Efficient AVB-aware Scheduling for Critical Traffic in Time-sensitive Networks

Daniel Bujosa*, Silviu S. Craciunas†*, Saad Mubeen‡
*Technical University of Denmark, Kongens Lyngby, Denmark
†NXP Semiconductors, Vienna, Austria
‡Mälardalen University, Västerås, Sweden
Emails: dbuma@dtu.dk, silviu.craciunas@nxp.com, saad.mubeen@mdu.se

***Abstract*—Time-Sensitive Networking (TSN) enables deterministic mixed-criticality communication over standard Ethernet through time-aware scheduling (IEEE 802.1Qbv) for Scheduled Traffic (ST) and via Credit-Based Shaping (CBS) for Audio-Video Bridging (AVB) traffic. Existing scheduling methods often prioritize ST schedules and require iterative feedback loops between ST-schedule generation and AVB schedulability analysis to ensure the timing of both classes. We propose a novel approach called EAST (Efficient AVB-aware Scheduling for TSN) that improves runtime and schedulability by introducing a constraint-driven approach that decouples AVB analysis from ST schedule synthesis. We first perform a Worst-Case Response Time (WCRT) analysis to determine the maximum ST interference each AVB stream can tolerate without missing its deadline. This interference bound is then translated into temporal constraints on the ST scheduler, ensuring that both ST and AVB traffic remain schedulable without requiring multiple iterations of generating ST schedules and checking AVB schedulability. The resulting formulation guarantees end-to-end latency bounds for both ST and AVB streams while drastically reducing schedule synthesis time. Synthetic and real-world test cases show that EAST reduces runtime by several orders of magnitude compared to prior methods, while maintaining or improving overall schedulability.**

I. Introduction

Time-sensitive Networks (TSN) encompasses a variety of amendments and protocols based on IEEE 802.1. These enhancements introduce a series of features that enable real-time compliance, including time synchronization (IEEE 802.1AS-rev), preemption (IEEE 802.1Qbu), frame replication and stream redundancy (IEEE 802.1CB), and time-aware scheduling (IEEE 802.1Qbv). Safety-critical communication that requires guaranteed latency and bounded jitter, usually belonging to the periodic isochronous traffic class [1], is categorized as scheduled traffic (ST) and utilizes the timed-gate mechanism introduced in IEEE 802.1Qbv. This ST traffic is managed according to a schedule known as the Gate-Control List (GCL), which is computed offline through exact algorithms (e.g., SMT-based synthesis [13]) or heuristics (e.g., [28], [19], [9]). In contrast, periodic or sporadic communication that requires bounded end-to-end latency but does not require the stringent jitter control of ST traffic is classified as audio-video-bridging (AVB) stream traffic [10], [6]. Both ST and AVB streams are real-time communication protocols that transmit a predefined payload from a talker (sender) to one or more listeners (receivers). For AVB streams, usually, a schedulability analysis is employed to determine (pessimistic) worst-case upper bounds on the latency that a stream experiences, for instance, using a network calculus (NC) approach [38] or a worst-case response time analysis (WCRTA) [8]. Although the two traffic classes are handled using different mechanisms, the placement of ST traffic transmission windows directly affects the latency experienced by AVB traffic. Typically, ST traffic is prioritized and placed in higher-priority queues, or the timed gates for AVB traffic are closed when ST traffic is actively transmitting (guard band). Moreover, the frame preemption mechanism introduced in IEEE 802.1Qbu enables the configuration of specific traffic types as express (non-preemptable), while other traffic types can be configured as preemptable. Typically, ST traffic is defined as express, while AVB queues are preemptable. Preemption allows express traffic to interrupt AVB traffic within certain bounds (c.f. [3]), thereby reducing latency and jitter for ST traffic. However, this comes at the cost of increased latency for preemptable AVB traffic and introduces some overhead from frame fragmentation.

Most scheduling algorithms for the ST class tend to overlook the impact on AVB and best-effort (BE) streams (see, e.g., [13], [30], [26]). However, some approaches seek to formulate ST schedules that ensure AVB traffic's latency requirements are met (e.g., [28], [19], [4]) or aim to enhance quality of service (QoS) metrics for lower-priority, best-effort, or AVB traffic (e.g., [21]). Integrating the scheduling of the ST class with an analysis of the AVB class to minimize interference between the two is a complex challenge. The algorithms used to create ST schedules via ILP or SMT solvers cannot be readily adapted to incorporate network calculus (NC) or worst-case response time analysis (WCRT) formulations [16]. These algorithms typically rely on a feedback loop that directs the solver based on specific optimization criteria [16], [4]. On the other hand, while not optimal, heuristic approaches can be easily extended to include analysis methods for AVB based on NC [38], [4], [35] or WCRTA [22], [21], [19]. For heuristic solutions, most approaches prioritize scheduling ST traffic first, then evaluate AVB traffic based on that schedule. If the analysis shows that AVB traffic is not schedulable, the process is repeated until a suitable configuration is found. This approach is inefficient in terms of both scheduling and analysis time. Another strategy focuses on optimizing the distribution of ST traffic throughout the schedule, for example, by introducing porosity in the ST schedule [32]. However,

this can adversely affect ST traffic latency, as some frames must be scheduled as late as possible to maximize distribution. Furthermore, it results in more ST transmission windows than necessary, leading to wasted resources due to excessive guard bands and preemptions.

In this paper, we introduce **EAST** - **E**fficient **A**VB-aware **S**cheduling for **T**SN - an efficient scheduling method for TSN networks that feature both AVB and ST traffic classes. In our approach, we eliminate the need to repeatedly reschedule ST traffic when the current GCL configuration results in unfeasible AVB transmissions. First, we determine the maximum ST interference that each AVB frame can tolerate without missing its deadline, based on a Worst-Case Response Time (WCRT) analysis of AVB traffic. Next, we define a sliding window constraint for ST transmissions on each link such that, regardless of the AVB frame's release time, the cumulative ST interference within any window remains within the derived bound. Finally, this constraint is integrated into the ST schedule synthesis. Note that, the computation of the maximum ST interference is independent of the specific ST schedule synthesis algorithm and can be combined with any approach, including heuristic, ILP- or SMT-based methods.

By first computing the maximum admissible ST interference that preserves AVB schedulability, we decouple the two classes; i.e., AVB schedulability no longer depends on iterative ST schedule refinements, allowing us to compute schedulable ST configurations in a single pass. We are therefore much faster than traditional methods (e.g., [4], [19]) that must check AVB schedulability at each critical instant of an ST schedule candidate and repeatedly modify the ST schedule (often unguided) if AVB streams are not schedulable. In contrast, our method requires only a single computation to extract the maximum ST interference, achieving several orders of magnitude in speedup over classical methods. Furthermore, as shown in the experimental evaluation using both synthetic and real-world benchmarks, our approach achieves schedulability comparable to or better than that of state-of-the-art techniques. This makes the proposed method scalable to large TSN networks and suitable for industrial design loops where both correctness and synthesis time are critical. Due to the order-of-magnitude improvements in runtime, without sacrificing schedulability, EAST is hence also suitable for online reconfiguration of TSN networks (c.f. the industrial challenge at ECRTS 2025 [5]) in more dynamic industrial scenarios, where communication is changed at runtime (e.g. [29], [11], [20]).

The remainder of the paper is organized as follows. Sec. II surveys related work on integrated ST+AVB scheduling and analysis in TSN. Sec. III introduces the network and traffic models, followed by the AVB WCRT formulation in Sec. IV. Sec. V presents the proposed method, including the computation of maximum ST interference, the derivation of interference windows, and the adaptation of the ST scheduler. Sec. VI describes the experimental setup and the obtained results, followed by conclusions in Sec. VII.

II. Related Work

There has been extensive research on ST-only scheduling for TSN and other deterministic networks such as TTEthernet. This research employs either exact methods based on Integer Linear Programming (ILP) or Satisfiability Modulo Theory (SMT) [31], [13], [30], [15], or heuristic algorithms [28], [27], [26], [34], [2]. A comprehensive survey and a comparison of TSN scheduling methods can be found in [33] and [37], respectively. A significant limitation of ST-only scheduling is that they do not consider lower-priority sporadic or periodic traffic, such as AVB or ARINC 664-P7 rate-constrained (RC) traffic, which also have maximum latency requirements.

One of the first attempts to consider sporadic traffic was by Steiner [32] for TTEthernet networks, introducing the concept of porosity that effectively leaves gaps in the ST schedule to increase the likelihood (but not guarantee) that lower-priority sporadic traffic can be transmitted in a timely manner. Finzi et al. [16] introduced a feedback loop that imposes optimization criteria in the SMT-based scheduling formulation for TTEthernet, including the schedulability of lower-priority RC traffic. For TSN networks, some recent approaches aim at including AVB schedulability (e.g., [28], [19], [4]) in the ST scheduling step or enhancing quality of service (QoS) metrics for lower-priority traffic (e.g., [21]). Most feature a heuristic ST scheduler enhanced with a worst-case AVB latency bound analysis based on NC [38], [4] or WCRTA [22], [21], [19]. Other strategies focus on spacing out ST slots, similar to the porosity concept [32], to create space for AVB frames or use a hybrid offline-online approach [23] that uses meta-heuristic optimization and NC analysis to preconfigure delay budgets and priority queues to support dynamic flow arrivals. While this somewhat decouples the generation of ST schedules from the AVB analysis, the methods require a feedback loop if AVB traffic cannot be guaranteed, which may adversely affect the latency of ST traffic, as some frames must be scheduled as late as possible to maximize distribution. Additionally, it creates more ST transmission windows than necessary, resulting in resource waste due to unnecessary guard bands and preemptions. Another strategy is first to create ST schedules and then, in a feedback loop, evaluate AVB traffic latency based on the ST schedule candidate. If the analysis shows that AVB traffic is not schedulable, the process is repeated until a suitable configuration is found. This approach is inefficient in terms of both scheduling and analysis time, and there are no good metrics for the feedback loop that would guide the generation of new ST schedule candidates.

