 2010.

[8] Mohammad Ashjaei, Gaetano Patti, Moris Behnam, Thomas Nolte, Giuliana Alderisi, and Lucia Lo Bello. Schedulability analysis of Ethernet Audio Video Bridging networks with scheduled traffic support. *Real-Time Systems*, 53(4):526–577, 2017.

[9] Mohammad Ashjaei, Lucia Lo Bello, Masoud Daneshtalab, Gaetano Patti, Sergio Saponara, and Saad Mubeen. Time-Sensitive Networking in Automotive Embedded Systems: State of the Art and Research Opportunities. *Journal of Systems Architecture*, 110:1–47, September 2021.

[10] G. Alderisi, G. Patti, and L. Lo Bello. Introducing support for Scheduled traffic over IEEE Audio Video Bridging networks. In *Conf. Emerging Technologies Factory Automation*, 2013.

[11] S. Kehrer, O. Kleineberg, and D. Heffernan. A comparison of fault-tolerance concepts for IEEE 802.1 Time Sensitive Networks (TSN). In *IEEE Emerging Technology and Factory Automation*, 2014.

[12] Inés Álvarez, Ignasi Furió, Julián Proenza, and Manuel Barranco. Design and Experimental Evaluation of the Proactive Transmission of Replicated Frames Mechanism over Time-Sensitive Networking. *Sensors*, 21(3):756, 2021.

[13] K. S. Umadevi and R. K. Sridharan. Multilevel ingress scheduling policy for Time Sensitive Networks. In *Int. Conf. Microelectronic Devices, Circuits and Systems*, 2017.

[14] F. A. R. Arif and T. S. Atia. Load balancing routing in Time-Sensitive Networks. In *Int. Scientific-Practical Conference Problems of Infocommunications Science and Technology*, 2016.

[15] Marina Gutiérrez, Astrit Ademaj, Wilfried Steiner, Radu Dobrin, and Sasikumar Punnekkat. Self-configuration of IEEE 802.1 TSN networks.

In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2017.

[16] Marina Gutiérrez, Wilfried Steiner, Radu Dobrin, and Sasikumar Punnekkat. A Configuration Agent based on the Time-Triggered paradigm for Real-Time Networks. In *2015 IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–4. IEEE, 2015.

[17] Siwar Ben Hadj Said, Quang Huy Truong, and Michael Boc. SDN-based configuration solution for IEEE 802.1 Time Sensitive Networking (TSN). *ACM SIGBED Review*, 16(1):27–32, 2019.

[18] Seifeddine Nsaibi, Ludwig Leurs, and Hans D Schotten. Formal and simulation-based timing analysis of Industrial-Ethernet sercos III over TSN. In *2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, pages 1–8, 2017.

[19] Oscar Seijo, Zaloa Fernández, Iñaki Val, and Jesús A López-Fernández. SHARP: towards the integration of Time-Sensitive communications in legacy LAN/WLAN. In *2018 IEEE Globecom Workshops (GC Wkshps)*, pages 1–7, 2018.

[20] Sebastian Szancer, Philipp Meyer, and Franz Korf. Migration from SERCOS III to TSN-Simulation Based Comparison of TDMA and CBS Transportation. In *OMNeT++*, pages 52–62, 2018.

[21] Sebastian Schriegel and Juergen Jasperneite. A migration strategy for profinet toward ethernet TSN-based field-level communication: An approach to accelerate the adoption of converged IT/OT communication. *IEEE Industrial Electronics Magazine*, 15(4):43–53, 2021.

[22] Mohammadreza Barzegaran, Niklas Reusch, Luxi Zhao, Silviu S Craciunas, and Paul Pop. Real-time traffic guarantees in heterogeneous time-sensitive networks. In *Proceedings of the 30th International Conference on Real-Time Networks and Systems*, pages 46–57, 2022.

[23] Zhe Zhao, Lu Gao, Weitao Pan, Zhiliang Qiu, Xu Guo, Ya Gao, and Ling Zheng. Access mechanism for period flows of non-deterministic end systems for time-sensitive networks. *Computer Networks*, 231:109805, 2023.

[24] Sameer Chouksey, Hariram Selvamurugan Satheesh, and Johan Åkerberg. An experimental study of TSN-nonTSN coexistence. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1577–1584. IEEE, 2021.

[25] Voica Gavriluţ and Paul Pop. Traffic-Type Assignment for TSN-Based Mixed-Criticality Cyber-Physical Systems. *ACM Trans. Cyber-Phys. Syst.*, 4(2), 2020.

[26] Théo Docquier, Ye-Qiong Song, Vincent Chevrier, Ludovic Pontnau, and Abdelaziz Ahmed-Nacer. Iec 61850 over tsn: Traffic mapping and delay analysis of goose traffic. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 246–253. IEEE, 2020.

[27] Industrial Internet Consortium et al. Time sensitive networks for flexible manufacturing testbed characterization and mapping of converged traffic types, 2019.

[28] Chuanyu Xue, Tianyu Zhang, Yuanbin Zhou, Mark Nixon, Andrew Loveless, and Song Han. Real-time scheduling for 802.1 Qbv time-sensitive networking (TSN): A systematic review and experimental study. In *2024 IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 108–121. IEEE, 2024.

[29] Silviu S Craciunas, Ramon Serna Oliver, Martin Chmelík, and Wilfried Steiner. Scheduling Real-Time Communication in IEEE 802.1 Qbv Time Sensitive Networks. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, pages 183–192, 2016.

[30] Voica Gavriluţ and Paul Pop. Scheduling in Time Sensitive Networks (TSN) for mixed-criticality industrial applications. In *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 1–4, 2018.

[31] Mauricio GC Resende and Celso C Ribeiro. GRASP: Greedy Randomized Adaptive Search Procedures. In *Search methodologies*, pages 287–312. Springer, 2014.

[32] Yuanbin Zhou, Soheil Samii, Petru Eles, and Zebo Peng. Time-triggered scheduling for time-sensitive networking with preemption. In *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 262–267. IEEE, 2022.

[33] Voica Gavriluţ, Luxi Zhao, Michael L Raagaard, and Paul Pop. AVB-aware Routing and Scheduling of Time-Triggered traffic for TSN. *IEEE Access*, 6:75229–75243, 2018.

[34] Jin Y Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

[35] David Hellmanns, Lucas Haug, Moritz Hildebrand, Frank Dürr, Stephan Kehrer, and René Hummen. How to optimize joint routing and scheduling models for TSN using integer linear programming. In *Proceedings of the 29th International Conference on Real-Time Networks and Systems*, pages 100–111, 2021.

[36] Eike Schweissguth, Dirk Timmermann, Helge Parzyjegla, Peter Danielis, and Gero Mühl. ILP-based routing and scheduling of multicast realtime traffic in Time-Sensitive Networks. In *2020 IEEE 26th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–11. IEEE, 2020.

[37] Maryam Pahlevan and Roman Obermaisser. Genetic Algorithm for scheduling Time-Triggered traffic in Time-Sensitive Networks. In *2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA)*, volume 1, pages 337–344. IEEE, 2018.

[38] Ayman A Atallah, Ghaith Bany Hamad, and Otmane Ait Mohamed. Fault-resilient topology planning and traffic configuration for IEEE 802.1 Qbv TSN networks. In *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pages 151–156. IEEE, 2018.

[39] Maryam Pahlevan, Nadra Tabassam, and Roman Obermaisser. Heuristic list scheduler for Time Triggered traffic in Time Sensitive Networks. *ACM Sigbed Review*, 16(1):15–20, 2019.

[40] Ammad Ali Syed, Serkan Ayaz, Tim Leinmüller, and Madhu Chandra. Dynamic Scheduling and Routing for TSN based in-vehicle networks. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2021.

[41] Emmanuel Arzuaga and David R Kaeli. Quantifying load imbalance on virtualized enterprise servers. In *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*, pages 235–242, 2010.

[42] Leonard Kleinrock. Queueing systems, volume i: Theory, vol. I, 1975.

[43] Rina Panigrahy, Kunal Talwar, Lincoln Uyeda, and Udi Wieder. Heuristics for vector bin packing. *research.microsoft.com*, 2011.

[44] Yanzhou Zhang, Qimin Xu, Shouliang Wang, Yingxiu Chen, Lei Xu, and Cailian Chen. Scalable no-wait scheduling with flow-aware model conversion in time-sensitive networking. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pages 413–418. IEEE, 2022.

[45] Chuanyu Xue, Tianyu Zhang, Yuanbin Zhou, and Song Han. Real-time scheduling for time-sensitive networking: A systematic review and experimental study. 2023.

[46] Zitong Wang, Feng Luo, Yunpeng Li, Haotian Gan, and Lei Zhu. Schedulability Analysis in Time-Sensitive Networking: A Systematic Literature Review. *arXiv preprint arXiv:2407.15031*, 2024.

[47] Jonas Diemer, Jonas Rox, and Rolf Ernst. Modeling of Ethernet AVB networks for worst-case timing analysis. *IFAC Proceedings Volumes*, 45(2):848–853, 2012.

[48] Jonas Diemer, Daniel Thiele, and Rolf Ernst. Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching.

In *7th IEEE International Symposium on Industrial Embedded Systems (SIES'12)*, pages 1–10. IEEE, 2012.

[49] Lucia Lo Bello, Mohammad Ashjaei, Gaetano Patti, and Moris Behnam. Schedulability analysis of Time-Sensitive Networks with scheduled traffic and preemption support. *Journal of Parallel and Distributed Computing*, 144:153–171, 2020.

[50] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. Timing analysis of AVB traffic in TSN networks using network calculus. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 25–36. IEEE, 2018.

[51] Luxi Zhao, Paul Pop, Zhong Zheng, Hugo Daigmorte, and Marc Boyer. Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus. *IEEE Transactions on Industrial Electronics*, 68(10):10291–10302, 2020.

[52] Mohamed Seliem, Ahmed Zahran, and Dirk Pesch. Delay analysis of TSN based industrial networks with preemptive traffic using network calculus. In *2023 IFIP Networking Conference (IFIP Networking)*, pages 1–9. IEEE, 2023.

[53] Unmesh D Bordoloi, Amir Aminifar, Petru Eles, and Zebo Peng. Schedulability analysis of Ethernet AVB switches. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[54] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Tight worst-case response-time analysis for ethernet AVB using eligible intervals. In *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–8. IEEE, 2016.

[55] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent yet tight WCRT analysis for individual priority classes in ethernet AVB. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, pages 55–64, 2016.

[56] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent WCRT analysis for individual priority classes in Ethernet AVB. *Real-Time Systems*, 54:861–911, 2018.

[57] Maxim, Dorin and Song, Ye-Qiong. Delay analysis of AVB traffic in time-sensitive networks (TSN). In *Proceedings of the 25th International Conference on Real-Time Networks and Systems*, pages 18–27, 2017.

[58] Jetmir Haxhibeqiri, Xianjun Jiao, Muhammad Aslam, Ingrid Moerman, and Jeroen Hoebeke. Enabling TSN over IEEE 802.11: Low-overhead time synchronization for wi-fi clients. In *2021 22nd IEEE international conference on industrial technology (ICIT)*, volume 1, pages 1068–1073. IEEE, 2021.

[59] Haytham Baniabdelghany, Roman Obermaisser, et al. Extended synchronization protocol based on IEEE802. 1AS for improved precision in dynamic and asymmetric TSN hybrid networks. In *2020 9th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–8. IEEE, 2020.

[60] Alexey M Romanov, Francesco Gringoli, and Axel Sikora. A precise synchronization method for future wireless TSN networks. *IEEE Transactions on Industrial Informatics*, 17(5):3682–3692, 2020.

[61] Oscar Seijo, Xabier Iturbe, and Inaki Val. Tackling the Challenges of the Integration of Wired and Wireless TSN with a Technology Proof-of-Concept. *IEEE Transactions on Industrial Informatics*, 2021.

[62] Michael Gundall, Christopher Huber, Peter Rost, Rüdiger Halfmann, and Hans D Schotten. Integration of 5G with TSN as prerequisite for a highly flexible future industrial automation: Time synchronization based on IEEE 802.1 AS. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pages 3823–3830. IEEE, 2020.

[63] Zichao Chai, Wei Liu, Mao Li, and Jing Lei. Cross Domain Clock Synchronization Based on Data Packet Relay in 5G-TSN Integrated Network. In *2021 IEEE 4th International Conference on Electronics and Communication Engineering (ICECE)*, pages 141–145. IEEE, 2021.

[64] Jiajia Song, Makoto Kubomi, Jeffrey Zhao, and Daisuke Takita. Time synchronization performance analysis considering the frequency offset inside 5G-TSN network. In *2021 17th International Symposium on Wireless Communication Systems (ISWCS)*, pages 1–6. IEEE, 2021.

[65] Haochuan Shi, Adnan Aijaz, and Nan Jiang. Evaluating the Performance of Over-the-Air Time Synchronization for 5G and TSN Integration. In *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pages 1–6. IEEE, 2021.

[66] Tobias Striffler and Hans D Schotten. The 5G Transparent Clock: Synchronization Errors in Integrated 5G-TSN Industrial Networks. In *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, pages 1–6. IEEE, 2021.

[67] Junli Xue, Guochu Shou, Hongxing Li, and Yaqiong Liu. Enabling deterministic communications for end-to-end connectivity with software-defined time-sensitive networking. *IEEE Network*, 36(2):34–40, 2022.

[68] Daniel Bujosa Mateu, Daniel Hallmans, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, and Thomas Nolte. Clock Synchronization in Integrated TSN-EtherCAT Networks. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 214–221, 2020.

[69] Michael Lander Raagaard and Paul Pop. Optimization algorithms for the scheduling of IEEE 802.1 Time-Sensitive Networking (TSN). *Tech. Univ. Denmark, Lyngby, Denmark, Tech. Rep*, 2017.

[70] Gordana Dodig-Crnkovic. Scientific Methods in Computer Science. In *Proceedings of the Conference for the Promotion of Research in IT at New Universities and at University Colleges*, 2002.

[71] Bujosa, Daniel and Proenza, Julian and Papadopoulos, Alessandro V and Nolte, Thomas and Ashjaei, Mohammad. Introducing Guard Frames to Ensure Schedulability of All TSN Traffic Classes. In *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4. IEEE, 2023.

II

Included Papers

Chapter 8

Paper A
LETRA: Mapping Legacy Ethernet-Based Traffic into TSN Traffic Classes.

Daniel Bujosa, Mohammad Ashjaei, Alessandro V. Papadopoulos, Julián Proenza, Thomas Nolte.
In Proceedings of the 26^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2021).

Abstract

This paper proposes a method to efficiently map the legacy Ethernet-based traffic into Time Sensitive Networking (TSN) traffic classes considering different traffic characteristics. Traffic mapping is one of the essential steps for industries to gradually move towards TSN, which in turn significantly mitigates the management complexity of industrial communication systems. In this paper, we first identify the legacy Ethernet traffic characteristics and properties. Based on the legacy traffic characteristics we presented a mapping methodology to map them into different TSN traffic classes. We implemented the mapping method as a tool, named Legacy Ethernet-based Traffic Mapping Tool or LETRA, together with a TSN traffic scheduling and performed a set of evaluations on different synthetic networks. The results show that the proposed mapping method obtains up to 90% improvement in the schedulability ratio of the traffic compared to an intuitive mapping method on a multi-switch network architecture.

8.1 Introduction

New technologies often offer new solutions or improvements for companies that can lead to an advantage over the competitors; reducing costs or offering a better product, or lead to an environmental improvement; improving performance, optimization of resources, and/or emission reduction. However, new opportunities imply new challenges. These challenges are often related to their integration with existing legacy solutions and their implementation. For many industrial applications, it may not be cost-effective to adopt the new technologies, as it may require redeveloping all previous solutions and systems.

One of the new technologies, which can change the current paradigm of industrial communications and seems to be a key to the transition to Industry 4.0, is Time-Sensitive Networking (TSN). Everything started when, in 2005, the IEEE Audio-Video Bridging (AVB) Task Group (TG) was created. The purpose was to provide Ethernet with soft real-time capabilities oriented to audio/video streaming. The AVB TG developed three projects: (i) the IEEE Std 802.1AS [5], dedicated to clock synchronization, (ii) the IEEE Std 802.1Qav, which standardized the Credit-Based Shaper (CBS) [1]; and, finally, the IEEE Std 802.1Qat, which standardized the Stream Reservation Protocol (SRP) [2]. Additionally, another profile with series of rules, called IEEE Std 802.1BA-2011: Audio Video Bridging Systems [3], was created to ensure a minimum QoS when using the aforementioned standards. These standards together are commonly referred to as AVB standards. Over time, areas of applications, such as automotive [18], automation [21] and energy distribution [17], were interested in the work done by the TG so in 2012 the group was renamed to TSN TG and its objective was expanded to meet the needs of these new applications. The set of standards developed by the TG is usually referred to as TSN standards and it presents several interesting features. Specifically, TSN seems to provide Ethernet with proper support for mixed hard and soft real-time communications, flexibility of the traffic requirements and fault tolerance mechanisms. For these characteristics, TSN seems promising to enable new solutions within the context of modern industrial systems and solutions enabling, among other things, the integration of multiple legacy networks onto one TSN network. However, current TSN networks do not support all Ethernet-based legacy system message implementation characteristics, such as jitter in some legacy network devices, while currently used legacy technologies do not meet all TSN requirements. Moreover, it is cost-effective and beneficial for companies if they gradually move towards new technologies instead of completely replacing existing ones. Therefore, solutions to integrate a legacy system into a TSN network are essential so that services are not disturbed. An example of that is when a network consists of 4 nodes connected via different protocols, would be able to replace the end-to-end links with a TSN switch connected to all 4 nodes. This would allow the communications between networks while letting previous communications benefit from TSN features hence improving their real-time behavior, synchronization, and fault tolerance, to name a few.

Contributions: To allow industry to adopt TSN solutions, and the desire integration, a proper migration methodology of the legacy Ethernet-based traffic to TSN traffic classes should be designed. In this work we propose three steps to achieve this goal, as follows:

1. We develop a Legacy Ethernet-based Traffic model that can describe messages of any Ethernet-based communication protocols. Moreover, apart from the model, we require a methodology to identify the parameters of the messages inherited from the legacy system. The identification methodology is out of the scope of this paper. However, some works like [12] have already addressed this issue.

2. We develop a mapping methodology, and its corresponding implementation as a mapping tool, named Legacy Ethernet-based Traffic Mapping Tool or LETRA, that can map the legacy Ethernet-based messages characterized by the proposed model into different TSN traffic classes. To the best of our knowledge, this is the first attempt to map the Ethernet-based legacy traffic into TSN traffic classes considering a full spectrum of message characteristics. To evaluate the mapping methodology we implemented it as a mapping tool and compared its performance with an intuitive mapping methodology on different networks.

3. We integrated a pre-existing TSN scheduling method and a TSN schedulability analysis method into LETRA, which can map the messages, schedule the TSN traffic and evaluate their real-time behavior.

Paper outline: The paper is organized as follows. Section 8.2 presents the related work. Section 8.3 presents the legacy Ethernet-based traffic model. Section 8.4 describes the background of TSN and TSN traffic classes. Then, Section 8.5 proposes the traffic mapping methodology and development of LETRA, while Section 8.6 presents the experiments and evaluations. Finally, Section 8.7 concludes the paper and gives future directions.

8.2 Related work

Due to the great relevance of the work done by the TSN TG since 2012, the community has carried out a significant amount of work related to their study, application, and improvement. For example, the work in [6] studied the effects of the time-aware shaper, the work in [13] analyzed the fault tolerance issues, while the work in [7] proposed time redundancy to tolerate temporary faults, the work in [20] studied the scheduling policies and the load balancing was studied in [8]. Moreover, the work in [9] provided an up-to-date comprehensive survey of the TSN-related research.

Within the context of TSN traffic mapping, the work in [12] presented a network monitoring method to obtain the traffic properties based on measurements. A meta-heuristic method is proposed in [11] that maps mixed-criticality applications into the TSN traffic classes. Although the aim is similar to this paper, the proposed method does not cover all cases that are studied and exist in industrial applications. The method, thus, becomes suitable for cases where only very few mixed-criticality levels are assumed in the legacy system with no extensive timing information, whereas in this paper we consider different traffic characteristics including many timing characteristics and constraints when we map them into the TSN traffic classes.

There are also few works on integrating legacy networks into TSN networks. For example, the work in [15] integrated a few of the TSN standards into Sercos III, which is a closed system that allows standard Ethernet devices to be plugged, to improve its performance. Moreover, an integration methodology of wireless TSN (802.11) was proposed in [19]. Finally, an integration that focuses on the clock synchronization for EtherCAT and TSN was proposed in [14].

Nevertheless, to be the best of our knowledge, the proposed mapping methodology and the tool LETRA in this paper is the first attempt to map messages from any Ethernet-based legacy network into TSN traffic classes considering a full spectrum of message characteristics.

8.3 Legacy Ethernet-based traffic model

This section introduces a message model to describe the legacy Ethernet-based messages. To handle legacy messages from different Ethernet-based protocols, we identified a set of characteristics with which we can model the Ethernet-based messages. The model is used by the mapping tool to map the messages into different TSN traffic classes. Note that the extraction of values in the model is out of the scope of this paper, which can be done by measurements in the legacy networks as described in [12].

The parameters used to characterize the Ethernet messages are divided into three categories, including: *common parameters*, *periodic message parameters* and *non-periodic message parameters*. Note that not all messages need to have all parameters to map them. Following is the description of the parameters.

Common parameters

The common parameters are those that are independent of the behavior of the messages that are presented by $\{S, D, ML, DL, LRT, FLR, Prec\}$. In the above set, S and D represent the source and destination(s) of the message. The length of the message in bytes is denoted by ML which can be a range of sizes to consider variable message sizes. The message deadline is shown by DL. Moreover, LRT shows if the message is soft or hard real-time, i.e., missing deadlines lead to a degradation of functionality or a complete system failure, respectively. FLR denotes the message lost percentage, and $Prec$ denotes the precedence constraints between two or more messages which identifies if some messages should be transmitted or received in a specific sequential order.

Periodic message parameters

The periodic message parameters are those dependent on the periodicity of the messages, that is presented by $\{P, O, JI, JO\}$. P are the message period, while O is the offset, i.e., time shift of the message release time. JI represents the maximum jitter on the message release, while JO is the maximum jitter in the message reception. Note that jitter is the variation of delays that can be on transmission and/or reception of the message.

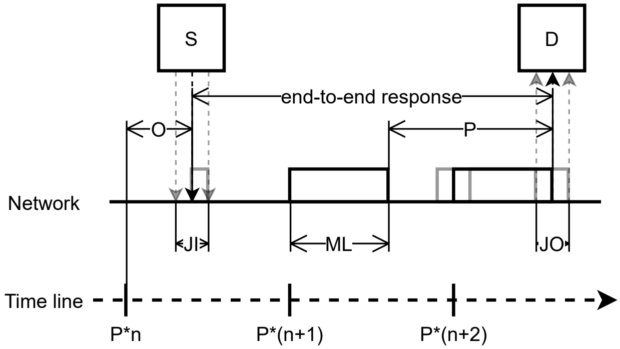


Fig. 8.1: Graphical representation of the presented model.

Non-periodic message parameters

The main parameter for non-periodic messages is the minimum inter-arrival time, identified by MIT, which is the minimum time between two consecutive releases of a message in non-periodic messages.

Figure 8.1 illustrates a graphical representation of some of the presented parameters in the model.

8.4 TSN traffic characteristics

This section gives a brief background about the TSN standards and TSN traffic classes. Communication in a TSN network is done among end-stations through routes of links and switches through Ethernet messages. A port in a TSN switch supports eight FIFO queues. A typical port with four TSN traffic classes (queues) is shown in Fig. 8.2. The TSN standards define three traffic classes including Scheduled Traffic (ST), Audio-Video Bridging (AVB) traffic, and Best Effort (BE) traffic. The AVB traffic is named by classes A and B, where class A has higher priority than class B. Following, we describe the details of TSN traffic classes.

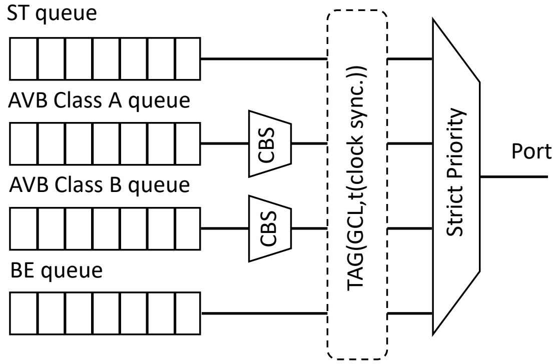


Fig. 8.2: A TSN egress port.

8.4.1 ST traffic class

The ST traffic is scheduled offline, which makes them fully deterministic with zero jitter in the message delivery. The TSN standard [4] defines a Time-Aware Gates (TAG) that is controlled by a Gate Control List (GCL). The GCL specifies at which specific time of the network-wide reference time gates are open, and thus, the link is available for a queue to send messages. Note that the reference clock is achieved thanks to the synchronization protocol defined in [5] which allows clock synchronization between end-stations and switches.

8.4.2 AVB traffic class

The AVB TG [1] introduced the CBS that defines credits to AVB queues. The credit is consumed when a message in that queue is sent, otherwise, it is replenished when there is a pending message in the queue (or the credit is still negative). The AVB queues can only transmit when their credit is positive or zero and their gate is open according to the GCL. CBS defines priority classes (classes A and B) but allowing transmission of low-priority traffic even if high-priority traffic is waiting according to their credits. This reduces buffering and improves lower-priority traffic QoS. Even though the activation time of AVB traffic is unknown due to possible blocking from other AVB classes or

ST queues, there are analysis methods to calculate their worst-case response time. For example, the analysis used in the experiments of this paper is the one presented in [10].

8.4.3 BE traffic class

BE has no real-time guarantees and is the lowest priority. This queue is not shaped by CBS and can only be sent if its gate is open and all other AVB queues have negative credit or there is no AVB traffic ready for transmission.

8.5 Proposed traffic mapping methodology

To map the legacy Ethernet-based traffic characterized by the parameters modeled in Section 8.3 into the TSN traffic classes, we developed three logic-based equations explained in the following sub-sections. As the equations are logic-based, in this section all parameters are treated as logical Boolean variables. In this sense, a parameter is True if the frame is affected by the parameter, otherwise False. For example, if one frame is non-periodic and has deadline of 100ms, as $P = NULL$, $P = 0$ in the equation. On the other hand, as $DL = \mathit{100ms}$, $DL = 1$ in the equation.

8.5.1 Mapping to the ST traffic class

The mapping first checks whether the legacy Ethernet traffic should be mapped into the ST traffic, according to the following expression:

$$ST = P \ \& \ (JO \ \| \ (!JI \ \& \ DL)) \tag{8.1}$$

where $\&$ is the logical "and" operator, $\|$ is the logical "or" operator, and ! is the logical "not" operator. First of all, Eq. (8.1) checks whether the message is periodic (P). This is mandatory as, otherwise, it would be impossible to implement a proper GCL. That is because, as described in Section 8.4, GCL is a list of open/close instructions executed repeatedly at certain times. Therefore, even if it is possible to schedule the message instances offline it would imply generating a long GCL table that makes the schedule impractical. Secondly, Eq. (8.1) checks if the message has JO constraints, and it is periodic, it must be transmitted as ST traffic as it is the only way to ensure meeting the JO requirement. That is because, as mentioned in Section 8.4, ST traffic is the only fully deterministic TSN traffic with zero JO. However, if the message has no JO requirements but it has DL and no JI then it can be also sent as an ST message. The reason for having D is to benefit from the ST characteristics while the JI conditions are to prevent waste of resources. JI implies considering larger open gates for the ST traffic to ensure the message instance to be transmitted within that time. This intuitively means allocating more bandwidth, which can be a waste of network bandwidth resources. Note that this happens only with JI and not with JO as once the message reaches the first TSN switch the messages will be sent just when the window starts.

8.5.2 Mapping to the AVB traffic class

Eq. (8.2) indicates whether the message can be sent as an AVB message. First, the message should have DL constraints to benefit from being sent as an AVB message. Moreover, the message must not have JO constraints unless it is not a hard real-time message. If the message has JO constraints but it is not hard real-time, another type of analysis, such as utilization-based analysis [11] may be needed. Note that Eq. (8.2) only specifies if the message can be transmitted as AVB traffic but it says nothing about the AVB possible priority classes. In this work, we consider only one AVB class queue for mapping as currently LETRA only identifies the messages as suitable or not for each traffic class. Later, it will be integrated or improved through constraint programming and/or metaheuristics to reach the desired specification level. In Eq. (8.2), HRT indicates the hard real-time requirement for the message.

$$AVB = DL\,\&\,(!JO \parallel !HRT) = DL\,\&\,!(JO\,\&\,HRT) \tag{8.2}$$

8.5.3 Mapping to the BE traffic class

Eq. (8.3) indicates whether the message can be sent as a BE class message. It checks whether the message has real-time requirements or not, which is the only requirement to be in the BE class.

$$BE = !JO \,\&\, !DL = !(JO \parallel DL) \tag{8.3}$$

8.5.4 Resulting truth table

As a result of the presented mapping methodology, we summarized the equations with a truth table shown in Table 8.1. As it can be seen, we just use P, JI, JO, DL, and LRT to map legacy Ethernet messages. The other parameters in the model do not affect the mapping but on the scheduling and analysis of the traffic after the mapping is performed to verify the timing properties.

To show the performance of the proposed mapping methodology we also define an intuitive mapping methodology. The intuitive mapping methodology classifies all periodic messages as ST traffic class and all non-periodic messages as AVB traffic class to still have a level of timing guarantee for them.

Note that the presented mapping methodology cannot be compared with the mapping presented in [11], which is only based on the criticality level of messages. The reason is that we consider more specific variables to map the messages, which means that [11] cannot map 90% of the messages considered in this work. However, [11] maps messages between ST class and AVB class which, according to our tool, are suitable for both classes. In this sense, the combination of both tools would expand the number of mappable messages, while resolving some ambiguities in LETRA.

8.5.5 Evaluation tool

To have a complete evaluation tool, we implemented LETRA, a TSN traffic scheduling, and a schedulability analysis to be able to evaluate the solution. Moreover, we developed a network generator and an intuitive mapping tool which implements the intuitive mapping methodology described in Section 8.5. The tools that are used for this evaluation from previous works include the ST scheduling tool in [16] and the AVB traffic schedulability analysis in [10].

The integration of the mentioned tools is shown in Fig. 8.3. First, the Network Generator generates the network messages according to the network configuration specified in the presented model. The generated messages are used as inputs for LETRA and the intuitive mapping tool, thus, obtaining two different

P	JI	JO	DL	HRT	ST	AVB	BE
0	X	X	0	0	0	0	1
0	X	X	0	1	0	0	1
0	X	X	1	0	0	1	0
0	X	X	1	1	0	1	0
1	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1
1	0	0	1	0	1	1	0
1	0	0	1	1	1	1	0
1	0	1	0	0	1	0	0
1	0	1	0	1	1	0	0
1	0	1	1	0	1	1	0
1	0	1	1	1	1	0	0
1	1	0	0	0	0	0	1
1	1	0	0	1	0	0	1
1	1	0	1	0	0	1	0
1	1	0	1	1	0	1	0
1	1	1	0	0	1	0	0
1	1	1	0	1	1	0	0
1	1	1	1	0	1	1	0
1	1	1	1	1	1	0	0

Table 8.1: Truth table of the mapping methodology.

classifications for the messages. The ST messages of both mapping tools are scheduled through the ST traffic scheduler [16] and, finally, the AVB traffic is checked through the AVB analyzer [10], which checks whether the AVB traffic are schedulable considering the transmission algorithms based on a response time analysis. The results of the scheduler and the AVB analysis are used to compare the mapping performance.

The Network Generator tool uses the parameters in the model explained in Section 8.3 as inputs to generate the messages randomly. Besides the parameters in the model, the network topology is an input.

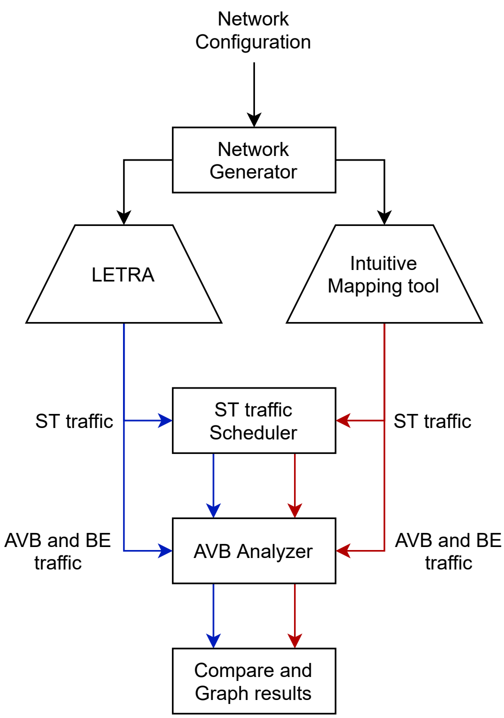


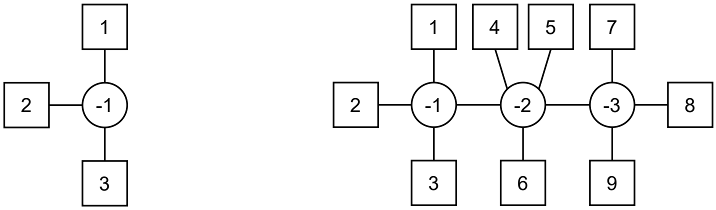
Fig. 8.3: Integration of the tools for evaluation.

8.6 Experiments and results

This section presents the experiments that we conducted to evaluate the proposed mapping methodology using the developed integrated tools (Fig. 8.3). We first present the experimental setup and then we illustrate and discuss the results.

8.6.1 Experimental setup

For the evaluation in this paper, we considered two network architectures, including a single-switch and a three-switch network. The topology is a line-star topology with the switches connected in a line and nodes connected to the



(a) A single-switch network. (b) A three-switches network.

Fig. 8.4: Experimental network architectures.

switches in the form of stars, as shown in Fig. 8.4. The line topology, apart from being widely used in the industry in many layers of the automation pyramid, is simpler and share many similarities with a tree topology. This allows us to extend the results of these experiments to a greater number of communication networks that could be developed in the future.

The network generator is designed such that we can select the input probabilities to uniformly distribute the probability among all possible messages, which is listed in Table 8.1. To achieve that we set the probability of all parameters to 50% except for periodicity. This means, as an example, that the messages have a 50% chance to have deadline constraints. With this setup, we ensure that all possible combinations with the same probability will be generated.

The parameters to generate the messages are set as follows. The network bandwidth is set to 10Mbps to prevent generating too many messages in case of generating a high load. The maximum link utilization is varied within the range [10%,90%] with the interval of 5%. Note that the messages will be generated such that the utilization in all links will be the one selected as an input, i.e., when we select 10% utilization the message generator selects the message sizes and routes to obtain 10% on all links. The maximum number of generated messages is set to 100, however, depending on the selected link utilization the message number can be different. The message length is selected within the range [64,1530] Bytes. The maximum allowed period and minimum inter-arrival time for messages are set to 1000μs. We also allowed arbitrary deadlines, which can be selected within the range [500,1000]μs. The input and

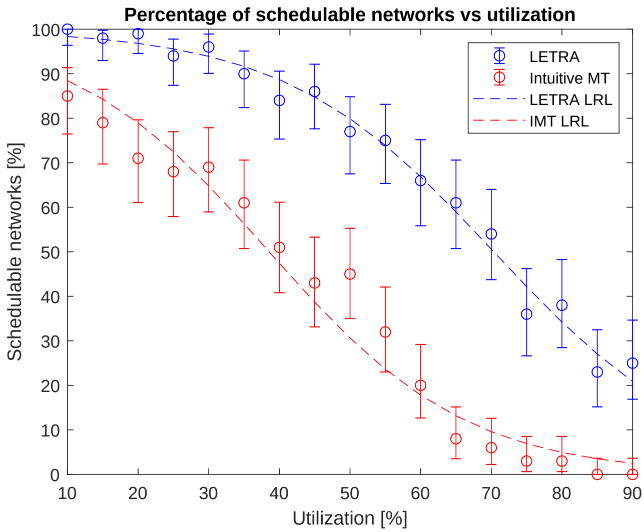


Fig. 8.5: Percentage of schedulable networks with respect to the bandwidth utilization for the single-switch network where the schedulability percentage is represented by a circle with error bars and the dashed curve corresponds to its logistical regression tendency line (LRL).

output jitter values are also selected within the range [1,100]μs. We generated 100 networks for each link utilization, e.g., 100 networks with the load of 10% on links, hence 1700 networks are generated for each network architecture shown in Fig. 8.4.

The next sub-sections present the results of the experiments. We compare the results of mapping the generated messages with LETRA and an intuitive mapping tool based on three different variables, including the link utilization, the number of messages, and the time it takes to schedule the ST messages.

8.6.2 Results of the single-switch network

Fig. 8.5 shows the percentage of schedulable networks with respect to the utilization. The horizontal axis shows the utilization of the generated networks,

Mean IMP [%]	Min IMP [%]	Max IMP [%]
86.65	37.52	149.47

Table 8.2: Performance improvements with respect to the percentage of bandwidth utilization for the single-switch network.

while the vertical axis shows the percentage of networks that are schedulable with two different traffic mapping tools. The circles in the figure are the schedulable percentage of generated networks with specific network utilization and the error bars are calculated through the binomial analysis with 95% certainty. In addition, the dashed lines show the trend of the data as logistic regression. As it can be seen, LETRA results in more schedulable networks in all generated utilization compared to the intuitive mapping tool. For instance, when we generated traffic with 90% utilization on all links, which is a very high network utilization, LETRA gives just below 30% of the networks schedulable, whereas the intuitive mapping results in very few schedulable networks. Table 8.2 shows the mean improvement by using LETRA compared to the intuitive mapping tool while Min IMP and Max IMP are the maximum and minimum possible improvements due to the result errors. LETRA results in 86.65% more schedulable networks compared to the intuitive mapping on average for the single-switch network architecture.

Fig. 8.6 shows the percentage of schedulable networks by varying the number of messages. Similar to the previous results, the vertical axis is the percentage of schedulable networks, and the horizontal axis is the number of generated messages. The figure also shows the error bars calculated through the binomial analysis with 95 % certainty and the tendency line calculated through logistic regression. Again, LETRA shows a significant improvement in the schedulability of networks compared to the intuitive mapping tool. For example, the intuitive mapping tool cannot schedule the networks when the number of messages is more than 35, however, LETRA can schedule a few of the generated networks up to 45 messages in the single-switch network. Table 8.3 shows the mean improvement between the two mapping tools, where it shows that on average 77.75% improvement by using LETRA.

Fig. 8.7 illustrates the time that it takes to schedule the ST messages in a

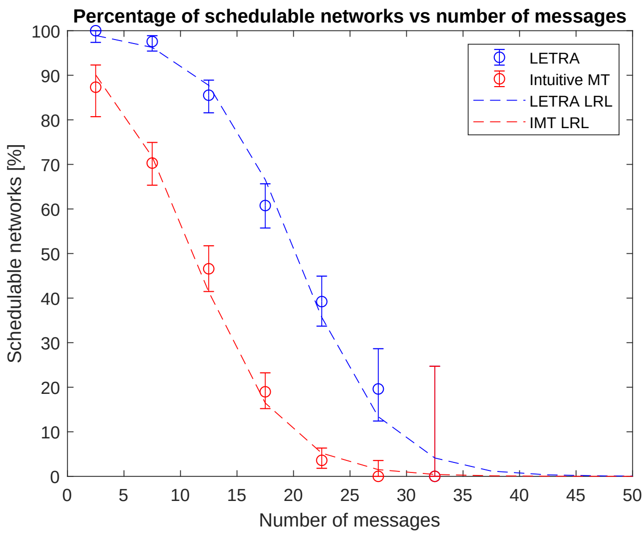


Fig. 8.6: Percentage of schedulable networks with respect to the number of messages for the single-switch network where the schedulability percentage is represented by a circle with error bars and the dashed curve corresponds to its logistical regression tendency line (LRL).

generated message after mapping. In the figure, times are shown in ms and the error bars are calculated through the gamma distribution analysis with 95% certainty. We also used a linear trend line to show the overall trend of data. In this figure, the values present high variability and they do not follow any basic tendency curve due to the number of parameters that can affect the scheduling time, such as memory or CPU utilization, which could not be monitored during the execution of the experiments due to hardware limitations. However, it can show, in general, apart from having better performance, LETRA is also delivering the ST schedules faster compared to the intuitive mapping tool. The reason is that the intuitive mapping tool selects more messages to be ST messages, while LETRA decides based on many timing requirements which in general leads to less number of ST messages. This means that the legacy messages are not unnecessarily mapped into the ST class.

Mean IMP [%]	Min IMP [%]	Max IMP [%]
77.57	35.93	120.85

Table 8.3: Performance improvements with respect to number of messages for the single-switch network.

Mean IMP [%]	Min IMP [%]	Max IMP [%]
90.74	32.30	165.28

Table 8.4: Performance improvements with respect to the percentage of bandwidth utilization for the three-switch network.

8.6.3 Results of the three-switch network

Fig. 8.8 shows the percentage of schedulable networks with respect to the bandwidth utilization for the three-switch network. Again, LETRA exhibits better performance compared to the intuitive mapping tool for all ranges of network utilization in larger networks. Table 8.4 shows mean improvement, which is 90.74% better performance with LETRA on average compared to the intuitive mapping tool. Although the performance of both mapping tools decreases with the size of the network, according to the results, we can conclude with 95% certainty that the percentage of improvement remains constant with the size of the network.