Two state-of-the-art tools for AVB-aware scheduling for TSN with preemption are CONAN-TSN and ARIEL. CONAN-TSN [7] integrates the WCRTA for AVB traffic from [8] with the ST HERMES scheduling algorithm [9] by generating ST schedules and then computing AVB worst-case response times under the resulting ST schedule to ensure end-to-end schedulability across both traffic types. Similar to EAST, CONAN-TSN only does a single iteration between ST scheduling and AVB analysis. Moreover, the WCRTA from [8]

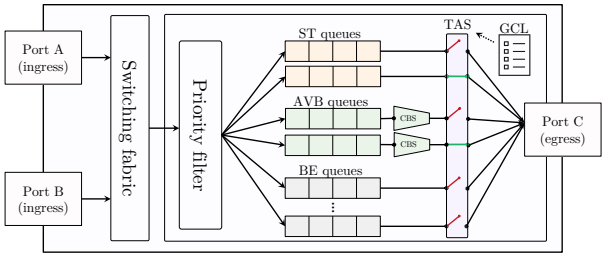


Fig. 1: Example TSN switch.

is the basis for computing the Maximum ST Interference (Max_STI) proposed in this work. Compared to [8], the novelty presented in this paper is not a new WCRTA formulation, but in how the existing WCRTA is used. We derive from the analysis in [8] an analytical bound on the maximum admissible ST interference per AVB stream, and transform this bound into scheduling constraints for ST synthesis.

ARIEL [4] integrates ST schedule synthesis with a new AVB analysis based on NC and uses a feedback loop between the AVB analysis and a heuristic routing-and-scheduling algorithm that accounts for HOLD/RELEASE behavior to reduce ST-induced interference and maximize AVB schedulability.

The idea of decomposing end-to-end latency requirements of streams in TSN networks has been used in [24], [39] and [36] under NC analysis to address reconfiguration issues for the Credit-Based Scheduling (CBS) and Deficit Round Robin (DRR) methods, respectively. Using this decomposition, global end-to-end latencies are respected as long as local deadlines are met during reconfiguration. In the context of task scheduling, B3LF [17] decouples the generation of time-triggered (TT) task schedules from the analysis of sporadic event-triggered (ET) tasks by using real-time calculus to derive a constraint on the generation of slots for TT tasks. In this paper, we use a similar strategy by decoupling the AVB analysis from the ST schedule generation, by performing a WCRT analysis to obtain end-to-end laxity for AVB traffic. This laxity then serves as a constraint for the generation of TT schedules. With this novel approach, we achieve multiple orders-of-magnitude improvements in synthesis runtime and comparable or better schedulability.

III. System Model

A. Network model

We model our network consisting of end-stations (ES) and switches (SW) that are interconnected through *link*s represented by l similar to [8]. All devices have full-duplex ports; hence, each physical device port has two links, one for reception and one for transmission. Similar to [8], the reception, forwarding, and queueing delays within a switch are accrued into a common processing delay denoted with ϵ. Furthermore, without loss of generality, we assume 0 link delay and heterogeneous link transmission speed BW. Each egress port can support 8 queues for ST, AVB, and BE traffic. One example assignment is depicted in Fig. 1, with the highest 2 priority queues being reserved for ST traffic (scheduled using TAS) and the next 2 priority queues being assigned to AVB traffic handled via the credit-based shaper (CBS) mechanism. The remaining queues are reserved for BE traffic. We refer the reader to [8] for an in-depth description of the TAS and CBS shapers and the interaction between shapers. For CBS, we denote the credit replenishment rate (idleSlope) and credit consumption rate (sendSlope) for an AVB queue with priority X on a link l with $\alpha^{+}_{X,l}$ and $\alpha^{-}_{X,l}$, respectively. Furthermore, we denote the CBS credit value for AVB traffic assigned to priority queue X on link l with $CR_{X,l}$.

B. Traffic model

Similar to [8], we utilize the periodic traffic model in TSN networks [22], which defines streams as sequences of frames sent from a source to a destination with the same period and subject to a common deadline. Hence, the set Γ of N streams is defined by: $\Gamma = \{m_i(C_i, T_i, D_i, P_i, \mathcal{L}_i, \mathcal{O}_i) | i = 1, \ldots, N\}$, where C_i denotes the transmission time of a frame of the stream m_i (calculated based on frame size and the network bandwidth BW). Similar to [8], the transmission time of an Ethernet frame header (denoted with v) is included in the frame transmission time C_i, as is the guard band for ST when necessary. Please note that the guard band is only necessary when there are gaps between the transmissions of ST frames. T_i and D_i are the period and relative deadline of frames, respectively, with $D_i \leq T_i$. The deadline defines the maximum latency that a TSN frame can experience from the time it enters the transmission queue at the source end-station until it is received by the destination end-station.

We allow AVB and BE traffic classes to have periodic or sporadic activation, and in case of sporadic arrivals, T_i represents the minimum inter-arrival time between two frames of the stream m_i. As in [8], $P_i \in \mathbb{P}$ denotes the priority of a stream m_i with the highest priority queue being assigned to ST, the lowest to BE, and all intermediate priorities being reserved for AVB traffic. The ST priority is denoted as P_{ST}, BE priority as P_{BE}, and AVB priorities as $\{P \in \mathbb{P} \mid P_{ST} > P > P_{BE}\}$. The WCRT calculation from [8] does not require that ST and BE traffic are assigned to only one queue, respectively; however, a single ST queue and a single BE queue simplify the notation. As in [8], $\mathbb{L}$ 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

$$\mathcal{L}_i = \{\mathcal{L}_i(0) = l_j, \ldots, \mathcal{L}_i(n-1) = l_k\}.$$

Each ST frame has a scheduled offset per link, i.e., $\forall l \in \mathcal{L}_i$ traversed by m_i, we have e.g., $\mathcal{O}_i = \{O_i^{l_j}, \ldots, O_i^{l_k}\}$. AVB frames do not have offsets, i.e., $\mathcal{O}_i = \emptyset$.

In this paper, we support frame preemption as defined in IEEE 802.1Qbu, and define ST queues as express queues capable of preempting both AVB and BE frames. This allows us to establish a guard band of 124 Bytes, which reflects the portion of any non-ST frame that cannot be preempted. Additionally, when a non-ST frame is preempted, it continues transmission with an additional header.

IV. WCRT ANALYSIS

In this section we reiterate the key aspects of the WCRT analysis presented in [8], which serves as the foundation for our work. For a comprehensive understanding, we refer the reader to the original work [8]. The WCRTA considers interference caused by higher-priority streams, taking into account factors such as ST interference with preemption, interference from higher-priority AVB streams, same-priority interference, and blocking by lower-priority frames, including those from lower-priority AVB and BE. In particular, the definitions of same-priority interference (SPI), higher-priority interference (HPI), lower-priority blocking (LPI), and scheduled-traffic interference (STI) are adopted directly from [8].

The WCRTA considers various sources of delay for an AVB frame of stream m_i on link l during busy periods and eligible intervals. First, due to FIFO queue operation, delays arise from interference with same-priority traffic, represented as SPI_i^l. Second, higher-priority AVB frames can interfere with lower-priority ones, indicated by HPI_i^l, although this is limited by the CBS. Non-preemptive AVB classes also cause delays; an AVB frame must wait for the entire transmission of lower-priority traffic, denoted as LPI_i^l. Lastly, TAS gates can block AVB and BE queue transmissions according to a non-uniform schedule, leading to interference noted as $STI_{i,c[k]}^l$ which needs to be calculated for every critical instant candidate $I_c^l[k]$ corresponding to the start of each transmission window on link l. The WCRT of an AVB frame of stream m_i traversing link l in the critical instant candidate $I_c^l[k]$ is:

$$WCRT_{i,c[k]}^l = STI_{i,c[k]}^l + HPI_i^l + SPI_i^l + LPI_i^l + C_i \quad (1)$$

The WCRT of a frame of stream m_i in link l is the maximum response time across all critical instant candidates: $WCRT_i^l = \max_{\forall m_c, \forall k}\{WCRT_{i,c[k]}^l\}$. Note that we do not need the iterative computation across critical instants from [8] to compute Max_STI, which is one of the reasons why EAST is faster and independent of the hyperperiod (c.f. Sec. VI-D.

Finally, since the analysis is compositional, the $WCRT_i^l$ for each link along the path of the stream m_i is accrued with the additional ϵ for each switch crossed. Hence, the overall WCRT for frame m_i is computed as follows in [8]:

$$WCRT_i = \sum_{\forall l \in \mathcal{L}_i} WCRT_i^l + (|\mathcal{L}_i| - 1) \times \epsilon \quad (2)$$

We now briefly present the same priority interference (SPI_i^l), higher-priority AVB interference (HPI_i^l), lower-priority blocking (LPI_i^l), and scheduled traffic interference ($STI_{i,c[k]}^l$), which make up the components of equation 1.

As detailed in [8], the delay for an AVB frame of stream m_i on link l, influenced by same-priority traffic $sp(m_i) = \{m_j | P_j = P_i, j \neq i\}$, is derived by the FIFO semantics of the queue. Due to the CBS behavior, we must consider both the transmission time of each interfering frame C_j and the credit recovery time needed for these frames. Same-priority frames can only be transmitted when the credit reaches 0; hence the credit consumed by each frame takes $C_j \times \frac{\alpha_{P_i,l}^-}{\alpha_{P_i,l}^+}$ to replenish to 0. Given the deadline-constrained model, only one frame of each same-priority stream in the FIFO queue can interfere with the frame under analysis [14], and hence the interference for m_i of class P_i on link l is thus computed as:

$$SPI_i^l = \sum_{\substack{\forall m_j \in sp(m_i), \\ i \neq j \\ \wedge l \in \mathcal{L}_j}} C_j \times \left(1 + \frac{\alpha_{P_i,l}^-}{\alpha_{P_i,l}^+}\right) \quad (3)$$

The same priority interference is also related to buffer occupancy and capacity. The SPI_i^l in Eq. (3) assumes that the egress queue of priority P_i can hold at most one frame from each same-priority stream $m_j \in sp(m_i)$ traversing link l in the worst case. Given the available buffer capacity for priority P_i on link l, denoted $B_{P_i,l}^{\max}$, the following must hold:

$$B_{P_i,l}^{\max}[bytes] \geq \sum_{\substack{m_j : P_j = P_i \\ l \in \mathcal{L}_j}} \text{size}(m_j).$$

where $\text{size}(m_j)$ denotes the frame length in bytes of a frame of stream m_j, corresponding to the transmission time C_j at link rate BW_l in bits/s, i.e., $\text{size}(m_j) = \lceil C_j \cdot BW_l / 8 \rceil$. In case of arbitrary deadlines ($D_i > T_i$), the following must hold:

$$B_{P_i,l}^{\max}[bytes] \geq \sum_{\substack{m_j : P_j = P_i \\ l \in \mathcal{L}_j}} \lceil D_j / T_j \rceil * \text{size}(m_j).$$

As shown in [12], the higher-priority AVB interference and lower-priority blocking correspond to the time required to reach maximum CBS credit CR^{max} for the AVB class of m_i:

$$HPI_i^l + LPI_i^l = \frac{CR_{P_i,l}^{max}}{\alpha_{P_i,l}^+} \quad (4)$$

In [12] it is also shown that $CR_{P_i,l}^{max}$, and consequently $HPI_i^l + LPI_i^l$, remains bounded if the bandwidth of all streams in $\mathbb{H} = \{H \in \mathbb{P} \,|\, ST > H > P_i\}$ is less than or equal to BW. Hence, the non-ST interference $HPI_i^l + LPI_i^l$ is:

$$HPI_i^l + LPI_i^l = C_{\mathbb{L},l}^{max} \times \left(1 + \frac{\alpha_{\mathbb{H},l}^+}{\alpha_{\mathbb{H},l}^-}\right) - \frac{CR_{\mathbb{H},l}^{min}}{\alpha_{\mathbb{H},l}^-}, \quad (5)$$

where $C_{\mathbb{L},l}^{max}$ is the largest frame size from the set $\mathbb{L} = \{L \in \mathbb{P} \,|\, L < P_i\}$ and $CR_{\mathbb{H},l}^{min}$ is the minimum value that the combined credit of the highest priority queues can reach on link l. In [12], this is computed using the recursive formula

$$CR_{\mathbb{H}=\{H_1,\ldots,H_n\},l}^{min} = -\max(\alpha_{\mathbb{H},l}^- \times C_{H_1,l}^{max} - CR_{\mathbb{H}-H_1,l}^{min}, \ldots, \alpha_{\mathbb{H},l}^- \times C_{H_n,l}^{max} - CR_{\mathbb{H}-H_n,l}^{min}) \quad (6)$$

As proven in [22], each ST transmission slot within the hyperperiod can be a critical instant candidate, i.e., :

$$I_j^l = \left\{ (k-1)T_j + O_j^l \;:\; k = 1, \ldots, n \;,\; n = \frac{\Omega_l}{T_j} \right\} \quad (7)$$

Next, the phase difference between each ST frame in $m_j \in ST$ and each potential critical instant $I_c^l[k]$ is calculated. For a complete description and proofs, we refer the reader to [25]

and [8]. $\Phi^l_{jc[k]} = (O^l_j - I^l_c[k]) \bmod T_j$. Then, for every $I^l_c[k]$, the ST interference over a time t is:

$$W^l_{c[k]}(t) = \sum_{\forall j \in ST \wedge l \in \mathcal{L}_j} \left(\left\lceil \frac{t - \Phi^l_{jc[k]}}{T_j} \right\rceil \right) C_j \quad (8)$$

Due to preemption, the WCRTA also needs to consider additional headers of size v for every ST frame segment, as they are needed to recover the preempted AVB frames. As described in [8], in the worst-case scenario, we have:

$$V^l_{c[k]}(t) = \sum_{\forall j \in ST \wedge l \in \mathcal{L}_j} \left(\left\lceil \frac{t - \Phi^l_{jc[k]}}{T_j} \right\rceil \right) v \quad (9)$$

The total interference from ST and the additional headers resulting from preemption is: $STI^l_{c[k]}(t) = W^l_{c[k]}(t) + V^l_{c[k]}(t)$

Finally, the response time of an AVB frame sent over link l during the critical instant candidate $I^l_c[k]$, denoted with $WCRT^{l,(x)}_{i,c[k]}$, is iteratively calculated as:

$$WCRT^{l,(x)}_{i,c[k]} = STI^l_{c[k]} \left(WCRT^{l,(x-1)}_{i,c[k]} \right) + HPI^l_i + SPI^l_i + LPI^l_i + C_i \quad (10)$$

The calculation starts with $WCRT^{l,(0)}_{i,c[k]} = HPI^l_i + SPI^l_i + LPI^l_i + C_i$ and ends when $WCRT^{l,(x)}_{i,c[k]} = WCRT^{l,(x-1)}_{i,c[k]}$ or $WCRT^{l,(x)}_{i,c[k]} > D_i$.

V. Proposed Solution

Our solution first determines the maximum ST interference that each AVB frame can tolerate by means of a modified version of the WCRTA presented in [8]. Specifically, by considering the impact of AVB and BE traffic and by constraining the WCRT to be less than or equal the deadline, we can compute the maximum amount of ST interference that each AVB frame can tolerate without violating its deadline. Then, a sliding window is defined for the ST traffic on each link to ensure that, regardless of when an AVB frame is released, the ST interference it experiences does not exceed this maximum allowable value. Finally, this sliding window is imposed as a constraint on the ST scheduler. Consequently, if a feasible ST schedule can be found under these constraints, we can guarantee that both ST and AVB traffic are schedulable. Conversely, if no feasible ST schedule exists, the ST traffic does not have sufficient resources to be scheduled without causing unacceptable interference to AVB traffic.

A. Maximum ST Interference

We assume that AVB traffic mapping (priority assignment), and AVB bandwidth reservation are given. Optimizing the AVB traffic configuration by determining the mapping and bandwidth allocation that maximizes the maximum ST interference AVB traffic can experience without missing the deadline may further improve overall schedulability. However, such a optimization is orthogonal to the contribution of this paper and is left for future work.

Similar, we assume that ST and AVB routing is predefined or computed before the analysis and synthesis step (similar to e.g., [13], [15], [30]) and remains static, which is preferred for predictability and certification compliance [18]. The fixed route can be obtained using existing routing techniques such as shortest-path, k-shortest paths, or redundancy-aware methods [40], [18], [19]. Routing affects the WCRT and the derived Maximum ST Interference in two ways. First, the number of traversed links $|\mathcal{L}_i|$ determines the accumulated per-link delay and switch latency ϵ in Eq. 2, directly influencing the admissible Max_STI_i. Second, routing determines which streams share each link, thereby affecting the per-link interference terms (HPI^l_i, SPI^l_i, LPI^l_i, STI^l_i) and the resulting $Non\_STI^l_i$. Consequently, different routing decisions may alter the feasible interference bounds. EAST is agnostic to the routing algorithm used, as Max_STI_i is derived from the WCRT formulation for a given path. However, this enables future joint routing, AVB configuration, and scheduling optimization aimed at finding the best configuration in terms of overall schedulability.

To compute the maximum ST interference for each AVB stream, we remove the ST contribution from Eq. (10) to obtain the Non-ST interference contribution ($Non\_STI^l_i$) and set the $WCRT_i$ in Eq. (2) equal to the deadline to obtain Max_STI_i, based on the assumption that a stream is schedulable only if its WCRT does not exceed its deadline. Applying this variable substitution yields the following result:

$$Non\_STI^l_i = HPI^l_i + SPI^l_i + LPI^l_i + C_i \quad (11)$$

$$Max\_STI_i = D_i - \sum_{\forall l \in \mathcal{L}_i} Non\_STI^l_i - (|\mathcal{L}_i| - 1) \times \epsilon \quad (12)$$

Note that $Non\_STI^l_i$ is computed using the parameters that characterize both AVB and BE traffic and that Max_STI_i accounts not only for the transmission time of the ST frames but also includes, for each ST transmission window, a guard band and the additional time for the transmission of an extra header in case of preemption. Given the compositionality of the analysis, which guarantees that the other terms (LPI, SPI, and HPI) are already maximized independently of the STI value, if each AVB stream experiences an STI lower than Max_STI_i, we can guarantee that the WCRT experienced by that AVB frame will be less than or equal to its deadline.

B. Sliding Window Configuration

We assign to each link l a period (T^l_{STI}) and an active time (A^l_{STI}) for the transmission of ST traffic. These parameters define the maximum duration of ST traffic (A^l_{STI}) that can be transmitted within each period T^l_{STI}.

Lemma V.1. *The AVB traffic is schedulable if the parameters A^l_{STI} and T^l_{STI} satisfy the following conditions:*

$$T^l_{STI} \geq \max_{\forall m_j : l \in \mathcal{L}_j} (Non\_STI^l_j) + A^l_{STI} \quad (13)$$

$$Max\_STI_i \geq \sum_{\forall l \in \mathcal{L}_i} A^l_{STI} \quad (14)$$

Proof. By satisfying Eq. (13), it is guaranteed that an AVB frame m_i will not remain on the output port of the link

for longer than T^l_{STI}. Otherwise, either the $Non\_STI^l_i$ experienced by that frame on the link l would have to exceed $\max_{\forall m_j : l \in \mathcal{L}_j}(Non\_STI^l_j)$, or the ST interference during the interval T^l_{STI} would have to exceed A^l_{STI}. Both cases are impossible, as they would violate the ST scheduler's constraints. Consequently, the maximum interference that any AVB frame can experience on link l is bounded by A^l_{STI}.

Furthermore, according to Eq. (14), if the sum of A^l_{STI} along the path of any AVB frame m_i is less than Max_STI_i, then m_i is guaranteed to be schedulable. This follows because the ST interference on each link is at most A^l_{STI}, and since the total accumulated interference along the path is less than Max_STI_i, the frame cannot experience a total ST interference exceeding its schedulability bound.