Fig. 8.9 shows the percentage of schedulable networks with respect to the number of messages. LETRA exhibits better performance compared to the intuitive mapping tool for the entire range of the number of messages analyzed. Table 8.5 shows that the amount of improvement in the larger network is 83.41% on average when using LETRA to map the traffic.

Mean IMP [%]	Min IMP [%]	Max IMP [%]
83.41	27.19	157.84

Table 8.5: Performance improvements with respect to the number of messages for the three-switch network.

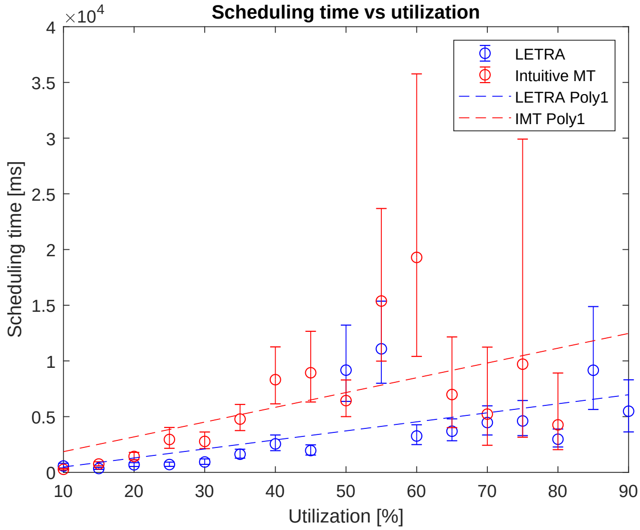


Fig. 8.7: Scheduling time with respect to bandwidth utilization for the single-switch network where the scheduling time is represented by a circle with error bars and the dashed line corresponds to a linear regression (Poly1).

Finally, Fig. 8.10 presents the time that it takes to schedule the ST messages after the mapping. Again, the results show that, besides LETRA having better performance, it is also faster in scheduling the ST messages compared to the intuitive mapping tool. Another interesting observation is that in the bigger network with high utilization, the ST messages can be scheduled within a reasonable time, while it is not the case if we map the traffic with the intuitive mapping tool where after 50% network utilization the ST messages cannot be scheduled. We believe that the reason is mainly due to the high amount of messages and utilization of the network.

8.7 Conclusions and Future Work

We argued that one of the essential steps towards migrating from legacy Ethernet-based networks to TSN-based networks in industries is efficiently

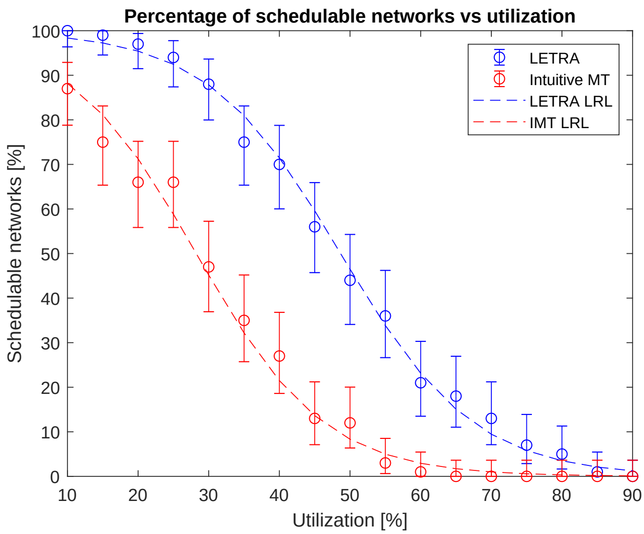


Fig. 8.8: Percentage of schedulable networks with respect to the bandwidth utilization for the three-switch network where the schedulability percentage is represented by a circle with error bars and the dashed curve corresponds to its logistical regression tendency line (LRL).

mapping the traffic into TSN traffic classes. Therefore, in this paper, we took a three-step strategy to achieve such a missing step. The steps include: (i) identifying the properties of legacy Ethernet-based messages by modeling them, (ii) map the Ethernet messages into different TSN classes, including ST, AVB, and BE classes, according to several timing properties, and (iii) develop a set of tools to evaluate the proposed mapping methodology, including the mapping tool, called LETRA, an ST scheduling tool and a schedulability analysis for the AVB messages. We also developed an intuitive mapping tool to show the performance of LETRA compared with that. We performed a set of experiments using two network architectures, being single-switch and three-switch architectures. We generated a set of messages randomly with a specific network utilization to show which mapping tool can result in more schedulable networks. The results show that in both network sizes LETRA performs much better, in

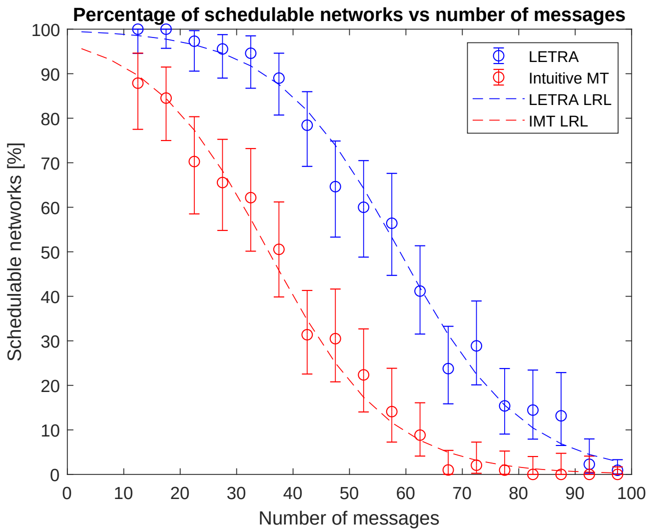


Fig. 8.9: Percentage of schedulable networks with respect to the number of messages for the three-switch network where the schedulability percentage is represented by a circle with error bars and the dashed curve corresponds to its logistical regression tendency line (LRL).

concrete, 86.65% better performance in the smaller network and 90.74% in the larger network in average.

All these results were obtained for a specific network topology and traffic configuration. In future work, we plan to run similar evaluations for other kinds of networks and schedulers to better define the mapping criteria and their performance. Moreover, we want to integrate it with the scheduler so the messages that can be placed in different traffic classes according to the mapping criteria can be specified while running the scheduler. On the other hand, we also plan to continue working on the other steps of the legacy integration. In this sense, we plan to develop a formal standard legacy Ethernet-based traffic model and adapt the schedulers to those message characteristics.

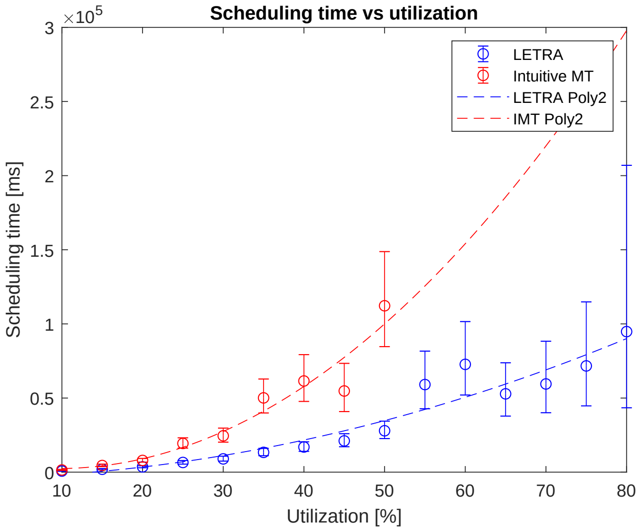


Fig. 8.10: Scheduling time with respect to the bandwidth utilization for the three-switch network where the scheduling time is represented by a circle with error bars and the dashed curve corresponds to a quadratic regression (Poly2).

Bibliography

[1] IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams. *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pages C1–72, 2010.

[2] IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP). *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, 2010.

[3] IEEE Standard for Local and Metropolitan Area Networks–Audio Video Bridging (AVB) Systems. *IEEE Std 802.1BA-2011*, pages 1–45, 2011.

[4] IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic. *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)*, pages 1–57, 2016.

[5] IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.

[6] G. Alderisi, G. Patti, and L. Lo Bello. Introducing support for scheduled traffic over IEEE audio video bridging networks. In *Conf. Emerging Technologies Factory Automation*, 2013.

[7] Inés Álvarez, Ignasi Furió, Julián Proenza, and Manuel Barranco. Design and experimental evaluation of the proactive transmission of replicated frames mechanism over time-sensitive networking. *Sensors*, 21(3):756, 2021.

[8] F. A. R. Arif and T. S. Atia. Load balancing routing in time-sensitive networks. In *Int. Scientific-Practical Conference Problems of Infocommunications Science and Technology*, 2016.

[9] Mohammad Ashjaei, Lucia Lo Bello, Masoud Daneshtalab, Gaetano Patti, Sergio Saponara, and Saad Mubeen. Time-sensitive networking in automotive embedded systems: State of the art and research opportunities. *Journal of Systems Architecture*, 110:1–47, September 2021.

[10] Mohammad Ashjaei, Gaetano Patti, Moris Behnam, Thomas Nolte, Giuliana Alderisi, and Lucia Lo Bello. Schedulability analysis of ethernet audio video bridging networks with scheduled traffic support. *Real-Time Systems*, 53(4):526–577, 2017.

[11] Voica Gavriluţ and Paul Pop. Traffic-type assignment for tsn-based mixed-criticality cyber-physical systems. *ACM Trans. Cyber-Phys. Syst.*, 4(2), 2020.

[12] Marina Gutiérrez, Wilfried Steiner, Radu Dobrin, and Sasikumar Punnekkat. Learning the parameters of periodic traffic based on network measurements. In *2015 IEEE International Workshop on Measurements & Networking (M&N)*, pages 1–6. IEEE, 2015.

[13] S. Kehrer, O. Kleineberg, and D. Heffernan. A comparison of fault-tolerance concepts for IEEE 802.1 Time Sensitive Networks (TSN). In *IEEE Emerging Technology and Factory Automation*, 2014.

[14] Daniel Bujosa Mateu, Daniel Hallmans, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, and Thomas Nolte. Clock synchronization in integrated tsn-ethercat networks. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 214–221, 2020.

[15] Seifeddine Nsaibi, Ludwig Leurs, and Hans D Schotten. Formal and simulation-based timing analysis of industrial-ethernet sercos iii over tsn. In *2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, pages 1–8, 2017.

[16] Niklas Reusch, Luxi Zhao, Silviu S Craciunas, and Paul Pop. Window-based schedule synthesis for industrial ieee 802.1 qbv tsn networks. In *2020 16th IEEE International Conference on Factory Communication Systems (WFCS)*, pages 1–4. IEEE, 2020.

[17] R. Salazar, T. Godfrey, L. Winkel, N. Finn, C. Powell, B. Rolfe, and M. Seewald. Utility Applications of Time Sensitive Networking White Paper (D3). Technical report, IEEE, 2018.

[18] S. Samii and H. Zinner. Level 5 by Layer 2: Time-Sensitive Networking for Autonomous Vehicles. *IEEE Communications Standards Magazine*, 2(2):62–68, 2018.

[19] Oscar Seijo, Zaloa Fernández, Iñaki Val, and Jesús A López-Fernández. SHARP: towards the integration of time-sensitive communications in legacy LAN/WLAN. In *2018 IEEE Globecom Workshops (GC Wkshps)*, pages 1–7, 2018.

[20] K. S. Umadevi and R. K. Sridharan. Multilevel ingress scheduling policy for time sensitive networks. In *Int. Conf. Microelectronic Devices, Circuits and Systems*, 2017.

[21] M. Wollschlaeger, T. Sauter, and J. Jasperneite. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Industrial Electronics Magazine*, 11(1):17–27, 2017.

Chapter 9

Paper B
HERMES: Heuristic Multi-queue Scheduler for TSN Time-Triggered Traffic with Zero Reception Jitter Capabilities.

Daniel Bujosa, Mohammad Ashjaei, Alessandro V. Papadopoulos, Julián Proenza, Thomas Nolte.
In Proceedings of the 30^{th} International Conference on Real-Time Networks and Systems (RTNS 2022).

Abstract

The Time-Sensitive Networking (TSN) standards provide a toolbox of features to be utilized in various application domains.The core TSN features include deterministic zero-jitter and low-latency data transmission and transmitting traffic with various levels of time-criticality on the same network. To achieve a deterministic transmission, the TSN standards define a time-aware shaper that coordinates transmission of Time-Triggered (TT) traffic. In this paper, we tackle the challenge of scheduling the TT traffic and we propose a heuristic algorithm, called HERMES. Unlike the existing scheduling solutions, HERMES results in a significantly faster algorithm run-time and a high number of schedulable networks. HERMES can be configured in two modes of zero or relaxed reception jitter while using multiple TT queues to improve the schedulability. We compare HERMES with a constraint programming (CP)-based solution and we show that HERMES performs better than the CP-based solution if multiple TT queues are used, both with respect to algorithm run-time and schedulability of the networks.

9.1 Introduction

Data communication in industrial systems has dealt with many challenges during recent years, such as scalability in data transmission, high volume of data exchange, the coexistence of diverse applications with different time-criticality requirements, and guaranteeing deterministic transmission for hard real-time traffic. These challenges are mainly due to recent demands for increasing functionalities in industrial systems that impose further pressure on the data communication design of such systems. For instance, in several application domains, e.g., autonomous vehicles and smart automation, many sophisticated smart sensors and cameras are utilized to perform newly added functionalities that require a high amount of communication bandwidth and at the same time meet their timing requirements. Besides the timing requirements, the rise of adaptive industrial systems imposes another criterion for designing data communication systems in which the network should be reconfigured due to changes in the environment. Therefore, in such systems, the configuration of the network is not seen as a one-time configuration in the initialization phase, but as a dynamic reconfiguration during the run-time (and operational) phase.

IEEE Audio-Video Bridging (AVB) Task Group (TG) was established in 2005 to provide Ethernet with soft real-time capabilities oriented to audio/video streaming. The three main projects developed by this TG are: (i) the IEEE Std 802.1AS [3] for clock synchronization, (ii) the IEEE Std 802.1Qav, which standardized the Credit-Based Shaper (CBS) [1]; and finally (iii) the IEEE Std 802.1Qat, which standardized the Stream Reservation Protocol (SRP) [2]. The latter standard is particularly interesting in the context of dynamic networks as it allows adding and removing streams at run-time. As the features that were developed by the AVB TG became relevant to other application areas, such as automotive [23], automation [27], and energy distribution [22], new requirements emerged. Therefore, in 2012, the TG broadened its objectives to meet the demands and was renamed to Time-Sensitive Networking (TSN) TG. Specifically, TSN TG's work was developed as a set of standards to provide transmission of hard and soft real-time traffic on the same network, deterministic zero-jitter and low-latency transmission, precise clock synchronization, fault tolerance mechanisms, and advanced network management allowing dynamic reconfiguration.

Motivation: One of the main features developed within TSN TG is the zero-jitter traffic transmission, known as the Time-Aware Shaper (TAS), which is particularly utilized in applications that require low-latency and low-jitter data transmission, e.g., in embedded control systems. The TAS allows transmission of Time-Triggered (TT) traffic while preventing any interference from other traffic via a gate mechanism on the ports of the switches. Therefore, TAS requires the synthesis of the Gate Control Lists (GCL) that are specifying at which point in time each frame should be transmitted. A GCL is defined for each switch port which contains 8 queues, in such a way that the GCL identifies the moments in which the gate of each queue will be open. The scheduling of TT traffic, and its synthesis in GCLs, is known to be an NP-complete problem [19]. Several solutions are proposed in the literature to schedule TT traffic in TSN networks that are mainly based on Integer Linear Programming (ILP) and Constrained Programming (CP) [7]. These solutions are known to have high time complexity, i.e., they require a long time to schedule large networks, thus they are not generally scalable. In addition, these solutions are not suitable for systems that require dynamic reconfigurations as the new configuration should be created relatively fast. Few heuristic schedulers are also proposed, e.g., [17], whose performance is not properly compared with the ILP and CP solutions.

Paper contributions: In this paper, we propose a heuristic scheduler for TT traffic in TSN networks, called Heuristic Multi-queue Scheduler (HERMES), that takes advantage of multiple queues for TT traffic to provide high schedulability with very low scheduling times. Frames in HERMES can be configured to be scheduled in two modes of zero or relaxed reception jitter, which provides better control for users. Through a set of experiments, we show that HERMES can perform better than CP-based solutions, i.e., it results in more schedulable networks, by allowing it to use multiple queues, and at the same time, it provides the results within 17 to 800 times faster. In our experiments with two sizes of networks, we obtained schedules in less than $1ms$, which shows that HERMES is suitable for dynamic reconfiguration of networks.

Paper outline: The paper is organized as follows. Section 9.2 presents the related work. Section 9.3 presents the background. Section 9.4 presents the proposed algorithm, i.e., HERMES. Sections 9.5 analyzes the HERMES performance. Finally, Section 9.6 concludes the paper and indicates future directions.

9.2 Related Work

There have been many works on various TSN topics, including investigation of time-aware shaper mechanisms [4], proposing fault tolerance techniques [12], techniques to tolerate temporary faults in TSN networks with the use of retransmissions [5], and schedulability analysis of traffic with different TSN features [29], [14]. A recent comprehensive survey [7] presents the status of research within TSN, including schedulability and scheduling problems, safety and security issues, and evaluation models and tools.

Within the context of TT traffic scheduling in TSN networks, the work in [26] present a scheduling algorithm formalized as an ILP while the works in [24] and [16] present a joint routing and scheduling algorithm formalized as an ILP and as a meta-heuristic scheduling approach based on a Genetic Algorithm (GA) approach, respectively. The work in [9] presents an SMT-based scheduler capable of scheduling networks with several TT queues. The work in [10] proposes a GCL synthesis approach based on Greedy Randomized Adaptive Search Procedure (GRASP) meta-heuristic [20], which takes AVB traffic into account, whereas the work in [11] proposes a joint routing and scheduling approach for TT and AVB traffic by means of an integrated heuristic and meta-heuristic strategy. In the latter work, the K-Shortest Path (KSP) method [28] is utilized for routing, and GRASP is used to schedule both TT and AVB at the same time. Moreover, the work in [8] synthesizes a network topology that supports seamless redundant transmission for TT traffic by proposing a greedy heuristic algorithm for joint topology, routing, and scheduling synthesis.

Protocol	Routing	Multi-queuing	Schedule	ZRJ
HLS	Yes	No	per frame	No
MML	Yes	No	per frame	No
BN	Yes	No	per frame	No
CV	Yes	No	per frame	No
MDP	Yes	No	per frame	No
HERMES	No	Yes	per link	Yes

Table 9.1: Comparison between heuristic schedulers.

The above-mentioned solutions are mostly based on ILP or constraint programming, while some of them exploit the use of meta-heuristics, e.g., GA. However, these solutions normally are highly time-complex, which makes them not scalable. Few works target heuristic solutions with lower time complexity. For instance, the work in [17] proposes a heuristic routing and scheduling algorithm called Heuristic List Scheduler (HLS) that is limited to a single TT queue, while the work in [25] compares 4 heuristic algorithms combining routing and scheduling (Modified Most Loaded Heuristic (MML), Bottleneck Heuristic (BN), Coefficient of Variation Heuristic (CV) [6][13], and Modified Dot Product Heuristic (MDP) [18]), all with scheduling times greater than 100 ms and unable to handle multiple queues. In this work, we propose a heuristic algorithm, called HERMES, with scheduling times lower than 10 ms that uses multiple TT queues to improve schedulability. Moreover, the proposed algorithm provides two modes, one with zero reception jitter and relaxed reception jitter in the receiver end-station, see Section 9.3 for detailed description of zero and relaxed reception jitter. The zero reception jitter mode is configurable which is helpful for the applications in which the feature is not essential. Table 9.1 shows the main differences between the heuristic schedulers mentioned above, including HERMES. The features that are analyzed in this comparison include routing, multi-queuing for TT traffic, scheduling process, and support for zero reception jitter (ZRJ).

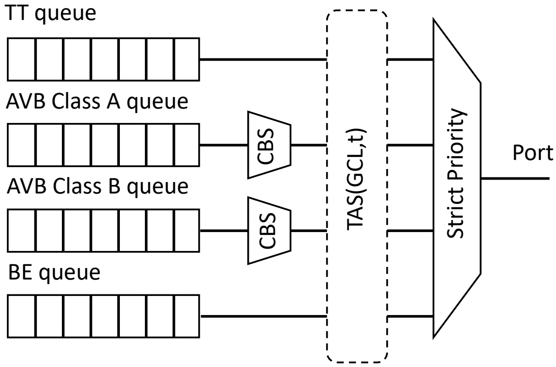


Fig. 9.1: A TSN egress port with four FIFO queues: one TT queue, two AVB queues, and one BE queue.

9.3 Background

TSN end-stations communicate by transmitting Ethernet frames through routes consisting of links and time-sensitive switches. In TSN, Ethernet frames belong to one of the eight possible priorities. The traffic is classified as one of the three available traffic classes, including TT traffic, AVB traffic, and BE traffic, where TT traffic has higher priority than other traffic classes and BE has the lowest priority. Note that several priorities may cover one traffic class, e.g., AVB can consist of classes A, B, and C, each associated with one priority level. A port of a TSN switch supports up to eight FIFO queues each of them associated with one priority level. Figure 9.1 shows an example of a time-sensitive device output port with four queues configured as TT traffic with the highest priority, AVB classes A and B traffic with the medium priority, and a BE traffic class as the lowest priority.

9.3.1 Time-Triggered Traffic

TT traffic is scheduled offline, which allows to know exactly in which time slot each TT frame is transmitted. This requires that interference between frames must be prevented. This is achieved through the TAS mechanism (see Fig-

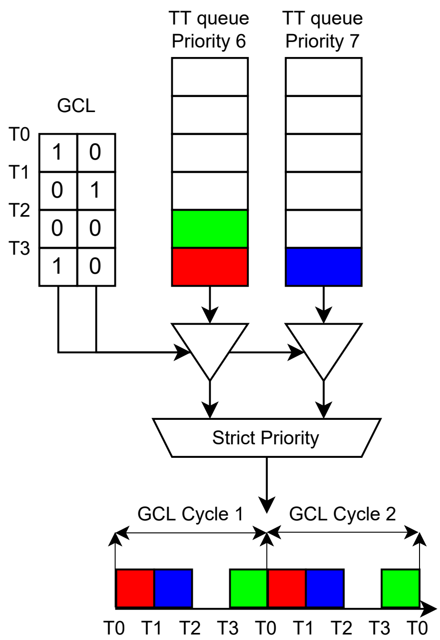


Fig. 9.2: TSN TAS gate mechanism.

ure 9.1). According to this mechanism, each queue has an associated gate that can be open or closed. The frames in a queue can be transmitted when the gate is open, otherwise, the frames are blocked for transmission.The gates are controlled by the GCL, which specifies at which point in time gates should be open, and it is a cyclic list that repeats the schedule. The time that gates are open or closed can be specified at the nanosecond level for each entry of the GCL and we refer to the opening time of a gate as *window*.

Figure 9.2 shows an example of TAS operation for two TT queues with two different priorities, i.e., priority 6 and 7. In this example, we assume that three TT frames with a period of 4 time units are transmitted through a switch port where one of the TT frames is set to the highest priority 7 (blue frame), while the other two frames (red and green) are set to priority 6. As the periods of the frames are equal, the hyper-period (the least common multiple) of them is

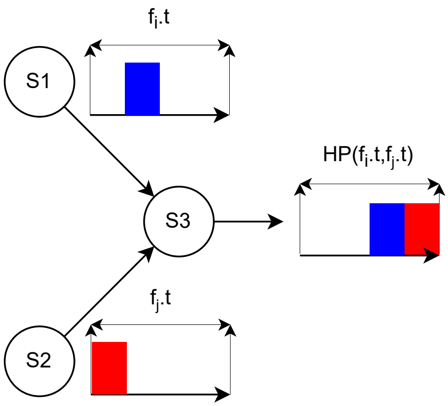


Fig. 9.3: Multi-hop behavior of TSN and GCLs.

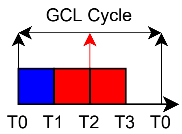
4 time units. Therefore, the GCL cycle is defined as 4 time units allowing gates operation in each time slot and repeating every 4 time units. According to the schedule in this example, which is set in the GCL, at time T0 till T1 the gate for priority 6 queue is open (shown as 1 in GCL), whereas the gate for priority 7 is closed (shown as 0 in GCL) allowing transmission of the red frame. Further, in the time slot between T1 and T2, the blue frame can be transmitted as the gate for priority 7 is open. Between T2 and T3 both gates are closed, thus no transmission can occur, and finally, the gate of priority 6 queue is open in the last time slot that allows the transmission of the last frame, i.e., the green frame. Two cycles of frame transmissions are represented at the bottom of Figure 9.2.

On the other hand, TSN supports multi-hop communication which imposes other restrictions when scheduling. For a frame to be transmitted on a link, it must have been previously transmitted through the preceding links in the route of the frame. Furthermore, the order of frames in transmission is also important. Considering that the queues in the switches are FIFO, if a frame arrives to a switch before another one, it will also be transmitted first. Figure 9.3 shows an example of a multi-hop schedule. The figure shows three switches (S1, S2, and S3) and three links, two of them connecting S1 and S2 with S3 and one in the S3 output. Two frames are exchanged between these 3 switches, one blue frame is sent by S1 and one red frame is sent by S2 and both frames are forwarded by S3 through the output link. As we can see, both are sent after they are received by S3. However, according to the schedule that is decided for this case (for whatever the reason), the red frame arrives before the blue frame to S3 but the blue frame is sent by S3 before the red frame, thus this schedule would not be possible with a single queue. In this case, the red and blue frames must have been assigned to different queues and hence have different priorities.

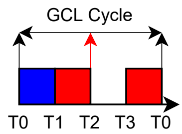
Finally, interference-free transmission of frames ensures their zero jitter transmission. This means that the variations in the transmission and reception of each frame with respect to the schedule will be zero, assuming that the clock drift is zero. However, in this work, not only the jitter but also the reception jitter has been considered. The reception jitter is defined as the variability of the instant of reception by the receiver end-station of a frame with respect to its period. For example, Figure 9.4 shows the schedule over the GCL cycle of a TSN output port connected to the input port of the receiving end-station. In this schedule two frames are shown, a blue frame with a period of 4 time units and a red frame with a period of 2 time units. According to the schedule example in Figure 9.4a, the blue frame has zero reception jitter since it is always transmitted at time unit 0, whereas the red frame has reception jitter since it is transmitted at time unit 1 in its first instance (T1) of the hyper-period and at time 0 in its second instance (T2). Zero reception jitter is particularly interesting in heterogeneous systems combining TSN components and legacy components that cannot adopt TSN synchronization mechanisms. In these cases, a frame with zero reception jitter helps the synchronization of the applications even if the devices are not properly synchronized. Throughout the paper, we denote zero reception jitter by ZRJ and reception jitter by RJ. Note that in this case, as shown in Figure 9.4b, it would be enough to delay the transmission of the second instance of the red frame by one time unit for both frames to have ZRJ.

9.3.2 AVB and BE Traffic

AVB frames are not scheduled offline, i.e., they are scheduled via CBS once they arrive at the switch port. The gates are normally open for them unless TT traffic is to be transmitted. The CBS aims at improving the Quality-of-Service (QoS) of lower priority traffic while ensuring a minimum of bandwidth



(a) Schedule of two frames with and without reception jitter.



(b) Schedule of two frames, both with zero reception jitter.

Fig. 9.4: Difference between reception jitter and zero reception jitter.

utilization. Finally, BE frames have no real-time guarantees, thus they will be transmitted when their gate is open and the CBS is negative for all higher priority traffic.

9.4 Proposed scheduling algorithm

We developed what we call Heuristic multi-queue Scheduler (HERMES) which generates the global schedule for the transmission of TT traffic. Our goal is to reduce the scheduling time while achieving high schedulability through the use of different numbers of TT queues and providing zero jitter. Moreover, HERMES can provide zero reception jitter.

HERMES calculates the GCL of each egress port of the end-stations and TSN switches. To do this, unlike other heuristic algorithms that schedule frame by frame, our algorithm decides the schedule link by link (and then for each link deciding the schedule of each frame to be transmitted in that link) starting with the destination links and ending with the source links scheduling each frame as late as possible according to their timing constraints. The reason why we design HERMES to calculate the schedule link by link instead of frame by frame is that in this way the conflicts between frames are detected before the entire frame has been scheduled. On the contrary, when the network is scheduled frame by frame, the schedule of one frame may hinder the scheduling of the other frames and the algorithm will have to schedule that frame again throughout all its links. On the other hand, links are scheduled from destination to source because the destination link is the most restrictive link especially if the frame has more restrictive reception time constraints such as frames that must be received with zero reception jitter. In addition, in each links, each frame is scheduled as late as possible so that the preceding links have enough time margin between the frame's period start and the offset of the same frame in the last scheduled link. However, this does not imply that the frames will have the highest possible latencies because an improvement as simple as looking for the minimum offset of all frames and moving all frames earlier by that amount can be applied.

The use of multiple queues for TT traffic to improve the schedulability is thoroughly explained in Section 9.4.2. However, it is worth to mention that HERMES does not consider relative priorities between TT queues, i.e., all queues have the same priority. The isolation of frames by only opening the gate of a single queue at a time eliminates the arbitration among TT frames which is performed by the Strict Priority module (see Figure 9.1) if more than one queue has its gate open. The queues are only used to help the scheduler meeting the order condition explained in Section 9.3.

Although HERMES uses only unicast frames, the algorithm would work equally well with multicast frames. As we will see in the algorithm description, HERMES only schedules those links whose frames have already been scheduled in the subsequent links. In this way, in the case of multicast frames, when scheduling the link before the fork, the following links will already be scheduled and therefore the algorithm can continue to function normally only taking into account the offset and restrictions of several following links instead of a single following one.

9.4.1 System Model

In this work the communication network model consists of two main sets, one for the links $\mathcal{L}$ and one for the TT-frames $\mathcal{F}_{\text{TT}}$. Each link $l \in \mathcal{L}$ is unidirectional, is defined by its identifier, and has a parameter $l.\phi$ indicating in which phase of the execution of HERMES the link will be scheduled. Indeed, the execution of the scheduling algorithm presented in this paper is divided into phases, with a total of Φ phases. In the case of full-duplex links, two links with opposite directions are instantiated. On the other hand, each frame $f \in \mathcal{F}_{\text{TT}}$ is characterized by seven parameters $f = \langle t, w, d, q, u, n, \mathcal{S} \rangle$, i.e.,

1. the period $f.t$,
2. the length of the frame or the size of the window needed to transmit the frame $f.w$,
3. the deadline $f.d$,
4. the queue of the frame in all egress ports of the whole route $f.q$,
5. a parameter to decide in which order the frames in the links are scheduled, which in this case is the frame utilization $f.u$ (see Algorithm 2 in Section 9.4.2),
6. the number of links in the route $f.n$, and
7. a set $f.\mathcal{S}$ containing the route and the schedule of each link in the route.

Each element $s \in f.\mathcal{S}$ includes three parameters $s = \langle \zeta, \iota, \mathcal{O} \rangle$:

1. the link $s.\zeta$ of the route assigned in reverse order, i.e. $s_1.\zeta$ being the destination link and $s_{f.n}.\zeta$ the source link,
2. the number of instances $s.\iota$ of the frame in the link, and
3. a set $s.\mathcal{O}$ indicating the offset of each instance according to the period start of the instance.

9.4.2 HERMES

The input to the scheduler consists of a list of TT frames characterized by their period, frame length, deadline, and routing expressed as a vector of unidirectional link identifiers. With this list, HERMES executes the following steps to obtain the schedule.

Algorithm 1, DivPhases: First of all, as mentioned before, links are divided into phases where all frames in all links assigned to that phase are scheduled togther. The only condition for a link l_i to be assigned to a phase $l_i.\phi$ is that all frames transmitted through that link $f \in \mathcal{F}_{\text{TT}} : f.s_j.\zeta = l_i$ must have all previous links (links closer to destination) assigned to previous phases

Algorithm 1: HERMES - DivPhases

Data: $\mathcal{L}, \mathcal{F}_{\text{TT}}$

Result: $\forall l \in \mathcal{L}$ return $l.\phi$

procedure *DivPhases*

  for $\forall l \in \mathcal{L}$ **do** $l.\phi \leftarrow NULL$ **end**

  $\Phi \leftarrow 1$

  while $\exists l \in \mathcal{L} : l.\phi = NULL$ **do**

    for $f_i.s_j, f_k.s_x \in \mathcal{F}_{TT}, f.\mathcal{S} : f_i.s_{j-1}! = NULL \wedge f_k.s_{x-1} = NULL$ **do**

      if $\exists! f_i.s_j.\zeta = f_k.s_x.\zeta$ **then**

        $f_i.s_j.\zeta.\phi \leftarrow \Phi$

      end

    end

    $\Phi \leftarrow \Phi + 1$

  end

Algorithm 2: HERMES - AssignFrameUtilization

Data: $\mathcal{F}_{\text{TT}}$

Result: $\forall f \in \mathcal{F}_{\text{TT}}$ return $f.u$

procedure *AssignFrameUtilization*

  for $\forall f \in \mathcal{F}_{TT}$ **do**

    $f.u \leftarrow \frac{f.w}{f.d} \cdot f.n$

  end

$\forall f.s_k.\zeta : k < j | f.s_k.\zeta.\phi < f.s_j.\zeta.\phi$. To do that, if there are links not assigned to any phase, the algorithm adds a new phase and checks if links not assigned are ready to be assigned in the new phase according to previous conditions. Note that, as mentioned before, links in the route are ordered from destination to source, i.e., routes are scheduled backwards and in each phase all links which, fulfilling the above conditions, are independent can be scheduled. Figure 9.5 presents a network example where the squares with positive numbers represent end-stations and the circles with negative numbers represent switches. For

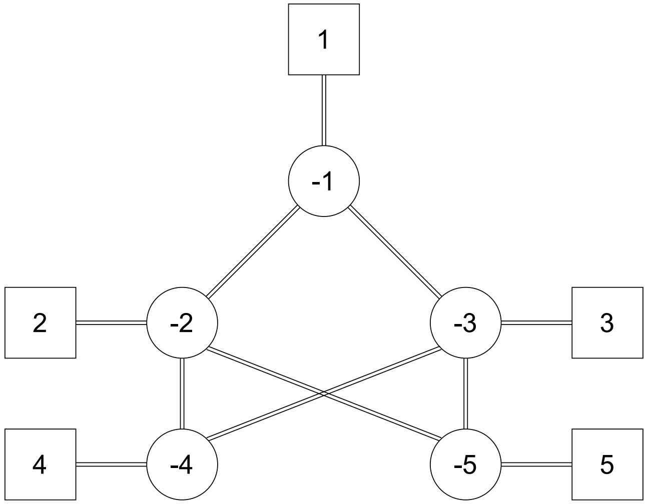


Fig. 9.5: TSN Network example.

this example, Table 9.2 shows a list of frames and their corresponding routes expressed as a list of pairs of end-station/switch identifiers indicating the link and its direction. Moreover, Table 9.3 shows how links in reverse order (from destination to source) are assigned to phases and how these links are delayed (represented by arrows) in the scheduling process until the preceding links are assigned to a previous phase. Finally, for this example, Table 9.4 shows the resulting distribution of the links in phases. The reason for scheduling the links in reverse order is that, due to the need for determinism in the reception link, this link becomes the most restrictive. This is particularly important to obtain a zero reception jitter schedule as mentioned in Section 9.4. On the other hand, note that, although this scheduler can be used in feed-forward networks, the routes cannot present triple dependencies in a loop. For example, the routes of the frames in Table 9.5 cannot be distributed in phases. The reason is that, as it can be seen in Table 9.6, the triple dependency in the loop causes a deadlock in the DivPhases algorithm as some links in the route will be indefinitely delayed.

Algorithm 3: HERMES – Schedule

```
Data: 𝓛, 𝓕_TT
Result: ∀f ∈ 𝓕_TT return f.S
procedure Schedule
    for ∀f ∈ 𝓕_TT do f.q ← 1 end
    for p = 1..Φ do
        for ∀l ∈ 𝓛 : l.ϕ = p do
            HP ← LCM({f.t : f.s.ζ = l})
            for ∀f.s ∈ f.𝓢 : f.s.ζ = l do f.s.ι ← HP/f.t end
            for ∀f ∈ 𝓕_TT, f.s_x ∈ f.𝓢 : f.s_x.ζ = l from highest to
             lowest f.u do
                for i = f.s_x.ι..1 do
                    f.s_x.o_i ← min(f.s_{x−1}.𝓞, f.d) − f.w
                    while Collision(f.s_x.o_i) ∨ Order1(f.s_x.o_i) ∨
                     Order2(f.s_x.o_i) do
                        if Collision(f.s_x.o_i) then
                            f.s_x.o_i ← f.s_x.o_i − 1
                        else
                            if Order1(f.s_x.o_i) then
                                f.q ← f.q + 1
                                if f.q > Q then
                                    f.s_x.o_i ← f.s_x.o_i − 1
                                end
                            end
                            if Order2(f.s_x.o_i) then
                                f.q ← f.q + 1
                                if f.q > Q then
                                    return Unschedulable
                                end
                            end
                        end
                        if f.s_x.o_i < 0 then
                            return Unschedulable
                        end
                    end
                end
            end
        end
    end
end
```

Frame	Route
f1	1 -1 ; -1 -2 ; -2 2
f2	2 -2 ; -2 -1 ; -1 -3 ; -3 3
f3	2 -2 ; -2 -4 ; -4 4
f4	3 -3 ; -3 -5 ; -5 5
f5	3 -3 ; -3 -1 ; -1 1
f6	3 -3 ; -3 -4 ; -4 -2 ; -2 2
f7	4 -4 ; -4 -2 ; -2 -1 ; -1 1
f8	5 -5 ; -5 -2 ; -2 -1 ; -1 1
f9	5 -5 ; -5 -3 ; -3 3

Table 9.2: Example of frame routes for Figure 9.5

Frame	ϕ 1	ϕ 2	ϕ 3	ϕ 4	ϕ 5	ϕ 6
f1	-2 2	-1 -2	1 -1			
f2	-3 3	-1 -3	-2 -1	2 -2		
f3	-4 4	-2 -4	$\rightarrow$	2 -2		
f4	-5 5	-3 -5	$\rightarrow$	$\rightarrow$	$\rightarrow$	3 -3
f5	-1 1	-3 -1	$\rightarrow$	$\rightarrow$	$\rightarrow$	3 -3
f6	-2 2	$\rightarrow$	$\rightarrow$	-4 -2	-3 -4	3 -3
f7	-1 1	$\rightarrow$	-2 -1	-4 -2	4 -4	
f8	-1 1	$\rightarrow$	-2 -1	-5 -2	5 -5	
f9	-3 3	-5 -3	$\rightarrow$	$\rightarrow$	5 -5	

Table 9.3: DivPhases procedure

Algorithm 2, AssignFrameUtilization: To decide which frames in the link will be scheduled first, the utilization of the frame along its route is calculated according to the formula $f.u = \frac{f.w}{f.d} \cdot f.n$. This formula can change depending on the needs. However, in this case, we sought to prioritize the frames that required the most bandwidth at specific time periods (between the period start and deadline) as this causes the free space for scheduling to be consumed very quickly, so it is important to prioritize them to provide them with the space they

Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6
-1 1	-1 -2	1 -1	2 -2	-3 -4	3 -3
-2 2	-1 -3	-2 -1	-4 -2	4 -4	
-3 3	-2 -4		-5 -2	5 -5	
-4 4	-3 -1				
-5 5	-3 -5				
	-5 -3				

Table 9.4: Resulting link distribution in phases

Frame	Route
f1	1 -1 ; -1 -3 ; -3 -4 ; -4 4
f2	3 -3 ; -3 -4 ; -4 -2 ; -2 -1 ; -1 1
f3	4 -4 ; -4 -2 ; -2 -1 ; -1 -3 ; -3 3

Table 9.5: Example of frame routes with feed-forward dependencies for Figure 9.5

Frame	ϕ 1	ϕ 2	ϕ 3	ϕ 4
f1	-4 4	$\rightarrow$	$\rightarrow$	...
f2	-1 1	$\rightarrow$	$\rightarrow$	...
f3	-3 3	$\rightarrow$	$\rightarrow$	...

Table 9.6: Infinite DivPhases procedure

need.

Algorithm 3, Schedule (l.2-8): Firstly, we initialize the queues by assigning queue 1 to all frames. Then, HERMES goes through all the phases and, within each phase, all the links, and calculates the hyper-period (HP) by calculating the Least Common Multiple (LCM) of all frames of each link and the number of instances of each frame in the link by dividing the HP by the frame period $f.t$. In this way, HERMES schedules phase by phase, where in each phase the links can be scheduled in parallel because they are independent. Finally, in each link, the schedule of the frames is done in order of descending

$f.u$, instance by instance from last ($f.s_x.\iota$) to the first instance in the HP. This is to prevent, in case of having frames with deadlines bigger than periods, later instances to be scheduled before previous instances.