In conclusion, an arbitrary AVB frame m_i cannot remain on each link of its path for more than T^l_{STI}. Moreover, over any interval of duration T^l_{STI}, the amount of STI is less than or equal to A^l_{STI}. Therefore, in the worst case, the stream m_i will experience a WCRT that is less than its deadline, i.e. since $\sum_{\forall l \in \mathcal{L}_i} A^l_{STI} \leq Max\_STI_i$:

$$WCRT_i \leq (|\mathcal{L}_i| - 1) \times \epsilon + \sum_{\forall l \in \mathcal{L}_i} (Non\_STI^l_i + A^l_{STI})$$

$$WCRT_i \leq (|\mathcal{L}_i| - 1) \times \epsilon + Max\_STI_i + \sum_{\forall l \in \mathcal{L}_i} Non\_STI^l_i$$

Therefore $WCRT_i \leq D_i$

□

Although there are various ways to define A^l_{STI} and T^l_{STI}, one of the simplest is to make them proportional to the utilization of ST traffic, since A^l_{STI}/T^l_{STI} represents the fraction of bandwidth available for its scheduling. However, A^l_{STI} should be at least as large as the maximum size of the ST frames transmitted over that link to ensure that, within each period T^l_{STI}, there is sufficient time to allocate at least one frame from any ST stream. Therefore, we define A^l_{STI} as the sum of: (i) a fixed component corresponding to the maximum transmission time among the ST frames sent through link l plus the interference caused by its preemption v, and (ii) a component proportional to the utilization of the ST traffic on that link ($\widehat{A}^l_{STI}$). For (i) we have:

$$A^l_{STI} = \widehat{A}^l_{STI} + \max_{\forall m_j \in ST}(C_j) + v \tag{15}$$

For (ii), we introduce a non-negative scaling factor γ_l which represents the admissible fraction of the nominal ST utilization on link l. Intuitively, γ_l scales the total ST utilization on the link such that the resulting sliding-window parameters A^l_{STI} and T^l_{STI} satisfy the Max_STI constraint of all AVB streams traversing that link. Note that γ_l is link-specific and does not represent a global scaling factor across the network.

$$\gamma_l \times \sum_{\substack{\forall j \in ST, \\ l \in \mathcal{L}_j}} \frac{C_j}{T_j} = \frac{\widehat{A}^l_{STI}}{T^l_{STI}} \tag{16}$$

Since we aim to maximize A^l_{STI} and minimize T^l_{STI} to allocate as much bandwidth as possible and thereby enhance the schedulability of ST traffic, we can derive $T^l_{STI} = \max(Non\_STI^l_i) + A^l_{STI}$ from equation Eq. (13). Applying this variable substitution and simplifying Eq. (16), we get:

$$A^l_{STI} = \frac{\gamma_l \times \sum_{\substack{\forall j \in ST, \\ l \in \mathcal{L}_j}} \frac{C_j}{T_j} \times \max(Non\_STI^l_i) + \max_{\forall m_j \in ST}(C_j) + v}{1 - \left(\gamma_l \times \sum_{\substack{\forall j \in ST, \\ l \in \mathcal{L}_j}} \frac{C_j}{T_j}\right)} \tag{17}$$

By substituting A^l_{STI} in equation Eq. (14) with the expression obtained from equation Eq. (17), we derive a new inequality with a single variable γ_i for the considered AVB stream. We first compute a per-stream candidate γ_i (applied uniformly across links in Eq. (18)), then assign per-link γ_l by taking the minimum across streams that traverse link l. This allows us to compute the largest feasible value of γ_i, and hence, the ratio $\widehat{A}_S TI^l / T_S TI^l$, in a weighted manner with respect to the utilization of ST traffic on each link in the path:

$$Max\_STI_i \geq \sum_{\forall l \in \mathcal{L}_i} \frac{\gamma_i \times \sum_{\substack{\forall j \in ST, \\ l \in \mathcal{L}_j}} \frac{C_j}{T_j} \times \max(Non\_STI^l_i) + \max_{\forall m_j \in ST}(C_j) + v}{1 - \left(\gamma_i \times \sum_{\substack{\forall j \in ST, \\ l \in \mathcal{L}_j}} \frac{C_j}{T_j}\right)} \tag{18}$$

Note that, since A^l_{STI} cannot be negative, the solution to Eq. (18) is the largest value of γ_i that satisfies the inequality while ensuring $A^l_{STI} \geq 0$. Moreover, $\gamma_i > 0$. Once γ_i has been determined for each AVB frame m_i, the minimum γ_i value is assigned to the route of its corresponding AVB stream. For each link l, we assign $\gamma_l = \min_{i : l \in \mathcal{L}_i} \gamma_i$, ensuring that the ST allocation on link l respects the interference bounds of all AVB streams traversing that link. The process is then repeated until all links have an associated γ_l, using a fixed value of A^l_{STI} for links where γ_l has already been defined. The system is considered unschedulable if the analysis yields negative values of A^l_{STI}. Algorithm 1 shows the described process. For example, consider a very simple network consisting of three end-stations (ES1, ES2, and ES3) and a single switch (SW1) arranged in a star topology, and three AVB streams (m_1, m_2, and m_3) with sources ES1, ES1, and ES2, and destinations ES2, ES3, and ES3, respectively.

In the first iteration, the value of γ_i is computed over the entire path of each AVB stream m_i. Next, the minimum γ_i is identified; assume it corresponds to stream m_1. In this case, the value of γ_1 is fixed for the links ES1–SW1 and SW1–ES2.

Then, γ_2 and γ_3 are recomputed for streams m_2 and m_3 while taking into account the previously fixed value of γ_1. For stream m_2, which shares the ES1–SW1 link with m_1, the value for this link is already fixed and is lower than the value previously computed. As a result, a larger amount

Algorithm 1: Computation of γ_l for Each Link l

Input: Set of AVB frames $\{\forall m_i \in AVB\}$, set of links $\mathcal{L}$
Output: Value of γ_l for each link l

while $\exists\, l:\ \gamma_l$ *not defined* **do**
  foreach *AVB frame* m_i **do**
    compute γ_i according to Eq. (18);
  find $\min(\gamma_i)$ corresponding to AVB stream m_j;
  foreach *link* $l \in \mathcal{L}_j$ **do**
    $\gamma_l \leftarrow \min(\gamma_i)$;
    compute A^l_{STI} based on Eq. (17);
  if $\exists A^l_{STI} < 0$ **then**
    mark system as *unschedulable*;

of Max_STI_2 can be allocated when computing γ_2 for the SW1–ES3 link. A similar situation occurs for stream m_3, which shares the SW1–ES2 link with m_1.

Once again, the new minimum γ_j is selected and assigned to the corresponding links. This process is repeated iteratively until all links have an assigned γ_l.

C. ST Scheduler Configuration

The calculation of the maximum ST interference and the definition of the sliding window are independent of the specific ST synthesis method used. Consequently, any algorithm that ensures the resulting GCL schedules no more than A^l_{STI} ST transmissions within any interval of duration T^l_{STI} can be employed. This formulation is also compatible with ILP- and SMT-based scheduling approaches, such as [13], as follows. We can introduce a boolean variable b_t for each macrotick $t \in \{0, \ldots, HP-1\}$ of duration Δ^l, where $b_t = 1$ indicates that the egress link is transmitting a TT frame at time t. Then, we convert both the window length T^l_{STI} and the maximum allowed TT occupancy A^l_{STI} to a macrotick-based, discretized time model as follows:

$$A^{l,\text{ticks}}_{\text{STI}} = \left\lfloor \frac{A^l_{\text{STI}}}{\Delta^l} \right\rfloor, \quad T^{l,\text{ticks}}_{\text{STI}} = \left\lceil \frac{T^l_{\text{STI}}}{\Delta^l} \right\rceil.$$

The sliding-window constraint then is:

$$\forall\, i \in \{0, .., HP-1\}: \sum_{k=0}^{T^{l,\text{ticks}}_{\text{STI}}-1} b_{\left((i+k) \bmod HP\right)} \leq A^{l,\text{ticks}}_{\text{STI}}.$$

An alternative realization of EAST is to integrate the sliding-window constraint into exact ST synthesis methods, such as SMT-based approaches [13], [30] or ILP-based formulations [15]. In such designs, the existing correctness constraints for collision freedom, precedence, frame/stream isolation, and end-to-end latency would remain unchanged, and the sliding-window constraint from above can be added as an additional linear (ILP) or logical (SMT) constraint over the schedule slot variables. Since the sliding-window bound is compositional with respect to the ST schedule, it does not require modifying the core structure of the solver, but only augmenting the constraint set. While compatible and easy to integrate with existing exact ILP- and SMT-based schedule synthesis methods, this would require a potentially large number of binary variables (one for each macrotick slot until the hyperperiod) and assertions (or quantifiers) in the SMT context. Both of these will most likely make the scheduler prohibitively slow. SMT/ILP-based realizations may nevertheless be preferable in scenarios where global optimality objectives are required (e.g., minimizing latency) or the network size is small, while EAST's heuristic implementation (see below) prioritizes scalability and runtime efficiency.

In this work, we adopt a credit-based approach within a heuristic method (Algorithm 2), assigning each link a credit value A^l_{STI}. This credit is decreased by an amount $C_j + v$ whenever an ST frame $m_j \in ST$ is scheduled and replenished by the same amount after a period of T^l_{STI}. Moreover, scheduling ST frames must guarantee that the credit value never becomes negative. We implement the Max_STI credit constraint in HERMES [9]. Unlike other ST schedulers, HERMES operates on a link-by-link basis rather than scheduling frame by frame from the destination link to the source link. This approach allows parallelized scheduling and enables each scheduled frame to consider the reception requirements of other frames. Algorithm 2 presents a simplified version of the scheduler that integrates the credit mechanism proposed in this work. The updated version of HERMES introduces credit constraints and starts by dividing links into scheduling phases. All links within the same phase can be scheduled simultaneously. A link is assigned to a specific phase only after all ST streams that traverse it have had their following links in the path allocated to earlier phases. Thus, in the first phase, all destination links are scheduled; in the second phase, the links leading to those destinations are scheduled; and the process continues, ensuring that dependencies between frame paths are maintained. Once all links have been assigned to a phase, frame scheduling proceeds for each link within its phase. Frames are scheduled as close to their deadlines as possible to maximize the schedulability of links in later phases. The additional latency this may introduce is subsequently mitigated through an optimization process that shifts offsets closer to the beginning of the period.

When scheduling each frame, we verify 3 conditions, the first two similar to HERMES [9]: (1) no link collisions occur (checkCollision), (2) the order of arrival at the next transmission queue is consistent with the scheduled transmissions on the subsequent link (checkOrder), and (3) the credit constraint is satisfied (checkCredit). If any condition fails, the scheduler either increases the ST stream's priority or reduces its offset.

Once a frame is successfully scheduled, the corresponding credit is updated, and the process continues until all ST streams across all links are schedulable. If, however, the offset of any stream must be reduced below 0, the set of ST streams is deemed unschedulable under the defined credit constraints.

Regarding the credit mechanism, it is mapped over the entire hyper-period (HP), i.e., the LCM of the periods of all ST streams transmitted through the link. The credit value starts at $\widehat{A}^l_{STI}$ for the entire HP of link l. Each time a frame of ST stream m_j is scheduled, the credit within the interval [$\mathcal{O}^l_j$, $\mathcal{O}^l_j + T^l_{STI}$] is reduced by $C_j + v$, where v accounts for the

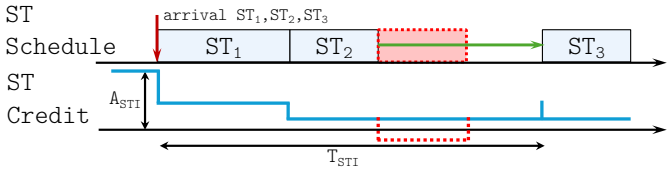


Fig. 2: Example of ST credit.

additional delay that the preemption of AVB traffic may incur due to the corresponding ST transmission.

To determine whether a frame from an ST stream is schedulable at offset $\mathcal{O}_j^l$, we check that the available credit within $[\mathcal{O}_j^l,\ \mathcal{O}_j^l + T_{STI}^l]$ remains greater than $C_j + v$. Therefore, in addition to checking link collisions and the order in which streams are received in the FIFO queues, as specified in paper [9], it is now also necessary to verify the available credit. Fig. 2 illustrates an example of scheduling three ST frames based on the credit mechanism. As shown in the figure, both ST frame ST_1 and ST_2 have sufficient credit to be scheduled consecutively immediately after their arrival times. However, since there is not enough remaining credit for ST frame ST_3, its transmission must be deferred until part of the previously consumed credit is recovered. This mechanism guarantees that no time interval of duration T_{STI} results in an ST interference greater than A_STI.

Algorithm 2: Credit-Based Scheduling of ST Frames

Input: Set of ST frames $\{m_j \in ST\}$, link set $\mathcal{L}$, frame costs C_j, constant v, and parameters A_{STI}^l, T_{STI}^l
Output: Valid GCL for each link l that satisfies the Max_STI

```
DivPhases() ∀l ∈ L assign a phase l.φ = p;
Φ ← {unique(l.φ)};
foreach p ∈ Φ do
    foreach l ∈ L : l.φ = p do
        foreach m_j ∈ ST : ∃l ∈ L_j do
            maximize(O_j^l);
            while release ≠ O_j^l or queue ≠ P_j do
                release ← O_j^l;
                queue ← P_j;
                if checkCollision ∨ checkOrder ∨ checkCredit
                  then
                    if P_j == max(P) then
                        reduce O_j^l;
                    else
                        increase P_j;
            updateCredit();
```

D. Optimized Credit Assignment

EAST, similar to CONAN-TSN [7], applies a bandwidth allocation (RBW) to AVB traffic proportional to the traffic load, as follows:

$$RBW_l^X = \left(1 - U_l^{BE}\right) \times \left(\frac{U_l^X}{U_l^{AVB}}\right) \tag{19}$$

where:

- U_l^{BE} is the utilization of BE traffic on link l,
- U_l^X is the utilization of AVB class X on link l, and
- U_l^{AVB} is the total utilization of all AVB traffic classes on link l.

However, as discussed above, the Max_STI analysis of EAST opens the possibility for new ways to optimize AVB traffic configuration. Through the Max_STI computation, EAST establishes a framework for the analytical optimization of AVB parameters, including credit assignment, priority allocation, and routing. This opens the door to more efficient strategies than those based on heuristics or analyses requiring prior knowledge of the ST schedule. The ultimate objective of such optimization would be to maximize the Max_STI value, thereby enhancing the schedulability of ST traffic and, consequently, improving the overall schedulability of the network. That is because, as shown in Sec. VI-B, the Max_STI constraint ensures that the schedulability of ST traffic inherently guarantees the schedulability of AVB traffic as well. This optimization lies beyond the scope of this paper; hence, we use the proportional credit assignment and leave the exploration of this optimization for future research.

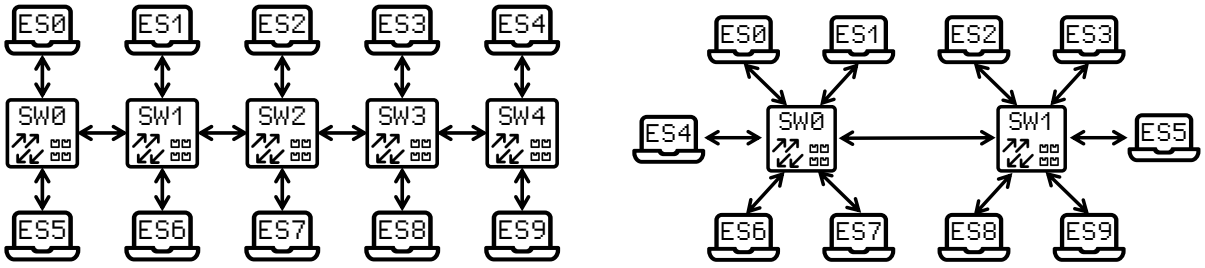
VI. Evaluation

A. Experimental Setup

To evaluate the proposed solution, EAST was compared against two previous configuration and analysis approaches: CONAN-TSN [7][1], a configuration and analysis tool that incorporates one of the latest WCRTA algorithms, which forms the basis for the maximum ST interference calculation proposed in this work, and ARIEL [4][2], an AVB-aware heuristic scheduling algorithm which, like EAST, also supports AVB integration and preemption. ARIEL operates through an iterative loop between an ST scheduler that creates transmission gaps for AVB traffic scheduling and an AVB analysis module based on Network Calculus.

We examine two network topologies, illustrated in Figures 3a and 3b, which follow a line-star topology. Network N1 consists of a network with 5 switches, each connected to 2 end-stations, while Network N2 features a more compact network with 2 switches, each connected to 5 end-stations. The selected topologies provide several advantages that justify their use in the evaluation. First, line-star topologies do not require the use of routing algorithms, which allows us to avoid introducing a routing strategy that could unintentionally favor one solution over another. This design choice ensures that the comparison focuses solely on the characteristics of the evaluated analysis approaches rather than on routing-related optimizations. Second, existing Network Calculus–based analyses, such as ARIEL, are not able to handle circular dependencies. Since our evaluation relies on randomly generated traffic, avoiding loops in the topology prevents the introduction of circular dependencies and enables a fairer and more robust comparison with the other considered approaches. In addition, line-star topologies are widely used in industrial networking scenarios,

[1]CONAN-TSN open source https://github.com/DanielBujosa/CONAN-TSN
[2]We thank the authors of [4] for providing the source code to ARIEL.



(a) Experimental network topology N1.

(b) Experimental network topology N2.

Fig. 3: Experimental network topologies used in the evaluation.

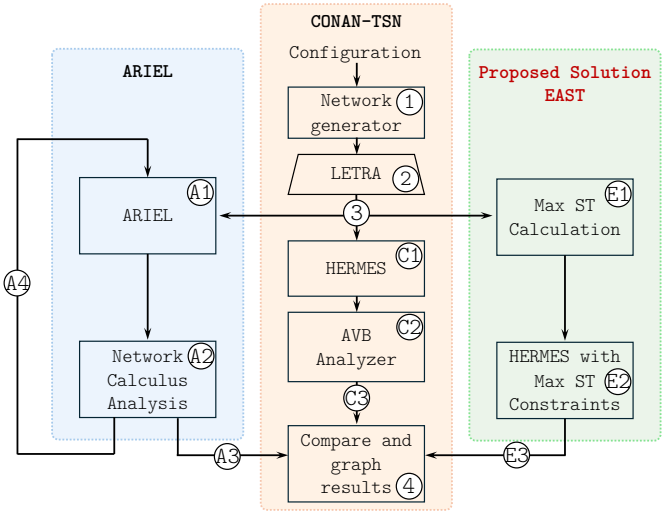


Fig. 4: Experimental setup.

making the evaluation representative of realistic deployments and increasing the practical relevance of the results. Finally, by considering two topologies with the same number of end-stations but different structural characteristics—one more compact and one more distributed—we are able to study the impact of different network parameters. In the compact topology, performance is more strongly influenced by bottlenecks on the link connecting the two switches, whereas in the more distributed topology, streams experience longer paths and higher accumulated delays. These contrasting characteristics allow us to identify whether any of the methods performs better than the others for bottlenecks or longer paths.

The network bandwidth was configured to 100 Mbps. Frame lengths were randomly selected within the standard Ethernet range, from 500 to 1500 bytes. The minimum and maximum allowed periods were set to 10,000 µs and 30,000 µs, respectively. Deadlines were set equal to the corresponding periods for streams with deadline constraints, while jitter was set to zero for streams with jitter constraints.

For both networks, N1 and N2, the traffic was distributed as follows: 12.5% ST traffic, 25% BE traffic, and the remaining traffic divided between AVB Class A and Class B. Moreover, to evaluate the impact of ST traffic on schedulability, N1 was also evaluated with traffic comprising 25% ST, 25% BE, and 50% AVB, evenly distributed across AVB class A and B.