Algorithm 3, Schedule (l.9-10): Secondly, we initialize the offset of each instance of the frame in the link to the minimum between the deadline of the frame and the release of the frame in the previous link (the following link if we consider order from source to destination) if any. Then we check if the offset assigned to the instance of the frame makes it collide with another previously scheduled instance and if the order of reception and transmission in the switch between this link and the previous one is adequate. **Algorithm 3, Schedule (l.11-34):** Finally, if there is a collision, defined as

$$\begin{aligned} &Collision(f_i.s_j.o_k) = \exists f_m.s_j.o_n : \\ &[f_m.s_j.o_n + f_m.t \cdot (n-1) \leq f_i.s_j.o_k + f_i.w + f_i.t \cdot (k-1)] \wedge \\ &[f_m.s_j.o_n + f_m.w + f_m.t \cdot (n-1) \geq f_i.s_j.o_k + f_i.t \cdot (k-1)]\,, \end{aligned} \tag{9.1}$$

the frame moves backward until it encounters an empty space. Once an empty space is found, the reception and transmission order in the switch is checked. If the frame arrives at the switch later than another frame with which shares the transmission link and which is also transmitted later than the frame under scheduling in the shared transmission link, i.e.,

$$\begin{aligned} &Order1(f_i.s_j.o_k) = \exists f_m.s_j.o_n : \\ &[f_m.s_{j-1}.\zeta = f_i.s_{j-1}.\zeta] \wedge \\ &[f_m.s_j.o_n + f_m.t \cdot (n-1) < f_i.s_j.o_k + f_i.t \cdot (k-1)] \wedge \\ &\left[f_m.s_{j-1}.o_{n'} + f_m.t \cdot (n'-1) > f_i.s_{j-1}.o_{k'} + f_i.t \cdot (k'-1)\right] \end{aligned} \tag{9.2}$$

the switch must change the queue or move backward in the schedule so that it arrives earlier than the frame instance in order conflict. On the other hand, if the frame arrives at the switch earlier than another frame that shares the transmission link and is transmitted earlier than the frame under scheduling in the shared transmission link, i.e.,

$$\begin{aligned}
&Order2(f_i.s_j.o_k) = \exists f_m.s_j.o_n : \\
&[f_m.s_{j-1}.\zeta = f_i.s_{j-1}.\zeta] \wedge \\
&[f_m.s_j.o_n + f_m.t \cdot (n-1) > f_i.s_j.o_k + f_i.t \cdot (k-1)] \wedge \\
&\left[f_m.s_{j-1}.o_{n'} + f_m.t \cdot (n'-1) < f_i.s_{j-1}.o_{k'} + f_i.t \cdot (k'-1)\right]
\end{aligned} \tag{9.3}$$

the only option is to change the queue or the network configuration will be unschedulable for the ordering approach used. Moreover, if the offset becomes negative due to the frame advance, the configuration will also be unschedulable. Note that changing the queue is not enough, it is also necessary to check that such a change does not affect the order conditions of the previously scheduled links but for the sake of simplicity, this has not been included in the algorithm.

On the other hand, in HERMES frames can be configured as RJ or ZRJ. For the sake of simplicity, this is not reflected in the algorithm but it consists of forcing all offsets to be equal on the reception link of the frames configured as ZRJ. If a frame is configured as RJ, it can be received by the receiver at different points in time even if reception is deterministic. For example, if one frame is scheduled as RJ and has a period of 4s, HERMES may schedule it in a loop of 3 instances where the first instance has an offset of 1s, the second instance has an offset of 3s, and the third instance has an offset of 2s. In this way, the instances of this frame will be received at seconds 1, 7, 10, 13, 19, and so on. However, if a frame is configured as ZRJ, it will be received by the receiver at a constant rate equal to the period. For example, if one frame is scheduled as ZRJ and has a period of 4s, HERMES schedules it in a way that the offsets of all instances are the same. In this way, if the frame has an offset of 2s the frame will be received at seconds 2, 6, 10, 14, 18, and so on. As mentioned before, this behavior is especially interesting for legacy devices that cannot implement TSN synchronization protocols but want to execute applications in a TT fashion, or want to exchange their own legacy synchronization frame with other legacy devices through TSN.

9.5 Evaluation of HERMES

9.5.1 Experimental setup

For the evaluation of HERMES, in this paper, we used the LETRA evaluation toolset (ETS) developed in [15]. LETRA ETS provides a set of integrated tools capable of performing automated experiments regarding the scheduling and schedulability analysis of TSN networks. In this section, we will explain LETRA ETS and the modifications that have been done for this paper. The reader is referred to [15] for more information about LETRA ETS.

The toolchain of the LETRA ETS used in this work is depicted in Figure 9.6. The main input to the ETS is the network configuration, including the network topology and traffic characteristics. For this paper, we use two network topologies both following a line-star topology. The network topologies are depicted in Figure 9.7 and they are a small single-switch network consisting of 3 nodes (S1) and a larger three-switch network consisting of 9 nodes (S3).

For the traffic characteristics, we set the traffic to be only TT as we exclude the effect of HERMES on lower priority traffic in this stage. The network bandwidth is set to 10Mbps to prevent generating too many frames when reaching loads around 90% utilization to avoid taking more than a week to conduct each round of experiments due to the CP scheduler and the network generator which are the two most time-consuming tools. The maximum number of generated frames is set to 100, however, depending on the selected utilization, the frame number can be different. The frame length is selected within the range [500,1000] Bytes. The minimum and maximum allowed periods are set to $200\mu s$ and $1000\mu s$ respectively, while deadlines were assigned the same values as the corresponding periods. Note that the frames will be generated such that the utilization of all links will be the one selected as an input, e.g., when we select 10% utilization the traffic generator selects the frame sizes and routes to obtain 10% on all links if possible.

The first step of LETRA ETS is generating random traffic. We used the network configuration as input of the network generator to generate 1700 sets of TT traffic randomly for each network topology (100 TT traffic configurations for each of the values of the utilization, which are taken [from 10% to 90% in steps of 5% of utilization]).

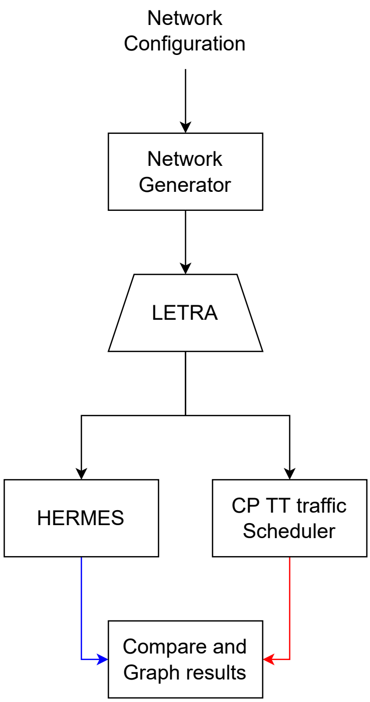
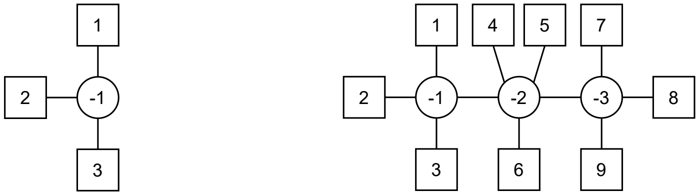


Fig. 9.6: LETRA evaluation tool set modification.

The next step is LETRA, as it can be seen in Figure 9.6, which is a mapping tool to map the generated traffic into TSN traffic classes, i.e., TT, AVB and BE. In this paper, we are only interested in TT traffic, thus, we skip the traffic mapping. However, ETS is integrated in a way that the output of each tool is the input of the next. For this reason, we used LETRA only as a translator between the output of the network generator and the input of the schedulers.

Finally, each generated network processed by LETRA is used as input to both HERMES and a CP scheduler [21]. The CP scheduler runs in the only available mode which is with one queue and RJ while HERMES, depending on the experiment, is configured with up to 4 TT queues and with the frames configured in either mode RJ or ZRJ.



(a) Single-switch network (S1).   (b) Three-switches network (S3).

Fig. 9.7: Experimental network topologies.

Once the scheduling of all the generated networks is done, we compare the two schedulers, i.e., HERMES and CP, with respect to the time that it takes for each of them to give a solution and the number of networks for which each of the schedulers is able to find a schedule (number of networks considered as schedulable by each scheduler). The experiments are done for different values of the network utilization, which is the same on all network links, e.g., 10% utilization is considered in all links of the network. We show the results for different network utilization in several graphs in the following sections. In the graphs that show the number of schedulable networks, the circles indicate the percentage of networks generated that could be scheduled while error bars represent the error in the percentage obtained through the binomial analysis with 95% certainty. Additionally, these graphs include dashed trend lines obtained through logistic regression adjustment to present the trend of changes. Moreover, Table 9.7 compares the schedulability between HERMES in modes RJ and ZRJ and the CP scheduler on networks S1 and S3. In the table, we analyze schedulability $S(u)$, as a function of the utilization u, for the range of utilizations under analysis ($u \in [10\%, 90\%]$). Since the schedulability $S(u)$ is sampled, we approximate it by a logistic regression, that we indicate with the notation $\widehat{S}(u)$. Then we define the accumulated schedulability for a network x as:

$$AS_x = \int_{0.1}^{0.9} \widehat{S}_x(u)du \tag{9.4}$$

which we use to compare schedulers using the accumulated scheduling ratio defined as:

$$ASR_{x,y}[\%] = \frac{AS_x}{AS_y} \cdot 100 \tag{9.5}$$

to measure the percentage of schedulable networks of x compared to y. On the other hand, in the graphs that show the time taken by each scheduler to give a solution, the circles show the average time needed to get the schedule in milliseconds for each utilization level, while error bars are calculated using the gamma distribution with 95% certainty. Moreover, the graphs include a dashed trend line obtained by fitting the data to an exponential function or a polynomial function of order 1 or 2.

9.5.2 Results of the scheduling time

We start with the evaluation by analyzing the time it takes to give a schedule for both network topologies. In Figs. 9.8 and 9.9 scheduling times for all HERMES modes and the number of queues as well as the scheduling time of the CP scheduler for networks S1 and S3 respectively are shown. In both graphs, we can see how the scheduling time of the CP scheduler is exponentially increasing with the percentage of utilization and the number of frames while HERMES remains with scheduling times below 10ms. This implies that HERMES exhibits scheduling times from tens of times lower than the CP scheduler to thousands of times lower for high utilization values. Furthermore, we can see how for 50% utilization the scheduling time in the S1 network for the CP scheduler is 3000ms while for the S3 network it is 8000ms, which also shows a large increase in scheduling time with the size of the network.

On the other hand, Figs. 9.10 and 9.11 show a detail of the scheduling times specifically for HERMES in RJ mode for networks S1 and S3 respectively. Both graphs show an increase in scheduling time proportional to the square of the utilization, which is related to the number of frames. On the other hand, the scheduling time is proportional to the longest path between two end-stations. In this case, as shown in Figure 9.7, the ratio between the longest routes is 4/2. For a utilization of 60%, we observe that in the S1 network HERMES with 2 and 3 queues takes 1 and 2 ms respectively while in the S3 network for the same number of queues the scheduling time is 2 and 4 ms, which corresponds to the ratio calculated above. Finally, it is also possible to identify that scheduling time doubles with every extra queue, for example, for a 60% utilization in the S3 network, HERMES takes 2, 4, and 8 ms to get a schedule with 2, 3, and 4 queues respectively. Although the complexity doubles with each extra queue used, the fact that the queues are limited to 8 reduces its impact and allows HERMES to remain scalable.

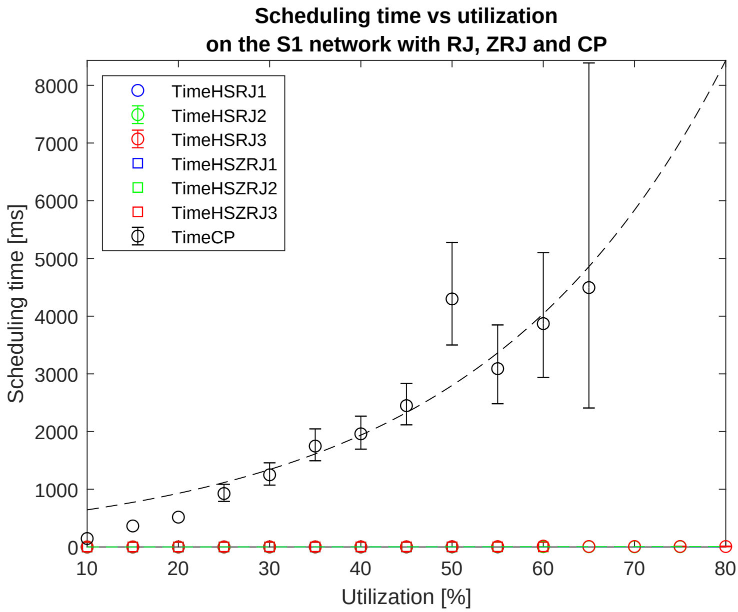


Fig. 9.8: Scheduling time for different levels of network utilization on network S1 of a CP scheduler and HERMES with and without zero reception jitter with 1, 2 and 3 queues.

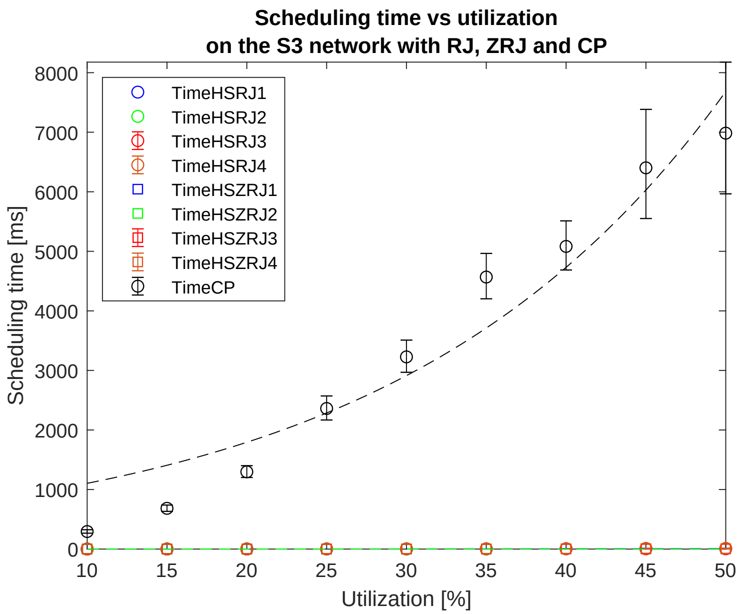


Fig. 9.9: Scheduling time for different levels of network utilization on network S3 of a CP scheduler and HERMES with and without zero reception jitter with 1, 2, 3 and 4 queues.

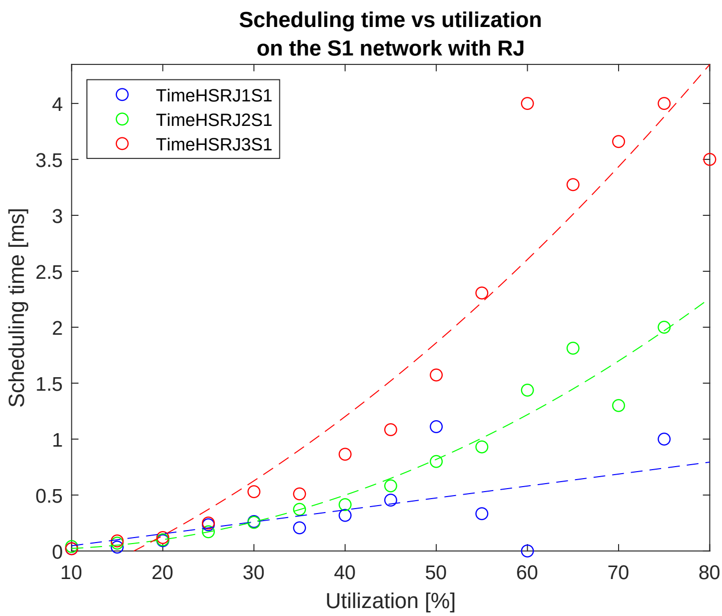


Fig. 9.10: Scheduling time for different levels of network utilization on network S1 of HERMES with reception jitter with 1, 2 and 3 queues.

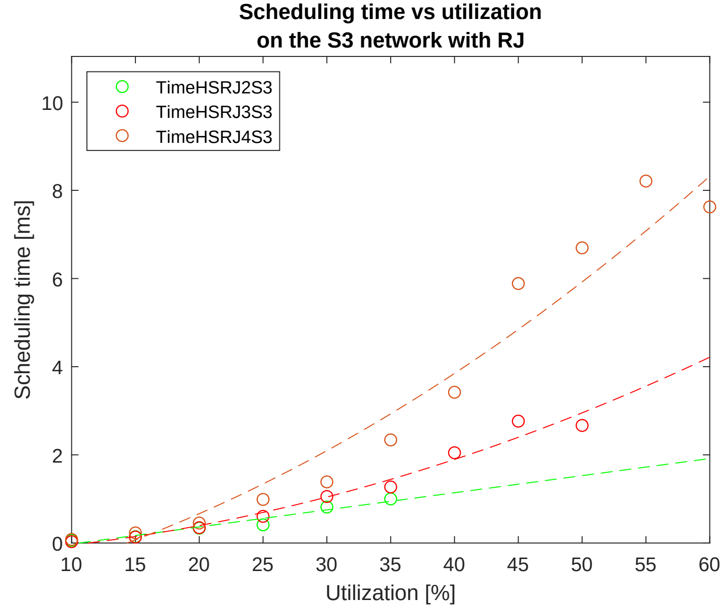


Fig. 9.11: Scheduling time for different levels of network utilization on network S3 of HERMES with reception jitter with 2, 3 and 4 queues.

9.5.3 Results of the schedulability

Figure 9.12 shows in black the schedulability of the CP scheduler for a single-switch network (S1) with the set of generated networks. On the other hand, in blue, green, and red we can see the HERMES schedulability with 1, 2, and 3 TT queues respectively in mode RJ (with jitter in the reception). The first observation is that it would be enough to increase the number of queues to 2 to obtain the same schedulability as the CP scheduler with 1 queue but, in addition, with three queues it is even possible to exceed by more than 32% the schedulability achieved by the CP scheduler, as shown in the first column of Table 9.7. These results, together with those shown in the previous subsection, show the usefulness of HERMES in contexts where the number of queues is not a constraint but the schedulability time is, e.g., run-time configurations.

Figure 9.13 shows the HERMES schedulability in zero reception jitter (ZRJ) mode for the S1 network. This graph shows that in this case, from 2 queues onwards, the schedulability stagnates due to the tough constraint that the ZRJ mode imposes. However, in the second column of Table 9.7 it can be seen how the schedulability is lower than in RJ mode, except for the case of one queue where the ZRJ mode restriction facilitates the scheduling of certain cases. Since, in general, the schedulability in ZRJ mode is lower than the schedulability in RJ mode, it is recommended to limit this mode to frames that really require it.

Figure 9.14 shows in black the schedulability of the CP scheduler for a

	Network S1		Network S3	
N° Q	$ASR_{RJ,CP}$	$ASR_{ZRJ,RJ}$	$ASR_{ZRJ,CP}$	$ASR_{ZRJ,RJ}$
1	51.04	101.58	17.37	102.59
2	98.50	80.54	55.09	104.62
3	131.99	62.73	81.59	96.33
4	–	–	101.35	81.98

Table 9.7: Schedulability comparison between HERMES with different number of queues (Q) in mode RJ and ZRJ and the CP scheduler on networks S1 and S3.

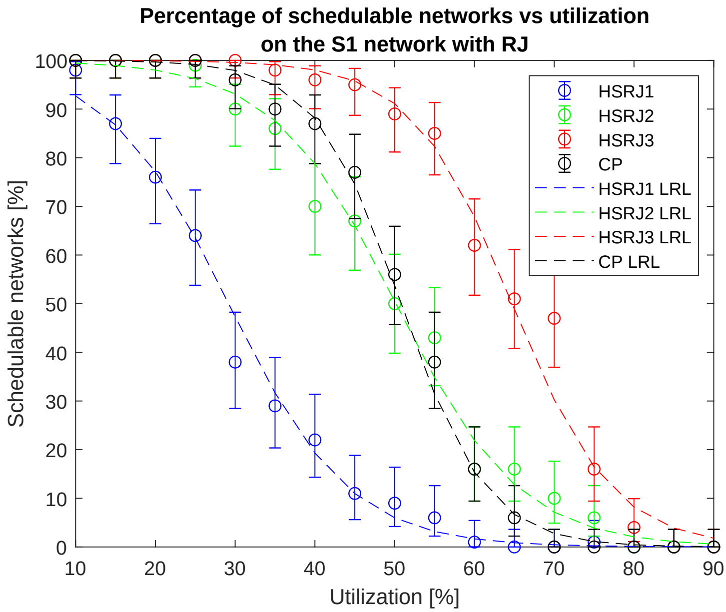


Fig. 9.12: Schedulability for different levels of network utilization on network S1 of a CP scheduler and HERMES allowing reception jitter with 1, 2 and 3 queues.

three-switches network (S3) with the set of generated networks. On the other hand, in blue, green, red, and orange we can see the HERMES schedulability with 1, 2, 3, and 4 TT queues respectively in mode RJ. In this graph, we can see how HERMES scales worse the longer the route of the frame, being necessary up to 4 queues to surpass the schedulability levels that are attainable with the CP scheduler, as shown in the third column of Table 9.7. However, it can also be noticed that with 3 queues 81% schedulability is achieved so the improvement in scheduling time can still be worthwhile. For example, different approaches can be tried to order the frames, apart from frame utilization, so that, although each has a lower schedulability, together can cover all the cases covered by the CP scheduler even in less time since different frame scheduling orders in the links will provide different schedules.

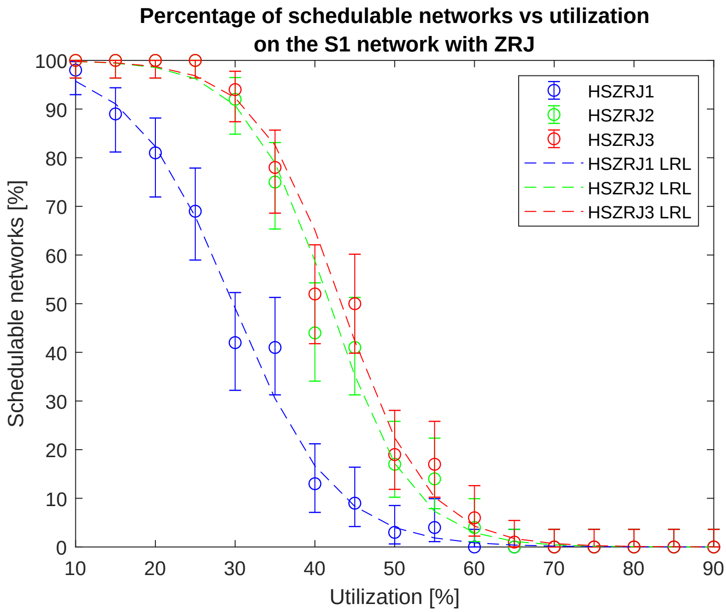


Fig. 9.13: Schedulability for different levels of network utilization on network S1 of HERMES in zero reception jitter mode with 1, 2 and 3 queues.

Figure 9.15 shows the HERMES schedulability in ZRJ mode for the S3 network. Similar to what happened in the S1 network, in this case, the schedulability in ZRJ mode also stagnates. However, the stall occurs after the third queue. On the other hand, as we can see in the fourth column of Table 9.7, with fewer queues the ZRJ constrain may slightly improve schedulability but for more queues, the difference is greater and increasing so, since more queues are needed to achieve high values of schedulability for this kind of traffic, again, this mode should be left for very specific frames if high schedulability wants to be achieved.

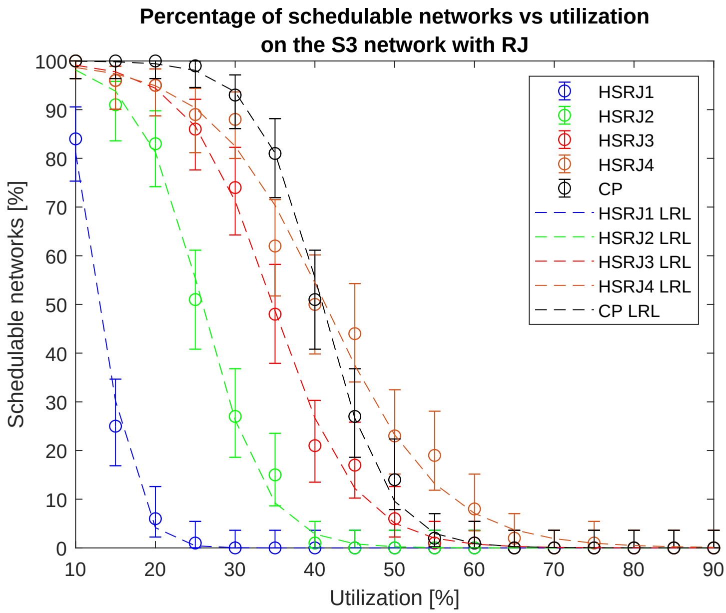


Fig. 9.14: Schedulability for different levels of network utilization on network S3 of a CP scheduler and HERMES allowing reception jitter with 1, 2, 3 and 4 queues.

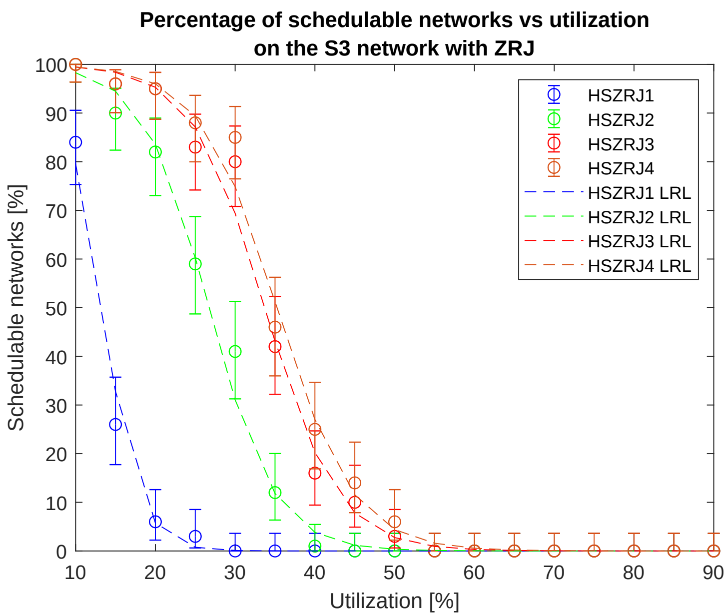


Fig. 9.15: Schedulability for different levels of network utilization on network S3 of HERMES in zero reception jitter mode with 1, 2, 3 and 4 queues.

9.6 Conclusion and Future Work

We argued that developing fast scheduling algorithms are crucial specially for adaptive and evolutionary systems. Therefore, in this work, we have developed a fast heuristic scheduler for TT traffic in TSN networks called HERMES that can match the level of schedulability of reference schedulers by using several TT queues. We also use the LETRA ETS to evaluate HERMES performance showing that by using several queues HERMES can outperform the schedulability of CP schedulers with a single queue but with HERMES exhibiting scheduling times of less than 10 ms, which implies that HERMES is hundreds or thousands of times faster. In addition, HERMES supports the integrative capability of TSN by providing a more restrictive ZRJ mode that facilitates the integration into TSN networks of legacy devices that cannot implement TSN's own synchronization mechanisms.

In this work, we focus on TT traffic scheduling. However, in previous works, we have developed a TSN mapping tool and an AVB analyzer. Therefore, the next step is to integrate all these tools to create a toolset capable of mapping and scheduling traffic taking into account the real-time requirements of all kinds of traffic.

Bibliography

[1] IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams. *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pages C1–72, 2010.

[2] IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP). *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, 2010.

[3] IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.

[4] G. Alderisi, G. Patti, and L. Lo Bello. Introducing support for scheduled traffic over IEEE audio video bridging networks. In *Conf. Emerging Technologies Factory Automation*, 2013.

[5] Inés Álvarez, Ignasi Furió, Julián Proenza, and Manuel Barranco. Design and experimental evaluation of the proactive transmission of replicated frames mechanism over time-sensitive networking. *Sensors*, 21(3):756, 2021.

[6] Emmanuel Arzuaga and David R Kaeli. Quantifying load imbalance on virtualized enterprise servers. In *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*, pages 235–242, 2010.

[7] Mohammad Ashjaei, Lucia Lo Bello, Masoud Daneshtalab, Gaetano Patti, Sergio Saponara, and Saad Mubeen. Time-sensitive networking in automotive embedded systems: State of the art and research opportunities. *Journal of Systems Architecture*, 110:1–47, September 2021.

[8] Ayman A Atallah, Ghaith Bany Hamad, and Otmane Ait Mohamed. Fault-resilient topology planning and traffic configuration for ieee 802.1 qbv tsn networks. In *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pages 151–156. IEEE, 2018.

[9] Silviu S Craciunas, Ramon Serna Oliver, Martin Chmelík, and Wilfried Steiner. Scheduling real-time communication in ieee 802.1 qbv time sensitive networks. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, pages 183–192, 2016.

[10] Voica Gavriluţ and Paul Pop. Scheduling in time sensitive networks (tsn) for mixed-criticality industrial applications. In *14th IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 1–4. IEEE, 2018.

[11] Voica Gavriluţ, Luxi Zhao, Michael L Raagaard, and Paul Pop. Avb-aware routing and scheduling of time-triggered traffic for tsn. *IEEE Access*, 6:75229–75243, 2018.

[12] S. Kehrer, O. Kleineberg, and D. Heffernan. A comparison of fault-tolerance concepts for IEEE 802.1 Time Sensitive Networks (TSN). In *IEEE Emerging Technology and Factory Automation*, 2014.

[13] Leonard Kleinrock. Queueing systems, volume i: Theory, vol. i, 1975.

[14] Lucia Lo Bello, Mohammad Ashjaei, Gaetano Patti, and Moris Behnam. Schedulability analysis of time-sensitive networks with scheduled traffic and preemption support. *Journal of Parallel and Distributed Computing,*, 144,, 2020.

[15] Daniel Bujosa Mateu, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, and Thomas Nolte. Letra: Mapping legacy ethernet-based

traffic into tsn traffic classes. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2021.

[16] Maryam Pahlevan and Roman Obermaisser. Genetic algorithm for scheduling time-triggered traffic in time-sensitive networks. In *2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA)*, volume 1, pages 337–344. IEEE, 2018.

[17] Maryam Pahlevan, Nadra Tabassam, and Roman Obermaisser. Heuristic list scheduler for time triggered traffic in time sensitive networks. *ACM Sigbed Review*, 16(1):15–20, 2019.

[18] Rina Panigrahy, Kunal Talwar, Lincoln Uyeda, and Udi Wieder. Heuristics for vector bin packing. *research. microsoft. com*, 2011.

[19] Michael Lander Raagaard and Paul Pop. Optimization algorithms for the scheduling of ieee 802.1 time-sensitive networking (tsn). *Tech. Univ. Denmark, Lyngby, Denmark, Tech. Rep*, 2017.

[20] Mauricio GC Resende and Celso C Ribeiro. Grasp: Greedy randomized adaptive search procedures. In *Search methodologies*, pages 287–312. Springer, 2014.

[21] Niklas Reusch, Silviu S Craciunas, and Paul Pop. Dependability-aware routing and scheduling for time-sensitive networking. *arXiv preprint arXiv:2109.05883*, 2021.

[22] R. Salazar, T. Godfrey, L. Winkel, N. Finn, C. Powell, B. Rolfe, and M. Seewald. Utility Applications of Time Sensitive Networking White Paper (D3). Technical report, IEEE, 2018.

[23] S. Samii and H. Zinner. Level 5 by Layer 2: Time-Sensitive Networking for Autonomous Vehicles. *IEEE Communications Standards Magazine*, 2(2):62–68, 2018.

[24] Eike Schweissguth, Dirk Timmermann, Helge Parzyjegla, Peter Danielis, and Gero Mühl. Ilp-based routing and scheduling of multicast realtime

traffic in time-sensitive networks. In *2020 IEEE 26th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–11. IEEE, 2020.

[25] Ammad Ali Syed, Serkan Ayaz, Tim Leinmüller, and Madhu Chandra. Dynamic scheduling and routing for tsn based in-vehicle networks. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2021.

[26] Marek Vlk, Kateřina Brejchová, Zdeněk Hanzálek, and Siyu Tang. Large-scale periodic scheduling in time-sensitive networks. *Computers & Operations Research*, 137:105512, 2022.

[27] M. Wollschlaeger, T. Sauter, and J. Jasperneite. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Industrial Electronics Magazine*, 11(1):17–27, 2017.

[28] Jin Y Yen. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716, 1971.

[29] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. Timing analysis of avb traffic in tsn networks using network calculus. In *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 25–36, 2018.

Chapter 10

Paper C
An Improved Worst-Case Response Time Analysis for AVB Traffic in Time-Sensitive Networks.

Daniel Bujosa, Julián Proenza, Alessandro V. Papadopoulos, Thomas Nolte, Mohammad Ashjaei.
In the proceedings of the 45^{th} IEEE Real-Time Systems Symposium (RTSS 2024).

Abstract

Time-Sensitive Networking (TSN) has become one of the most relevant communication networks in many application areas. Among several traffic classes supported by TSN networks, Audio-Video Bridging (AVB) traffic requires a Worst-Case Response Time Analysis (WCRTA) to ensure that AVB frames meet their time requirements. In this paper, we evaluate the existing WCRTAs that cover various features of TSN, including Scheduled Traffic (ST) interference and preemption. When considering the effect of the ST interference, we detect optimism problems in two of the existing WCRTAs, namely (i) the analysis based on the busy period calculation and (ii) the analysis based on the eligible interval. Therefore, we propose a new analysis including a new ST interference calculation that can extend the analysis based on the eligible interval approach. The new analysis covers the effect of the ST interference, the preemption by the ST traffic, and the multi-hop architecture. The resulting WCRTA, while safe, shows a significant improvement in terms of pessimism level compared to the existing analysis approaches relying on either the concept of busy period or the Network Calculus model.

10.1 Introduction

Since its inception in 2012, Time-Sensitive Networking (TSN) has become one of the most relevant sets of standards for real-time communications. TSN provides Ethernet with various features, including precise clock synchronization, traffic flexibility, fault tolerance, and advanced network management. These characteristics make TSN relevant to several application areas, such as automation [18], automotive [12], and power distribution [10].

This paper focuses on two of TSN's most relevant features: its traffic flexibility and the real-time features it provides. Traffic flexibility allows mixing traffic with different characteristics and levels of real-time requirements on the same network. Traditionally, this has been addressed using several networks with different characteristics connected through gateways with limited inter-network connectivity. However, thanks to TSN, this separation is no longer necessary, as a single network can handle all types of traffic.

To achieve this, TSN defines three types of traffic: Scheduled Traffic (ST), Audio-Video Bridging Traffic (AVB), and Best-Effort Traffic (BE). ST can meet its requirements by construction as it is scheduled offline, and BE traffic has no Real-Time (RT) requirements. However, AVB traffic, despite being predictable, requires a Worst-Case Response Time Analysis (WCRTA) to determine whether it can meet its time requirements. WCRTAs can be optimistic, exact, or pessimistic. An optimistic analysis is unacceptable, as it may fail to meet the time requirements, potentially leading to catastrophic consequences in critical systems. In contrast, a pessimistic WCRTA, albeit safe, leads to wasted resources.

An exact WCRTA is preferred for analysis of any real-time systems; however, it is challenging to achieve and often impossible due to considering worst-case scenarios. To the authors' best knowledge, currently, there is no exact WCRTA for TSN's AVB traffic. Nevertheless, providing a WCRTA with as little pessimism as possible is essential when the exact analysis is unavailable. This paper proposes a WCRTA and shows that it is less pessimistic than the existing WCRTA for AVB traffic in literature.

Contributions: In this paper, we closely evaluate all the existing AVB WCRTAs that support AVB traffic analysis under ST interference with and without preemption of the ST since this is the most common configuration. Note that there are other WCRTAs that compute the Worst-Case Response Times (WCRT) of AVB traffic without ST interference or preemption, or WCRTAs of TSN, such as [15], that do not consider AVB traffic. Those works are considered limited in their capacity to support the TSN features, thus out of the scope of this evaluation. To the best of our knowledge, there are only two AVB WCRTAs in the literature that consider ST interference and preemption according to the standards, being an analysis based on the busy period calculation [11] and an analysis based on the Network Calculus model [19]. There is also a work based on the notion of eligible interval [4, 3, 5] that shows a significant improvement in terms of reducing pessimism compared to other analyses, but that is limited to a single TSN output port and that does not consider the ST interference. Further, there was an attempt to add the ST interference without preemption to the eligible interval analysis [14]. However, this paper shows that the approach presented in [14] contains a potential optimism in corner cases. The concrete contributions in this paper are as follows:

- First, we show that although the eligible interval approach for AVB traffic analysis presents less pessimism than the existing analysis approaches, the extended analysis presented in [14] to include the ST interference contains optimism in corner cases.
- Second, we show that the WCRTA based on the busy period calculation presented in [11] also contains optimism in calculating the ST interference, potentially making the analysis unsafe.
- Third, we propose a new calculation of the ST interference, solving the optimism problems in both analysis approaches.
- Fourth, we present a new WCRTA which extends the one based on the eligible interval approach [4, 3, 5] to consider ST interference with and without preemption and the effect of the multi-hop behavior instead of being limited to a single output port.
- Finally, we compare the three WCRTAs, i.e., the analysis based on the busy period calculation [11], the Network Calculus model [19], and the

new analysis presented in this paper, for two different network sizes. We show that our analysis, while safe, significantly outperforms the other two by exhibiting much less pessimism, on average around 100% compared to the busy period [11] and 15% compared to the Network Calculus model [19].

Paper organization The paper is organized as follows. Section 10.2 introduces the concepts and features of TSN. Section 10.3 reviews the related works, while Section 10.4 defines the system model. Section 10.5 summarizes the WCRTA based on the eligible interval, and Section 10.6 extends it by adding the proposed ST interference calculation and the multi-hop behavior effect. Section 10.7 presents the experimental setup, while Section 10.8 shows the results. Finally, Section 10.9 concludes the paper.

10.2 Recap on TSN Traffic Shapers

In TSN, communication between end-stations relies on transmitting Ethernet frames through Ethernet links and TSN switches. TSN switches and end-stations output ports have up to 8 First-In-First-Out (FIFO) queues. Each queue corresponds to one of the 8 priorities available in TSN. Each of these queues can apply different traffic shapers. Depending on the queue configuration, it can handle one of the three types of TSN traffic (ST, AVB, or BE). As many queues as needed from the 8 available priorities can be allocated to each type of traffic, resulting in more than one class in AVB and BE.

Fig. 10.1 shows an example of a simplified 4-queue output port consisting of an ST queue, two AVB queues (class A and B), and a BE queue. The following sections will explain each TSN traffic shaper and how they are configured.

10.2.1 Time-Aware Shaper

ST is pre-scheduled offline to ensure precise transmission with high accuracy and zero jitter. To meet the determinism requirements of ST and prevent inter-frame interference, TSN employs the Time-Aware Shaper (TAS) mechanism. TAS operates by isolating queues eligible for transmission and assigning to each queue a gate that can be toggled between open and closed states. The state of the gate is determined by the Gate Control List (GCL), specifying, at the nanosecond level, the duration for which a gate should be open or closed in a cyclically repeating list. The cycle length is determined by the hyper-period of the ST frames transmitted through the corresponding output port. This is a limitation of the ST traffic due to the potential for certain period combinations to cause an explosion in the length of the GCL, underscoring the significance of AVB traffic. An open gate allows traffic transmission from the queue, whereas a closed gate prevents frames from being transmitted. The open gate period is a *transmission window*. For a scheduled transmission of an ST frame at a designated time, the frame must reside in the assigned queue, and the gate of that queue must remain closed until the scheduled transmission time. Before the transmission window starts, all gates of other queues are closed to prevent interference. This time when all gates are closed is called *guard band*. Then, the ST queue gate opens precisely at the scheduled time for the transmission of the ST frame. This mechanism is referred to as the HOLD/RELEASE mechanism in the standards.

Additionally, TSN can be configured so that ST traffic can preempt the non-scheduled traffic (AVB and BE traffic), commonly known as non-ST traffic. That is, the transmission of the non-ST frame halts temporarily and resumes after the ST frame completes its transmission. The guard band size varies based on whether TSN is configured with or without preemption. With preemption, the guard band needs to exceed the minimum preemptible message size (124 Bytes). Whereas, without preemption, the guard band must be greater than the maximum size of a TSN frame, which is 1518 Bytes. Alternatively, not adding guard bands would result in jitter for the ST equivalent to the transmission time.

Fig. 10.1 shows an example with two ST frames (1 and 4). This example does not include guard bands or preemption for simplicity. According to the GCL, these frames are scheduled to be sent at times T0 and T3, respectively. Thus, as depicted in the figure, at time T0, the ST gate is open, and all other gates are closed, enabling the transmission of frame 1 without interferences. Then, between T1 and T3, the ST gate is closed, and the other gates are open, enabling the transmission of the non-ST traffic. At T3, the ST gate is reopened, and the other gates are closed to allow the transmission of frame 4 without

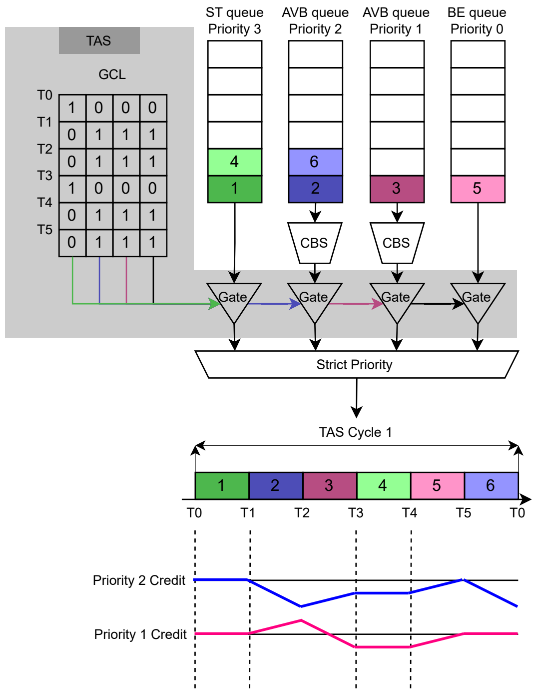


Fig. 10.1: Example of TSN output port.

interference. Finally, the ST gate is closed, and the non-ST gates are opened to enable the transmission of the remaining non-ST frames.