In terms of credit allocation, ARIEL does not optimize the credit assignment based on the utilization of each AVB class along individual links. Instead, it distributes bandwidth evenly across all links in the network. We believe that ARIEL could also support per-link traffic reservation, as implemented in CONAN-TSN and EAST; however, the version from [4], which we used in our comparison, does not include this feature. For this reason, we compare schedulability using both optimized and non-optimized credit assignment of EAST and CONAN-TSN. In the non-optimized (NO) configuration, since 25% of the total bandwidth is reserved for BE traffic and the remaining non-ST traffic is evenly divided between AVB classes A and B, an idleSlope of $(1 - 0.25)/2$ was assigned to both AVB class A and class B traffic.

The performance of the WCRTAs was evaluated under varying network utilizations, ranging from 5% to 45% for the N1 network and from 10% to 50% for the N2 network. This configuration ensured that up to 300 streams on N1 and up to 350 streams on N2 consistently achieved the target utilization across almost all links. For each utilization level, 100 traffic sets were generated, resulting in 900 networks analyzed per experiment and a total of 2700 network use-cases.

Fig. 4 shows the experimental setup. The networks are generated according to the parameters described earlier through the Network generator ① and classified into different types of TSN traffic using the LETRA mapping tool ②, both included in CONAN-TSN [7]. These two tools generate sets of traffic streams classified as ST, AVB, and BE, which are subsequently configured and analyzed using the different approaches. Hence, the generated traffic serves as input to the 3 evaluated approaches: CONAN-TSN, ARIEL, and EAST ③.

CONAN-TSN follows the conventional approach of first scheduling ST traffic (C1) and subsequently analyzing AVB traffic (C2). Hence, the ST traffic generated in the previous steps is used by HERMES to synthesize the ST schedule. Next, the ST traffic, the resulting ST schedule, the AVB traffic, and the BE traffic are used as inputs to the WCRTA. If a feasible schedule is obtained for the ST traffic and the WCRT of all AVB frames is less than or equal to their deadlines, the stream set is considered schedulable; otherwise, it is deemed non-schedulable under CONAN-TSN (C3).

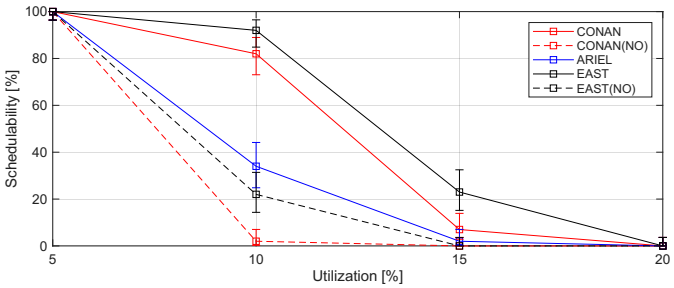


Fig. 5: Schedulability comparison for network N1 with 12.5% ST traffic under different network utilizations.

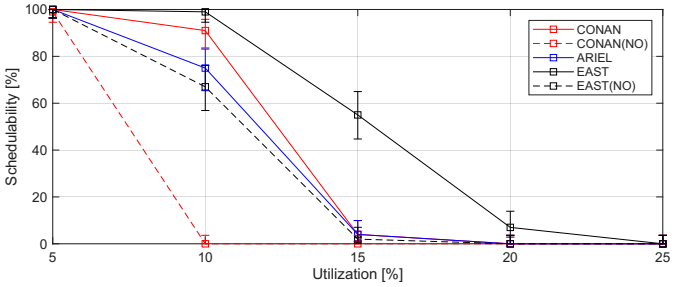


Fig. 6: Schedulability comparison for network N1 with 25% ST traffic under different network utilizations.

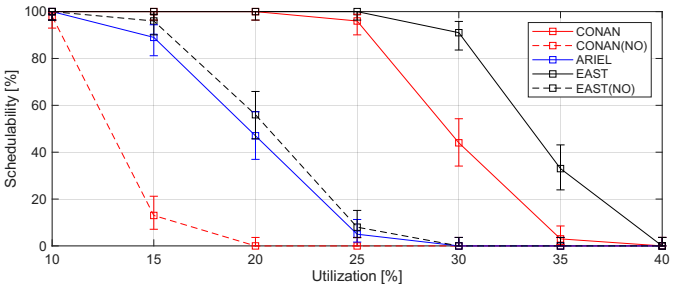


Fig. 7: Schedulability comparison for network N2 with 12.5% ST traffic under different network utilizations.

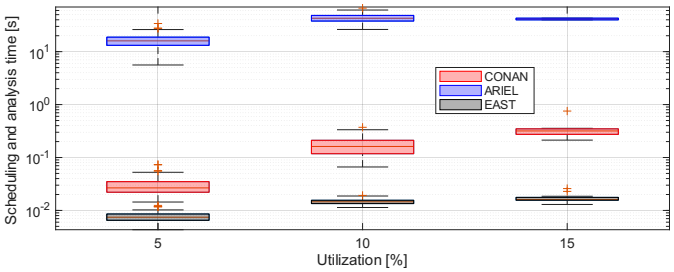


Fig. 8: Scheduling and analysis time for network N1 with 12.5% ST traffic under different network utilizations.

ARIEL, on the other hand, iteratively adjusts the ST schedule based on the schedulability of AVB traffic. Specifically, ARIEL starts in the same way as CONAN-TSN by scheduling the ST traffic generated in the previous steps using an ASAP heuristic scheduler (A1). It then analyzes all traffic together with the schedule, as in CONAN-TSN, but using a Network Calculus–based WCRTA (A2). If a feasible ST schedule is obtained and all AVB frames meet their deadlines, the stream set is considered schedulable (A3). Otherwise, the process is repeated by progressively spacing the ST transmission windows until the stream set becomes schedulable or the ST schedule can no longer be further dispersed (A4).

In contrast, EAST begins by analyzing the AVB traffic without considering either the ST traffic or the ST schedule in order to determine the maximum allowable ST interference (E1). This parameter is then applied as a constraint to the ST scheduler (E2). If a feasible ST schedule is obtained under this constraint, the entire traffic set is considered schedulable (E3). Note that, although not depicted in the figure since it is not part of EAST itself, we additionally execute the CONAN WCRTA a posteriori to verify that, in all cases where EAST provides a feasible ST schedule, the remaining traffic also meets its requirements. This additional check is not required by EAST and is performed solely for validation purposes. At the end all results from all methods are compared and graphed ④.

B. Results and Discussion

This section presents the experimental results comparing the three approaches: CONAN-TSN with and without AVB credit optimization per link (denoted as *CONAN* and *CONAN (NO)*), ARIEL, and EAST with and without AVB credit optimization per link (denoted as *EAST* and *EAST (NO)*). Figures 5 – 7 show the *Schedulability* achieved by each approach for different network utilization levels. The y-axis shows the schedulability ratio, and the x-axis shows the evaluated traffic utilizations. Each figure corresponds to a distinct network configuration: (i) network N1 with 12.5% of the traffic utilization being ST, (ii) network N1 with 25% of the traffic being ST, and (iii) network N2 with 12.5% ST traffic. Figures 8 – 10 show the *Scheduling & Analysis Time* for the same 3 configurations. The log y-axis shows the synthesis time (including AVB analysis time) and the x-axis corresponds to the different network utilizations. The plots exclude the scheduling and analysis time of *CONAN (NO)* and *EAST (NO)* since their values overlap with those of *CONAN* and *EAST*, respectively. Finally, Figures 11 – 13 focus exclusively on *EAST* and *EAST (NO)* to emphasize that the runtimes overlap almost perfectly, demonstrating that AVB credit optimization per link has a negligible effect on the scheduling time.

Several observations can be drawn from the results. First, *EAST* consistently achieves higher schedulability than *CONAN-TSN*, i.e., more networks are schedulable with *EAST* than with *CONAN-TSN*. Moreover, both *CONAN-TSN* and *EAST* outperform *ARIEL* in terms of schedulability, mainly due to the per-link credit optimization. However, when comparing *CONAN (NO)* with *ARIEL*, it exhibits lower schedulability, whereas *EAST (NO)* shows only minimal differences and is even superior in some cases.

This behavior is expected, as *ARIEL* is a heuristic that explores all possible gap alternatives within the ST schedule,

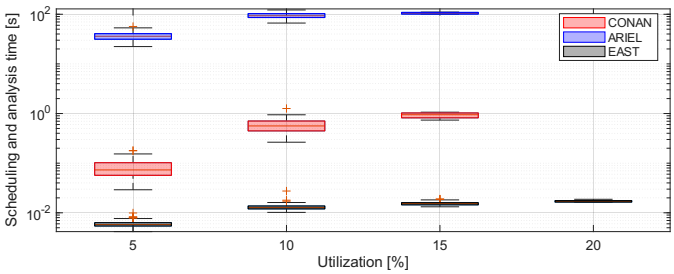


Fig. 9: Scheduling and analysis time for network N1 with 25% ST traffic under different network utilizations.

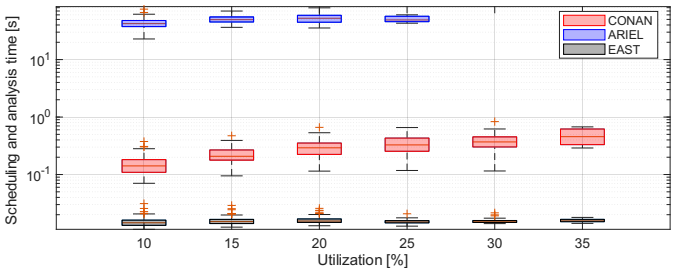


Fig. 10: Scheduling and analysis time for network N2 with 12.5% ST traffic under different network utilizations.

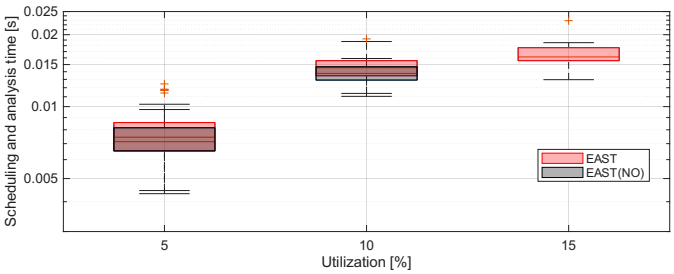


Fig. 11: Comparison of EAST and EAST (NO) scheduling and analysis time for network N1 with 12.5% ST traffic.