10.2.2 Credit-Based Shaper

While ST is transmitted according to a fixed schedule with zero jitter, its reliance on off-line scheduling considerably reduces its effectiveness and efficiency in handling other traffic types, such as event-triggered traffic. Additionally, the cost of obtaining a schedule substantially increases with the number of frames to be scheduled and may not always be affordable or even feasible. This is where non-ST traffic comes into play. Non-ST can be transmitted when its gate is open, i.e., when no ST frames are transmitted.

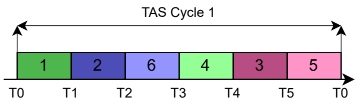


Fig. 10.2: TSN output port transmission considering only TAS and strict priority.

Without additional mechanisms, non-ST traffic would be subject to arbitration through strict priority, severely impacting the quality of service for lower-priority queues and significantly impairing their real-time response guarantees. Fig. 10.2 illustrates the transmission outcome under arbitration solely by TAS and strict priority. Frames from lower-priority queues experience delayed transmission, with the delay increasing as the queue's priority decreases. In the most extreme scenario, a high-priority queue may indefinitely interfere with low-priority queues, causing an infinite delay in their transmissions.

TSN incorporates a Credit-Based Shaper (CBS) mechanism to overcome this limitation. The traffic assigned to queues implementing CBS is the AVB traffic, while the lowest-priority queue with no CBS is the BE traffic. CBS limits the maximum percentage of bandwidth a queue can utilize, ensuring a minimum bandwidth allocation for lower-priority queues. In CBS, credit is assigned to each AVB queue. The credit is progressively replenished when an AVB frame awaits to be transmitted or the credit is negative and is consumed when a frame is being sent. The replenishment and consumption rates of the credit are denoted by the terms *idleSlope* and *sendSlope*, respectively. A queue is eligible for transmission only if its credit is zero or a positive value. According to the standard, the credit of a queue freezes if its gate is closed.

Additionally to the ST queue, the example in Fig. 10.1 includes two AVB queues and one BE queue. Two frames (2 and 6) are assigned to the higher-priority AVB queue, while the lower-priority AVB queue and the BE queue present a single frame (frames 3 and 5, respectively). At time T1, following the transmission of the ST frame, both AVB queues become eligible for transmission. Consequently, the strict priority mechanism selects the priority 2 AVB queue, leading to the transmission of frame 2. While the frame is transmitted, its credit is consumed. At the same time, the credit of the lower-priority AVB queue is replenished as it awaits transmission.

Moving to T2, the higher-priority AVB queue, having negative credit, becomes ineligible for transmission. Subsequently, the lower-priority AVB queue is chosen, resulting in the transmission of frame 3. Remarkably, this occurs even when a higher-priority non-ST frame (frame 6) awaits the transmission. Similarly to T1, priority 1 credit is consumed during the transmission of frame 3, while the higher-priority AVB credit is replenished.

At T3, another ST frame is transmitted, and the gates of the AVB queues close, causing their credit to freeze. By T4, the credit of both AVB queues is negative, prompting the BE queue to be selected for transmission. This choice occurs even though a higher-priority AVB frame awaits the transmission. Finally, at T5, the credit of the higher-priority AVB queue becomes zero, allowing for the transmission of the last frame (frame 6). Note that BE has no timing requirement and can only be transmitted when its gate is open, and no other higher-priority frame is eligible to be transmitted.

10.3 Related Work

Since the introduction of the AVB standard in 2011, numerous papers have delved into the challenge of analyzing its WCRT. However, almost all of them can be categorized into three distinct approaches, including busy period analysis, Network Calculus model, and eligible interval analysis.

The busy period analysis relies on finding the critical instant producing the WCRT. Diemer et al. [8] introduced one of the first AVB WCRTAs based on busy period analysis. Nevertheless, it was constrained to a single output port and a single AVB queue and did not account for ST interference. Subsequently, the work in [9] extended the previous work to include the analysis of two AVB queues (class A and class B). However, the consideration of ST was still omitted and confined to only one output port. Finally, in the work by Lo Bello et al. [11], one of the last AVB WCRTA based on busy period analysis is introduced. Apart from considering interference with other AVB classes, this paper also incorporates the impact of ST interference with and without preemption of non-ST traffic and the blocking of lower-priority frames. Moreover, it includes a multi-hop calculation, i.e., it is not constrained to a single output port. However, it is important to note that the paper is limited to two AVB classes (Class A and B). Furthermore, this paper shows that the analysis contains optimism in the ST interference calculation.

The second approach leverages Network Calculus, a theoretical framework designed for the performance analysis of communication networks. Network Calculus allows for calculating the maximum delay each frame may experience, i.e., its WCRT. Zhao et al. [20, 19] introduce the WCRTA of AVB implemented using Network Calculus. This method enables the calculation of the WCRT from the transmitter to the receiver for multiple AVB queues considering ST interference with and without preemption of non-ST traffic. However, limitations in Network Calculus arise when dealing with loop networks that involve circular dependencies. Furthermore, this approach allocates the same bandwidth for an AVB class on all links, reducing flexibility in configuration and potentially leading to situations where more bandwidth than necessary is reserved. This solution, along with the one presented in [11], is unique in providing WCRT from the transmitter to the receiver, encompassing ST interference and preemption of non-ST traffic by the ST. However, they have never been compared as they were developed in parallel and published at almost the same time. As part of the experiments in this paper, we also compare these two analysis approaches. While other WCRTAs based on Network Calculus exist, they do not meet the criteria for inclusion in this paper. For instance, [16] introduces a WCRTA based on Network Calculus designed for a highly specific setup featuring a single transmission window for each traffic type. Moreover, it restricts the AVB Class A and B gates from being opened simultaneously, significantly constraining the arbitration capabilities of CBS.

The third and final approach defines the time interval in which each frame can be eligible to be transmitted (*eligible interval*). The WCRT for a frame can be calculated by analyzing the maximum interference it may encounter throughout its eligible interval. Bordoloi et al. [1] introduced one of the earliest versions of the WCRTA, although the term had not yet been established. In the work by Cao et al. [4, 3, 5], the authors extend Bordoloi et al. [1] AVB's WCRTA and introduce the term eligible interval. They demonstrate that the maximum delay an AVB frame can experience, i.e., its WCRT, depends solely on the maximum credit attainable by its AVB class, while this maximum credit is bounded and can be calculated. This analysis makes WCRTs notably less pessimistic than those based on the busy period analysis or the Network Calculus model. Moreover, this analysis can be applied to any number of AVB classes. However, this analysis does not consider the interference caused by ST and is limited to a single output port. Maxim et al. [14] extended the AVB's WCRTA based on the eligible interval to incorporate ST interference, yet without preemption. However, as we will see later in this paper, the solution contains optimism when considering the ST interference.

As reviewed above, the existing analysis approaches either exhibit limitations or contain optimism. Therefore, this paper addresses such limitations and proposes a new ST interference calculation on the AVB traffic on multi-hop networks that results to a significantly less pessimistic analysis compared to the state-of-the-art analysis solutions.

10.4 System Model

10.4.1 Network model

In the network topology, unidirectional connections between any end-station and any TSN switch and between two TSN switches are done through *link*s denoted by l. The TSN ports are considered full-duplex, i.e., the inputs and outputs are isolated. Therefore, reception and transmission do not interfere on the same physical port. This means that for each physical port, there are two links, one for sending and one for receiving. Moreover, the time that elapses from the reception of a frame through a TSN switch port until it is inserted into the queue at the output port is specific to each TSN switch and is denoted by ϵ. The link delay due to the wire and its physical characteristics is assumed to be negligible for the analysis, and the total network bandwidth is denoted by BW and is the same in all network links. Finally, for AVB traffic of priority X, on a link l, the credit replenishment rate (idleSlope) is denoted by $\alpha^{+}_{X,l}$, the credit consumption rate (sendSlope) is indicated as $\alpha^{-}_{X,l}$, and the credit value is denoted by $CR_{X,l}$.

10.4.2 Traffic model

In this paper, we use the real-time periodic model for all traffic. This model defines frames as a sequence of frame instances that share common characteristics, such as source and destination addresses, periods, deadlines, etc. In this sense, a set of N frames is characterized as follows:

$$\Gamma = \{m_i(C_i, T_i, D_i, P_i, \mathcal{L}_i, \mathcal{O}_i) | i = 1, \ldots, N\} \tag{10.1}$$

In this model, C_i represents the transmission time of a m_i frame, which depends on the frame size and network bandwidth. Moreover, the header of each Ethernet frame is included in the frame transmission time C_i, and the header transmission time is denoted by v. This value will be considered constant and independent of the frame class. Regarding the ST, the guard band is also included in C_i when necessary. Note that when multiple ST frames are transmitted consecutively, only the first frame requires a guard band. Moreover, T_i and D_i denote the period and relative deadline of the frame. We also assume a constrained deadline model, i.e., $D_i \leq T_i$. Note that the AVB traffic classes can be initiated periodically or sporadically. In the case of sporadic transmissions, T_i represents the minimum inter-arrival time, i.e., the minimum time between initiations. The traffic priority of a frame is denoted by $P_i \in \mathbb{P}$. The priority level is the highest for the ST, the lowest for BE, and all middle ones are for AVB. In this regard, ST priority is denoted by P_{ST}, BE priority is denoted by P_{BE}, and AVB priorities are denoted by ($\{P \in \mathbb{P} \mid P_{ST} > P > P_{BE}\}$). Moreover, $\mathbb{L}$ and $\mathbb{H}$ are the sets of non-ST frames with lower and higher priority than m_i, respectively. As a frame may cross several links, the set of n links that m_i traverses is specified by $\mathcal{L}_i = \{l_1, \ldots, l_n\}$.

Offsets are used to accommodate ST frames in the transmission schedule. The offset for each ST frame is defined per link, and the set of offsets for all links that m_i crosses is specified by $\mathcal{O}_i$, e.g., $\mathcal{O}_i = \{O_i^l\}$. We assume that the offsets are given, as scheduling optimization for ST frames is out of the scope of this paper. Other works have already addressed this topic, such as [2]. Note that no offset is defined for AVB frames; hence, for AVB frames, $\mathcal{O}_i$ is an empty set, i.e., $\mathcal{O}_i = \{\emptyset\}$.

In this model, we consider the TSN configuration in which ST traffic can preempt the non-ST traffic. This implies that certain frames might experience interruptions during their transmission, only to be resumed later. Secondly, ST frames arriving at a switch will stay in the output queue until their transmission window becomes active. Specifically, the traffic classes can be classified as *express* or *preemptable*. Our system model considers the ST frames to be express, whereas all the other classes are preemptable. This means that the guard band can be sized as 124 Bytes since this is the non-preemptable segment of any non-ST frame, i.e., is the maximum time a preemptable frame (non-ST frame) can block an express frame (ST frame). This is the most common TSN configuration when all traffic types are considered.

Finally, when an ST frame preempts a non-ST frame, it is resumed after preemption but with a new header. This means that when a frame is preempted and is split into two segments, the second segment also gets a header, i.e., two headers should be accounted for. This affects the analysis for AVB traffic, which will be discussed later in Section 10.6.

10.5 Worst-Case Response Time Analysis

This section will give a detailed overview of the WCRTA based on the eligible interval [4, 3, 5]. As mentioned above, this WCRTA only considers interference from AVB frames of higher and same priority, as well as lower-priority frames blocking, including lower-priority AVB traffic and BE traffic. In addition, it considers only one output port. Therefore, in the next section, we will extend this analysis to consider interference with ST and the contribution of multi-hop behavior.

10.5.1 WCRTA overview

We will begin by introducing each of the sources of delay that an AVB frame m_i in link l may experience according to the WCRTA based on the eligible interval. First, given that all queues in TSN are FIFO, a frame's transmission depends on the preceding frames within the same queue. Therefore, the first source of delay comes from interference with the same-priority traffic, denoted by SPI_i^l. Secondly, higher-priority AVB classes can also interfere with lower-priority

AVB traffic, denoted by HPI_i^l, albeit to a restricted extent, thanks to the CBS mechanism. Finally, since AVB Classes are not preemptive, if frames from the lower-priority classes are being transmitted, an AVB frame should wait for its entire transmission. This entails a blocking by the lower-priority traffic, denoted by LPI_i^l. Summarizing, both the higher- and same-priority AVB traffic, as well as the lower-priority traffic (lower-priority AVB traffic and BE traffic), must be considered to compute the WCRT of any AVB frames, i.e., for a frame m_i crossing link l, the $WCRT_i^l$ considering only non-ST interference is:

$$WCRT_i^l = HPI_i^l + SPI_i^l + LPI_i^l + C_i \tag{10.2}$$

In the following subsections, we present the worst-case scenarios for each of the above components that build the WCRT for an AVB frame m_i on link l.

10.5.2 Same Priority Interference

According to [4, 3, 5], the delay experienced by an AVB frame m_i in link l interfered solely by its same-priority traffic, i.e., $sp(m_i) = \{m_j | P_j = P_i, j \neq i\}$, is calculated as a basic FIFO schedule. However, due to CBS behavior, not only the interference corresponding to the transmission of each same-priority frame C_j must be considered, but also the time required to recover the credit consumed by such interfering frame should be considered. This occurs because when only same-priority interference is considered, the credit can only increase when negative. The credit cannot become positive in this case because no other traffic is interfering with or blocking the traffic with the same priority as the frame under analysis. This behavior follows the CBS mechanism. As a result, the maximum attainable credit is zero, i.e., the credit decreases when a same-priority frame is being transmitted and increases only until zero. Consequently, the interference of every same-priority frame m_j will drop the credit to a negative value equal to $C_j \times \alpha^-_{P_i,l}$. Since an AVB class cannot transmit unless its credit is positive or zero, the next same-priority frame in the queue has to wait for the credit to recover. The recover time is calculated by $C_j \times \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}$. Therefore, the interference is the summation of the interfering frame transmission time and the time to recover the credit. In the worst case, frame m_i is interfered by all the same-priority frames. Since we consider the deadline-constrained model, each same-priority frame in the FIFO queue can interfere with m_i only once if all frames meet their deadlines, as it is already discussed in the Controller Area Network (CAN) domain [7]. In this context, as is common with most deadline-constrained analyses, the result lacks reliability if the WCRT exceeds the deadline of at least one same-priority frame. Therefore, the interference of the same-priority frames on m_i belonging to the class P_i on link l is calculated as in Eq. (10.3).

$$SPI_i^l = \sum_{\substack{\forall m_j \in sp(m_i), i \neq j \\ \wedge l \in \mathcal{L}_j}} C_j \times \left(1 + \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}\right) \tag{10.3}$$

10.5.3 Higher-Priority AVB Interference and Lower-Priority Blocking

Although the higher-priority AVB interference and the lower-priority blocking are two different contributions, the authors in [5] demonstrated that these delay contributions correspond to the time needed to reach the maximum credit CR^{max} achievable by the AVB class of the frame under analysis m_i with priority P_i, i.e.

$$HPI_i^l + LPI_i^l = \frac{CR^{max}_{P_i,l}}{\alpha^+_{P_i,l}} \tag{10.4}$$

The authors also proved that $CR^{max}_{P_i,l}$, and therefore the $HPI_i^l + LPI_i^l$, is bounded as long as the sum of the bandwidth allocated to P_i and all the higher priority queues $\mathbb{H} = \{H \in \mathbb{P} \mid ST > H > P_i\}$ does not exceed the available bandwidth. i.e.:

$$\sum_{\forall P \in \mathbb{H} \cup P_i} \alpha^+_{P,l} \leqslant BW. \tag{10.5}$$

Under these conditions, the non-ST interference $HPI_i^l + LPI_i^l$ that a frame can experience is computed as below:

$$
\begin{aligned}
HPI_i^l + LPI_i^l &= \frac{CR_{P_i,l}^{max}}{\alpha_{P_i,l}^{+}} \\
&= C_{\mathbb{L},l}^{max} \times \left(1 + \frac{\alpha_{\mathbb{H},l}^{+}}{\alpha_{\mathbb{H},l}^{-}}\right) - \frac{CR_{\mathbb{H},l}^{min}}{\alpha_{\mathbb{H},l}^{-}}
\end{aligned}
\tag{10.6}
$$

where $C_{\mathbb{L},l}^{max}$ is the size of the largest frame of all lower priority queues $\mathbb{L} = \{L \in \mathbb{P} \mid L < P_i\}$ and $CR_{\mathbb{H},l}^{min}$ is the minimum value that the joint credit of the highest priority queues can reach in link l. The latter value is calculated recursively as follows:

$$
\begin{aligned}
&CR_{\mathbb{H}=\{H_1,\ldots,H_n\},l}^{min} = \\
&\quad -\max(\alpha_{\mathbb{H},l}^{-} \times C_{H_1,l}^{max} - CR_{\mathbb{H}-H_1,l}^{min}, \\
&\quad \ldots, \alpha_{\mathbb{H},l}^{-} \times C_{H_n,l}^{max} - CR_{\mathbb{H}-H_n,l}^{min})
\end{aligned}
\tag{10.7}
$$

10.6 Proposed ST Interference

This section outlines the primary contributions of our work. Firstly, we introduce a novel method for calculating ST interference. Secondly, we incorporate this calculation along with the delay associated with multi-hop behavior into the WCRTA based on the eligible interval.

10.6.1 ST Interference

Excluding the WCRTAs based on the Network Calculus model, there have been 2 main attempts to calculate the ST interference on the AVB frames.

The work in [14] introduced a modified version of the WCRTA based on the eligible interval for a single link, considering the presence of ST. However, this study did not establish the independence of the eligible interval from ST interference. Additionally, the paper suggested that the maximum ST interference an AVB frame might experience is limited to all ST transmissions in a TAS Cycle.

$$
STI^l = \sum_{\forall j \in ST} C_j \tag{10.8}
$$

Nonetheless, this viewpoint is optimistic as nothing prevents an AVB frame from being interfered by more than one TAS cycle, as we see below.
Optimism in ST interference calculation in [14]: Consider an output port with a single ST frame of C = 1 tu and T = 2 tu and two AVB frames (frames 2 and 3) of C = 1 tu and T = 4 tu. Note that while the analysis considers two AVB queues (Class A and B), a single AVB queue example suffices to illustrate the optimism problem in the analysis. Regarding the GCL, i.e., the TAS cycle, it will be 2 tu long because there is only one ST frame with T=2 tu (TAS cycle equals the hyper-period of all ST frames). In each TAS cycle, 1 tu is allocated to the ST frame transmission and one is free for an AVB frame transmission. In this example, since there is only one AVB queue, we can assume that the credit is 100% for this queue, i.e., $\alpha^+_{P_0} = 1$. Therefore, no credit is consumed, hence there is no need to wait for its replenishment. According to [14], the delay experienced by either of the two AVB frames would be:

$$WCRT_i^l = STI_i^l + SPI_i^l + C_i = 1 + 1 + 1 = 3tu \tag{10.9}$$

However, as shown in Fig. 10.3, frame 3 experiences a WCRT:

$$WCRT_3^l = STI_3^l + SPI_3^l + C_3 = 2 + 1 + 1 = 4tu \tag{10.10}$$

The analysis in [14] considers the ST interference in this example 1 tu, while it should be 2 tu, thus imposing optimism.

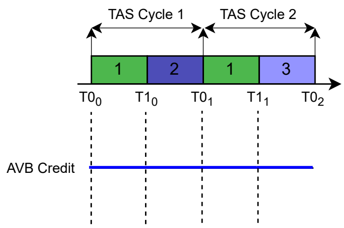


Fig. 10.3: Example of optimism on ST interference with AVB traffic.

Besides the work introduced above, the work in [11] proposed that the ST interference calculation consists of identifying the critical instant that generates the maximum interference on the AVB frame under analysis which in turn creates the maximum WCRT.

As proven in their work, the starting time of every ST transmission window in the hyper-period is a critical instant candidate. Since each link has a distinct hyper-period of length Ω_l, and given that the hyper-period of a set of frames is determined by the least common multiple of their periods, the instants to consider for the ST interference of $m_j \in ST$ on link $l \in \mathcal{L}_j$ are:

$$I_j^l = \{(k-1)T_j + O_j^l \ : k = 1, \ldots, n \ , \ n = \frac{\Omega_l}{T_j}\} \tag{10.11}$$

After obtaining all potential critical instants across the hyper-period on link l, it is necessary to calculate the phase difference between each ST frame $m_j \in ST$ and each potential critical instant $I_c^l[k]$. These phase differences are the offsets that the different ST frames would have if the start of the hyper-period were at the critical instant candidate. For more information and the proofs, the reader is referred to [13].

$$\Phi_{jc[k]}^l = (O_j^l - I_c^l[k]) \bmod T_j \tag{10.12}$$

Finally, for each critical instant candidate $I_c^l[k]$, the amount of ST interference that an AVB frame would experience over time t is :

$$W_{c[k]}^l(t) = \sum_{\forall j \in ST \ \wedge l \in \mathcal{L}_j} \left(\left\lfloor \frac{\Phi_{jc[k]}^l}{T_j} \right\rfloor + \left\lceil \frac{t - \Phi_{jc[k]}^l}{T_j} \right\rceil \right) C_j \tag{10.13}$$

Additionally, AVB traffic will be preempted for each ST frame interfering, resulting in additional headers being transmitted. Therefore, for each preemption by an ST frame, an additional interference by the header size v should be considered as shown in Eq. (10.14).

$$V_{c[k]}^l(t) = \sum_{\forall j \in ST \ \wedge l \in \mathcal{L}_j} \left(\left\lfloor \frac{\Phi_{jc[k]}^l}{T_j} \right\rfloor + \left\lceil \frac{t - \Phi_{jc[k]}^l}{T_j} \right\rceil \right) v \tag{10.14}$$

Therefore, the maximum ST interference an AVB frame m_i can experience in link l at instant $I_c^l[k]$ over a time t is computed in Eq. (10.15), which is the summation of interference by ST and the extra headers due to preemption.

$$\begin{aligned} &STI^l_{c[k]}(t) = W^l_{c[k]}(t) + V^l_{c[k]}(t) = \\ &\sum_{\forall j \in ST \wedge l \in \mathcal{L}_j} \left(\left\lfloor \frac{\Phi^l_{jc[k]}}{T_j} \right\rfloor + \left\lceil \frac{t - \Phi^l_{jc[k]}}{T_j} \right\rceil \right) (C_j + v) \end{aligned} \tag{10.15}$$

In this way, the response time of an AVB frame enqueued in the output port of link l in the critical instant candidate $c[k]$, denoted by $STRT^{l,(x)}_{i,c[k]}$, is calculated iteratively as follows.

$$\begin{aligned} STRT^{l,(x)}_{i,c[k]} =& STI^l_{c[k]} \left(STRT^{l,(x-1)}_{i,c[k]} \right) + \\ & HPI^l_i + SPI^l_i + LPI^l_i + C_i. \end{aligned} \tag{10.16}$$

The iteration starts from $STRT^{l,(0)}_{i,c[k]} = HPI^l_i + SPI^l_i + LPI^l_i + C_i$ and terminates when $STRT^{l,(x)}_{i,c[k]} = STRT^{l,(x-1)}_{i,c[k]}$.

The worst-case response time of the frame m_i in link l is denoted by $STWCRT^l_i$. This interference is determined by selecting the largest response time among all critical instant candidates, i.e., the interference for all critical instant candidates should be evaluated and the maximum should be selected as follows:

$$STWCRT^l_i = \max_{\forall m_c, \forall k} \{STRT^l_{i,c[k]}\} \tag{10.17}$$

Optimism in ST interference calculation in [11]: While this solution does cover all potential interferences, unlike the one proposed in [14], it remains optimistic about the magnitude of these interferences which potentially can make the analysis unsafe in corner cases. We show this optimism in a counter-example as follows.

Fig. 10.4(c) illustrates the response time of an AVB frame m_i with C_i = 4 tu, interfered by another same-priority frame with the same frame size (C_j = 4 tu) and an ST frame with C_k = 5 tu, taking into account the guard band and frame size. Moreover, for the sake of simplicity, both credit slopes $\alpha^+_{P_i,l}$ and $\alpha^-_{P_i,l}$ are assigned a value of 0.5, and the header is assigned a value of 1 tu. In

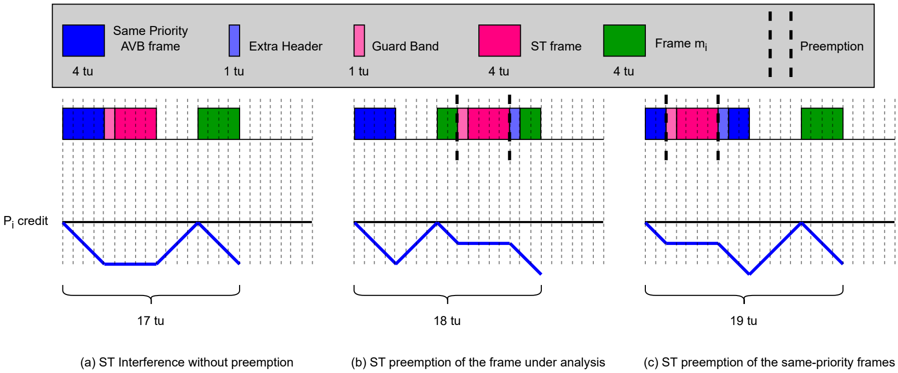


Fig. 10.4: Example of worst-case ST preemption.

this scenario, the analysis by [11] - Eqs. (10.15), (10.16), and (10.17) - yields the following calculated value:

$$\begin{aligned} STWCRT_i^l =& STI_i^l + SPI_i^l + C_i \\ =& 5 + 1 + 8 + 4 = 18tu \end{aligned} \tag{10.18}$$

However, as depicted in the diagram, m_i experiences a delay of 4 tu due to the transmission of the same-priority frame, another 5 tu for the transmission of the ST frame along with the guard band, an additional 1 tu for the header required to resume the transmission of the same-priority frame, 5 tu to recover the consumed credit, and finally, 4 tu of C_i, resulting in a WCRT of 19 tu, proving the optimism in the analysis. Although some analyses, such as [6], consider that the credit freeze during the transmission of the bytes related to the preemption overhead, this only applies to the end-of-frames transmitted during the guard band. That is because during the guard band, the gates of the AVB queues are closed and, according to the most common TSN implementation, the credit of a queue is frozen while its gate is closed. However, the additional headers are transmitted once the AVB gate is reopened so their credit is consumed.

The main issue with these ST interference calculations is the absence of analysis concerning their interaction with other types of traffic, i.e., considering the ST interference as an independent term in the analysis. In the following subsections, we will delve into the interactions between ST and other types of traffic. Initially, we will focus on same-priority traffic before integrating higher- and lower-priority traffic into the analysis.

10.6.2 ST and Same-Priority Interference Interaction

We show the effect of ST preemption with same-priority interference via a lemma. In this subsection, we consider the interference of ST when there is interaction with only same-priority interference.

Lemma 10.6.1. *The WCRT of m_i in link l in the presence of ST and only same-priority interference with ST preemption in the critical instant candidate $I_c^l[k]$ is calculated as follows:*

$$\begin{aligned} STRT_{i,c[k]}^{l,(x)} =& W_{c[k]}^{l}\left(STRT_{i,c[k]}^{l,(x-1)}\right) + \\ & V_{c[k]}^{l}\left(STRT_{i,c[k]}^{l,(x-1)}\right) \times \left(1 + \frac{\alpha_{P_i,l}^{-}}{\alpha_{P_i,l}^{+}}\right) + \\ & SPI_i^l + C_i \end{aligned} \tag{10.19}$$

Proof. In this scenario, there are only three possible cases of ST interference as follows: (i) when the ST interference occurs during credit recovery, thus not preempting any frame (as shown in Fig. 10.4(a)); (ii) when the ST interference occurs during the transmission of the frame under analysis (as shown in Fig. 10.4(b)); and (iii) when the ST interference happens while a frame of the same priority as the one under analysis is being transmitted (as shown in Fig. 10.4(c)).

As depicted in Fig. 10.4(a), the scenario without preemption produces the shortest response time. This is because this type of ST interference does not introduce any extra delay beyond the time associated with the transmission window of the ST frame, i.e., $W_{c[k]}^{l}(t)$ in Eq. (10.13). Conversely, the case shown in Fig. 10.4(b) is the scenario considered in [11]. In this scenario, the ST frame preempts the frame under analysis. In this case, alongside the contribution related to the transmission window of the ST frame $W_{c[k]}^{l}(t)$, it is necessary to add the delay corresponding to the header required to resume the transmission of the frame under analysis, i.e., $V^l_{c[k]}(t)$ in Eq. (10.14). This result corresponds to the one shown in Eqs. (10.15), (10.16), and (10.17). Finally, Fig. 10.4(c) shows the case in which the ST frame preempts frames of the same priority as the frame under analysis. In this case, additionally to the contribution related to the transmission window of the ST frame $W^l_{c[k]}(t)$, and the delay corresponding to the header required to resume the transmission of the same-priority frame, $V^l_{c[k]}(t)$, it is necessary to add the delay corresponding to the recovery of the credit consumed during the transmission of the extra header.

The third case, also shown in Fig. 10.4(c), results in the worst-case situation that can occur to the frame under analysis, which was not considered in the previous work [11]. When ST preempts the same-priority traffic, the transmission time of frames of the same priority is extended by v for each preemption, i.e., $V^l_{c[k]}(t)$. The credit consumed during the transmission of $V^l_{c[k]}(t)$ is $V^l_{c[k]}(t) \times \alpha^-_{P_i,l}$. As explained in Section 10.5.2, all consumed credit must be recovered to allow the transmission of the same-priority traffic. The time required to recover the credit consumed by $V^l_{c[k]}(t)$ is calculated by $V^l_{c[k]}(t) \times \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}$ which by adding $V^l_{c[k]}(t)$ we obtain $V^l_{c[k]}(t) \times \left(1 + \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}\right)$. Therefore, the term for calculating the contribution of multiple headers due to the ST preemption should be modified, proving the lemma. ■

10.6.3 ST and Non-ST Interference Interactions

In this subsection, we consider the interference of ST when there is interaction with all non-ST frame interference, including the same-priority interference.

Lemma 10.6.2. *The WCRT of m_i in link l in the presence of ST interference, higher-priority AVB traffic interference, lower-priority blocking, and same-priority interference in the critical instant candidate $I^l_c[k]$ is calculated as fol-*

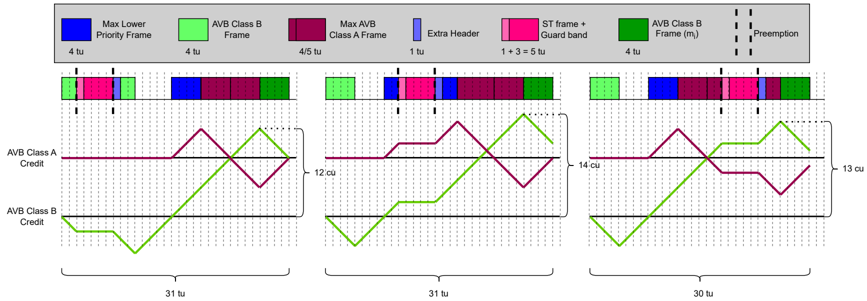
lows:

$$
\begin{aligned}
&STRT_{i,c[k]}^{l,(x)} = \\
&\quad W_{c[k]}^{l}\left(STRT_{i,c[k]}^{l,(x-1)}\right) \\
&\quad +V_{c[k]}^{l}\left(STRT_{i,c[k]}^{l,(x-1)}\right) \times \left(1+\max\left(\frac{\alpha_{P_i,l}^{-}}{\alpha_{P_i,l}^{+}}, \frac{\alpha_{\mathbb{H},l}^{+}}{\alpha_{\mathbb{H},l}^{-}}\right)\right) \\
&\quad +HPI_i^l + LPI_i^l + SPI_i^l + C_i
\end{aligned}
\tag{10.20}
$$

Proof. Alongside the three cases of ST interference in the interaction between ST and same-priority interference, we must incorporate two additional cases: (i) when an ST frame preempts a higher-priority frame, and (ii) when an ST frame preempts a lower-priority frame. However, The ST preemption of the frame under analysis and the case with no preemption have already been discarded as worst-case scenario candidates in Lemma 10.6.1, thus they will not be analyzed again.

To prove Lemma 10.6.2, we will refer to the example in Fig. 10.5, where we analyze the three new candidates of maximum interference due to preemption: (a) same-priority preemption by an ST frame, (b) lower-priority preemption by an ST frame, and (c) high-priority preemption by an ST frame. These figures illustrate the worst-case response time of an AVB Class B frame m_i under the three scenarios. In this example, the frame under analysis m_i with C_i = 4 tu is interfered with by another same-priority frame with the same frame size (C = 4 tu) and an ST frame with C = 5 tu, taking into account the guard band and frame size. Also, we will consider one higher-priority queue (AVB Class A), and we will assume that there are constantly higher- and lower-priority frames queued for transmission in the output port, with a maximum frame size of C = 4 tu in both cases. Moreover, for simplicity, the credit slopes of both AVB Classes ($\alpha_{P_A,l}^{+}$, $\alpha_{P_A,l}^{-}$, $\alpha_{P_B,l}^{+}$, and $\alpha_{P_B,l}^{-}$) are assigned a value of 0.5, and the header is assigned a value of 1 tu.

According to the eligible interval approach, non-ST interference is constrained by the maximum achievable credit of the AVB Class of the frame under analysis, which can be calculated through Eqs. (10.6) and (10.7). Eq.(10.6) further categorizes the contributions into two parts: one dependent on the



(a) ST preemption of the same-priority frames (b) ST preemption of the lower-priority frames (c) ST preemption of the higher-priority frames

Fig. 10.5: Example of worst-case ST preemption with lower and higher priority traffic interference.

lower-priority interference, $LPI_i^l = C_{\mathbb{L},l}^{max} \times \left(1 + \frac{\alpha_{\mathbb{H},l}^+}{\alpha_{\mathbb{H},l}^-}\right)$, and another one corresponding to the higher-priority interference, $HPI_i^l = \frac{CR_{\mathbb{H},l}^{min}}{\alpha_{\mathbb{H},l}^-}$. Since $W_{c[k]}^l(t)$ does not affect the credit, it does not affect the maximum achievable P_i credit. However, the extra headers $V_{c[k]}^l(t)$ can affect the maximum achievable credit. In cases where ST preemption affects same-priority traffic, as shown in Fig. 10.5(a), the extra headers do not affect the maximum credit, as the cumulative credit over the same-priority interference is zero, as demonstrated in Lemma 10.6.1. Conversely, if the largest lower-priority frame ($C_{\mathbb{L},l}^{max}$) is preempted, as shown in Fig. 10.5(b), its size is increased by $V_{c[k]}^l(t)$. According to the eligible interval analysis, in the worst-case scenario, all credit accumulated due to $C_{\mathbb{L},l}^{max}$ in the highest priority AVB classes, i.e. $C_{\mathbb{L},l}^{max} \times \alpha_{\mathbb{H},l}^+$, will be consumed by higher-priority classes before higher-priority interference starts. This implies that if frame $C_{\mathbb{L},l}^{max}$ is preempted by ST, its duration will be increased by $V_{c[k]}^l(t)$, resulting in an additional delay experienced by frame m_i equal to $V_{c[k]}^l(t) \times \left(1 + \frac{\alpha_{\mathbb{H},l}^+}{\alpha_{\mathbb{H},l}^-}\right)$.

Lastly, preempting the higher-priority frames does not produce the worst-case scenario. The higher-priority interference contribution, according to the eligible interval analysis and Eqs. (10.6) and (10.7), involves a summation of $C^{max}_{H_i,l} \times \frac{\alpha^-_{H_j,l}}{\alpha^-_{\mathbb{H},l}}$, where $\frac{\alpha^-_{H_j,l}}{\alpha^-_{\mathbb{H},l}}$ is always less than or equal to 1. If ST preempts a higher-priority frame, it implies an increase in $C^{max}_{H_i,l}$ by $V^l_{c[k]}(t)$, leading to its contribution to the delay being $V^l_{c[k]}(t) \times \frac{\alpha^-_{H_j,l}}{\alpha^-_{\mathbb{H},l}} \leq V^l_{c[k]}(t)$. Considering that $\left(1 + \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}\right)$ corresponding to ST preemption of same-priority frames and $\left(1 + \frac{\alpha^+_{\mathbb{H},l}}{\alpha^-_{\mathbb{H},l}}\right)$ corresponding to ST preemption of lower-priority frames are always greater than 1, ST preemption of higher-priority frames does not produce the WCRT. In conclusion, the WCRT of frame m_i occurs either when ST preempts the same-priority traffic, increasing the delay by $V^l_{c[k]}(t) \times \left(1 + \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}\right)$, or when ST preempts the largest lowest-priority frame, increasing the delay by $V^l_{c[k]}(t) \times \left(1 + \frac{\alpha^+_{\mathbb{H},l}}{\alpha^-_{\mathbb{H},l}}\right)$, depending on which preemption leads to the biggest delay, i.e. depending on $\max\left(\frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}, \frac{\alpha^+_{\mathbb{H},l}}{\alpha^-_{\mathbb{H},l}}\right)$. Therefore, we obtain $V^l_{c[k]}(t) \times \left(1 + \max\left(\frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}, \frac{\alpha^+_{\mathbb{H},l}}{\alpha^-_{\mathbb{H},l}}\right)\right)$. ■

Finally, recall that to obtain the WCRT of m_i in link l it is necessary to select the largest response time among all critical instant candidates through Eq. (10.17) where $STRT_i$ for link l and critical instant candidate $I^l_c[k]$ is calculated by Eq. (10.20).

10.6.4 Multi-hop Calculation

Finally, since the analysis is compositional in the sense that a frame passing through every hop is buffered, we sum the $STWCRT^l_i$ of each link in the path from the source to the destination of the frame under analysis m_i and add the ϵ factor for each traversed switch. This factor represents the delay suffered by the frame from being received at the input port of the switch to being enqueued in the TSN queue at the output port. The WCRT for frame m_i is calculated in

Eq. (10.21)

$$WCRT_i = \sum_{l=1,\ldots,|\mathcal{L}_i|} STWCRT_i^l + (|\mathcal{L}_i| - 1) \times \epsilon \tag{10.21}$$

where, $STWCRT_i^l$ is determined by Eq. (10.17), which relies on the calculation of $STRT_{i,c[k]}^l$ by Eq. (10.20). Finally, $W_{c[k]}^l(t)$, $V_{c[k]}^l(t)$, HPI_i^l, LPI_i^l, and SPI_i^l are computed using Eqs. (10.13), (10.14), (10.3), (10.6), (10.7), respectively. The compositional structure of the WCRTA allows for its application even when certain delay sources are absent (i.e., their contribution is zero). Therefore, by considering all potential sources of delay, the WCRT can be accurately determined. However, note that each contribution in the formula is not independent. For instance, when quantifying ST interference, the interaction with other traffic types is considered, including the effect of extra headers on credit consumption or accumulation. Consequently, each component in the equations (STI, HPI, SPI, and LPI) accounts not only for the contribution of the specific traffic type (e.g., ST traffic) but also for the cumulative effect of all types of interference as a result of the interactions of that type of traffic with them.

10.7 Experimental Setup

This paper uses an Evaluation Toolset (ETS) to evaluate the proposed WCRTA. The ETS comprises a comprehensive suite of integrated tools tailored for conducting automated experiments focused on the scheduling and schedulability analysis of TSN networks. This section explains the used ETS, including the modifications made specifically for this paper. The configuration of the ETS used for this study is shown in Fig. 10.6. The input for the ETS consists of the network's configuration, involving its topology and traffic characteristics.

We consider two network topologies, as shown in Figs. 10.7 and 10.8, following a line-star topology. This topology is adequate for the analysis, as the only missing element that could impact the results is the presence of loops. However, since some of the compared WCRTAs do not support circular dependencies, this element has been excluded to ensure a fair comparison. Network

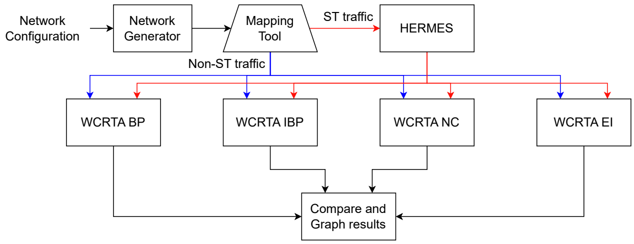


Fig. 10.6: Evaluation Toolset configuration.

N1 consists of a compact network with 2 switches, each with 10 end-stations, while network N2 presents a larger network with 5 switches and 4 end-stations connected to each switch. These network topologies are the input to the ETS.

To maintain manageable experiment durations, the network bandwidth was set to 100Mbps. This choice also ensures that the maximum allowed 100 frames can consistently reach the target utilization in almost every link. The frame length was selected from the range [500,1500] Bytes. Minimum and maximum allowed periods were set at 2000μs and 10000μs, respectively. Note that the number of frames generated can vary depending on the maximum allowed utilization of the links. For example, running an experiment that limits link utilization to 10% of the bandwidth will generate fewer frames than running an experiment that allows up to 90%.