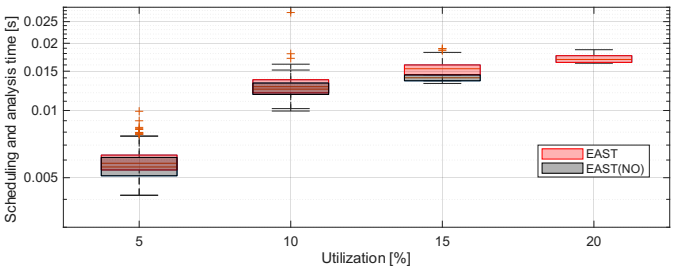


Fig. 12: Comparison of EAST and EAST (NO) scheduling and analysis time for network N1 with 25% ST traffic.

while *EAST* enforces gap placements based on the ST traffic distribution across links Eq. (15). Consequently, *EAST* does not guarantee an optimal gap distribution, but it provides a highly efficient one. Furthermore, the use of different analysis methods (NC for *ARIEL* and WCRTA for *EAST*), whose performance depends on the network characteristics, accounts for the performance discrepancies observed between the two.

The true improvement of *EAST*, however, becomes evident when analyzing the scheduling time. *EAST* requires several orders of magnitude less time than *ARIEL*. This is because *ARIEL* requires numerous iterations and evaluations, which are compute expensive. Even more remarkably, *EAST* is also significantly faster than *CONAN-TSN*, despite both performing a single iteration of ST scheduling and AVB analysis.

Method	**Max (s)**	**Average (s)**	**Min (s)**
Experiment 1 – Network N1 12.5% ST, Fig. 8			
ARIEL	66.4400	23.4598	5.6000
CONAN-TSN	0.7543	0.1028	0.0144
EAST	0.0259	0.0116	0.0043
Experiment 2 – Network N1 25% ST, Fig. 9			
ARIEL	122.0600	61.9915	22.3300
CONAN-TSN	1.2620	0.3316	0.0291
EAST	0.0275	0.0110	0.0042
Experiment 3 – Network N2 12.5% ST, Fig. 10			
ARIEL	79.4900	47.7862	22.8400
CONAN-TSN	0.8330	0.2668	0.0704
EAST	0.0312	0.0155	0.0111

TABLE I: Max, average, and min execution times (in seconds) for each experiment and approach.

Table I shows the maximum, minimum, and average execution times of each approach across all experiments. As shown in the table, ARIEL produces results in the range of tens or even hundreds of seconds, while CONAN-TSN operates within the range of hundreds of milliseconds, and EAST rarely exceeds 30 milliseconds. This implies a difference of 5 orders of magnitude between EAST and ARIEL, and 1 order of magnitude between EAST and CONAN-TSN.

This efficiency stems from the fact that it is much faster to compute the *Maximum ST Interference* and apply it as a constraint to the ST scheduler than to first calculate the ST schedule and then analyze it. When analyzing AVB schedulability based on a given ST schedule, all critical instant candidates (Eq. (7)) must be evaluated to determine which yields the maximum delay. This process is far more time-consuming than directly constraining the ST scheduler with interference limits. Furthermore, the experiments confirmed that obtaining an ST schedule that satisfies the *Maximum ST Interference* constraint is sufficient to guarantee the schedulability of AVB traffic; therefore, it is not necessary to perform a new AVB analysis afterwards. The high schedulability, combined with the fast configuration and analysis times of EAST, makes it an ideal tool for the online reconfiguration of TSN networks.

C. THALES Industry Challenge

We evaluated EAST using the THALES Industrial Challenge [5]. The THALES use-case features a network of 15 end-stations (ES) and 5 switches (SW) connected in a highly redundant mesh configuration. There are 241 streams with frame sizes ranging from 138 to 1503 bytes. Stream periods are in the set $\{200\mu s, 320\mu s, 400\mu s, 800\mu s, 1.6ms, 3.2ms, 6.4ms\}$. The overall network load is light to medium, with some back-

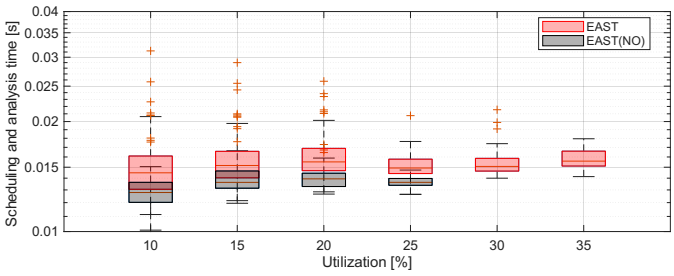


Fig. 13: Comparison of EAST and EAST (NO) scheduling and analysis time for network N2 with 12.5% ST traffic.

bone links having higher utilization. This benchmark specifies a highly demanding avionics network in which only methods that incorporate the optimized credit assignment described in Sec. V-D can achieve feasible configurations. Note that ARIEL may also be able to find such a configuration when given a suitable per-link credit allocations; however, the version from [4] does not incorporate this optimization. Table II shows the scheduling and analysis times for both CONAN-TSN and EAST with per-link AVB credit optimization.

EAST requires significantly less runtime than CONAN-TSN, being about 38 times faster while providing better schedulability. The credit constraint introduces additional complexity for the scheduler, increasing scheduling time by roughly a factor of 3. However, EAST does not need to analyze the interference of the entire ST schedule on AVB traffic, resulting in a substantial reduction in AVB analysis time, specifically 85 times faster than CONAN-TSN. In addition, EAST improves schedulability by generating the necessary gaps in the ST schedule to ensure AVB traffic becomes schedulable, based on the *ex ante* calculated Max_STI interference.

D. Hyperperiod scalability

We performed an experiment where we increased the hyperperiod (HP) duration from $60ms$ to $9.26min$ (Fig. 14). As expected, the total runtime becomes dominated by the ST synthesis as the HP increases since the underlying schedule construction is an NP-complete problem. Any ST synthesis algorithm will suffer from the same exponential behavior in the worst case (unless $P = NP$). However, the analysis step of EAST (i.e. Max_STI calculation) is independent of the HP explosion, remaining nearly constant across all HP values, since it relies solely on per-link interference bounds (SPI, HPI, LPI) and does not enumerate ST transmission windows over the HP. Consequently, its complexity depends on the number of streams and links, but not on the HP.

Steps	**CONAN-TSN**	**EAST**
Max ST analysis	-	6.676 ms
ST Scheduling	2.589 ms	8.179 ms
AVB analysis	565.788 ms	-
TOTAL time	568.377 ms	14.855 ms

TABLE II: THALES industrial challenge synthesis time.

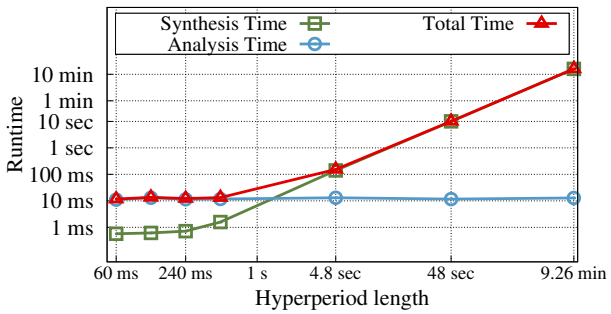


Fig. 14: EAST scalability in function of hyperperiod duration.

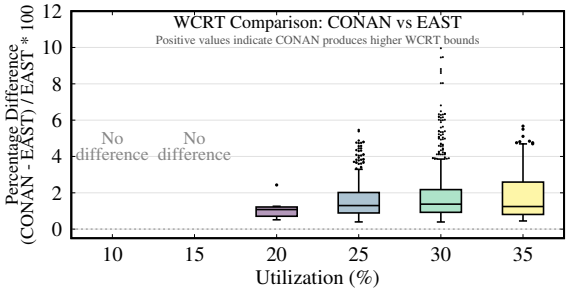


Fig. 15: WCRT comparison between CONAN and EAST.

E. WCRTA bound comparison

Finally, for test cases where both CONAN and EAST provide feasible schedules, we compare the WCRTA bounds of each analysis at different utilization levels. The percentage difference is computed as $((CONAN - EAST)/EAST) * 100$ for each sample where the values differ. Fig. 15 shows the distribution of these percentage differences for each utilization level. At lower utilization levels $(10\%, 15\%)$, both methods produce identical WCRT bounds. Here, no difference is observed because the ST scheduler constraint does not actually limit the packing of ST windows, so the ST schedule remains unaffected. In all cases where differences exist, CONAN produces more pessimistic WCRT bounds than EAST, with typical differences ranging from 0.5% to 2% and outliers reaching up to 10%. With increasing utilization, EAST enables dense packing of ST windows within AVB-imposed limits and thus reduces latency and guardband bandwidth waste.

VII. Conclusion

We presented EAST, a constraint-driven approach for scheduling mixed ST and AVB traffic in TSN networks under end-to-end timing constraints. By computing the maximum admissible ST interference through WCRT analysis and translating AVB timing requirements into constraints for ST synthesis, EAST eliminates iterative feedback loops between scheduling and analysis. This enables single-pass synthesis while guaranteeing schedulability for both traffic classes. Experiments on synthetic and real-world benchmarks demonstrate orders-of-magnitude runtime improvements over state-of-the-art methods, with equal or better schedulability. Hence, EAST is well suited for online reconfiguration in dynamic industrial and automotive TSN-networked systems.

ACKNOWLEDGEMENTS

This research was co-funded by the European Union and Innovation Fund Denmark as part of the Shift2SDV project (Grant Agreement No. 101194245). In addition, this work was supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) through the INTERCONNECT and FLEXATION projects, and by the Swedish Knowledge Foundation (KKS) through the SEINE project. The authors would also like to thank the industrial partners HIAB, Arcticus, ABB, and Westermo for their support.

REFERENCES

[1] A Ademaj, D Puffer, D Bruckner, G Ditzel, L Leurs, MP Stanica, P Didier, R Hummen, R Blair, and T Enzinger. Industrial automation traffic types and their mapping to QoS/TSN mechanisms. *TSN mechanisms*, 3, 2019.

[2] Anna Arestova, Wojciech Baron, Kai-Steffen J. Hielscher, and Reinhard German. ITANS: Incremental task and network scheduling for time-sensitive networks. *IEEE Open Journal of Intelligent Transportation Systems*, 3:369–387, 2022. doi:10.1109/OJITS.2022.3171072.

[3] Mohammad Ashjaei, Mikael Sjödin, and Saad Mubeen. A novel frame preemption model in TSN networks. *Journal of Systems Architecture*, 114(101914):1–30, June 2021. doi:10.1016/j.sysarc.2021.102037.

[4] Aldin Berisa, Luxi Zhao, Silviu S. Craciunas, Mohammad Ashjaei, Saad Mubeen, Masoud Daneshtalab, and Mikael Sjödin. Avb-aware routing and scheduling for critical traffic in time-sensitive networks with preemption. In *Proc. RTNS*. ACM, 2022. doi:10.1145/3534879.3534926.

[5] Marc Boyer and Rafik Henia. Embedded reconfiguration of TSN (industrial challenge description). In Renato Mancuso, editor, *Proc. ECRTS*, volume 335 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. URL: https://doi.org/10.4230/LIPIcs.ECRTS.2025.22, doi:10.4230/LIPICS.ECRTS.2025.22.

[6] Dietmar Bruckner, Marius-Petru Stănică, Richard Blair, Sebastian Schriegel, Stephan Kehrer, Maik Seewald, and Thilo Sauter. An introduction to OPC UA TSN for industrial communication systems. *Proceedings of the IEEE*, 107(6):1121–1131, 2019. doi:10.1109/JPROC.2018.2888703.

[7] Daniel Bujosa, Mohammad Ashjaei, and Saad Mubeen. CONAN-TSN: An Integrated Toolchain for CONfiguration and ANalysis of TSN Networks. In *Proc. ETFA*, pages 1–8. IEEE, 2025. doi:10.1109/ETFA65518.2025.11205713.

[8] Daniel Bujosa, Julian Proenza, Alessandro V Papadopoulos, Thomas Nolte, and Mohammad Ashjaei. An improved worst-case response time analysis for AVB traffic in time-sensitive networks. In *Proc. RTSS*. IEEE, 2024. doi:10.1109/RTSS62706.2024.00021.

[9] 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 *Proc. RTNS*, 2022. doi:10.1145/3534879.3534906.

[10] Martin Böhm and Diederich Wermser. Multi-domain time-sensitive networks—control plane mechanisms for dynamic inter-domain stream configuration. *Electronics*, 10(20), 2021. doi:10.3390/electronics10202477.

[11] Nurefşan Sertbaş Bülbül, Doğanalp Ergenç, and Mathias Fischer. Towards SDN-based dynamic path reconfiguration for time sensitive networking. In *Proc. NOMS*, pages 1–9, 2022. doi:10.1109/NOMS54207.2022.9789890.

[12] 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. doi:10.1007/s11241-018-9321-z.

[13] Silviu S. Craciunas, Ramon Serna Oliver, Martin Chmelik, and Wilfried Steiner. Scheduling real-time communication in IEEE 802.1Qbv Time Sensitive Networks. In *Proc. RTNS*. ACM, 2016. doi:10.1145/2997465.2997470.

[14] Robert I Davis, Steffen Kollmann, Victor Pollex, and Frank Slomka. Controller Area Network (CAN) Schedulability Analysis with FIFO Queues. In *Proc. ECRTS*. IEEE, 2011. doi:10.1109/ECRTS.2011.13.

[15] Frank Dürr and Naresh Ganesh Nayak. No-wait packet scheduling for ieee time-sensitive networks (TSN). In *Proc. RTNS*. ACM, 2016. doi:10.1145/2997465.2997494.

[16] Anais Finzi and Silviu S. Craciunas. Integration of SMT-based scheduling with RC network calculus analysis in TTEthernet networks. In *Proc. ETFA*, 2019. doi:10.1109/ETFA.2019.8869365.

[17] Anaïs Finzi, Silviu S. Craciunas, and Marc Boyer. Integrating sporadic events in time-triggered systems via affine envelope approximations. In *Proc. RTAS*, 2024. doi:10.1109/RTAS61025.2024.00010.

[18] Voica Gavrilut, Bahram Zarrin, Paul Pop, and Soheil Samii. Fault-tolerant topology and routing synthesis for ieee time-sensitive networking. In *Proceedings of the 25th International Conference on Real-Time Networks and Systems*, RTNS '17, page 267–276, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3139258.3139284.

[19] 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. doi:10.1109/ACCESS.2018.2883644.

[20] Christoph Gärtner, Amr Rizk, Boris Koldehofe, René Guillaume, Ralf Kundel, and Ralf Steinmetz. Fast incremental reconfiguration of dynamic time-sensitive networks at runtime. *Computer Networks*, 224:109606, 2023. doi:10.1016/j.comnet.2023.109606.

[21] Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, and Saad Mubeen. Synthesising schedules to improve qos of best-effort traffic in tsn networks. In *Proc. RTNS*. ACM, 2021. doi:10.1145/3453417.3453423.

[22] 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. doi:10.1016/j.jpdc.2020.06.001.

[23] Lisa Maile, Kai-Steffen Hielscher, and Reinhard German. Combining Static and Dynamic Traffic with Delay Guarantees in Time-Sensitive Networking. In *Proc. ValueTools*. Springer, 2023. doi:10.1007/978-3-031-48885-6_8.

[24] Lisa Maile, Dominik Voitlein, Alexej Grigorjew, Kai-Steffen Hielscher, and Reinhard German. On the validity of credit-based shaper delay guarantees in decentralized reservation protocols. In *Proc. RTNS*, pages 108–118. ACM, 2023. doi:10.1145/3575757.3593644.

[25] J. Maki-Turja and M. Nolin. Fast and tight response-times for tasks with offsets. In *Proc. ECRTS*, 2005. doi:10.1109/ECRTS.2005.15.

[26] N. G. Nayak, F. Dürr, and K. Rothermel. Incremental flow scheduling and routing in time-sensitive software-defined networks. *IEEE Trans Industr Inform*, 14(5), 2018. doi:10.1109/TII.2017.2782235.

[27] M. Pahlevan and R. Obermaisser. Genetic algorithm for scheduling time-triggered traffic in time-sensitive networks. In *Proc. ETFA*, 2018. doi:10.1109/ETFA.2018.8502515.

[28] Paul Pop, Michael Lander Raagaard, Silviu S. Craciunas, and Wilfried Steiner. Design optimization of cyber-physical distributed systems using IEEE time-sensitive networks (TSN). *IET-CPS*, 1(1):86–94, 2016. doi:10.1049/iet-cps.2016.0021.

[29] Mario Qosja, Utkarsh Raj, Simon Meckel, and Roman Obermaisser. Dynamic TSN reconfiguration for time-triggered organic computing. *Procedia Computer Science*, 257:364–373, 2025. doi:10.1016/j.procs.2025.03.048.

[30] Ramon Serna Oliver, Silviu S. Craciunas, and Wilfried Steiner. IEEE 802.1Qbv Gate Control List Synthesis using Array Theory Encoding. In *Proc. RTAS*. IEEE, 2018. doi:10.1109/RTAS.2018.00008.

[31] Wilfried Steiner. An evaluation of SMT-based schedule synthesis for time-triggered multi-hop networks. In *Proc. RTSS*. IEEE, 2010. doi:10.1109/RTSS.2010.25.

[32] Wilfried Steiner. Synthesis of static communication schedules for mixed-criticality systems. In *Proc. ISORCW*, pages 11–18, 2011. doi:10.1109/ISORCW.2011.12.

[33] Thomas Stüber, Lukas Osswald, Steffen Lindner, and Michael Menth. A survey of scheduling algorithms for the time-aware shaper in time-sensitive networking (tsn). *IEEE Access*, 11:61192–61233, 2023. doi:10.1109/ACCESS.2023.3286370.

[34] Marek Vlk, Kateřina Brejchová, Zdeněk Hanzálek, and Siyu Tang. Large-scale periodic scheduling in time-sensitive networks. *Comput.*

Oper. Res., 137(C), January 2022. doi:10.1016/j.cor.2021.105512.

[35] Meng Wang and Yiqin Lu. An improved avb-aware scheduling of time-triggered traffic in time-sensitive networks. *Computer Communications*, 242:108292, 2025. doi:10.1016/j.comcom.2025.108292.

[36] Anlan Xie, Feng He, and Luxi Zhao. Optimizing quantum assignment for DRR in TSN: A network calculus-based method. In *Proc. RTSS*, 2024. doi:10.1109/RTSS62706.2024.00020.

[37] Chuanyu Xue, Tianyu Zhang, Yuanbin Zhou, Mark Nixon, Andrew Loveless, and Song Han. Real-time scheduling for 802.1Qbv time-sensitive networking (TSN): A systematic review and experimental study. In *Proc. RTAS*, 2024. doi:10.1109/RTAS61025.2024.00017.

[38] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. Timing analysis of AVB traffic in TSN networks using network calculus. In *Proc. RTAS*, 2018. doi:10.1109/RTAS.2018.00009.

[39] Luxi Zhao, Yida Yan, and Xuan Zhou. Minimum bandwidth reservation for CBS in TSN with real-time QoS guarantees. *IEEE Transactions on Industrial Informatics*, 20(4):6187–6198, 2024. doi:10.1109/TII.2023.3342466.

[40] Yuanbin Zhou, Soheil Samii, Petru Eles, and Zebo Peng. Reliability-aware scheduling and routing for messages in time-sensitive networking. *ACM Trans. Embed. Comput. Syst.*, 20(5), May 2021. doi:10.1145/3458768.