The ETS starts with the Network Generator generating random traffic based on the topology and the traffic characteristics used as input. We enforced the distribution of 25% of the traffic being ST and 75% being AVB Class A and AVB Class B. For the experiments, the BE class and lower AVB priorities than class B were omitted due to limitations in the compared WCRTAs. We also evaluate the performance of the compared WCRTAs across various network utilizations, ranging from 5% to 60% for the small network and from 5% to 45% for the large network. We conducted 100 traffic generations for each utilization, making 2100 experiments. Each experiment involved analyzing up to 100 frames, resulting in the analysis of almost 210,000 frames.

The generated traffic is mapped into the different TSN traffic classes (ST, AVB Class A, and AVB Class B) in the next step, i.e., in the Mapping Tool

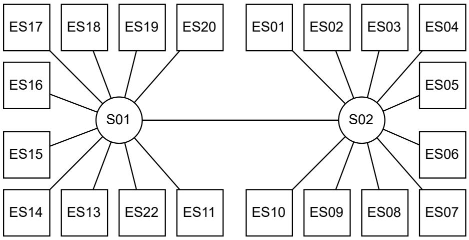


Fig. 10.7: Experimental network topology N1.

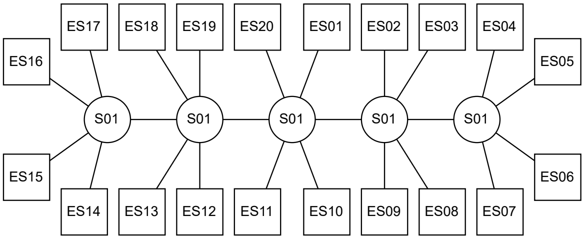


Fig. 10.8: Experimental network topology N2.

(Fig. 10.6). The ST traffic will be scheduled using an existing heuristic algorithm [2]. Note that any other scheduling algorithm for ST traffic can be utilized, and a recent survey [17] identified their various features and characteristics. We selected a heuristic algorithm as it provides a good trade-off between fast and feasible results.

The AVB traffic and the ST schedule are used as input for each of the compared WCRTAs. In Fig. 10.6, WCRTA Busy Period (BP) and WCRTA Improved Busy Period (IBP) are the WCRTA based on busy period analysis [11], before and after removing the optimism introduced in Lemmas 1 and 2, respectively. WCRTA Network Calculus (NC) corresponds to the WCRTA implemented in Network Calculus [19], while WCRTA Eligible Interval (EI) represents the analysis proposed in this paper, i.e., the WCRTA based on the eligible interval extended with the new ST interference calculation and multi-hop behaviors. All WCRTAs were configured with both AVB Classes' credit slopes ($\alpha^+_{P_A,l}$, $\alpha^-_{P_A,l}$, $\alpha^+_{P_B,l}$, and $\alpha^-_{P_B,l}$) set to 0.5, which divides the available bandwidth equally between AVB Classes A and B corresponding to the experimental setup distribution of the AVB traffic. Additionally, a switch traversal factor ϵ of 0 was used for a fairer comparison since some analyses do not consider this factor.

Finally, the WCRTAs are compared using two methods. First, the schedulability of each WCRTA is determined for each bandwidth utilization. This comparison consists of calculating the percentage of generated networks that meet their time requirements according to each WCRTA. Second, the pessimism ratio between each previous WCRTA and the one proposed in this paper is calculated for each bandwidth utilization. Specifically, for each AVB-generated frame m_i, the pessimism ratio of WCRTA X to WCRTA EI is computed as follows.

$$\textit{Pessimism Ratio}_i = \frac{WCRTX_i}{WCRTEI_i} \tag{10.22}$$

This value illustrates how much pessimism is reduced by our proposed analysis (WCRTA EI) compared to the analysis named by X (BP, IBP, NC). Using this ratio, a pessimism ratio below 1 shows how much the proposed WCRTA is pessimistic compared to the previous analysis. Whereas, the ratio above 1 shows how much the proposed analysis in this paper performs better with respect to reducing pessimism compared to the existing WCRTA.

Due to the random traffic generation and the extensive number of experiments conducted, we can confidently assert that these experiments do not bias any specific WCRTA. Therefore, the results accurately reflect the performance differences among the compared WCRTAs.

10.8 Results

This section presents the outcomes derived from the experiments outlined in Section 10.7. We assess the schedulability of the networks in relation to the bandwidth utilization percentage. To this end, we compare the additional pessimism introduced by the existing WCRTAs (WCRTA BP, IBP, and NC) in relation to the WCRTA EI proposed in this work.

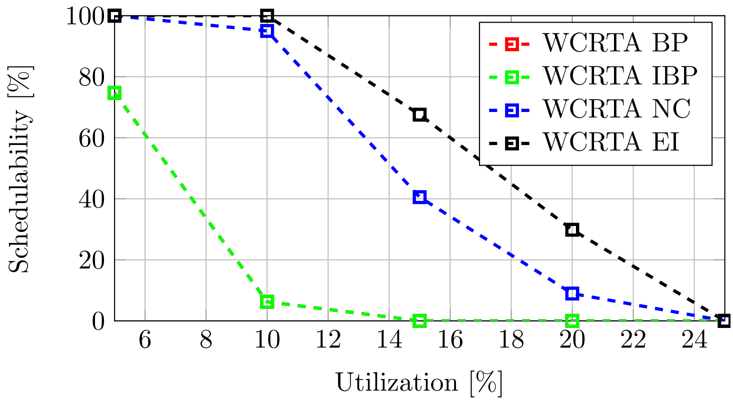


Fig. 10.9: Schedulability of the WCRTAs in network N1.

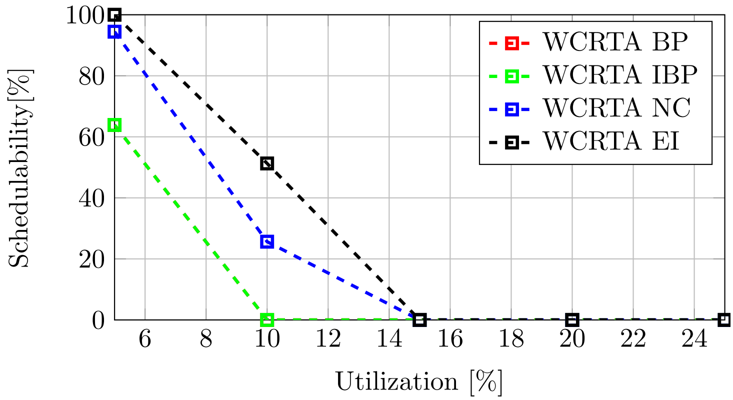


Fig. 10.10: Schedulability of the WCRTAs in network N2.

10.8.1 Schedulability Comparison

Figs. 10.9 and 10.10 depict the percentage of schedulability achieved by each evaluated WCRTA with respect to the bandwidth utilization for N1 and N2, respectively. Note that the results for WCRTA BP and IBP overlap on the graph.

These graphs illustrate how the WCRTA introduced in this study consis-

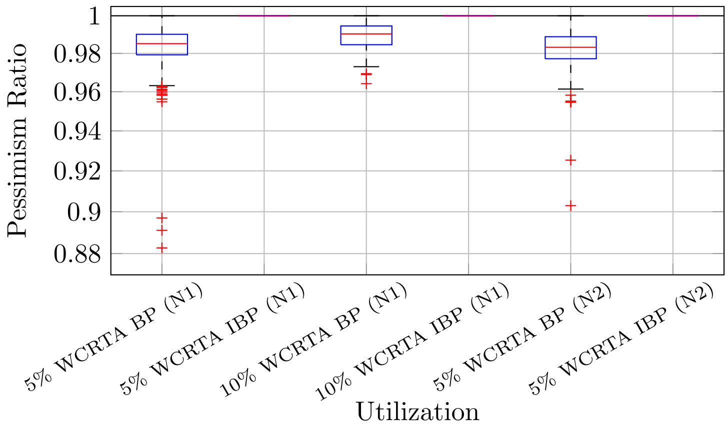


Fig. 10.11: Pessimism Ratio of the AVB Class A traffic of WCRTA BP and IBP in the networks N1 and N2.

tently outperforms all prior WCRTAs in terms of schedulability despite the optimism present in one of them. The proposed WCRTA EI exhibits up to 90% higher schedulability compared to the WCRTAs BP and IBP based on busy periods with and without optimism and up to 25% higher schedulability at certain utilization levels, when compared to the WCRTA NC based on the Network Calculus model. Additionally, it is noteworthy that, despite the reduction in network schedulability with an increase in network hops (Fig. 10.10), the proposed WCRTA EI still demonstrates better performance than the previous solutions in terms of schedulability. All WCRTAs show zero schedulability after 25% network utilization for N1 and after 15% network utilization for N2. This is due to the accumulation of delays over multiple hops, a source of pessimism that is not in the scope of this work.

10.8.2 Additional Pessimism Comparison

In this subsection, we examine the WCRTs of only schedulable configurations as the analysis is deadline-constrained, i.e., the WCRT exceeding the deadline may not be correct.

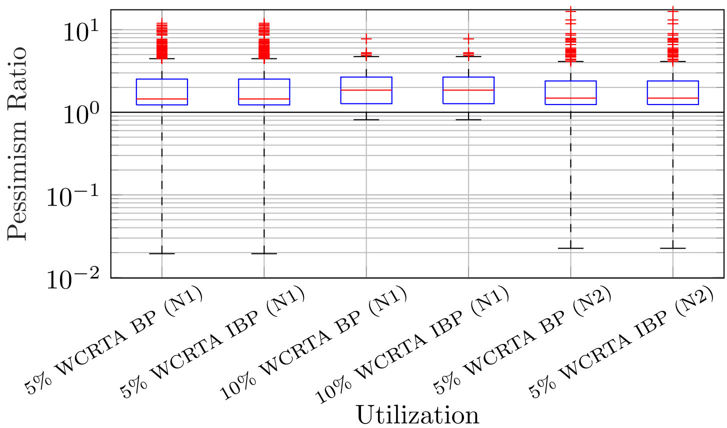


Fig. 10.12: Pessimism Ratio of the AVB Class B traffic of WCRTA BP and IBP in networks N1 and N2.

Fig. 10.11 illustrates the pessimism ratio between the WCRTs of the AVB Class A traffic provided by WCRTA BP and IBP compared to WCRTA EI on networks N1 and N2. As can be seen, WCRTA BP shows better results (pessimism ratio lower than 1) compared to WCRTA EI. This is due to the optimism we identified in WCRTA BP, which shows an incorrect WCRT for class A traffic. After solving the optimism, shown as WCRTA IBP, both WCRTA IBP and WCRTA EI have identical values for WCRT of class A traffic. The figure shows how much optimism existed in the analysis based on the busy period [11] for class A traffic.

On the other hand, Fig. 10.12 depicts the pessimism ratio of AVB Class B traffic between WCRTAs BP and IBP compared to WCRTA EI for networks N1 and N2. As anticipated, despite WCRTA BP exhibiting optimistic ST interference, the pessimism associated with WCRTAs BP and IBP for AVB Class B traffic far surpasses that of WCRTA EI, potentially resulting in WCRTs tenfold higher. This discrepancy explains to a large extent the low schedulability of WCRTAs BP and IBP shown in Figs. 10.9 and 10.10.

Figs. 10.13 and 10.14 illustrate the pessimism ratio between the WCRTs of AVB Class A traffic provided by WCRTA NC compared to WCRTA EI on

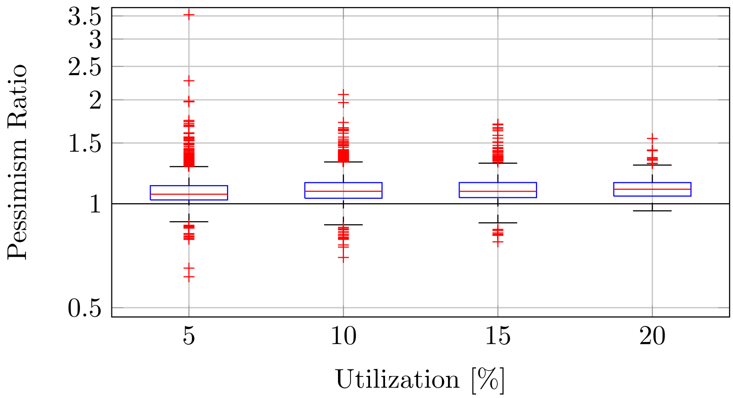


Fig. 10.13: Pessimism Ratio of the AVB Class A traffic of WCRTA NC in network N1.

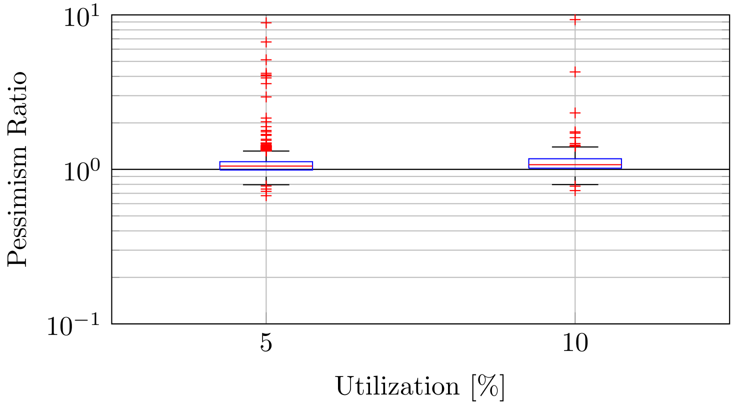


Fig. 10.14: Pessimism Ratio of the AVB Class A traffic of WCRTA NC in network N2.

networks N1 and N2, respectively. These figures demonstrate that WCRTA NC yields WCRTs around 1.1 times larger than those achieved through WCRTA EI, resulting in an additional pessimism of approximately 10-15%.

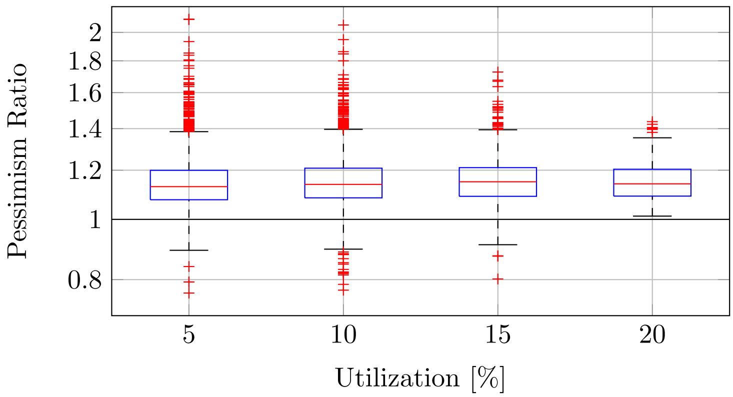


Fig. 10.15: Pessimism Ratio of the AVB Class B traffic of WCRTA NC in network N1.

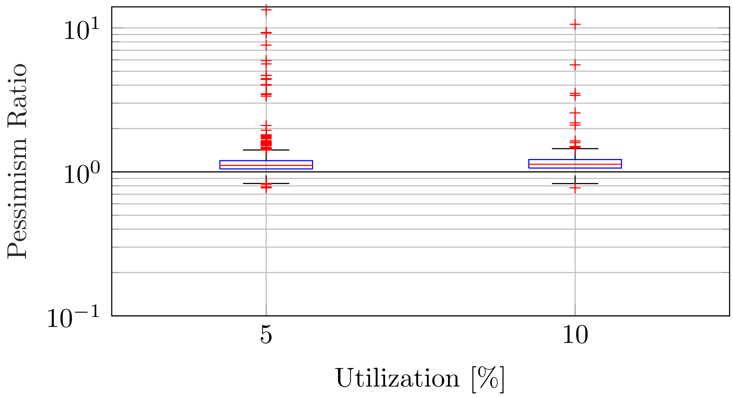


Fig. 10.16: Pessimism Ratio of the AVB Class B traffic of WCRTA NC in network N2.

Similarly, Figs. 10.15 and 10.16 show the pessimism ratio of AVB Class B traffic provided by WCRTA NC compared to WCRTA EI for networks N1 and N2, respectively. The results show an additional pessimism of around 10-15%, depending on the network size. The higher WCRT values provided by WCRTA NC account for its lower schedulability compared to WCRTA EI in

Figs. 10.9 and 10.10.

Finally, upon comparing the outcomes of WCRTAs BP and IBP with those of WCRTA NC, it becomes evident that the latter exhibits higher schedulability. This is primarily attributed to the fact that, while WCRTA BP and IBP display lower pessimism in the WCRT of AVB Class A traffic, the considerably higher pessimism in the WCRT of AVB Class B traffic significantly hampers their effectiveness.

10.9 Conclusion

For the successful adoption of Time-Sensitive Networking (TSN) in industry, developing an increasingly accurate and less pessimistic Worst-Case Response Time Analysis (WCRTA) is crucial. In this paper, we thoroughly reviewed existing WCRTAs, evaluating their strengths, weaknesses, and potential errors. Based on this examination, we introduced a novel WCRTA, for Audio-Video Bridging (AVB) traffic, that eliminates the optimism found in some prior works and yields less pessimistic results compared to the existing counterparts. Notably, our WCRTA demonstrated a reduction of around 15% in pessimism compared to the WCRTA based on the Network Calculus model [19] and around 100% improvement compared to the WCRTA based on busy period analysis [11]. The results provide strong evidence that our proposed WCRTA offers a better solution for meeting TSN's time requirements.

Bibliography

[1] Unmesh D Bordoloi, Amir Aminifar, Petru Eles, and Zebo Peng. Schedulability analysis of Ethernet AVB switches. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[2] Bujosa, Daniel and Ashjaei, Mohammad and Papadopoulos, Alessandro V and Nolte, Thomas and Proenza, Julián. HERMES: Heuristic multi-queue scheduler for TSN time-triggered traffic with zero reception jitter capabilities. In *Proceedings of the 30th International Conference on Real-Time Networks and Systems*, pages 70–80, 2022.

[3] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent yet tight WCRT analysis for individual priority classes in ethernet AVB. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, pages 55–64, 2016.

[4] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Tight worst-case response-time analysis for ethernet AVB using eligible intervals. In *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–8. IEEE, 2016.

[5] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent WCRT analysis for individual priority classes in Ethernet AVB. *Real-Time Systems*, 54:861–911, 2018.

[6] Hugo Daigmorte, Marc Boyer, and Luxi Zhao. Modelling in network calculus a TSN architecture mixing Time-Triggered, Credit Based Shaper and Best-Effort queues. 2018.

[7] Robert I Davis, Steffen Kollmann, Victor Pollex, and Frank Slomka. Controller Area Network (CAN) Schedulability Analysis with FIFO Queues. In *2011 23rd Euromicro Conference on Real-Time Systems*, pages 45–56. IEEE, 2011.

[8] Jonas Diemer, Jonas Rox, and Rolf Ernst. Modeling of Ethernet AVB networks for worst-case timing analysis. *IFAC Proceedings Volumes*, 45(2):848–853, 2012.

[9] Jonas Diemer, Daniel Thiele, and Rolf Ernst. Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching. In *7th IEEE International Symposium on Industrial Embedded Systems (SIES'12)*, pages 1–10. IEEE, 2012.

[10] Jover, Mateu and Barranco, Manuel and Proenza, Julián. Opportunities and Specific Plans for Migrating from PRP to TSN in Substation Automation Systems. In *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, 2023.

[11] Lucia Lo Bello, Mohammad Ashjaei, Gaetano Patti, and Moris Behnam. Schedulability analysis of Time-Sensitive Networks with scheduled traffic and preemption support. *Journal of Parallel and Distributed Computing*, 144:153–171, 2020.

[12] Lucia Lo Bello, Gaetano Patti, and Giancarlo Vasta. Assessments of real-time communications over TSN automotive networks. *Electronics*, 10(5):556, 2021.

[13] J. Maki-Turja and M. Nolin. Fast and tight response-times for tasks with offsets. In *17th Euromicro Conference on Real-Time Systems (ECRTS'05)*, 2005.

[14] Maxim, Dorin and Song, Ye-Qiong. Delay analysis of AVB traffic in time-sensitive networks (TSN). In *Proceedings of the 25th International Conference on Real-Time Networks and Systems*, pages 18–27, 2017.

[15] Mubarak Adetunji Ojewale, Patrick Meumeu Yomsi, and Borislav Nikolić. Worst-case traversal time analysis of tsn with multi-level preemption. *Journal of Systems Architecture*, 116:102079, 2021.

[16] Mohamed Seliem, Ahmed Zahran, and Dirk Pesch. Delay analysis of TSN based industrial networks with preemptive traffic using network calculus. In *2023 IFIP Networking Conference (IFIP Networking)*, pages 1–9. IEEE, 2023.

[17] Chuanyu Xue, Tianyu Zhang, Yuanbin Zhou, and Song Han. Real-time scheduling for time-sensitive networking: A systematic review and experimental study. 2023.

[18] Jiayi Zhang, Lihao Chen, Tongtong Wang, and Xinyuan Wang. Analysis of tsn for industrial automation based on network calculus. In *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 240–247, 2019.

[19] Luxi Zhao, Paul Pop, Zhong Zheng, Hugo Daigmorte, and Marc Boyer. Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus. *IEEE Transactions on Industrial Electronics*, 68(10):10291–10302, 2020.

[20] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. Timing analysis of AVB traffic in TSN networks using network calculus. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 25–36. IEEE, 2018.

Chapter 11

Paper D
Reducing Pessimism in Response Time Analysis of AVB Traffic in TSN.

Daniel Bujosa, Julián Proenza, Alessandro V. Papadopoulos, Thomas Nolte, Mohammad Ashjaei.
MRTC Report, MDU 2024..

Abstract

Time-Sensitive Networking (TSN) is a set of standards with significant industrial impact potential, primarily due to its ability to integrate multiple traffic types with different requirements, offering great network flexibility. Among these traffic types, Audio-Video Bridging (AVB) stands out for its real-time guarantees and dynamic scheduling. In order to guarantee a specific set of AVB frames meet their timing requirements a Worst-Case Response Time Analysis (WCRTA) is essential. Unfortunately, current WCRTAs are often overly conservative, failing to guarantee schedulability for TSN systems operating even under low bandwidth conditions. This limits the practical usefulness of these analyses. Since TSN utilizes a multi-hop architecture, most WCRTAs analyze each link independently and then add the contributions. This compartmental analysis introduces pessimism, particularly when calculating the interference caused by other AVB frames with the same priority as the frame under analysis. In this paper, we address this issue by refining the Same-Priority Interference (SPI) calculation, leading to a significant improvement in the schedulability of WCRTAs and, consequently, the overall efficiency of TSN networks.

11.1 Introduction

Time-Sensitive Networking (TSN) has become one of the most relevant sets of standards for the industry. Among its many features, TSN enables Ethernet to integrate diverse traffic types with different characteristics and requirements simultaneously, providing significant network flexibility. This flexibility is achieved through TSN's three traffic classes: Scheduled Traffic (ST), Audio-Video Bridging (AVB) traffic, and Best-Effort (BE) traffic. AVB traffic is particularly valued for its real-time guarantees and adaptability. This is made possible by mechanisms such as the Credit-Based Shaper (CBS) and the Stream Reservation Protocol (SRP) described in the IEEE Std 802.1Q [1], which support online scheduling through bandwidth reservations and dynamic traffic configuration. However, while ST traffic is scheduled offline and guarantees its time requirements by construction, and BE traffic does not have real-time guarantees, to ensure that AVB traffic meets its timing requirements a Worst-Case Response Time (WCRT) Analysis (WCRTA) is required.

Since TSN operates on a multi-hop architecture, most existing WCRTAs analyze each link independently and aggregate the contributions along the path from source to destination. This approach of compartmental link analysis introduces unnecessary pessimism, especially when calculating the interference caused by same-priority frames, i.e. the interference caused by other AVB frames with the same priority as the frame under analysis. In this paper, we improve the Same-Priority Interference (SPI) calculation by considering the interactions between the input and output links of a switch. This refinement leads to a tighter analysis with reduced pessimism, resulting in improved schedulability and greater efficiency in TSN.

Contributions: In this paper, we focus on analyses that account for ST traffic, with and without preemption of non-ST traffic (i.e., AVB and BE), as this configuration is the most commonly used. However, any analysis that calculates the WCRT from source to destination can benefit from the enhanced STI calculation presented here. Therefore, we limit our scope to analyses that evaluate the WCRT across the entire path from source to destination. Specifically, we apply our solution to the analysis based on eligible intervals [4], as it is one of the most recent and has demonstrated significantly lower pessimism compared to earlier approaches, such as the WCRTA based on busy period [11] and the one based on Network Calculus [16], which meet the criteria for being considered in this paper. To the best of our knowledge, these are the only three AVB WCRTAs in the literature that consider ST interference and preemption according to the standards and compute the WCRT from the source to the destination. The concrete contributions in this paper are as follows:

- First, we identify a source of pessimism arising from the independent analysis of the output ports and their links, without considering the input links of the switch.
- Next, we extend the analysis to consider the input and output links at the switches, leading to a refined calculation of the SPI.
- Finally, we compare the improved SPI computation with results obtained using the WCRTAs that meet the properties considered in this paper, i.e. ST interference, both with and without preemption, and source-to-destination results.

Outline The paper is organized as follows. Section 11.2 describes the different mechanisms and features of TSN considered in this paper and Section 11.3 reviews the related works. Then, Section 11.4 defines the system model, while Section 11.5 summarizes the WCRTA based on the eligible interval, and Section 11.6 presents the pessimism problem identified in the previous analysis. Section 11.7 extends the analysis through an improved SPI calculation. Finally, Section 11.8 presents the experimental setup, Section 11.9 shows the results, and Section 11.10 concludes the paper.

11.2 TSN's Shapping Mechanisms and Features

In TSN, frames exchanged between end-stations are transmitted over paths composed of Ethernet links and TSN switches. The output ports of these switches and end-stations incorporate traffic-shaping mechanisms that support the various TSN traffic types. Specifically, each output port on both TSN end-stations and switches is equipped with up to 8 First-In-First-Out (FIFO) queues, each corresponding to one of the 8 priority levels available in TSN. Depending on the queue configuration and the integrated traffic-shaping mechanisms, each queue is assigned to handle one of the three TSN traffic types (ST, AVB, or BE). Each traffic type can utilize one or more of the available priorities, and thus, queues, resulting in multiple ST priorities and potentially multiple AVB and BE classes. For instance, the network can be configured to include three AVB classes (AVB class A, B, and C), each class having a lower priority than the previous one.

Fig. 11.1 illustrates a simplified 4-queue output port, comprising one ST queue, two AVB queues (Class A and B), and one BE queue. The following sections will provide a more detailed explanation of each TSN traffic shaper and their respective configurations.

11.2.1 Time-Aware Shaper

ST traffic is scheduled offline, ensuring fully deterministic transmission with zero jitter. To enforce this schedule, TSN utilizes the Time-Aware Shaper (TAS) mechanism, which assigns each queue a gate that can either be opened or closed. The gate's state is governed by the Gate Control List (GCL), a cyclically repeating list that specifies, at the nanosecond level, how long each gate remains open or closed. The cycle length is determined by the hyper-period of the ST frames transmitted through the corresponding output port, specifically the least common multiple of the periods of the ST frames passing through the gate. This characteristic of ST traffic is a limitation, as certain period combinations may cause an excessive increase in the GCL duration, which makes the use of AVB traffic interesting even for periodic frames.

When the gate is open, traffic from the queue can be transmitted, while a closed gate blocks transmission. The period during which the gate remains open is referred to as the *transmission window* . TSN also supports configuring ST traffic to preempt non-ST traffic, meaning that the transmission of a non-ST frame is temporarily halted and resumed after the ST frame has completed its transmission. Additionally, TAS can be configured to prevent the start of a new frame transmission if there is not enough time left until the gate closes to complete the transmission. However, this configuration will not be considered in our analysis, as it is rarely implemented in commercial switches.

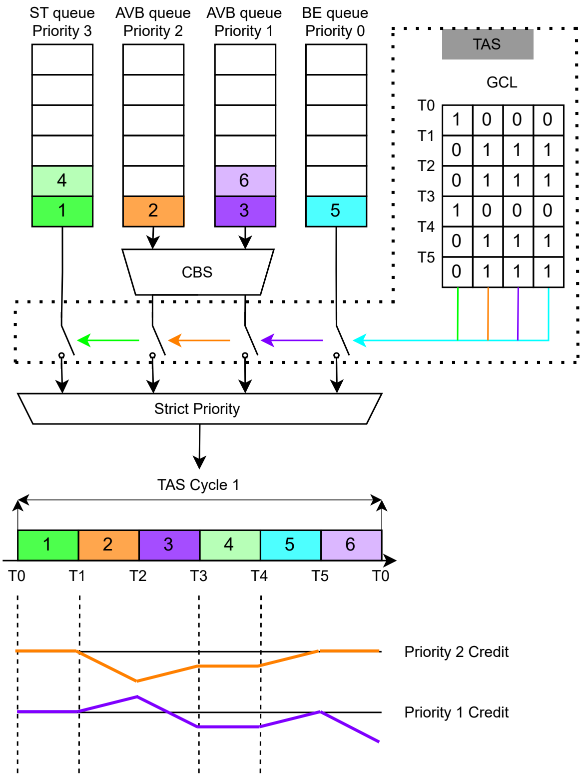


Fig. 11.1: Example of TSN output port.

For the transmission of an ST frame, the frame must reside in the designated queue, and its gate must remain closed until the scheduled transmission time. Before this time, all other queue gates are closed, during a certain time interval, to prevent interference. This time interval, when all gates are closed, is called the *guard band*. The size of the guard band depends on whether TSN is configured with or without preemption of non-ST traffic by ST frames. With preemption, the guard band must be at least the minimum preemptable frame size (124 bytes). Without preemption, the guard band must be larger than the maximum TSN frame size (1518 bytes). Omitting the guard band would introduce jitter in the transmission of ST frames. This jitter would be equivalent to the transmission time of the minimum preemptable frame size with preemption or the whole non-ST frame without preemption. Finally, the ST queue gate opens precisely at the scheduled transmission time, allowing the frame to be transmitted as scheduled. This mechanism is referred to as HOLD/RELEASE in the standards.

Fig. 11.1 provides an example with two ST frames (1 and 4). For simplicity, this example excludes guard bands and preemption. According to the GCL, these frames are scheduled for transmission at times T0 and T3, respectively. These transmission windows are marked by the dashed vertical lines in the figure. As shown, at T0, the ST gate is open while all other gates are closed, ensuring interference-free transmission of frame 1. Between T1 and T3, the ST gate is closed, and the other gates are open, allowing non-ST traffic to be transmitted. At T3, the ST gate is reopened, and the other gates are closed, enabling the transmission of frame 4 without interference. Finally, the ST gate closes again, and the non-ST gates open to allow the transmission of the remaining non-ST frames.

11.2.2 Credit-Based Shaper

While ST is transmitted on a precise, fixed schedule without jitter, its reliance on offline scheduling reduces its flexibility and limits its effectiveness in accommodating other types of traffic, such as event-triggered traffic. Additionally, the computational cost of obtaining a schedule rises considerably as the number of frames increases, potentially making scheduling infeasible or impractical in some scenarios. This is where AVB traffic proves advantageous, as it maintains real-time properties without being bound by the strict constraints of fixed scheduling.

To support real-time properties for AVB traffic, TSN employs the CBS mechanism. CBS restricts the maximum bandwidth percentage that a queue can utilize, ensuring that lower-priority queues receive a guaranteed minimum bandwidth allocation. In CBS, each AVB queue has a designated credit, which accumulates when an AVB frame is awaiting transmission or the credit is negative and is consumed while the queue is transmitting frames. Credit replenishment and consumption rates are defined by the terms *idleSlope* and *sendSlope*, respectively. A queue can transmit only when its credit is zero or positive, and its credit is frozen if the gate is closed.

In the example in Fig. 11.1 a single frame (frame 2) is allocated to the higher-priority AVB queue, while the lower-priority AVB queue holds two frames (frames 3 and 6), and the BE queue contains one frame (frame 5). At time T1, following the transmission of the ST frame, both AVB queues become eligible for transmission. The strict priority mechanism then selects the higher-priority AVB queue, resulting in the transmission of frame 2. During this transmission, the credit of the higher-priority AVB queue is consumed, while the credit of the lower-priority AVB queue is replenished as it awaits transmission.

At T2, the higher-priority AVB queue, now with negative credit, becomes ineligible for transmission. The lower-priority AVB queue is then selected, leading to the transmission of frame 3. Similar to T1, the credit of the active AVB queue is consumed during frame 3's transmission, while the credit for the higher-priority AVB queue is replenished.

At T3, another ST frame is transmitted, closing the gates of the AVB queues and freezing their credits. By T4, both AVB queues have negative credit, allowing the BE queue to transmit, even though a higher-priority AVB frame is awaiting transmission. Finally, at T5, the lower-priority AVB queue's credit reaches zero, permitting the transmission of the last frame (frame 6). Note that BE traffic lacks any timing requirements. Therefore, it can only transmit when its gate is open and no higher-priority frames are eligible for transmission.

11.3 Related Work

Since the introduction of the AVB standard in 2011, numerous works have tackled the challenge of analyzing its WCRT. These approaches are commonly classified into three main categories: busy period analysis, eligible interval analysis, and Network Calculus modeling.

Busy period analysis identifies the critical instant that produces the WCRT. Diemer et al. [9] pioneered AVB WCRT analysis based on this approach, although it was limited to a single output port and AVB queue, without accounting for ST interference. Later, in [10], they extended their work to handle two AVB queues (class A and class B) but remained constrained to a single output port and still omitted ST interference. In the more recent work of Lo Bello et al. [11], the authors present an AVB WCRTA based on busy period analysis that considers interference across multiple AVB classes and incorporates ST interference with and without preemption, as well as lower-priority frame blocking. This analysis also includes multi-hop calculations, extending the analysis from source to destination rather than limiting it to a single output port. However, it remains limited to two AVB classes (Class A and B).

The second approach, eligible interval analysis, defines the time interval in which each frame becomes eligible for transmission. The WCRT of a frame is calculated by analyzing the maximum interference it encounters over this interval. One of the earliest analyses of this type was introduced by Bordoloi et al. [2]. Cao et al.[6, 5, 7] later extended this work and formally introduced the term eligible interval, showing that the WCRT of an AVB frame depends solely on the maximum achievable credit of its AVB class, which is bounded and computable. This approach reduces WCRT pessimism compared to busy period analysis and applies to any number of AVB classes. However, it excludes ST interference and remains restricted to single output ports. Maxim et al. [14] extended the previous works to address ST interference without preemption.

The third approach employs Network Calculus, a mathematical framework for performance analysis in communication networks, to calculate the maximum delay, or WCRT, each frame can experience. Zhao et al. [17, 16] present a Network Calculus-based AVB WCRTA, allowing WCRT calculations from source to destination for multiple AVB queues with ST interference, both with and without preemption. Nevertheless, Network Calculus faces limitations when analyzing loop networks with circular dependencies and enforces equal bandwidth allocation for AVB classes across all links, reducing configuration flexibility. Other Network Calculus-based WCRTAs exist but were not included in this review due to limitations. For instance, [15] proposes a WCRTA tailored to a specific configuration with a single transmission window per traffic type and restrictions preventing concurrent AVB class A and B transmissions, limiting the CBS arbitration capabilities.

Finally, Bujosa et al. introduced a WCRTA in [4] that combines busy period and eligible interval analyses. This analysis, together with the proposals presented in [17], and [11], are the only ones that satisfy the requirements of this work, specifically providing source-to-destination WCRT calculations with the consideration for ST interference both with and without preemption. Due to its superior performance, we apply our improvement to the WCRTA proposed in [4] and compare the results with the previous WCRTA. Comparisons with the analysis based on busy periods are omitted, as the WCRTA combining busy periods and eligible intervals is an extension of it.

11.4 System Model

This section provides network and system models required for the response time analysis.

11.4.1 Network model

Connections between any end-station and any TSN switch, as well as between two TSN switches, are established through *link*s represented by l. The TSN ports operate in full-duplex mode, meaning that input and output operations are isolated. As a result, reception and transmission on the same physical port do not interfere with each other. Consequently, each physical port corresponds to two links: one for transmission and another for reception. Additionally, the duration from the reception of a frame at a TSN switch port to its queuing into the output port queue is unique to each TSN switch, denoted as ϵ. For the purposes of this analysis, the link delay attributed to the wire and its physical characteristics is considered negligible, while the overall network bandwidth, represented by BW, remains consistent across all network links. Lastly, for AVB traffic of priority X on a link l, the credit replenishment rate (idleSlope) is designated as $\alpha^{+}_{X,l}$, the credit consumption rate (sendSlope) is denoted by $\alpha^{-}_{X,l}$, and the credit value for traffic in priority X on link l is represented as $CR_{X,l}$.

11.4.2 Traffic model

We adopt a real-time periodic model for all types of traffic in TSN networks. This model defines a stream as a sequence of frames that share common attributes, including source and destination addresses, periods, and deadlines. Accordingly, a collection of N streams is characterized as follows:

$$\Gamma = \{m_i(C_i, T_i, D_i, P_i, \mathcal{L}_i, \mathcal{O}_i) | i = 1, \ldots, N\} \tag{11.1}$$

Within this model, C_i denotes the transmission time of a frame of the stream m_i, which is determined by the frame size and the network bandwidth. Note that the Ethernet frame header is included in the frame transmission time C_i, with the header transmission time indicated as v. This value is treated as constant and independent of frame type. For the ST, the guard band is included in C_i when necessary; notably, when multiple ST frames are transmitted sequentially without gaps between them, only the initial frame requires a guard band. Furthermore, T_i and D_i represent the period and relative deadline of the frames, respectively. We assume a constrained deadline model, meaning $D_i \leq T_i$. It is important to note that AVB traffic classes can be initiated either periodically or sporadically. In cases of sporadic transmissions, T_i signifies the minimum inter-arrival time, i.e. the shortest time between two frames of the stream m_i. The priority of a stream is indicated by $P_i \in \mathbb{P}$. The highest priority is assigned to ST, the lowest to BE, and all intermediate priorities are allocated to AVB. In this context, ST priority is denoted as P_{ST}, BE priority as P_{BE}, and AVB priorities as $\{P \in \mathbb{P} \mid P_{ST} > P > P_{BE}\}$. Additionally, $\mathbb{L}$ and $\mathbb{H}$ represent the sets of non-ST streams with lower and higher priority than m_i, respectively. Since a frame may traverse multiple links, the set of n links that m_i passes through is specified by $\mathcal{L}_i = \{\mathcal{L}_i(0), \ldots, \mathcal{L}_i(n)\}$.

Offsets are utilized to fit ST streams into the transmission schedule. The offset for each ST frame is defined per link, and the collection of offsets for all links traversed by m_i is given by $\mathcal{O}_i$, for example, $\mathcal{O}_i = \{O_i^l\}$. We assume that the offsets are predetermined, as the scheduling of ST streams is beyond the scope of this paper. Previous studies, such as [3], have already addressed this topic. It is important to note that AVB streams do not have defined offsets; thus, for AVB frames, $\mathcal{O}_i$ is represented as an empty set, i.e., $\mathcal{O}_i = \{\emptyset\}$.

In this model, we consider the TSN configuration where ST frames can preempt non-ST frames. This means that some frames may experience interruptions during transmission, which will be resumed later. Additionally, ST frames that arrive at a switch will remain in the output queue until their transmission window is activated. Specifically, traffic classes can be categorized as *express* or *preemptable*. In our system model, ST streams are classified as express, while all other classes are preemptable. This implies that the guard band can be set to 124 Bytes, representing the non-preemptable segment of any non-ST frame. This value indicates the maximum duration a preemptable frame (non-ST frame) can block an express frame (ST frame). This configuration is typical for TSN when all traffic types are considered, i.e. ST, AVB, and BE.

Lastly, when an ST frame preempts a non-ST frame, it resumes transmission with a new header. Thus, if a frame is preempted and divided into two segments, the second segment will also have a header, resulting in two headers needing to be accounted for. This has implications for the analysis of AVB traffic.

11.5 Worst-Case Response Time Analysis

This section offers an in-depth overview of the WCRTA based on eligible intervals [4]. As previously outlined, this WCRTA takes into account the interference from higher-priority streams, which encompasses ST interference with preemption and higher-priority AVB interference, same-priority interference, and blocking from lower-priority frames, including lower-priority AVB and BE. Furthermore, it considers the contributions of multi-hop behavior, providing the WCRT from the source to the destination.

11.5.1 WCRTA overview

The WCRTA based on busy periods and eligible intervals considers various sources of delay that an AVB frame of stream m_i may encounter on link l and adds them in a compositional way. First, since all queues in TSN operate on a FIFO basis, the transmission of a frame is contingent upon the preceding frames in the same queue. Consequently, the source of delay arises from interference with same-priority traffic, represented as SPI_i^l. Secondly, higher-priority AVB frames can interfere with lower-priority AVB frames, indicated by HPI_i^l, although this interference is somewhat limited due to the CBS. Furthermore, since AVB classes are non-preemptive, if frames from lower-priority classes are currently being transmitted, the AVB frame must wait for the entire transmission duration. This results in blocking caused by lower-priority traffic, denoted as LPI_i^l. Finally, the ST, via the TAS gates and according to the established schedule, can block the transmission of any AVB or BE queue. Given that the schedule is not uniform, it is essential to identify this interference, denoted as $STI_{i,c[k]}^l$, for all critical instant candidates $I_c^l[k]$ which correspond to the beginning of each transmission window of each frame k of each ST stream c on link l. In summary, to compute the WCRT of any AVB frames in the critical instant candidate $I_c^l[k]$, one must consider the contributions from ST, higher-priority AVB traffic, same-priority AVB traffic, and lower-priority traffic (both lower-priority AVB and BE). Thus, for a frame of stream m_i traversing link l at the critical instant candidate $I_c^l[k]$, the $WCRT_{i,c[k]}^l$ is expressed as:

$$WCRT_{i,c[k]}^l = STI_{i,c[k]}^l + HPI_i^l + SPI_i^l + LPI_i^l + C_i \tag{11.2}$$

Once the WCRT for a frame of m_i in link l is determined for each potential critical instant candidate $I_c^l[k]$, the WCRT of a frame of stream m_i in link l is established by identifying the maximum response time across all critical instant candidates. This means that the interference for each critical instant candidate must be assessed, and the highest value should be chosen as follows:

$$WCRT_i^l = \max_{\forall m_c, \forall k} \{ WCRT_{i,c[k]}^l \} \tag{11.3}$$

Finally, since the analysis is compositional, where a frame is buffered as it passes through each hop, we sum the $WCRT_i^l$ for each link along the path from the source to the destination of the frame m_i and add the ϵ factor for each switch crossed. This factor accounts for the delay incurred by the frame from its reception at the switch's input port until it is queued in the TSN output port. The overall WCRT for frame m_i is computed as follows:

$$WCRT_i = \sum_{l=1,\ldots,|\mathcal{L}_i|} WCRT_i^l + (|\mathcal{L}_i| - 1) \times \epsilon \tag{11.4}$$

In the subsequent subsections, we will discuss the worst-case scenarios for each of the components that contribute to the WCRT for an AVB frame m_i on link l.

11.5.2 Same Priority Interference

As outlined in [4], the delay experienced by an AVB frame of stream m_i on link l, when subjected solely to interference from same-priority traffic, i.e., $sp(m_i) = \{m_j | P_j = P_i, j \neq i\}$, is determined based on a basic FIFO scheduling approach. However, due to the behavior of the CBS, it is essential to account not only for the transmission time of each same-priority frame C_j, but also for the time needed to recover the credit consumed by those interfering frames. When only same-priority traffic is involved, the credit level cannot be bigger than 0, as that would imply interference or blocking from other traffic classes. Thus, same-priority frames can only be transmitted when the credit reaches 0. These frames will consume $C_j \times \alpha^-_{P_i,l}$ credit, which needs to be replenished over a duration of $C_j \times \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}$ for the credit to replenish to 0, allowing for the transmission of the next frame in the queue. Consequently, the total interference is the sum of the transmission time of the interfering frames and the time required to restore the credit consumed by each of these frames. In the worst-case scenario, a frame may be interfered with by all same-priority streams. However, given that the analysis operates under a deadline-constrained model, only one frame of each same-priority stream in the FIFO queue can interfere with the frame under analysis if all frames satisfy their deadlines, as discussed in the context of the Controller Area Network (CAN) [8]. In this way, similar to most analyses with constrained deadlines, the results lack reliability if any frame misses its deadline according to its WCRT. Therefore, the interference from same-priority frames on m_i of class P_i on link l is computed using Eq. (11.5).

$$SPI_i^l = \sum_{\substack{\forall m_j \in sp(m_i), i \neq j \\ \wedge l \in \mathcal{L}_j}} C_j \times \left(1 + \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}\right) \tag{11.5}$$

11.5.3 Higher-Priority AVB Interference and Lower-Priority Blocking

While higher-priority AVB interference and lower-priority blocking represent distinct contributions, the authors in [7] demonstrated that these delay contributions correspond to the time required to achieve the maximum credit CR^{max} for the AVB class of the analyzed stream m_i of priority P_i, expressed as:

$$HPI_i^l + LPI_i^l = \frac{CR_{P_i,l}^{max}}{\alpha_{P_i,l}^{+}}. \tag{11.6}$$

The authors also established that $CR_{P_i,l}^{max}$, and consequently $HPI_i^l + LPI_i^l$, remains bounded provided that the total bandwidth assigned to P_i and all higher-priority queues $\mathbb{H} = \{H \in \mathbb{P} \,|\, ST > H > P_i\}$ does not surpass the available bandwidth, i.e.:

$$\sum_{\forall P \in \mathbb{H} \cup P_i} \alpha_{P,l}^{+}[\%] \leqslant BW[\%]. \tag{11.7}$$

Given these conditions, the non-ST interference $HPI_i^l + LPI_i^l$ experienced by a frame can be computed as follows:

$$\begin{aligned} HPI_i^l + LPI_i^l &= \frac{CR_{P_i,l}^{max}}{\alpha_{P_i,l}^{+}} \\ &= C_{\mathbb{L},l}^{max} \times \left(1 + \frac{\alpha_{\mathbb{H},l}^{+}}{\alpha_{\mathbb{H},l}^{-}}\right) - \frac{CR_{\mathbb{H},l}^{min}}{\alpha_{\mathbb{H},l}^{-}} \end{aligned} \tag{11.8}$$

where $C_{\mathbb{L},l}^{max}$ represents the size of the largest frame from all lower-priority queues $\mathbb{L} = \{L \in \mathbb{P} \,|\, L < P_i\}$ and $CR_{\mathbb{H},l}^{min}$ is the minimum value that the combined credit of the highest priority queues can achieve on link l. This latter value is computed recursively as follows:

$$\begin{aligned} &CR_{\mathbb{H}=\{H_1,\ldots,H_n\},l}^{min} = \\ &\qquad -\max(\alpha_{\mathbb{H},l}^{-} \times C_{H_1,l}^{max} - CR_{\mathbb{H}-H_1,l}^{min}, \\ &\qquad\qquad \ldots, \alpha_{\mathbb{H},l}^{-} \times C_{H_n,l}^{max} - CR_{\mathbb{H}-H_n,l}^{min}) \end{aligned} \tag{11.9}$$

11.5.4 Scheduled Traffic Interference

As proven in [11], the starting time of each ST transmission window within the hyper-period must be considered as a critical instant candidate. Since every link has its unique hyper-period denoted as Ω_l, and the hyper-period for a set of frames is determined by the least common multiple of their respective periods, the specific instances relevant for assessing the ST interference of the frame $m_j \in ST$ on link $l \in \mathcal{L}_j$ are defined as:

$$I_j^l = \{(k-1)T_j + O_j^l \ : k = 1, \ldots, n \ , \ n = \frac{\Omega_l}{T_j}\} \tag{11.10}$$

After identifying all potential critical instants throughout the hyper-period for link l, the phase difference between each ST frame in $m_j \in ST$ and each potential critical instant $I_c^l[k]$ is calculated. These phase differences represent the offsets that different ST frames m_j would exhibit if the beginning of the hyper-period coincided with the critical instant candidate of frame k of stream $m_c \in ST$. For further details and supporting proofs, please refer to [12].

$$\Phi_{jc[k]}^l = (O_j^l - I_c^l[k]) \bmod T_j \tag{11.11}$$

Finally, for every critical instant candidate $I_c^l[k]$, the ST interference experienced by an AVB frame over time t is expressed as:

$$W_{c[k]}^l(t) = \sum_{\forall j \in ST \ \wedge l \in \mathcal{L}_j} \left(\left\lfloor \frac{\Phi_{jc[k]}^l}{T_j} \right\rfloor + \left\lceil \frac{t - \Phi_{jc[k]}^l}{T_j} \right\rceil \right) C_j \tag{11.12}$$

Additionally, AVB traffic may be preempted by each interfering ST frame, leading to the transmission of additional headers. Therefore, for every preemption caused by an ST frame, the added interference attributed to the header size v must be accounted for. Furthermore, these additional headers will consume credit that requires replenishment. Depending on whether the preemption affects a same-priority frame or a higher-/lower-priority frame, it will be weighted according to Eq. (11.5) or Eq. (11.8), respectively. In the worst-case scenario, the higher of the two cases will be selected, resulting in Eq. (11.13).

$$\begin{aligned} V_{c[k]}^{l}(t) = \sum_{\forall j \in ST \wedge l \in \mathcal{L}_j} & \left(\left\lfloor \frac{\Phi_{jc[k]}^{l}}{T_j} \right\rfloor + \left\lceil \frac{t - \Phi_{jc[k]}^{l}}{T_j} \right\rceil \right) v \\ & \times \left(1 + \max\left(\frac{\alpha_{P_i,l}^{-}}{\alpha_{P_i,l}^{+}}, \frac{\alpha_{\mathbb{H},l}^{+}}{\alpha_{\mathbb{H},l}^{-}} \right) \right) \end{aligned} \tag{11.13}$$

Consequently, the maximum ST interference that an AVB frame from stream m_i can encounter on link l at instant $I_c^l[k]$ over time t is computed in Eq. (11.14), which represents the total interference from ST and the additional headers resulting from preemption.

$$STI_{c[k]}^{l}(t) = W_{c[k]}^{l}(t) + V_{c[k]}^{l}(t) \tag{11.14}$$

In this manner, the response time of an AVB frame queued at the output port of link l during the critical instant candidate $I_c^l[k]$, represented as $WCRT_{i,c[k]}^{l,(x)}$, is iteratively calculated as follows:

$$\begin{aligned} WCRT_{i,c[k]}^{l,(x)} = & STI_{c[k]}^{l}\left(WCRT_{i,c[k]}^{l,(x-1)} \right) + \\ & HPI_i^l + SPI_i^l + LPI_i^l + C_i. \end{aligned} \tag{11.15}$$

The iteration starts with $WCRT_{i,c[k]}^{l,(0)} = HPI_i^l + SPI_i^l + LPI_i^l + C_i$ and concludes when $WCRT_{i,c[k]}^{l,(x)} = WCRT_{i,c[k]}^{l,(x-1)}$.

11.6 Problem Formulation

In most WCRTAs and the analysis presented in Section 11.5, a frame may be interfered with by one frame from each stream of the same priority under the assumption of constrained deadlines. However, this scenario can occur only at the transmitter's output port. In order to be interfered by all same-priority frames, those must arrive at the transmission queue simultaneously, just before the arrival of the frame under analysis. This situation is plausible at the talker's transmission queue, where applications may attempt to send frames concurrently, utilizing parallel resources. However, such simultaneous arrival of frames is unlikely at the switches, as frames are received sequentially through input ports. Consequently, frames of the same priority require reception times determined by the transmission rate non-null, leading to some frames being forwarded while the remaining frames are still being received. As a result, the maximum number of same-priority interferences will be less than or equal to the total number of frames of the same priority.

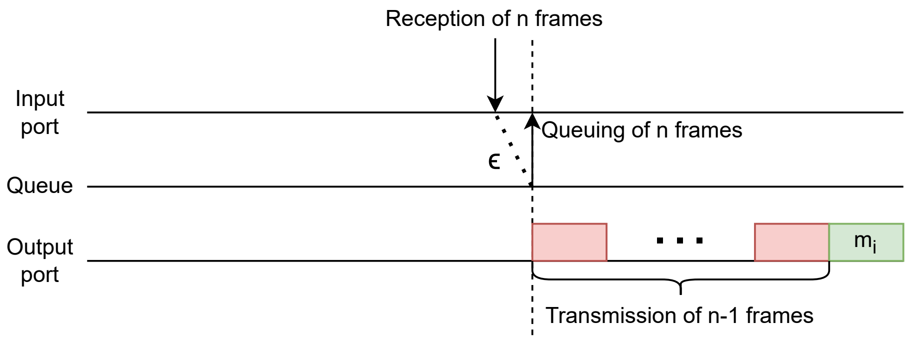


Fig. 11.2: WCRT of an AVB frame with a single input and output port and a single priority through traditional WCRTAs.

To analyze the problem, we will examine an extreme scenario involving a single switch with one input and output links. This switch receives and transmits traffic associated with a single AVB priority, i.e. all frames received are same-priority frames. Specifically, n streams of identical size C and period T are processed. Additionally, since all traffic is assigned the same priority, 100% of the bandwidth will be allocated to this priority, resulting in no credit recovery time. Fig. 11.2 shows the WCRT of a frame as calculated using the existing WCRTAs. In this figure, the horizontal lines represent the evolution of the input and output ports, along with the AVB queue. According to conventional analysis, one frame from each same-priority stream, including the frame under analysis, arrives almost simultaneously through the input link (indicated by the downward arrow). Consequently, by the time the frame under analysis is queued (indicated by the vertical dashed line), the $n-1$ preceding frames will have already entered (indicated by the upward arrow), resulting in a delay of $n \times C$. However, a closer examination of the switch's actual behavior reveals a different outcome, as illustrated in Fig. 11.3. This figure demonstrates that the reception of the n frames takes a finite amount of time, which the output link utilizes to retransmit those same-priority frames. Consequently, when the frame under analysis reaches the AVB queue, no same-priority frames are available for interference, leading to a delay of C, i.e., a delay n times smaller than the obtained through traditional WCRTA.

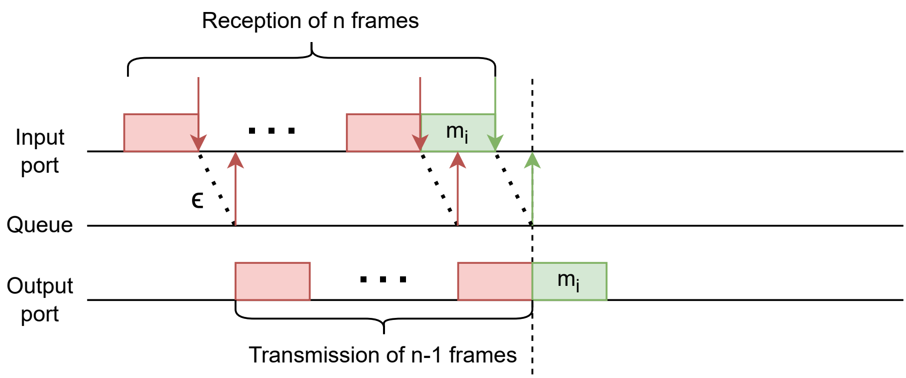


Fig. 11.3: WCRT of an AVB frame with a single input and output port and a single priority.

In the upcoming section, we will conduct a detailed analysis of the maximum same-priority interference that an AVB frame may encounter, reducing the pessimism of the calculation significantly.

11.7 Proposed Solution

This section outlines the main contributions of our work, presenting the lemmas and proofs that lead to the calculation of the maximum SPI.

The key concept in calculating the maximum SPI is to determine the minimum time the queue of the frame under analysis can transmit same-priority traffic before the frame under analysis' arrival time to the queue.

We begin the proof by analyzing the case without blocking or interference from other traffic types. Subsequently, we examine the interactions with other traffic to determine how they affect the calculation of the maximum SPI.

11.7.1 SPI without Blocking nor Interference from other Classes

To calculate the minimum time a queue can transmit in the absence of blocking and interference, two key aspects must be considered. First, we need to determine the minimum time required to receive all same-priority frames, with the frame under analysis being the last to be received. In the worst-case scenario, this represents the maximum time the output queue can transmit same-priority frames that would typically interfere with the frame under analysis.

Definition 11.7.1. *The Minimum Reception Time (MRT) is the shortest duration required to receive a set of frames across one or more communication links, considering transmission times and any dependencies between the frames, such as credit recovery in the case of AVB traffic.*

Second, it is essential to calculate the minimum elapsed time between any two frames of the same stream. If the interval between a frame of a same-priority stream and its predecessor is very short, it could result in the transmission of the preceding frame occurring during the reception time of the same-priority frames. This situation limits the transmission of same-priority frames that could interfere with the frame under analysis. A detailed analysis of this scenario is provided below.

Definition 11.7.2. *The Minimum Time Separation (MTS) is the shortest time interval that must elapse between the completion of a frame's transmission on a link and the start of the subsequent frame's transmission from the same stream on the same link.*

Lemma 11.7.1. *The minimum time necessary to receive the same priority frames as the frame under analysis (including the frame under analysis of stream m_i) through link $\{l' | \exists \mathcal{L}_j(x) = l' \,\&\, \mathcal{L}_j(x+1) = l\}$ that will be retransmitted by link l, referred to as Minimum Reception Time ($MRT_i^{l',l}$) is calculated* *as follows:*

$$\zeta = \Big(\sum_{\substack{\forall m_j \in sp(m_i) \\ \wedge l,l' \in \mathcal{L}_j}} C_j \Big) - C^{max}_{P_i,l'} \tag{11.16}$$

$$MRT_i^{l',l} = \max\left(\zeta \times \left(1 + \frac{\alpha^-_{P_i,l'}}{\alpha^+_{P_i,l'}} \right) - \frac{CR^{max}_{P_i,l'}}{\alpha^+_{P_i,l'}}, \zeta \right) \tag{11.17}$$

where $C^{max}_{P_i,l'} = \max\limits_{\forall m_j \in sp(m_i) \wedge l,l' \in \mathcal{L}_j} (C_j)$.

Proof. In the worst-case scenario, assuming no blocking nor interruptions, frames are received sequentially with an inter-frame interval that corresponds to the time required to recover the credit consumed during the transmission of each frame, as outlined in Section 11.5 and demonstrated in [6, 5, 7], i.e.: $\sum_{\substack{\forall m_j \in sp(m_i) \\ \wedge l,l' \in \mathcal{L}_j}} C_j \times \left(1 + \frac{\alpha^-_{P_i,l'}}{\alpha^+_{P_i,l'}} \right)$. Conversely, as illustrated in Fig. 11.3, the frame reception time for the transmission queue spans from the conclusion of the first frame's reception to the completion of the last frame's reception. Consequently, when calculating the total reception time, it is essential to exclude both the transmission time of the first frame and the credit recovery time of the last frame. In order to ensure a minimum frame reception time, the excluded times must be maximized. This is effectively equivalent to omitting the transmission and credit recovery time associated with the largest frame, denoted as $C^{max}_{P_i,l'}$, from the $MRT_i^{l',l}$ calculation. Additionally, the credit at the start of same-priority frame reception will be the maximum achievable by the queue, meaning that any credit already accumulated prior to transmission must be excluded from the credit recovery calculation, i.e. $-\frac{CR^{max}_{P_i,l'}}{\alpha^+_{P_i,l'}}$. However, this reduction in credit recovery time cannot result in a reception time shorter than the duration required to receive all same-priority frames, i.e. ζ. ■

In this regard, the minimum time necessary to receive all the same priority frames as the frame under analysis (including the frame under analysis of stream m_i) that will be forwarded by link l is calculated as follows:

$$MRT_i^l = \max(MRT_i^{l',l}) \tag{11.18}$$

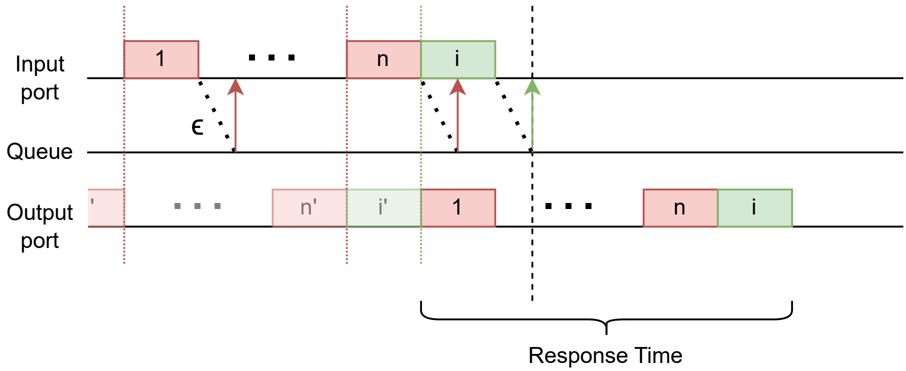


Fig. 11.4: WCRT of an AVB frame with a single input and output port and a single priority considering stream's previous frames.

During MRT_i^l, the link l can forward part of the same-priority traffic. However, a portion of this time might be used for transmitting previous frames of these same-priority streams. Figure 11.4 illustrates an extreme case where, just before the reception of each same-priority frame on the input link (frames $1..n, i$), a previous frame of the same stream is sent through the output link (frames $1'..n', i'$). This scenario consumes nearly all the MRT_i^l time in transmitting previous frames of the same-priority streams.

In this context, we must calculate the minimum temporal distance between two consecutive frames of the same stream. This calculation will determine whether a same-priority frame arriving at the queue before the frame under analysis could have a previous frame of the same stream transmitted during MRT_i^l. Specifically, we are interested in the temporal distance between the end of the transmission, including the recovery of the credit of a frame on link l, and the start of the transmission of the next frame of the same stream on the same link l.

Lemma 11.7.2. *The Minimum Temporal Separation (MTS_j^l) between the end of the transmission, including the recovery of the credit of a frame of stream m_j on link l, and the start of the transmission of the next frame of the same stream on the same link l occurs when one frame experiences the WCRT across the set* *of links $\mathcal{L}_j^l = \mathcal{L}_j(0), \ldots, \mathcal{L}_j(x) = l$, and the subsequent frame experiences the Best-Case Response Time (BCRT). This separation is calculated as follows:*

$$MTS_j^l = D_j + (|\mathcal{L}_j^l| - 1) \times C_j - C_j \times \frac{\alpha_{P_j,l}^-}{\alpha_{P_j,l}^+} - \sum_{\forall l'' \in \mathcal{L}_j^l} WCRT_j^{l''} \tag{11.19}$$

Fig. 11.5 shows a diagram of the MTS_j^l calculation.

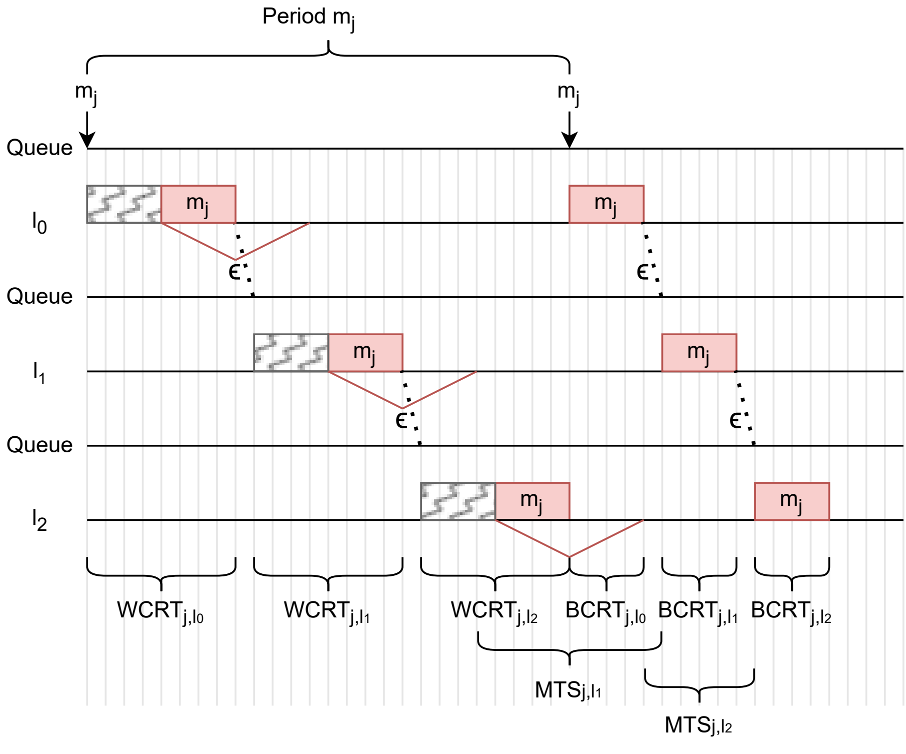


Fig. 11.5: Minimum temporal separation between two consecutive frames of the same stream.

Proof. First, since the time difference includes the same number of switches ($|\mathcal{L}_j^l| - 1$), we can exclude the ϵ factor of the switches from the calculation. On the other hand, $D_j - C_j \times \frac{\alpha^-_{P_j,l}}{\alpha^+_{P_j,l}} - \sum_{\forall l'' \in \mathcal{L}^l_j} WCRT^{l''}_j$ calculates the time between the worst-case transmission plus the replenishment of the credit on link l' and the end of the period, while $(|\mathcal{L}^l_j|-1) \times C_j$ calculates the best-case reception time of a frame by the output queue of link l. By combining both values we obtain the minimum temporal distance between two consecutive frames of the same stream, i.e. MTS^l_j. ■

Note that the transmission of same-priority streams before the arrival time of the frame under analysis does not apply to the first link in the path of the frame under analysis, i.e., $\mathcal{L}_i(0)$. For the first link, we will use the pessimistic assumption from the previous analysis (Fig. 11.2) since we cannot guarantee that the transmitter end-station application will not attempt to send all same-priority frames simultaneously. Consequently, it is also unnecessary to consider the MTS^l_j of frames that share the same source as the frame under analysis. Due to the constraint deadline condition, all their previous frames should have already been received by the time the transmission of the frame under analysis begins.

Lemma 11.7.3. *The minimum time the link l will be able to transmit same-priority traffic as the frame under analysis m_i (MTT^l_i) is:*

$$MTT^l_i = \min\left(MRT^l_i - C^{max}_{P_i,l} \times \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}, \min_{\substack{\forall m_j \in sp(m_i) \\ \wedge \exists \mathcal{L}_j(x) = \mathcal{L}_i(y) = l \\ \wedge \mathcal{L}_j(0) \neq \mathcal{L}_i(0)}} \left(MTS^l_j \right) \right) \tag{11.20}$$

Proof. Firstly, in the absence of previous frames from streams of the same priority as the frame under analysis, the minimum time that link l can transmit traffic of the same priority as the frame under analysis m_i is MRT^l_i minus the time necessary to recover the minimum credit of l, denoted as $C^{max}_{P_i,l} \times \frac{\alpha^-_{P_i,l}}{\alpha^+_{P_i,l}}$. This is because, in the worst-case scenario, we assume that at the beginning of the MRT^l_i, the credit is at its minimum, thereby limiting the transmission capacity of same-priority frames during the MRT^l_i. When same-priority streams with different sources converge on the path of the frame under analysis, part of the MRT_i^l will, in the worst case, be allocated to the retransmission of preceding frames.

For a frame to interfere with the frame under analysis, it must arrive at least just before the frame under analysis. Furthermore, Lemma 11.7.2 demonstrates that there is a minimum time interval between a frame and its predecessor. Consequently, in the worst-case scenario, the frame with the minimum MTS_j^l will have arrived just before the frame under analysis, implying that its predecessor frame was transmitted through link l at least MTS_j^l time units earlier. If the minimum MTS_j^l is bigger than $MRT_i^l - C_{P_i,l}^{max} \times \frac{\alpha_{P_i,l}^-}{\alpha_{P_i,l}^+}$ then link l will be able to transmit same-priority traffic during the whole $MTS_j^l - C_{P_i,l}^{max} \times \frac{\alpha_{P_i,l}^-}{\alpha_{P_i,l}^+}$; otherwise, previous frames of stream m_j leaves only MTS_j^l for the transmission of frames that may interfere with the frame under analysis. ■

Lemma 11.7.4. *In the absence of blocking and interference from other priorities, the transmission time of same-priority traffic MTT_i^l results in a reduction of SPI equivalent to its value, provided that it is either greater than or equal to 0 or less than or equal to SPI.*

Proof. During MTT_i^l in the absence of blocking and interference from other priorities, at least $MTT_i^l \times \frac{\alpha_{P_i,l}^+}{\alpha_{P_i,l}^+ + \alpha_{P_i,l}^-}$ same-priority frames will be forwarded, leading to a SPI delay reduction of

$$X \times \frac{\alpha_{P_i,l}^+}{\alpha_{P_i,l}^+ + \alpha_{P_i,l}^-} \times \left(1 + \frac{\alpha_{P_i,l}^-}{\alpha_{P_i,l}^+}\right) = X \tag{11.21}$$

of the frame under analysis. Consequently, MTT_i^l corresponds to SPI component reduction. ■

Therefore, the maximum SPI, assuming no blocking or interference, is:

$$NewSPI_i^l = SPI_i^l - \min\left(\max\left(MTT_i^l, 0\right), SPI_i^l\right) \tag{11.22}$$

11.7.2 SPI with Blocking and Interference from other Classes

The new analysis is divided into two phases: phase 1 (P1) corresponds to the transmission of same-priority frames (MTT_i^l) occurring before the reception of the frame under analysis, while the second phase (P2) corresponds to the blocking and interference affecting the frame under analysis after it has arrived at the transmission queue.

Lemma 11.7.5. *Any blocking or interference affecting link l during P1 has the same effect as if it had occurred during P2.*

Proof. Blocking or interference affecting link l during P1 will reduce the MTT_i^l transmission of same-priority frames by a certain amount of time. This reduction increases the $NewSPI_i^l$ (Eq. (11.22)) by the same amount, up to a maximum of MTT_i^l as demonstrated in the proof of Lemma 11.7.4. ■

As demonstrated in [7], the maximum blocking and interference from non-ST traffic experienced by the transmission queue, provided there is pending traffic, is defined by Eq. (11.6). Additionally, while same-priority traffic continues to accumulate in the transmission queue and until the frame under analysis is transmitted, the queue will consistently contain traffic, thus keeping the maximum levels of non-ST blocking and interference as in Eq. (11.6) for the combined phases P1 and P2. In other words, non-ST blocking and interference remain the same regardless of the phase in which it occurs (P1 or P2) and are therefore independent of the improvement.

On the other hand, it is necessary to account for the STI during MTT_i^l. Thus, when computing the $WCRT_{i,c[k]}^{l,(x)}$ in Eq. (11.15) using the new SPI value (i.e., $NewSPI$ from Eq. (11.22), the MTT_i^l value should be added to calculate the STI. After obtaining $WCRT_{i,c[k]}^{l,(x)}$ iteratively, the MTT_i^l value would then be subtracted again, as MTT_i^l occur before the reception of the frame under analysis and, therefore, do not count for the $WCRT_{i,c[k]}^{l,(x)}$ calculation. This is equivalent to calculating the $WCRT_i^l$ as in the previous analysis (Eq. (11.3)) and subsequently subtracting MTT_i^l, i.e.:

$$newWCRT_i^l = WCRT_i^l - \min\left(\max\left(MTT_i^l, 0\right), SPI_i^l\right) \tag{11.23}$$

However, MTT_i^l depends on MTS_j^l which depends on $newWCRT_i^l$. Therefore, we will start by calculating $WCRT_i^l$ and MRT_i^l for all AVB frames. Next, we will calculate $newWCRT_i^{l,(0)}$ without considering MTS_j^l for all AVB frames. Finally, we iterate using the formula

$$\begin{aligned} newWCRT_i^{l,(x)} &= WCRT_i^l \\ &- \min\left(\max\left(MTT_i^l\left(newWCRT_i^{l,(x-1)}\right), 0\right), SPI_i^l\right) \end{aligned} \tag{11.24}$$

for all AVB frames until $newWCRT_i^{l,(x)} = newWCRT_i^{l,(x-1)}$ for all frames. Once this is achieved, we calculate $WCRT_i$ as in Eq. (11.4) by substituting $WCRT_i^l$ with $newWCRT_i^l$.

11.8 Experimental Setup

This study utilizes the LETRA Evaluation Toolset [13] to assess the proposed SPI enhancement. LETRA is an extensive suite of integrated tools designed for automated experiments, focusing on the scheduling and schedulability analysis of TSN networks. This section outlines the LETRA configuration used in this research, including specific modifications made for this study. The configuration is depicted in Fig. 11.6. The input for the evaluation toolset includes the network's configuration, encompassing its topology and traffic characteristics.

We examine two network topologies, illustrated in Figs. 11.7 and 11.8, which follow a line-star topology. This topology is suitable for our analysis, as the only missing element that could affect the results is the presence of loops. However, since some of the compared WCRTAs do not support circular dependencies, loops were excluded to ensure a fair comparison. Network N1 consists of a compact network with 2 switches, each connected to 5 end-stations, while Network N2 features a larger network with 5 switches, each connected to 2 end-stations. These topologies are part of the LETRA input.

To keep experiment durations manageable, the network bandwidth was set to 100 Mbps. This setting ensures that the maximum allowed 300 frames can consistently reach the target utilization on nearly every link. Frame lengths were chosen from the range [500, 1500] B. The minimum and maximum allowed periods were set at 10,000 μs and 30,000 μs, respectively.

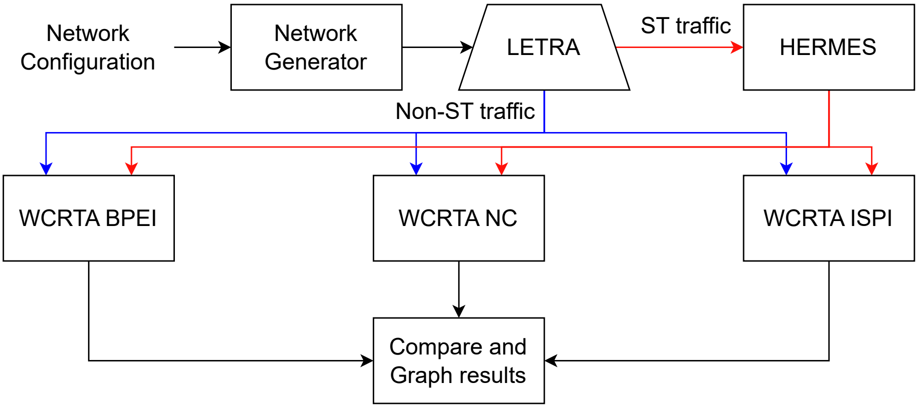


Fig. 11.6: LETRA configuration.

LETRA begins with the Network Generator, which creates random traffic based on the provided topology and traffic characteristics. We enforced a traffic distribution of 5% ST and 95% AVB Class A and Class B. For the experiments, the BE class and AVB priorities lower than Class B were omitted due to limitations in the compared WCRTAs. We evaluated the performance of the WCRTAs across various network utilizations, ranging from 5% to 45%. We conducted 100 traffic generations for each utilization level, resulting in 900 experiments. Each experiment involved analyzing up to 300 frames, totaling nearly 270,000 frames.

In the next step, the generated traffic is mapped into the different TSN traffic classes (ST, AVB Class A, and AVB Class B) using the Mapping Tool (Fig. 11.6). The ST traffic is scheduled using an existing heuristic algorithm [3], although any other ST scheduling algorithm could be used. We chose a heuristic algorithm for its balance between speed and feasibility.

The AVB traffic and the ST schedule serve as inputs for each of the compared WCRTAs. The WCRTAs compared are: the WCRTA based on busy period and eligible interval (BPEI) from [11], the WCRTA based on Network Calculus (NC) from [16], and the WCRTA with improved SPI (ISPI) proposed in this paper, which extends the BPEI method with a new SPI calculation. All WCRTAs were configured with AVB Classes' credit slopes ($\alpha^+_{P_A,l}$, $\alpha^-_{P_A,l}$,

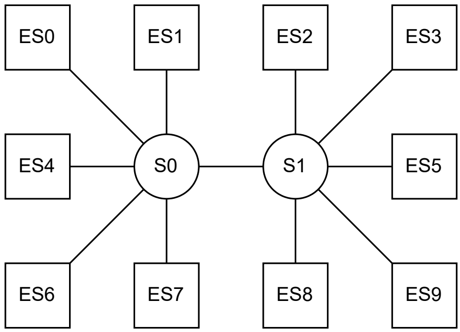


Fig. 11.7: Experimental network topology N1.

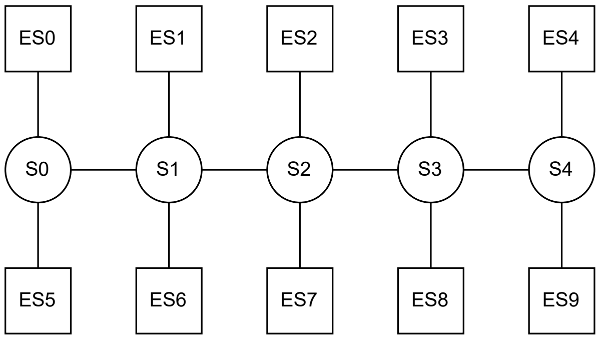


Fig. 11.8: Experimental network topology N2.

$\alpha^{+}_{P_B,l}$, and $\alpha^{-}_{P_B,l}$) set to 0.5, equally dividing the available bandwidth between AVB Classes A and B, in line with the experimental setup. Additionally, a switch traversal factor ϵ of 0 was used to ensure a fair comparison, as some analyses do not consider this factor.

Finally, the WCRTAs are compared using two methods. First, the schedulability of each WCRTA is determined for each bandwidth utilization, calculated as the percentage of generated networks that meet their time requirements according to each WCRTA. Second, the pessimism ratio between each previous WCRTA and the one proposed in this paper is calculated for each bandwidth utilization. Specifically, for each AVB-generated frame m_i, the pessimism ratio of WCRTA X to WCRTA ISPI is computed as follows:

$$\textit{Pessimism Ratio}_i = \frac{WCRT\ X_i}{WCRT\ ISPI_i} \tag{11.25}$$

This value demonstrates the extent to which our proposed SPI improvement (WCRTA ISPI) reduces pessimism compared to the analyses named X, specifically BPEI and NC. A pessimism ratio below 1 indicates an increase in pessimism in the SPI calculation, while a ratio above 1 indicates the degree to which the proposed SPI improvement reduces pessimism.

Given the random traffic generation and the large number of experiments conducted, we can confidently state that these experiments do not favor any particular WCRTA. Consequently, the results accurately reflect the performance differences among the compared WCRTAs under the specified topologies and traffic characteristics. Although the improvement over WCRTA BPEI is analytically validated, ensuring the improvement over WCRTA NC in all TSN topologies and traffic configurations is more challenging due to the inherent differences between the analyses.

11.9 Results and Discussion

This section presents the results of the experiments described in Section 11.8. We begin by presenting the schedulability results of the three WCRTAs (BPEI, NC, and ISPI) across the two network topologies for the various utilization percentages. Following this, we analyze the additional pessimism observed in WCRTA BPEI and NC compared to WCRTA ISPI, as this higher pessimism contributes to their lower schedulability.

11.9.1 Schedulability Results

Figs. 11.9 and 11.10 illustrate the schedulability percentages achieved by each evaluated WCRTA across different bandwidth utilizations for networks N1 and N2, respectively.

The results clearly demonstrate that the WCRTA incorporating the improved SPI calculation introduced in this study, consistently outperforms the

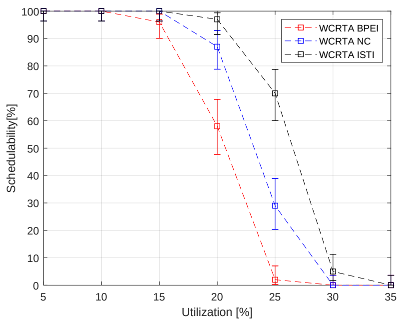


Fig. 11.9: Schedulability of the WCRTAs in network N1.

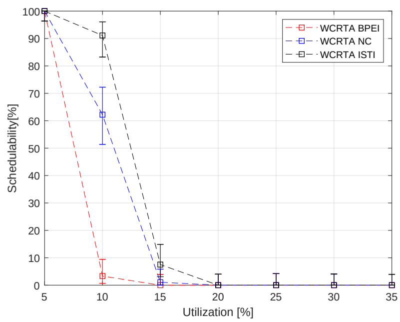
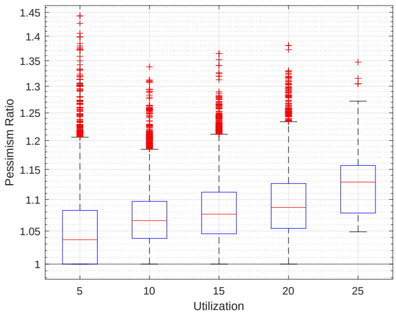


Fig. 11.10: Schedulability of the WCRTAs in network N2.

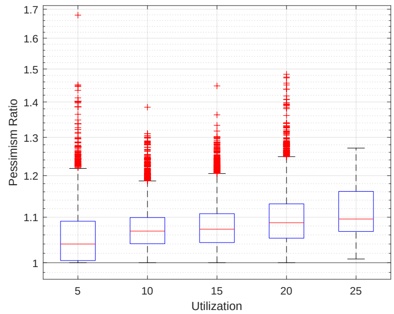
previous WCRTAs in terms of schedulability. Specifically, the proposed WCRTA ISPI achieves up to 90% higher schedulability compared to WCRTA BPEI and up to 40% higher schedulability at certain utilization levels when compared to WCRTA NC. It is also important to highlight that, despite the general decrease in network schedulability with an increase in network hops (Fig. 11.10), the proposed WCRTA continues to exhibit superior performance over the previous solutions.

11.9.2 Additional Pessimism

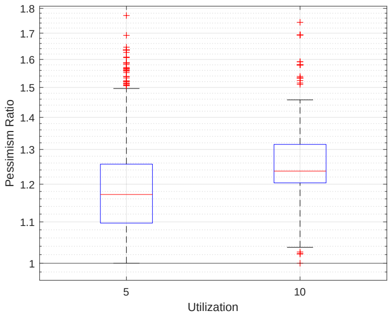
Figs. 11.11 and 11.12 illustrates the additional pessimism observed in the WCRTA BPEI and NC compared to the WCRTA ISPI across all analyzed networks and utilization. The figures show box plots formed using the pessimism ratios of all the frames analyzed for each schedulable utilization percentage. It is important to note that the WCRT values are only valid for schedulable networks, as the studied WCRTAs are based on the constrained deadline condition. Therefore, if any frame misses its deadline, the WCRT value becomes unreliable. The x-axis represents different utilization percentages, while the y-axis indicates the pessimism ratio. A horizontal dashed line is included at a pessimism ratio of 1, serving as a reference point. Values above 1 signify greater pessimism in previously proposed WCRTAs compared to the proposed



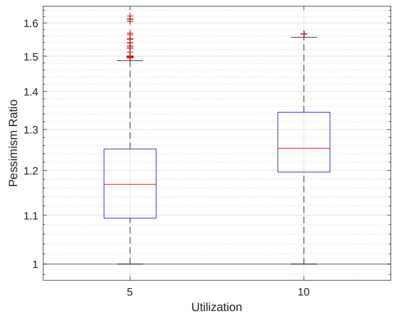
(a) AVB Class A traffic of WCRTA BPEI on networks N1



(b) AVB Class B traffic of WCRTA BPEI on networks N1



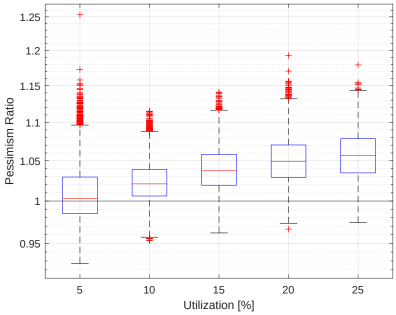
(c) AVB Class A traffic of WCRTA BPEI on networks N2



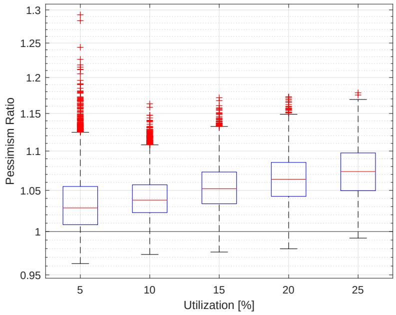
(d) AVB Class B traffic of WCRTA BPEI on networks N2

Fig. 11.11: Additional pessimism observed in the WCRTA BPEI compared to the WCRTA ISPI across all analyzed networks and utilizations.

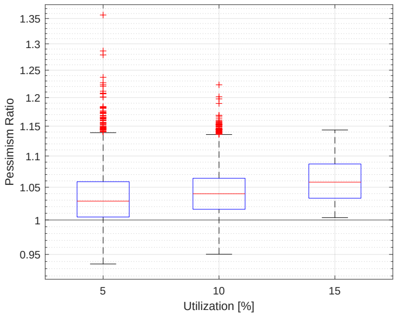
solution, whereas values below 1 indicate that the proposed solution introduces more pessimism relative to the previously compared WCRTA (BPEI or NC). The figure clearly demonstrates that, in all cases, the BPEI and NC WCRTAs exhibit greater pessimism than the WCRTA ISPI, which accounts for the higher schedulability of the latter.



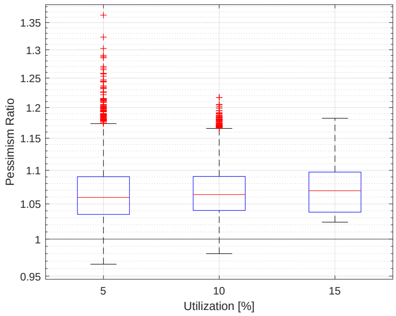
(a) AVB Class A traffic of WCRTA NC on networks N1



(b) AVB Class B traffic of WCRTA NC on networks N1



(c) AVB Class A traffic of WCRTA NC on networks N2



(d) AVB Class B traffic of WCRTA NC on networks N2

Fig. 11.12: Additional pessimism observed in the WCRTA NC compared to the WCRTA ISPI across all analyzed networks and utilizations.

11.10 Conclusion

Reducing pessimism in current Audio-Video Bridging (AVB) Worst-Case Response Time Analysis (WCRTA) is crucial for enhancing the practicality of Time-Sensitive Networks (TSN) and, consequently, its adoption by the industry. This paper addresses one of the primary sources of pessimism in existing analyses: the Same-priority Interference (SPI). We demonstrate that in TSN, AVB frames do not need to be interfered with by all same-priority frames at each switch, but rather by a smaller subset of them. This significant reduction in pessimism leads to an increase in the schedulability of the analysis when using the improved SPI calculation, compared to those that do not incorporate this improvement.

Bibliography

[1] IEEE 802.1Q-2022: IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks, 2022.

[2] Unmesh D Bordoloi, Amir Aminifar, Petru Eles, and Zebo Peng. Schedulability analysis of Ethernet AVB switches. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[3] Bujosa, Daniel and Ashjaei, Mohammad and Papadopoulos, Alessandro V and Nolte, Thomas and Proenza, Julián. HERMES: Heuristic multi-queue scheduler for TSN time-triggered traffic with zero reception jitter capabilities. In *Proceedings of the 30th International Conference on Real-Time Networks and Systems*, pages 70–80, 2022.

[4] Bujosa, Daniel and Proenza, Julián and Papadopoulos, Alessandro V and Nolte, Thomas and Ashjaei, Mohammad. An Improved Worst-Case Response Time Analysis for AVB Traffic in Time-Sensitive Networks. In *Proceedings-Real-Time Systems Symposium*, 2024.

[5] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent yet tight WCRT analysis for individual priority classes in ethernet AVB. In *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, pages 55–64, 2016.

[6] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Tight worst-case response-time analysis for ethernet AVB using

eligible intervals. In *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pages 1–8. IEEE, 2016.

[7] Cao, Jingyue and Cuijpers, Pieter JL and Bril, Reinder J and Lukkien, Johan J. Independent WCRT analysis for individual priority classes in Ethernet AVB. *Real-Time Systems*, 54:861–911, 2018.

[8] Robert I Davis, Steffen Kollmann, Victor Pollex, and Frank Slomka. Controller Area Network (CAN) Schedulability Analysis with FIFO Queues. In *2011 23rd Euromicro Conference on Real-Time Systems*, pages 45–56. IEEE, 2011.

[9] Jonas Diemer, Jonas Rox, and Rolf Ernst. Modeling of Ethernet AVB networks for worst-case timing analysis. *IFAC Proceedings Volumes*, 45(2):848–853, 2012.

[10] Jonas Diemer, Daniel Thiele, and Rolf Ernst. Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching. In *7th IEEE International Symposium on Industrial Embedded Systems (SIES'12)*, pages 1–10. IEEE, 2012.

[11] Lucia Lo Bello, Mohammad Ashjaei, Gaetano Patti, and Moris Behnam. Schedulability analysis of Time-Sensitive Networks with scheduled traffic and preemption support. *Journal of Parallel and Distributed Computing*, 144:153–171, 2020.

[12] J. Maki-Turja and M. Nolin. Fast and tight response-times for tasks with offsets. In *17th Euromicro Conference on Real-Time Systems (ECRTS'05)*, 2005.

[13] Mateu, Daniel Bujosa and Ashjaei, Mohammad and Papadopoulos, Alessandro V and Proenza, Julian and Nolte, Thomas. LETRA: Mapping legacy ethernet-based traffic into TSN traffic classes. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2021.

[14] Maxim, Dorin and Song, Ye-Qiong. Delay analysis of AVB traffic in time-sensitive networks (TSN). In *Proceedings of the 25th International Conference on Real-Time Networks and Systems*, pages 18–27, 2017.

[15] Mohamed Seliem, Ahmed Zahran, and Dirk Pesch. Delay analysis of TSN based industrial networks with preemptive traffic using network calculus. In *2023 IFIP Networking Conference (IFIP Networking)*, pages 1–9. IEEE, 2023.

[16] Luxi Zhao, Paul Pop, Zhong Zheng, Hugo Daigmorte, and Marc Boyer. Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus. *IEEE Transactions on Industrial Electronics*, 68(10):10291–10302, 2020.

[17] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. Timing analysis of AVB traffic in TSN networks using network calculus. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 25–36. IEEE, 2018.

Chapter 12

Paper E
TALESS: TSN with Legacy End-Stations Synchronization.

Daniel Bujosa, Julián Proenza, Alessandro V. Papadopoulos, Thomas Nolte, Mohammad Ashjaei.
In the 5^{th} Open Journal of the Industrial Electronics Society volume (OJIES 2024).

Abstract

In order to facilitate the adoption of Time Sensitive Networking (TSN) by the industry, it is necessary to develop tools to integrate legacy systems with TSN. In this paper, we propose a solution for the coexistence of different time domains from different legacy systems, each with its corresponding synchronization protocol, in a single TSN network. To this end, we experimentally identified the effects of replacing the communications subsystem of a legacy Ethernet-based network with TSN in terms of synchronization. Based on the results, we propose a solution called TALESS (TSN with Legacy End-Stations Synchronization). TALESS can identify the drift between the TSN communications subsystem and the integrated legacy devices (end-stations) and then modify the TSN schedule to adapt to the different time domains to avoid the effects of the lack of synchronization between them. We validate TALESS through both simulations and experiments on a prototype. We demonstrate that thanks to TALESS, legacy systems can synchronize through TSN and even improve features such as their reception jitter or their integrability with other legacy systems.

12.1 Introduction

Since the creation of the IEEE Time Sensitive Networking (TSN) Task Group (TG) in 2012, industry interest in TSN has continued growing. TSN seems to be essential for the incipient Industry 4.0 [29] as well as of interest in various areas such as automotive [21] and energy distribution [20]. The reason behind this growing interest is that TSN establishes a set of standards to provide deterministic zero-jitter and low-latency transmission, fault tolerance mechanisms, and advanced network management, allowing dynamic reconfiguration, precise clock synchronization, and flexibility in traffic transmission. The latter property is particularly relevant to the industry's adoption of TSN. This is so because the flexibility in the traffic transmission allows the transmission of different types of traffic over the same physical links, which enables the migration of all kinds of legacy traffic to TSN. Thereby, most legacy devices and implemented solutions could be kept, reducing adoption time and costs.

Furthermore, most current networks are composed of different sub-networks with different communication protocols to meet their specific requirements. This hinders communication between sub-networks and, therefore, their integrability. Moreover, this increases the complexity of the overall network due to the use of different technologies, cabling redundancy, etc. Thanks to the TSN's flexibility in traffic, it is possible to combine different types of traffic in the same network, facilitating the communication and integration of the sub-networks. This integration can be done in different ways, such as through the use of gateways. However, this would not allow sub-networks to take advantage of other TSN features, such as higher bandwidth or low jitter. Therefore, we propose to replace the communications subsystem of the legacy network directly, i.e., the set of devices exclusively responsible for communication, excluding the end-stations, with TSN but in such a specific way that the legacy end-stations can maintain their behavior and communication protocols (including their legacy synchronization protocol) agnostic to the change.

This approach improves the integrability of the different legacy systems and allows them to benefit from the advantages of TSN. For example, in recent years, the automotive industry has witnessed a substantial surge in complexity, driven by growing interest in vehicle automation. This complexity extends to the embedded networks within vehicles, as an increasing number of devices require increasing exchange of information with diverse and demanding timing requirements. Modern vehicles can now integrate hundreds of devices across various communication networks, with certain devices featuring multiple output ports for coordinating actions across different network layers like multimedia, data transmission, and comfort control. Integrating these networks through TSN offers a solution to reduce system complexity significantly. This integration streamlines manufacturing and maintenance processes and facilitates the addition of new devices to enhance functionality. Other benefits could be weight reduction and access to TSN advantages for both new and legacy devices, including fault-tolerance mechanisms and zero-jitter reception. This transformation paves the way for achieving unprecedented levels of automation and meeting previously unattainable requirements for legacy systems. However, certain types of TSN traffic, such as Time-Triggered (TT) traffic, require the path from the source to the destination, including all switches in the network, to be synchronized to have a common time view. This is because TT traffic is transmitted according to a fixed time schedule that needs to be known and respected by all network components. This requirement is not fulfilled in many industrial networks where non-TSN nodes (*end-stations* in TSN terminology) do not feature TSN synchronization mechanisms and may be unable to support them due to their hardware or software limitations.

For instance, in the automation pyramid depicted in Figure 12.1 [4], it can be seen how it is composed of the Field, the Supervisory Control And Data Acquisition (SCADA), and the Enterprise levels. Each of these levels entails distinct temporal constraints, data transmission volumes, and varying numbers of components, among other factors. Traditionally, this has been addressed using several networks with different characteristics connected through gateways with limited inter-network connectivity like PROFINET [11], EtherCAT [14], or Sercos [24] (Industrial Ethernet) for real-time traffic at the lower levels of the pyramid and Ethernet for higher levels of the pyramid. However, thanks to TSN, this separation is no longer necessary, as a single network can handle all types of traffic. Nonetheless, in an established factory, the cost-effectiveness of replacing all end-stations with TSN end-stations may not be cost-effective. Additionally, each sub-network mentioned above employs a distinct synchroniza-

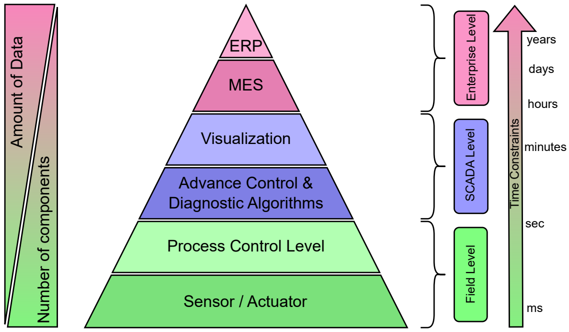


Fig. 12.1: Automation Pyramid.

tion protocol, none of which may be the Generalized Precision Time Protocol (gPTP) [2] used in TSN. Moreover, a recent study has quantified, in terms of loss of transmitted frames, the impact caused by mixing in the same network components that use different synchronization protocols [8]. Figure 12.2 depicts the number of frames lost over time in a heterogeneous network, wherein legacy end-stations synchronized through a legacy synchronization protocol other than gPTP communicate over a TSN network. These results highlight the need for a mechanism capable of harmonizing the different synchronization mechanisms that can coexist in a heterogeneous TSN network.

For the remainder of the paper, we will use the term *legacy network* for each original network, i.e., before replacing its communications subsystem with TSN. On the other hand, the term *legacy end-stations* will be used in reference to any node of any legacy network that has been integrated by replacing its communications subsystem with TSN, while the term *legacy system* will refer to the set of end-stations that were initially part of the same legacy network and are synchronized through its own legacy synchronization protocol, thus sharing a common time view. Figure 12.3 illustrates the terminology introduced with a diagram. It shows how two *legacy networks*, one with bus topology (Legacy Network 1) and one with ring topology (Legacy Network 2), are integrated into a single TSN network. This network is formed by the TSN communications subsystem and by two *legacy systems*, which in turn are created by the set of *legacy end-stations* from the *legacy networks*.

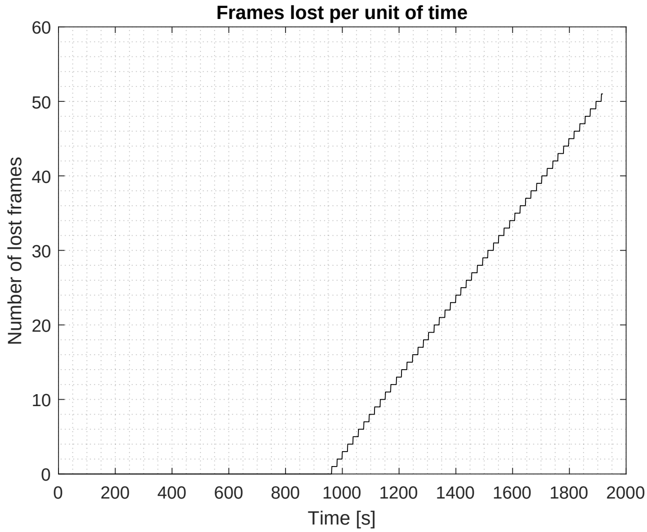


Fig. 12.2: Loss of frames per unit of time due to positive clock drift.

The critical piece for achieving the above-indicated integration of the legacy end-stations with the new TSN communication subsystem is a novel mechanism we propose in this paper. This mechanism is called TALESS (TSN with Legacy End-Stations Synchronization), and it is devised to prevent the negative effects resulting from the lack of synchronization between the TSN communications subsystem and the legacy systems integrated with it. TALESS transparently improves network performance without requiring modifications to legacy systems. To achieve this, rather than synchronizing the end-stations with TSN, which would necessitate modifications to the software or hardware of these devices, we tailor the TSN schedule to accommodate the unique time domains of each legacy system integrated into the TSN network. As we will see in Section IV, the main consequence of the lack of synchronization is clock drift, which is the root cause of the adverse effects. Clock drift causes two clocks to progress at different rates, leading to a clock skew, i.e. an accumulated discrepancy over time [15]. Figure 12.4 visually illustrates this behavior. Regarding frame transmission and reception, traffic drift refers to the variance

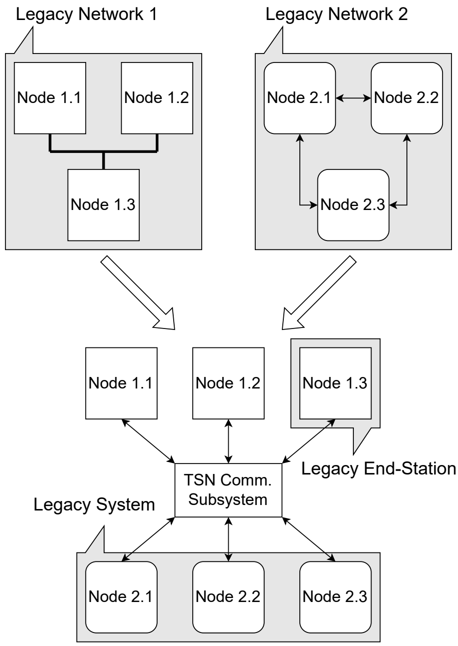


Fig. 12.3: Terminology used in the integration of legacy networks in TSN. Integration of two legacy networks, Legacy Network 1 with a bus topology and Legacy Network 2 with a ring topology, into a single TSN heterogeneous network combining two legacy systems consisting of the legacy end-stations from the two legacy networks.

between a frame's real transmission or reception period and its scheduled one, whereas traffic skew denotes the temporal disparity between the expected or scheduled reception time of a frame and its actual reception time. In this context, TALESS does not aim to remove the drift between the TSN clock and the legacy system clocks, but rather to eliminate the effects of this drift by adapting the TSN schedule.

This approach allows legacy systems to maintain their distinct time domains while benefiting from the enhancements offered by TSN, ensuring seamless operation without compromising its previous functionality and potentially

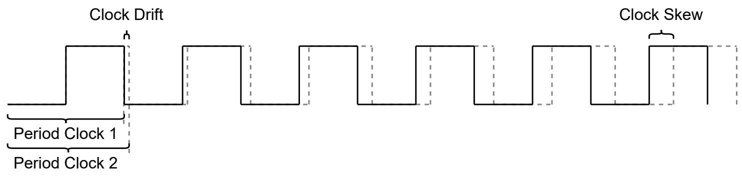


Fig. 12.4: Clock Drift and Clock Skew.

improving it. Preventing any modifications in the legacy end-stations makes TALESS a general solution that allows applying the proposed integration approach on any TSN network where several Ethernet-based legacy systems communicate with different communication protocols. On the one hand, we validate it using a model that simulates long executions (1 year) of a communication network. This model simulates the behavior of the TSN network with and without TALESS for different types of legacy end-station transmissions. On the other hand, we implement TALESS in a network prototype by which we experimentally validate the solution and verify both the implementation and its simulation model. However, in our experiments, we amplified certain clock parameters of the legacy system to magnify the effects of the solution and thereby be able to demonstrate its behavior in a reasonable run-time.

Contributions. As indicated, among the different requirements for integrating legacy systems into a TSN network, an essential aspect is clock synchronization. Maintaining proper communication behavior among devices is needed, especially if these devices require TT traffic transmissions. Thus, the main target of this paper is to develop a mechanism to avoid the adverse effects of carelessly putting together legacy systems with TSN in terms of clock synchronization. The main contributions of this paper are as follows:

- We identify problems caused by the lack of synchronization through experiments on a network prototype.
- We propose a mechanism named TALESS to remove the effects of lack of synchronization when including legacy systems into a TSN network.
- We model TALESS to validate the effectiveness of the proposed solution in a simulation environment with realistic network values.

- Finally, we implement TALESS in a network prototype to experimentally showcase its impact on utilizing legacy systems in a TSN network. We also compare the experiment results with the simulation model to verify the implementation and the simulation.

Outline. The paper is organized as follows. Section 12.2 presents the related work. Section 12.3 provides the necessary background to understand this paper better. Section 12.4 presents the effects of carelessly including legacy systems into a TSN network in terms of clock synchronization. Section 12.5 proposes TALESS. Section 12.6 presents the simulation model and experimental setup used to validate TALESS, while Section 12.7 presents the results obtained from both the simulations of the model and the experiments on the prototype. Finally, Section 12.8 concludes the paper and presents future directions.

12.2 Related Work

One of the most crucial aspects of TSN technology is clock synchronization. However, to our knowledge, no work has provided a solution to the adverse effects caused by the lack of synchronization in heterogeneous TSN networks that combine one or more Ethernet-based legacy systems through a TSN communications subsystem. On the contrary, most studies aim to integrate TSN with wireless and 5G networks. For example, a low-overhead beacon-based time synchronization method was implemented to provide precise synchronization in wireless networks in highly deterministic TSN networks, as outlined in [13]. Other research has focused on extending IEEE 802.1AS and IEEE 802.11 to enable TSN integration with wireless networks, as described in [5] and [19]. The challenges of integrating Wired TSN and WLAN technologies and a possible solution in a hybrid TSN device architecture were discussed in [23]. Moreover, the study in [12] presented TSN clock synchronization that aligns with 5G specifications. To solve cross-domain clock synchronization issues in 5G-TSN networks, a method based on data packet relay was proposed in [9]. Finally, the performance of 5G-TSN networks was also evaluated in terms of clock synchronization in several works such as in [26, 25], and in [27].

On the other hand, limited research explores synchronization in heterogeneous TSN networks, i.e., networks that incorporate TSN and non-TSN devices. For instance, [30] presents a method for preserving synchronization across TSN sub-networks connected through non-TSN switches. Their approach involves estimating the delays experienced by the synchronization messages passing through these devices and configuring TSN networks to minimize these delays. In contrast, our work focuses on integrating legacy systems into a single TSN network. Notably, there are even fewer studies addressing synchronization between legacy end-stations and TSN. [7] presents one methodology for integrating EtherCAT and TSN in terms of clock synchronization. However, this type of integration requires customized solutions for each integrated protocol, which can pose a challenge to the broader adoption of TSN by the industry. This is because designing and implementing these solutions take significant time and resources, and compatibility between solutions can also be demanding. Note that our approach differs from these solutions. In our proposed TSN heterogeneous networks, legacy systems are not synchronized with TSN, as they operate on distinct synchronization protocols. Instead, TSN adjusts to the clock timing of legacy systems to mitigate the negative impacts of the lack of synchronization.

Regarding the integration of legacy systems, several papers have proposed solutions for integrating TSN with different proprietary field buses. For example, paper [28] proposes a migration method for SERCOS III into TSN. However, as the synchronization mechanisms of both protocols are incompatible, the authors opted to adopt TSN's gPTP on SERCOS III devices. Furthermore, paper [17], also on integrating SERCOS III over TSN, limited the synchronization and integration to be only between the master and the TSN network, leaving the slaves disconnected to TSN. Regarding the integration of TSN with PROFINET, in paper [22], the authors propose a new type of switch that allows the mapping of PROFINET traffic on TSN. However, it does not prevent clock drift between TSN and PROFINET end-stations with the possible adverse effects that this would entail. Finally, [6] and [31] introduce TSN schedulers designed for unscheduled and unsynchronized legacy traffic exhibiting high jitter. However, they are unable to guarantee zero-jitter reception and do not consider clock drift. Both studies operate under the assumption of a constant period for legacy frames. Nevertheless, due to drift, this assumption may not hold true from the perspective of TSN. All these TSN integration works with different legacy systems could benefit from the solution proposed in this work since synchronization limitations would be avoided.

In a work presented in [8], the authors implemented a non-TSN network with its own synchronization protocols and replaced its communications subsystem with TSN. The work preliminary identified the effects of the lack of synchronization between the legacy system and the TSN network due to the lack of integration between the synchronization protocols used by the legacy system and the TSN's gPTP. Through several experiments, authors detected the causes and consequences of the network's lack of synchronization in the short and long term. However, the work was a short paper that merely suggested uncertain and indeterminate solutions that lacked implementation and proper validation.

12.3 Background

In TSN networks, communication between end-stations is achieved by transmitting Ethernet frames along Ethernet links and TSN switches. In TSN switches and end-stations, each output port has up to 8 FIFO queues, each corresponding to one specific priority level. TSN frames are assigned to one of the 8 priorities, or queues, configured as one of the 3 types of TSN traffic, including TT, Audio Video Bridging (AVB), and Best-Effort (BE) traffic. TT traffic is commonly given the highest priority, while BE traffic has the lowest priority. Several queues can be configured as the same type of traffic, thus giving different classes, for example, AVB class A, B, and C. An illustration of these concepts can be seen in Figure 12.5, which shows a TSN device (either an end-station or switch) output port with four queues configured to convey two TT traffic classes with the highest priority, one AVB traffic class with medium priority, and BE traffic with the lowest priority. As we will discuss later, TT traffic relies on the Time Aware Shaper (TAS) for transmission isolation, ensuring zero blocking and interference, resulting in the transmission according to the schedule with zero jitter. In contrast, AVB traffic utilizes both TAS to avoid blocking TT traffic and CBS to restrict the maximum bandwidth for each AVB queue, improving lower priority queues' quality of service. Lastly, BE traffic

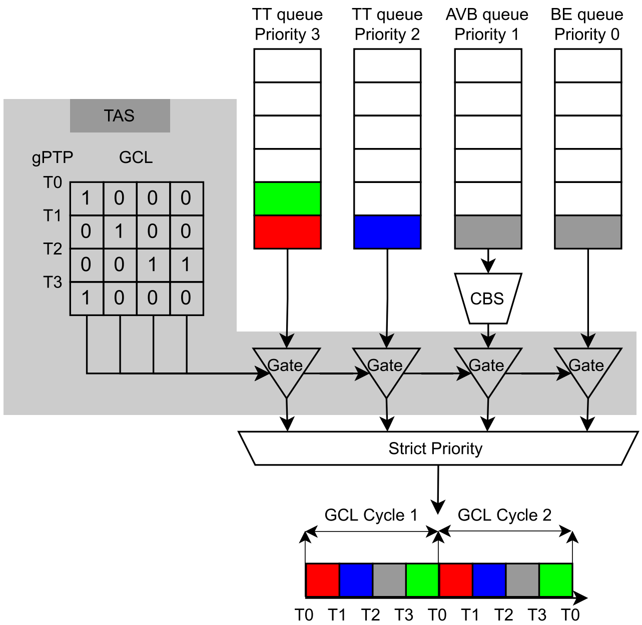


Fig. 12.5: A TSN egress port with four FIFO queues: two TT queues, one AVB queue, and one BE queue.

also utilizes TAS, hence it can transmit only when TT traffic is not transmitting and after AVB traffic has utilized its allocated bandwidth since it has the lowest priority.

Next, we explain three critical aspects of the background for this work. First, we will introduce the TAS and the gPTP since they are the main mechanisms responsible for TT transmission, which is the type of traffic most affected by the lack of synchronization. On the other hand, we will explain the Centralized Network Configuration (CNC) element, a key component for TALESS implementation. We will not delve deeper into CBS and the other traffic classes (AVB and BE) as they are not relevant to this study, given their synchronization-independent operation.

12.3.1 Time Aware Shaper

To provide the determinism required by TT traffic and, therefore, to know exactly when each TT frame is transmitted, TSN must be able to prevent inter-frame interference. To do this, TSN uses the TAS mechanism shown in Figure 12.5. This mechanism assigns a gate to each queue that can be open or close. The state of the gate is determined by the Gate Control List (GCL), which specifies at the nanosecond level how long a gate should be open or closed in a cyclically repeating list. If the gate of a queue is open, it can transmit the traffic in the queue. Otherwise, the frames in that queue are blocked from transmission. The opening period of a gate is called a *transmission window* or simply a *window*.

The operation of TAS for two TT queues is also depicted in Figure 12.5. In this example, three TT frames with a period of 4 time units and transmission time of 1 time unit are transmitted through a TSN switch port, where two of the frames are assigned the highest priority 3 (green and red), and one frame (blue) is assigned priority 2. The hyper-period, the least common multiple of the frames' periods, is calculated to schedule the transmissions. This value is used to define the GCL cycle, which controls the transmission of the frames by specifying the open or closed state of the gates associated with each priority queue. Thus, the GCL cycle in this example is set to 4 time units; hence, the list will be repeated every 4 time units. From time T0 to T1, the gate for priority 3 queue is open, allowing the transmission of the red frame, while the gate for the other queues remains closed. From T1 to T2, the blue frame, which has priority 2, can be transmitted as its gate is open. Both gates are closed between T2 and T3, resulting in no TT transmission but allowing lower priority queues to transmit even if higher priority frames are waiting for transmission. Finally, the gate for the priority 3 queue is open in the last transmission window, allowing the transmission of the green frame. The bottom of Figure 12.5 displays two cycles of frame transmissions, which shows the repetition of the GCL list.

12.3.2 Generalized Precision Time Protocol

The mechanism providing the TSN clock synchronization (gPTP) is described in the IEEE 802.1AS standard. It consists of three main parts, including the Best Master Clock Algorithm (BMCA), the Propagation Delay Measurement (PDM) mechanism, and the Transport of Time-synchronization Information (TTI). BMCA determines the grandmaster clock, which is the reference clock in the TSN network, and the hierarchy between the different TSN devices. The PDM mechanism is used once the hierarchy is established to measure the propagation delay between systems. Finally, the TTI mechanism is used to forward the grandmaster time, which, together with the measured propagation delay, synchronizes the other TSN devices updating their internal clocks.

This synchronization protocol can achieve a clock accuracy of tens of nanoseconds. However, it has stringent software and especially hardware requirements that, in most cases, legacy devices from Ethernet-based networks cannot support. Firstly, the absence of gPTP implementation in legacy devices poses a challenge, as modifying these devices implies high costs. Additionally, TSN requires network interfaces with hardware clocks capable of timestamping transmission and reception times, a feature lacking in most legacy devices. Even if legacy device software were modified to integrate gPTP, space constraints and hardware limitations would pose significant obstacles, potentially requiring the replacement of network interfaces and additional resources.

Furthermore, legacy systems employ diverse synchronization protocols. While the Network Time Protocol (NTP) [16], as a precursor of gPTP, shares similarities with it in functionality, other protocols like EtherCAT [14] or Flexible Time-Triggered (FTT) [18] utilize unique mechanisms unrelated to TSN. Modifying synchronization mechanisms in such cases would not only affect synchronization but also demand system-wide overhauls, potentially impacting application-level implementations. Hence, an independent synchronization mechanism for legacy end-stations is necessary for TSN adoption in legacy networks.

12.3.3 Centralized Network Configuration element

The CNC is a virtual component that can be placed in a designated node, an end-station, or a switch. Regardless of its placement, it can exchange information with network devices via NETCONF [10, 1]. This bidirectional communication allows end-stations to send user or network configuration requests to the CNC while switches can communicate their specifications. Finally, the CNC

Fig. 12.6: Heterogeneous TSN network with legacy end-stations topology.

can distribute new configurations to the entire network.

NETCONF utilizes a client-server approach for configuring the network, where the CNC acts as the client, responsible for collecting network information and initiating network device configurations. Note that all TSN network devices, e.g., TSN switches, must have a NETCONF server enabled to receive configurations from the CNC.

12.4 Problem statement

To observe the problems caused by the lack of synchronization between legacy systems and the TSN communication subsystems, we set up a small legacy network consisting of two single-board computers, i.e., Raspberry Pi (RPi) 3 Model B, running RPi Operating System (OS), connected point-to-point. Afterward, we add a Multiport TSN kit switch from System-on-Chip Engineering (SoC-e)[1] so that the RPIs behave as legacy end-stations in the new network, see Figure 12.6. The Raspberry Pi boards are configured to synchronize their software clocks with each other via NTP. Note that any clock synchronization protocol other than gPTP could be used between the legacy end-stations since they reproduce scenarios where the TSN switch cannot synchronize with the end-stations. We use NTP as a possible synchronization algorithm even if it is more common in industry the use of PTP, which typically provides a better synchronization accuracy.

In this experiment, we analyze the legacy network separately, i.e., without

[1]MTSN Kit: a Comprehensive Multiport TSN Setup. [Online]. Available: https://soc-e.com/mtsn-kit-acomprehensive-multiport-tsn-setup/

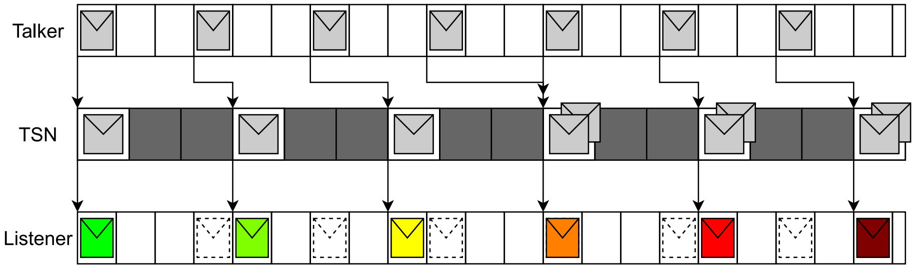


Fig. 12.7: Positive legacy system clock drift behavior.

the TSN network, to see its baseline behavior. Then, a TSN network is added to the legacy system to analyze the effects of putting both together. These experiments show that the only traffic affected by the lack of synchronization is the scheduled traffic; therefore, it is the one we will focus on in this paper. In this regard, thanks to the improved hardware and software capabilities of the TSN switches, the jitter of the legacy network TT traffic practically disappears. The reduced reception jitter would improve the system specifications and capabilities, enabling better service provision. Such enhancements would be challenging to achieve with the limitations of the communications subsystems previously used in the legacy network. However, due to the lack of synchronization, there is a drift between the clock time of TSN and the legacy system. This causes a deviation between the communication schedule of the legacy system and the TSN schedule that can be either positive or negative depending on which clock is faster or slower.

Regarding legacy synchronization, different protocols may require different configuration approaches. Traditional methods involve configuring synchronization traffic as AVB traffic to cap maximum latency or as TT traffic via the TAS for periodic configuration traffic. Alternatively, less conventional strategies like allocating a high-priority queue solely for synchronization traffic may be required. Nevertheless, these unconventional methods might compromise the maximum jitter experienced by TT traffic due to potential interference from synchronization traffic. Nonetheless, this jitter is expected to remain lower than that of the legacy network. However, these specific solutions fall beyond the scope of this paper. Below, we explain the findings of the experiment in detail.

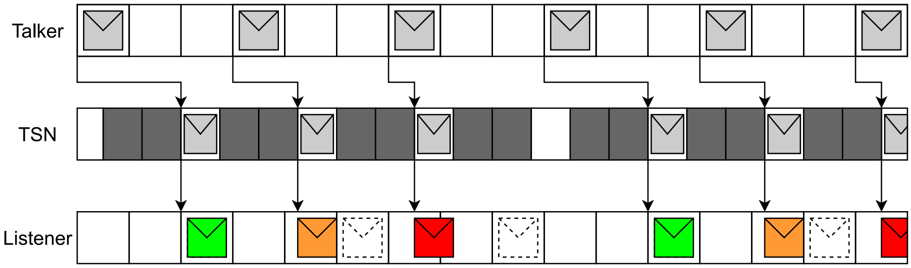


Fig. 12.8: Negative legacy system clock drift behavior.

Figure 12.7 shows the behavior of a heterogeneous TSN network in which the legacy system experiences a positive clock drift relative to the TSN communication subsystem. When the TSN clock is slower than the legacy system clock, the legacy system schedule exhibits a positive drift relative to the TSN schedule, causing frames to arrive at the receiver increasingly later than their legacy scheduled time. Moreover, since the transmission of frames by the TSN network to the legacy system receiver (*listener* in TSN terminology) is slower than the transmission by the legacy system transmitter (*talker* in TSN terminology) to the TSN network, the frames stack up in the buffers. However, the buffers are not infinite. Hence, frames that arrive once the buffer is full are discarded.

Figure 12.2 shows the number of frames lost (y-axis) per time unit (x-axis) during the experiment in which the legacy system experiences a positive clock drift (D) relative to the TSN communication subsystem. This experiment demonstrates that after a period of frame accumulation in the output queue, the queue starts to lose one frame out of every 100/(D [%]) frames.

Figure 12.8 shows the behavior of a heterogeneous TSN network in which the legacy system experiences a negative clock drift relative to the TSN communication subsystem. When the TSN clock is faster than the legacy system clock, the legacy system schedule exhibits a negative drift relative to the TSN schedule, causing frames to arrive at the receiver increasingly earlier than their legacy scheduled time. However, this effect cannot be infinitely extended over time since receiving a frame before it has been transmitted is impossible. When enough clock skew accumulates after a while, frames miss the transmission

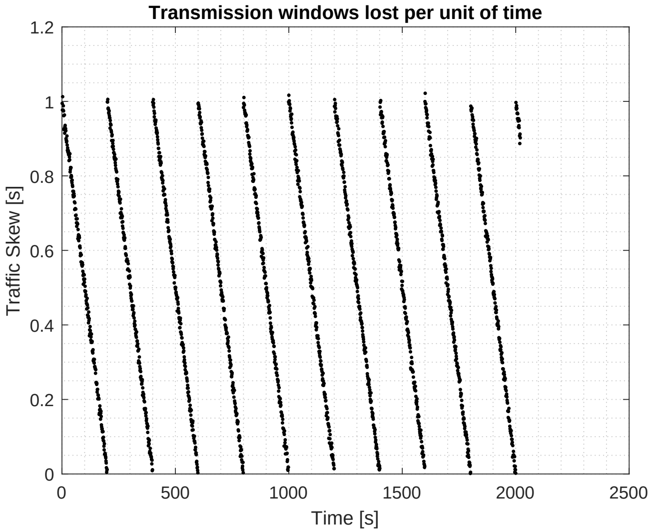


Fig. 12.9: Traffic skew due to negative clock drift.

window in which they are scheduled, leaving a period with no frames being transmitted.

Figure 12.9 shows the traffic skew observed during the experiment where the legacy system encounters a negative clock drift relative to the TSN communication subsystem. In this scenario, the loss of transmission windows becomes evident through the abrupt shifts in traffic skew observed in the graph. These findings reveal that frames undergo an entire period of clock drift before losing the transmission window and resetting the drift.

Through these experiments, which presented similar results to those in [8], we can observe that legacy systems can continue communicating through TSN and benefit from some of its features, such as improved reception jitter. However, due to the lack of synchronization, a clock drift appears, which not only causes a deviation in reception but can lead to empty transmission windows or even loss of frames. Therefore, this work aims to develop a mechanism that eliminates the drift between the TSN and legacy system schedules without requiring any modification in the legacy end-stations.

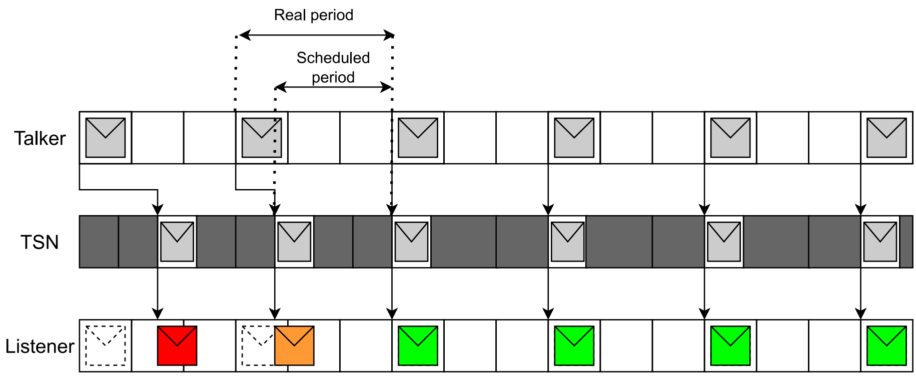


Fig. 12.10: TALESS operating diagram.

12.5 TALESS: TSN with Legacy End-Stations Synchronization

As we've discussed, drift is the primary source of errors in the absence of synchronization. However, when it comes to developing a solution, we must consider two crucial factors. First, the lack of synchronization among various legacy systems, each operating in its unique time domain, can lead to different drifts with respect to the TSN network. Secondly, these drifts are not constant over time, as environmental factors like temperature can impact the clocks in the network differently. Therefore, the proposed solution should eliminate the clock drift effects of different legacy systems that change over time.

One way to avoid the negative consequences of the drift caused by the lack of synchronization consists in eliminating the drift between the TSN network schedule and the legacy system rather than among clocks. As discussed in the previous section, the clock drift between the legacy system and the TSN communication subsystem leads to a disparity between the rates of frame reception and forwarding in the switches. When forwarding lags behind reception, a buffer overflow may occur, while faster forwarding than reception results in the loss of transmission windows, causing delays of nearly two periods between consecutive frames. Ensuring that the frame forwarding rate matches the reception rate through proper scheduling would solve these issues.

To achieve this, we propose to modify the size of the TSN GCL transmission windows when there is drift. This way, we can modify the TSN's transmission pace to match the legacy system one. Figure 12.10 shows an example of the operation of the proposed solution. This figure shows how, after detecting the drift, the TSN network changes the size of certain windows, specifically the lower priority BE queue, so that from that point onward, the frames arrive at the receiver according to the legacy system schedule. However, the TT traffic transmission windows should not be modified since the size of these windows is determined by the size of the frame and the link speed, where both parameters are independent of the clock drift. In this regard, TALESS would have no negative effect on any critical traffic unless a network reaches 100% utilization and the legacy system's clock becomes faster with respect to the TSN network. In this case, reducing any transmission windows would cause adverse effects on the network since there would not be sufficient resources in the TSN network. However, configuring to 100% utilization on the network is impractical, and industrial use cases commonly avoid that. Therefore, in TALESS, non-TT windows (NTTW) should be modified by a ratio equal to the drift between the legacy system and TSN (D) plus the cumulative variation in TT windows (TTW). Therefore, the new size of each NTTW ($NTTW_i.size'$) can be computed as:

$$\begin{aligned} NTTW_i.size' &= NTTW_i.size \ + \ D \times NTTW_i.size \ + \\ D &\times (NTTW_i.start \ - \ NTTW_{i-1}.end) \end{aligned} \tag{12.1}$$

where, to the previous NTTW size $NTTW_i.size$, we first add the variation of the window by multiplying the previous size $NTTW_i.size$ by the drift percentage D (either positive or negative) and secondly we add the cumulative variation of the TTW between the previous NTTW $NTTW_{i-1}$ and the current one. This last increment is because TTW cannot be modified, and the increment of these windows accumulates until the next NTTW. Note that the GCL is a list of transmission windows that specifies the size of each window. The start of each window is determined by the size of the windows preceding it. Consequently, while TTWs maintain a fixed size, they can be shifted forward or backward based on adjustments to the NTTW according to Eq. (12.1). Moreover, by exclusively adjusting the size of the NTTW windows, inherent rescheduling issues such as overlapping can be avoided. Figure 12.11 illustrates how modifying the NTTWs allows for the adjustment of the periods of the TTWs to align with the drift of legacy end-stations. The figure shows how TALESS adjusts the size of the NTTWs to align the periods of two TT frames initially set at 3 and 6 time units, respectively, to accommodate end-stations with approximately ± 8 % drift. Revisiting Figure 12.10, we can observe that the implementation of the Eq. (12.1) results in the expansion of the gray transmission windows, which correspond to the NTTWs, allowing them to match the transmission pace of the legacy system. Moreover, the NTTWs located after a TTW exhibit a longer extension due to their assimilation of the expansion corresponding to the TTW.

Consider a single TT frame with a size of 1 time unit and a period of 4 time units, forming the GCL of a TSN switch with a TTW of 1 time unit and an NTTW of 3 time units. Assuming a -10% drift of the legacy system relative to the TSN clock (i.e., D = -0.1), each TSN time unit corresponds to 0.9 time units of the legacy system.

Over 10 cycles of the GCL, there would be 10 transmission windows, but the legacy transmitter would have sent $(10 \times 4)/(4 \times 0.9) = 11$ frames due to the time conversion, resulting in an accumulation of one frame. With 20 cycles, it would accumulate 2 frames, with 30 cycles, 3 frames, and so on leading to a buffer overflow. However, applying the proposed solution, the resulting GCL would have a TTW of 1 time unit and an adjusted NTTW of $3 - 0.1 \times 3 - 0.1 \times 1 = 2.6$ time units. Consequently, regardless of the number of GCL cycles n, the number of transmission windows and legacy talker transmissions would remain equal $n \times (1 + 2.6)/4 \times 0.9 = n$.

Eq. (12.1) would be sufficient in a heterogeneous network where the drift between the legacy system and the TSN network is constant. In that case, it would be enough to calculate the drift and apply the formula to the TSN schedule only once offline. Drift can be measured by sampling the network traffic and comparing the real periodicity with the scheduled one. However, the previous solution will not be sufficient if the drift is variable or if several legacy systems with different time domains coexist in the same TSN network. Regarding variable drifts, constant monitoring and reconfiguration of the network is necessary. To do this, we propose a Drift Detector (DD) that continuously detects the drift

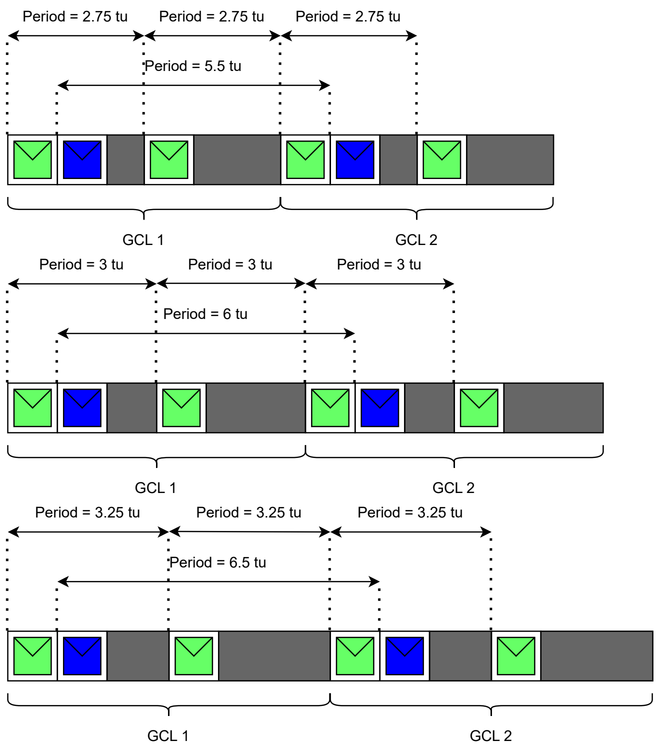


Fig. 12.11: Adjustment of the periods of 2 TTWs with 3 and 6 time units period (green frame and blue frame respectively) by modifying the size of NTTW (grey boxes) to accommodate end-stations' traffic with approximately ± 8 % drift.

between different clocks during run-time. Thus, we propose implementing a reconfiguration mechanism in the CNC, as shown in Figure 12.12. The DD, located on at least one reception port of a switch connected to a legacy system talker, samples the reception, i.e., the legacy system talker transmission. A single transmitter suffices because, under the assumption that the entire legacy system is synchronized using a legacy synchronization protocol, the drift between all legacy end-stations and the TSN communication subsystem is the same. The diagram in Figure 12.13 depicts a network that implements TALESS, in which end-stations T1 and T2, as well as L1 and L2, represent the Talkers and Listeners of legacy systems 1 and 2, respectively. The drift is determined based on the reception times of talker transmissions, which can be calculated using various methods. While this paper does not aim to provide the optimal or most efficient method, some options are outlined below.

The simplest approach involves utilizing a periodic frame with easily identifiable characteristics. By sampling the reception times, it becomes possible to compute the time interval between consecutive receptions of this periodic frame. Statistical inference techniques, such as T-Student analysis, can then be applied to ascertain, with a user-defined confidence level, whether the sampled frame adheres to the intended periodicity established in the TAS schedule within the TSN switch. The user-defined confidence level sets the threshold for drift detection by the DD. Specifically, in a heterogeneous TSN network incorporating several legacy systems, one transmitter is selected from each legacy system, and a TT frame is chosen from each of the selected transmitters. These frames can be, for example, periodic transmissions from any type of sensor such as temperature, pressure, revolutions, etc., or a combination thereof. Subsequently, a DD responsible for sampling the selected TT frames is deployed on the reception port of each TSN switch connected to the selected transmitters. Each DD knows the period of the corresponding frame since the TSN switch schedules the TT frames and therefore knows their periodicity. In this example, we assume a scheduled period of 1 second. Once the network is operational, any drift between the legacy systems and TSN may commence due to the lack of synchronization. Consequently, the sampled frames may arrive with an average period different than the 1 second expected. For instance, one of the sampled frames may arrive with a 0.9 second period. By continuously comparing this average period (0.9 sec) with the scheduled value (1 sec), the DD can identify the presence of drift. Notably, this comparison entails statistical inference, owing to reception jitter. Therefore, establishing a threshold for statistical inference becomes essential, whether it be a 90%, 95%, or 99% confidence level. This confidence level, in conjunction with the network jitter and drift, dictates the maximum achievable traffic skew before drift detection. The determination of this maximum traffic skew is driven either by user specifications or network requirements. For example, reducing the statistical confidence level may become necessary in systems with stringent delay and jitter requirements. This adjustment could lead to more false positives in drift detection, prompting additional reconfigurations. However, it ensures that traffic skew remains within acceptable limits set by jitter and delay constraints. Section 12.7 presents examples illustrating the maximum skew detected in the experiments conducted in this paper and provides a detailed explanation of the calculation process.

Every time the drift is detected, the average period of the last 'n' receptions of the sampled frame is calculated and divided by the expected frame period in the TSN switch to determine the drift percentage. This method was employed in the experiments discussed in Section 12.6.

Alternatively, other methods may involve calculating the reception rate per time unit. For instance, if the TSN switch expects to receive two TT frames from the talker, one with a period of 2 time units and the other with a period of 3 time units, it should ideally receive a total of 5 frames within every 6 time units interval. Through variations in the reception rate, it is possible to determine the drift. However, these methods are less accurate and require longer analysis periods to complete the determination.

Whenever a new drift is detected, a signal is sent to the CNC informing about the drift value. The CNC then updates the network configuration according to Eq. (12.1) and deploys it on the network to eliminate the drift between the legacy end-stations and the TSN schedule.

Finally, to allow the solution to work in networks combining different legacy systems, the only requirement is that TT traffic routes of different legacy systems cannot share output ports. This is because variations between the drifts of the legacy systems would invalidate TSN scheduling since the different drifts could cause some transmission windows to be advanced while others are delayed, causing them to collide. Moreover, given the small variability of the clocks, the resulting hyper-periods would be exponentially longer. For example, if two legacy systems transmit with 1 second period, but one has a 1% positive drift and the other one has 1% negative, instead of a GCL of 1

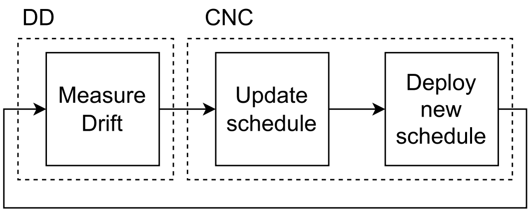


Fig. 12.12: TALESS task flow.

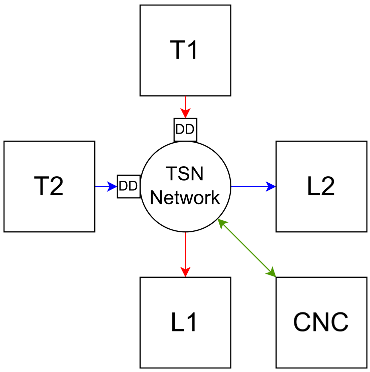


Fig. 12.13: TALESS architecture.

second with 3 transmission windows, the GCL would have an extension of lcm(1.01,0.99)=99.99 seconds with more than 200 transmission windows. Figure 12.14 illustrates an example of a heterogeneous TSN network following a tree topology combining two legacy systems with different drifts. The TSN communication sub-network comprises switches SW1, SW2, and SW3, while legacy systems 1 and 2 consist of end-stations ES1.1 and ES1.2, and ES2.1 and ES2.2, respectively. According to the requirement, TT traffic routes of each legacy system do not share output ports with the TT traffic routes of the other legacy system. To ensure meeting this condition, the TT traffic of each legacy system is grouped into separate branches of the tree topology, and the inter-system communication is restricted to AVB or BE traffic.

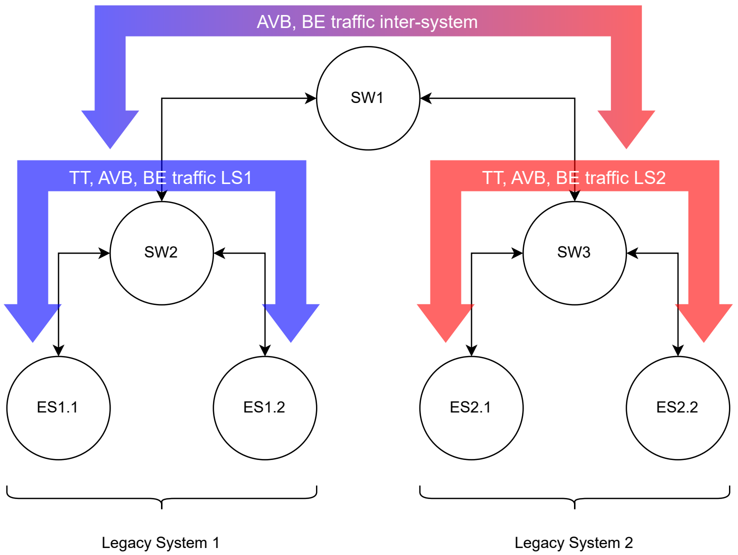


Fig. 12.14: Heterogeneous TSN network with tree topology using TALESS.

12.6 TALESS Validation Setup

In this paper, we validate the solution's effectiveness using two methods: a simulation model of the solution at the end-stations and an experimental implementation.

12.6.1 Simulation Model

Our model simulates the behavior of a TSN switch implementing TALESS. However, since TALESS solely eliminates drift, the results obtained in our experiments can be extrapolated to more extensive networks with any type of schedule, as long as the network architecture and schedule are functional in the absence of drift.

The model is implemented in Matlab and uses several parameters as inputs. These parameters include the period of the transmission to be modeled, the drift at the end of the experiment, and the jitter of the received transmission as the variance of a specified distribution. In addition, the modeled network run-time must be specified as an input. This is one of the main advantages of the model over the experimental implementation since, as real drifts are very small, the effects are noticeable only in the long term. In this sense, the model allows us to analyze long periods of time with realistic drift values in a reasonable model execution time.

The reception of frames is modeled as a list of timestamps (*ts*) generated by applying the drift variation (*dv*) and jitter (j) to the period (p), i.e.

$$ts_i = ts_{i-1} \; + \; p \times dv^i \; + \; \text{normrnd}(0, var) \tag{12.2}$$

where $\text{normrnd}(0, var)$ is a random value following a specific distribution, in this case, a normal distribution, with mean 0 and the variance *var* corresponding to the variance of the jitter used as an input. The DD module analyzes all *ts* values in the list individually. The DD module determines whether the period of the reception is equal to the initially scheduled one using a Student's t-test (`ttest`[2]). Once a significant difference is detected, i.e., the probability of the periods being equal is below a predetermined threshold, the period is updated based on the trend measured in the frames received since the last period update (`polyfit`[3]).

Finally, the model shows three different results for both positive and negative drift. The first result is the behavior of the reception with free transmission, i.e., without the intervention of the TSN switch, while the second result is the behavior with a fixed schedule without applying any solution. Finally, the last result is the effect of TALESS implementation. The results will be presented and discussed in Section 12.7.

12.6.2 Experimental Setup

We extended the network presented in Section 12.4 for the experimental implementation. More specifically, we use 4 Raspberry PIs and 4 TSN switches,

[2] One-sample and paired-sample t-test - MATLAB ttest [Online]. Available: `https://se.mathworks.com/help/stats/ttest.html`

[3] Polynomial curve fitting - MATLAB polyfit - MathWorks [Online]. Available: `https://se.mathworks.com/help/matlab/ref/polyfit.html`

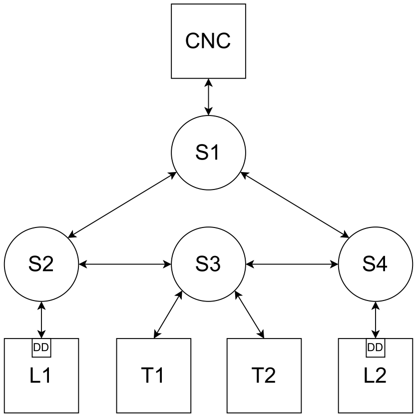


Fig. 12.15: Experimental network diagram showing TSN Switches (S) and legacy systems 1 and 2 represented by Talkers (T) and Listeners (L).

and a computer that will act as a CNC. The architecture of the new network is illustrated in Figure 12.15, where T1 and L1 represent the talker and listener of legacy system 1, and T2 and L2 the ones of legacy system 2. In addition, S1 to S4 and the CNC represent the TSN communications subsystem.

Each pair of Raspberry PIs (Ti, Li) forms an independent legacy system, i.e., they are not synchronized nor communicate with the end-stations of the other legacy system. For each talker, we implemented a synthetic clock with different drift values with respect to the TSN communications subsystem that changes throughout the experiment. This synthetic clock only changes the time perception of the legacy system by applying certain drift to the local clock synchronized through NTP. For example, if a 10% drift is applied, the synthetic clock will multiply all times by 1.1. These drift values were larger than those present in a normal network to magnify the effects in a reasonable duration of the experiments. In addition, the drift grew positively in one of the legacy systems, while in the other, it grew negatively. Each legacy system's synthetic clock is responsible for driving the transmission. To keep the talker and the listener synchronized with the drift changes, apart from the previously mentioned NTP, every time the synthetic clock drift changes, the talker sends a message to the listener with the new drift value so that the listener can update its synthetic clock.

According to the design sketched in Section 12.5, the DD should be implemented in the input port of switches to avoid modifications in the legacy end-stations. However, since we do not have access to the implementation of switches, we implemented the DDs in the legacy listener. Despite the change of the DDs location, neither the calculation method nor the obtained drift value changes. This is because the DDs can monitor the drifts on the ports, either connected to switches or the legacy end-stations. Once the DD measures a significant clock difference, it sends the drift value to the CNC. Note that the addition of the DD is the only modification made to the legacy end-stations with respect to their original implementation. This is required due to the limitations in modifying the commercial TSN switches. However, in a real TALESS implementation, no modifications to the legacy end-stations would be necessary.

The DD samples the frame reception time, either at the legacy end-station or at the TSN switch port connected to one, and compares it with the scheduled reception period. Using a t-test, it analyzes if there are variations in the periodicity. If so, the DD calculates the drift by dividing the period measured by the scheduled one and sends it to the CNC.

The CNC is based on the implementation proposed in [3], which was openly available to the research community. It uses a JSON file with the configuration to be deployed in the TSN network and NETCONF to deploy the configuration. The CNC is implemented to receive drift information from the DD, update the configuration based on Eq. (12.1), and automatically deploy the improved configuration in the TSN network. The results are presented and discussed in Section 12.7.

12.7 Simulation and experimental results

This section will show and analyze the results obtained using the simulation model and TALESS experimental implementation. In addition, we will compare the model with the experimental implementation to verify both the implementation and the simulation model.

We will use a metric called Synchronization Quality Metric (SQM) to analyze the obtained results. This is calculated by dividing the difference between the Reception Time (RT) of two consecutive frames minus the Scheduled Period (SP) for those frames by the SP, i.e.,

$$SQM_i = \frac{(RT_{i+1} - RT_i) - SP}{SP}. \tag{12.3}$$

The SQM allows us to analyze drift and jitter graphically. On the one hand, the mean SQM in a given interval provides information about the drift. Since the SQM gives the variation between the reception period and the scheduled period, if, for example, the scheduled period is 1 time unit and the average SQM is 0.1, then the system is receiving with a period of 1.1 time units. Therefore, there is a drift of 10%, i.e., the frames will arrive at times 1.1, 2.2, and 3.3 when they should arrive at times 1, 2, and 3. On the other hand, the maximum absolute value of SQM minus the mean SQM provides the ratio of jitter with respect to the period since by eliminating the drift from the variations in reception, we obtain the variation caused by the jitter. Note that this metric does not allow us to observe extreme cases such as frame loss, since the SQM cannot be quantified due to the missing RT.

Finally, all the analyses will be performed by comparing the reception of periodic frames in three different scenarios: (i) with free traffic flow through the TSN network, i.e., without applying TAS or any other scheduling mechanism, (ii) with the TSN communications subsystem scheduled without TALESS implementation, and (iii) with TALESS implementation.

12.7.1 Simulation Model Results

We simulate two different scenarios using the simulation model. In both cases, the model simulates a year of communications of a periodic transmission with an initial period of 1 second and with a variable drift that starts at 0% and grows progressively until reaching 10% at the end of the simulation in the first scenario and from 0 to -10% in the second one. These drift variations reflect the natural degradation of the end-station clocks, either positive or negative, caused by factors such as the passage of time or environmental influences like temperature, pressure, or electromagnetic interference. In addition, the jitter of the transmission is used as an input to the model and follows a normal distribution

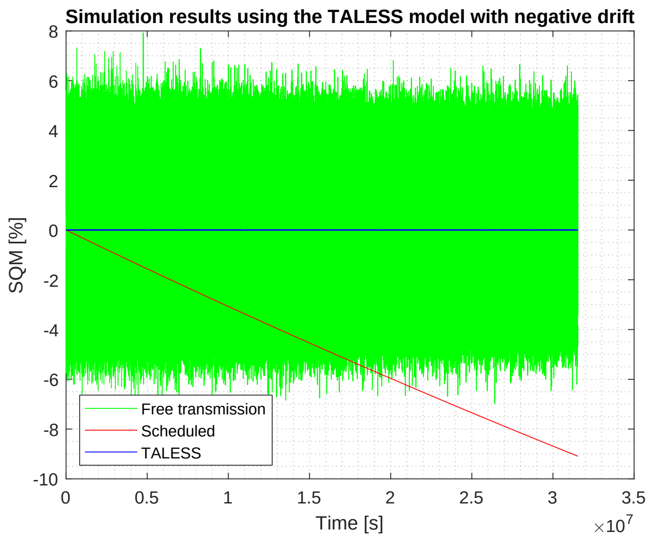


Fig. 12.16: Simulation results of one year of transmissions in a heterogeneous TSN network with negative clock drift in three different scenarios: free, scheduled, and TALESS transmission.

of variance 0.01. This distribution and variance are similar to the ones in Section 12.4. The results can be seen in Figures 12.16 and 12.17, both showing the SQM over the simulation time.

In both scenarios, the free reception has a jitter of 70 ms (as defined as input) and zero drift. When scheduling the TSN subsystem without TALESS, the jitter almost disappears, but the effects of the drift between the TSN schedule and the legacy transmission become evident. Finally, we observe that by applying TALESS, both the jitter and the drift almost drop to 0.

12.7.2 Real Network Implementation Results

Using the real network, we run an experiment similar to the model but with certain restrictions. Instead of a year of execution, only 2000 frames are transmitted in each legacy system, and the drift, instead of increasing and decreasing

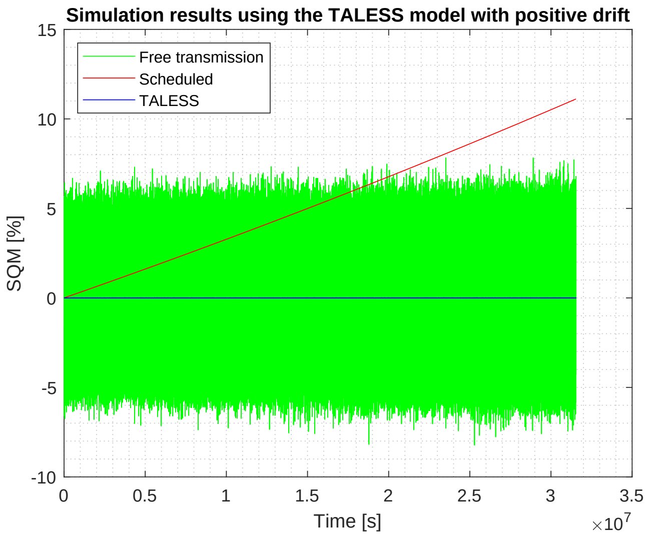


Fig. 12.17: Simulation results of one year of transmissions in a heterogeneous TSN network with positive clock drift in three different scenarios: free, scheduled, and TALESS transmission.

progressively up to $\pm 10\%$, varies by $\pm 5\%$ every 100 frames. Moreover, the legacy system with positive drift starts with a period of 1 second that is periodically shortened, while in the negative drift legacy system, it is the final period, which is equal to 1 second. All other characteristics are the same as in the simulation model. This experiment covers all the scenarios considered in this study, including the lack of synchronization between TSN and the legacy systems, the presence of drift due to the lack of synchronization, a time-varying drift due to environmental conditions (temperature, vibrations, power-supply, etc.), and the coexistence of two legacy systems with distinct drift characteristics. The results of these experiments can be seen in Figures 12.18 and 12.19.

As in the model, we can see how the free transmission has high jitter and no drift. Once the scheduling is applied without TALESS, the jitter disappears, but the drift occurs. In this case, the SQM (and therefore the drift) presents a stepwise behavior instead of a continuous one because, as previously mentioned,

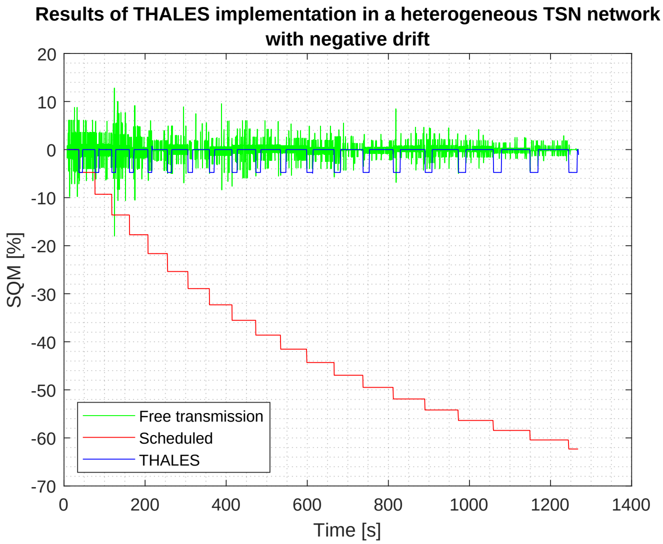


Fig. 12.18: Results of heterogeneous TSN network execution with negative drift in three different scenarios: free, scheduled, and TALESS transmission.

the drift variation is applied every 100 frames for simplicity in the experiment.

Finally, we see how TALESS eliminates jitter and drift yet leaves some drift remnants (the duty cycle observed in the figures). These are due to the time required by the solution to detect the change in the reception and are larger than what is observed in the simulation model due to the large synthetic drift applied to this experiment to allow us to visualize the effects of TALESS on the drift in a reasonable time. Although small periodic drifts can accumulate significant clock skew between the TSN network and the legacy system, there are ways to prevent this, e.g., by over-correcting the drift by creating equivalent drifts but of opposite sign to ensure an overall average drift equal to 0.

Also, as described in Section 12.6, in this experiment, both the positive and negative drift scenarios are performed simultaneously in the same network. This demonstrates that TALESS is capable of handling different drifts simultaneously. However, the difference in drift between legacy systems makes communication impossible through TT traffic. This can be achieved through other

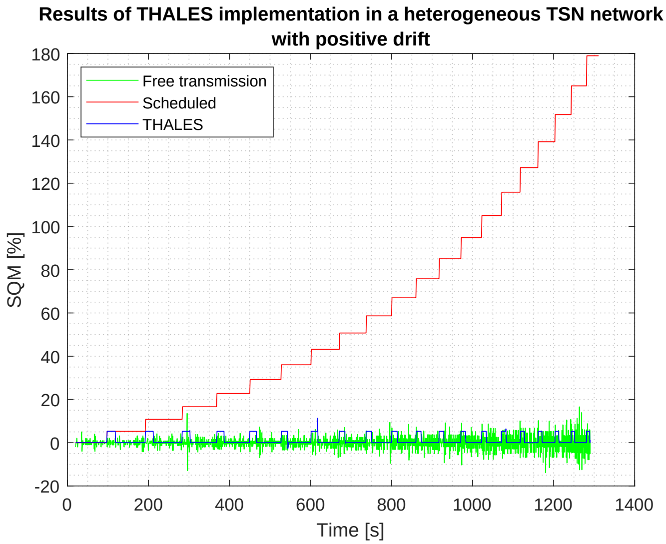


Fig. 12.19: Results of heterogeneous TSN network execution with positive drift in three different scenarios: free, scheduled, and TALESS transmission.

types of traffic not sensitive to clock drift, such as AVB or BE, improving the integrability of the different legacy systems integrated into TSN.

12.7.3 Comparison Results

Finally, to verify both the simulation model and the experimental implementation, we modified the model to simulate with the same conditions applied to the experimental implementation, i.e., execution of only 2000 frames with a variable drift of $\pm 5\%$ every 100 frames. Such simulations' results are shown in Figures 12.20 and 12.21.

As we can see, the simulations of implemented scenarios match the implementation results. Although the transmission by the legacy talker is not exactly the same since the real network does not strictly follow a normal distribution, the effects of both schedulings (with and without TALESS) on reception are essentially the same. This experiment provides evidence that the simulation

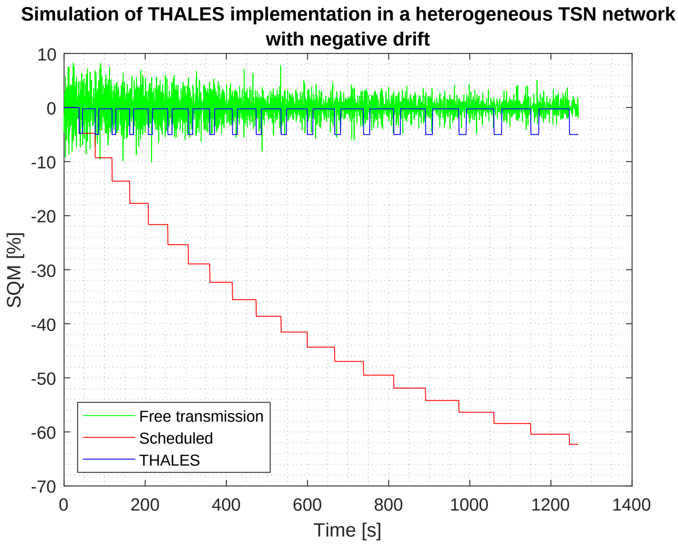


Fig. 12.20: Simulation results of the implemented heterogeneous TSN network with negative drift.

model follows the experimental results, ensuring the validity of the simulation model and, therefore, of TALESS.

In Table 12.1, we outline the absolute values of the jitter and drift obtained from both the simulation and the real network. Regarding drift, we calculate the difference between transmission and schedule time solely at the end of the experiments. At the start, the clock skew is presumed zero as insufficient time has elapsed for clock divergence. Note that the simulation spans one year, while the real network experiment lasts 2000 seconds of clock time for the end-station, under artificially amplified drift. In both simulated and real network experiments, the free transmission showcases the results previously discussed. Regarding the scheduled transmission without TALESS, both the real network and the simulation showed zero jitter. In the simulation, this occurs because we do not model TSN jitter since TSN time is directly equated with real-time, while real network results lack precision for direct jitter measurement, though TSN specifications suggest nanosecond-scale jitter. Notably, discrepancies be-

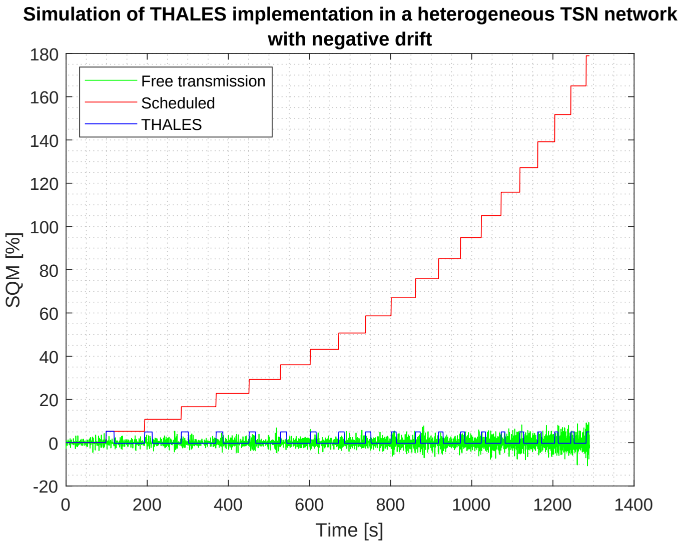


Fig. 12.21: Simulation results of the implemented heterogeneous TSN network with positive drift.

tween the end-station clock time and the TSN schedule due to clock drift are evident in both scenarios at the end of the experiments. Introducing TALESS effectively eliminates drift, yet simulation indicates minor jitter due to the time required for drift detection and application, while real network results show a 1 s jitter, attributed to the artificially amplified drift variation applied in the experiments.

12.7.4 Reconfiguration Time

Determining the network reconfiguration time is crucial for assessing the achievable jitter in the network. This time encompasses the duration needed to detect and address drift when it arises. By calculating this time, we can ascertain the traffic skew achieved before implementing the solution. This traffic skew, combined with the TSN jitter, determines the jitter experienced by the legacy system traffic when the legacy system experiences a drift change. This

Table 12.1: Results in absolute values.

Experiment		Jitter	Traffic Skew
Simulation	**Free Transmission**	70 ms	0 ms
	Scheduled	0 ms	18 h
	TALESS	2 ms	0 ms
Real Network	**Free Transmission**	70 ms	0 ms
	Scheduled	≈ 0 ms	1300 s
	TALESS	1 s	≈ 0 ms

parameter influences scheduling factors such as transmission window offsets. If the transmission window offset is bigger than the maximum drift plus the transmission jitter of the legacy end-station, we ensure that, in the absence of any other issues, TT frames will consistently transmit within their designated windows. Consequently, it becomes feasible to define latency by design by scheduling the TT traffic transmission windows via the TAS's GCL. Moreover, the reception jitter will be zero, while the latency jitter will be constrained to the transmission jitter of the legacy end-station plus the maximum traffic skew, which represents less than 2% of the jitter in the experiments as we will see below.

The reconfiguration time can be dissected into three components: drift detection time, rescheduling time, and new schedule deployment time. For drift detection, in our experimental network with approximately 70 ms jitter and around 1 µs drift per second, the DD requires sampling 2000 frames to detect the drift. Lower jitter and higher drift necessitate fewer samples. In our scenario, the 2000 frames needed imply a maximum traffic skew of 0.2% of the sampled frame period, corresponding to 2 ms for frames with a 1-second period.

As for rescheduling time, it can be considered negligible since TALESS modifies the existing schedule using Eq. 12.1 rather than creating a new one.

Regarding new schedule deployment time, the CNC enables background preparation of the new schedule, allowing the application at an opportune moment, such as the end of a GCL cycle. Thus, in the worst case, one clock cycle would be added to the network reconfiguration time. Given our legacy clock parameters, this translates to an additional traffic skew of 0.0001% of the GCL cycle.

In summary, the reconfiguration time equals the duration needed for drift identification plus one GCL cycle. For our experimental legacy network model, this time amounts to 2000 periods of the sampled frame plus one GCL cycle, totaling 2001 seconds. This corresponds to a maximum traffic skew of approximately 2 ms before implementing the solution.

12.8 Conclusions and Future work

This paper analyzed the effects of the lack of synchronization between the legacy systems and the TSN network. These effects are mainly due to the drift between TSN clocks and legacy systems, resulting in either delayed TT transmission or missing frames in the long term. Therefore, we designed, implemented, and validated a solution, TALESS, to remove the identified effects. Through simulation and implementation of TALESS, we demonstrated that TALESS efficiently enforces the reduction of jitter and removes the effects of clock drifts in legacy systems. This solution allows us to integrate several legacy systems into a TSN network without modifying their clock synchronization.

In future work, we aim to implement the proposed mechanism within a TSN switch to provide a complete tool for TSN adoption without any modification within the legacy systems.

Bibliography

[1] IEEE 802.1Qcc-2018 - IEEE Standard for Local and Metropolitan Area Networks - Bridges and Bridged Networks - Amendment 17: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements, 2018. DOI: 10.1109/IEEESTD.2018.8372875.

[2] IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.

[3] Inés Alvarez, Andreu Servera, Julián Proenza, Mohammad Ashjaei, and Saad Mubeen. Implementing a First CNC for Scheduling and Configuring TSN Networks. In *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, 2022.

[4] Marcin Bajer. Dataflow in modern industrial automation systems. Theory and practice. *Int. J. Appl. Control Electr. Electron. Eng*, 2(4), 2014.

[5] Haytham Baniabdelghany, Roman Obermaisser, et al. Extended synchronization protocol based on IEEE802. 1AS for improved precision in dynamic and asymmetric TSN hybrid networks. In *IEEE Mediterranean Conference on Embedded Computing (MECO)*, pages 1–8, 2020.

[6] Mohammadreza Barzegaran, Niklas Reusch, Luxi Zhao, Silviu S Craciunas, and Paul Pop. Real-time traffic guarantees in heterogeneous time-sensitive networks. In *Proceedings of the 30th International Conference on Real-Time Networks and Systems*, pages 46–57, 2022.

[7] Daniel Bujosa, Daniel Hallmans, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, and Thomas Nolte. Clock synchronization in integrated tsn-ethercat networks. In *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 214–221, 2020.

[8] Daniel Bujosa, Andreas Johansson, Mohammad Ashjaei, Alessandro V Papadopoulos, Julian Proenza, and Thomas Nolte. The Effects of Clock Synchronization in TSN Networks with Legacy End-Stations. In *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, 2022.

[9] Zichao Chai, Wei Liu, Mao Li, and Jing Lei. Cross Domain Clock Synchronization Based on Data Packet Relay in 5G-TSN Integrated Network. In *IEEE International Conference on Electronics and Communication Engineering (ICECE)*, pages 141–145, 2021.

[10] Rob Enns. RFC 4741: NETCONF Configuration Protocol, 2006.

[11] Joachim Feld. PROFINET-scalable factory communication for all applications. In *IEEE International Workshop on Factory Communication Systems, 2004. Proceedings.*, pages 33–38. IEEE, 2004.

[12] Michael Gundall, Christopher Huber, Peter Rost, Rüdiger Halfmann, and Hans D Schotten. Integration of 5G with TSN as prerequisite for a highly flexible future industrial automation: Time synchronization based on IEEE 802.1 AS. In *IECON Annual Conference of the IEEE Industrial Electronics Society*, pages 3823–3830. IEEE, 2020.

[13] Jetmir Haxhibeqiri, Xianjun Jiao, Muhammad Aslam, Ingrid Moerman, and Jeroen Hoebeke. Enabling TSN over IEEE 802.11: Low-overhead time synchronization for wi-fi clients. In *IEEE international conference on industrial technology (ICIT)*, volume 1, pages 1068–1073, 2021.

[14] Dirk Jansen and Holger Buttner. Real-time Ethernet: the EtherCAT solution. *Computing and Control Engineering*, 15(1):16–21, 2004.

[15] CA Latha and HL Shashidhara. Clock synchronization in distributed systems. In *2010 5th International Conference on Industrial and Information Systems*, pages 475–480. IEEE, 2010.

[16] David L Mills. Network time protocol (NTP). Technical report, 1985.

[17] Seifeddine Nsaibi, Ludwig Leurs, and Hans D Schotten. Formal and simulation-based timing analysis of Industrial-Ethernet sercos III over TSN. In *2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, pages 1–8. IEEE, 2017.

[18] Paulo Pedreiras, Luis Almeida, and Paolo Gai. The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency. In *Proceedings 14th Euromicro Conference on Real-Time Systems. Euromicro RTS 2002*, pages 152–152. IEEE Computer Society, 2002.

[19] Alexey M Romanov, Francesco Gringoli, and Axel Sikora. A precise synchronization method for future wireless TSN networks. *IEEE Transactions on Industrial Informatics*, 17(5):3682–3692, 2020.

[20] R. Salazar, T. Godfrey, L. Winkel, N. Finn, C. Powell, B. Rolfe, and M. Seewald. Utility Applications of Time Sensitive Networking White Paper (D3). Technical report, IEEE, 2018.

[21] S. Samii and H. Zinner. Level 5 by Layer 2: Time-Sensitive Networking for Autonomous Vehicles. *IEEE Communications Standards Magazine*, 2(2):62–68, 2018.

[22] Sebastian Schriegel and Juergen Jasperneite. A migration strategy for profinet toward ethernet TSN-based field-level communication: An approach to accelerate the adoption of converged IT/OT communication. *IEEE Industrial Electronics Magazine*, 15(4):43–53, 2021.

[23] Oscar Seijo, Xabier Iturbe, and Inaki Val. Tackling the Challenges of the Integration of Wired and Wireless TSN with a Technology Proof-of-Concept. *IEEE Transactions on Industrial Informatics*, 2021.

[24] SERCOS International e.V. SERCOS - the real-time ethernet communication standard for motion control. `https://www.sercos.org`, 2024. Accessed: 2024-04-15.

[25] Haochuan Shi, Adnan Aijaz, and Nan Jiang. Evaluating the Performance of Over-the-Air Time Synchronization for 5G and TSN Integration. In *IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pages 1–6, 2021.

[26] Jiajia Song, Makoto Kubomi, Jeffrey Zhao, and Daisuke Takita. Time synchronization performance analysis considering the frequency offset inside 5G-TSN network. In *IEEE International Symposium on Wireless Communication Systems (ISWCS)*, pages 1–6, 2021.

[27] Tobias Striffler and Hans D Schotten. The 5G Transparent Clock: Synchronization Errors in Integrated 5G-TSN Industrial Networks. In *IEEE International Conference on Industrial Informatics (INDIN)*, pages 1–6, 2021.

[28] Sebastian Szancer, Philipp Meyer, and Franz Korf. Migration from SERCOS III to TSN-Simulation Based Comparison of TDMA and CBS Transportation. In *OMNeT++*, pages 52–62, 2018.

[29] M. Wollschlaeger, T. Sauter, and J. Jasperneite. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Industrial Electronics Magazine*, 11(1):17–27, 2017.

[30] Junli Xue, Guochu Shou, Hongxing Li, and Yaqiong Liu. Enabling deterministic communications for end-to-end connectivity with software-defined time-sensitive networking. *IEEE Network*, 36(2):34–40, 2022.

[31] Zhe Zhao, Lu Gao, Weitao Pan, Zhiliang Qiu, Xu Guo, Ya Gao, and Ling Zheng. Access mechanism for period flows of non-deterministic end systems for time-sensitive networks. *Computer Networks*, 231:109805, 2023.