Article

Network Calculus-Based Latency for Time-Triggered Traffic under Flexible Window-Overlapping Scheduling (FWOS) in a Time-Sensitive Network (TSN)

Khaled M. Shalghum ^{1,2,3,*} , Nor Kamariah Noordin ^{1,3,*}, Aduwati Sali ^{1,3} and Fazirulhisyam Hashim ^{1,3}

¹ Department of Computer and Communication Engineering, Universiti Putra Malaysia, Serdang 43400, Malaysia; aduwati@upm.edu.my (A.S.); fazirul@upm.edu.my (F.H.)

² Department of Electrical and Electronic Eng., Azzaytuna University, Tarhuna 00218, Libya

³ Wireless and Photonic Networks Research Centre of Excellence (WiPNet), Faculty of Engineering, UPM, Serdang 40150, Malaysia

* Correspondence: kshalghum@gmail.com (K.M.S.); nknordin@upm.edu.my (N.K.N.)

Abstract: Deterministic latency is an urgent demand to pursue the continuous increase in intelligence in several real-time applications, such as connected vehicles and automation industries. A time-sensitive network (TSN) is a new framework introduced to serve these applications. Several functions are defined in the TSN standard to support time-triggered (TT) requirements, such as IEEE 802.1Qbv and IEEE 802.1Qbu for traffic scheduling and preemption mechanisms, respectively. However, implementing strict timing constraints to support scheduled traffic can miss the needs of unscheduled real-time flows. Accordingly, more relaxed scheduling algorithms are required. In this paper, we introduce the flexible window-overlapping scheduling (FWOS) algorithm that optimizes the overlapping among TT windows by three different metrics: the priority of overlapping, the position of overlapping, and the overlapping ratio (OR). An analytical model for the worst-case end-to-end delay (WCD) is derived using the network calculus (NC) approach considering the relative relationships between window offsets for consecutive nodes and evaluated under a realistic vehicle use case. While guaranteeing latency deadline for TT traffic, the FWOS algorithm defines the maximum allowable OR that maximizes the bandwidth available for unscheduled transmission. Even under a non-overlapping scenario, less pessimistic latency bounds have been obtained using FWOS than the latest related works.

Keywords: mixed-criticality real-time systems; time-sensitive network (TSN); scheduling algorithm; worst-case latency performance; network calculus

Citation: Shalghum, K.M.; Noordin, N.K.; Sali, A.; Hashim, F. Network Calculus-Based Latency for Time-Triggered Traffic under Flexible Window-Overlapping Scheduling (FWOS) in a Time-Sensitive Network (TSN). *Appl. Sci.* **2021**, *11*, 3896. <https://doi.org/10.3390/app11093896>

Academic Editor: Luigi Pomante

Received: 31 January 2021

Accepted: 17 March 2021

Published: 25 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

There is an urgent demand for high-speed and deterministic end-to-end communications to integrate several real-time applications. Missing latency deadlines may generate dangerous situations for humans or considerable economic waste in highly time-sensitive applications, such as autonomous vehicles and automation industries. Ethernet networking has sufficient bandwidth with reasonable costs for a wide range of application domains. It was used to build several relevant protocols, such as Audio/Video Bridging Ethernet (AVB-Ethernet) and time-triggered Ethernet (TT-Ethernet) networks. Although these technologies have provided several real-time functions, their capabilities are unable to manage the control flows in safety-critical systems (e.g., autonomous vehicles) and handle the continuous increase in such applications' intelligence.

A time-sensitive network (TSN) is a set of sub-standards that introduce an attractive solution to support safety-critical applications based on several TT-Ethernet extensions [1]. The amendments include timing and access control aspects to guarantee data transport with a deterministic low latency, extremely low delay variation, and zero congestion loss

for urgent traffic. These features have attracted many researchers and companies to the TSN framework for their interests [2].

Basically, the TSN framework serves mixed-criticality applications under three different traffic categories: time-triggered (TT), Audio/Video Bridging (AVB), and best effort (BE). The TT traffic represents the highest-priority class that requires a guaranteed zero jitter delay and a tightly bounded end-to-end latency. TT flows are configured by using the time-aware shaping (TAS) mechanism in the gate control list (GCL) schedule, as defined in IEEE 802.1Qbv [3]. The AVB traffic represents the second-highest priority class that is configured by using the credit-based shaping (CBS) mechanism to meet the deterministic end-to-end latency requirement, as presented in IEEE 802.1Qav [4]. The BE type represents the lowest-priority class, with no guaranteed timing requirements.

Implementing an appropriate GCL timing table for TT traffic in all selected switches while ensuring critical time streams' latency requirements is a complex and crucial problem. The complexity arises not only from the difficulties to satisfy TT demands but also because the other critical time flows, i.e., the AVB, must be considered. Excluding the soft real-time traffic from the predefined schedule (GCL) and allowing the TT flows to preempt them (as defined in IEEE 802.1Qbu [5]) may cause them to miss their latency deadlines, as many researchers have argued [6–9]. For these reasons, proper GCL designs are essential to guarantee safe latencies for all critical time flows.

Different scheduling solutions have been proposed to ensure latency requirements for hard real-time flows and enhance AVB latency performance at the same time. As the GCL implementation is window based, each priority class has to transmit related traffic through the assigned windows. Different window patterns have been proposed to enhance the overall system performance, as discussed in Section 2. Most of the relevant researchers in the literature, e.g., in [10–13], have suggested allowing TT open windows to interfere with each other, leading to more available bandwidth for unscheduled transmissions. Although the researchers in [11,12] have improved the overall solution space, no one has optimized how much TT windows can be overlapped without missing TT requirements. Accordingly, this paper focuses on presenting an optimal formal solution for the scheduling problem based on TT open windows' overlapping assumption.

In this paper, we present a complete end-to-end latency analysis for TT traffic based on the variable overlapping ratio (OR) between TT open windows in the predefined GCL. The GCL schedule is designed for all selected nodes, considering the relative positional relationship between window offsets for adjacent nodes. The GCLs are implemented according to three overlapping metrics: the priority of the overlapped window, the position of overlapping, and the overlapping ratio (OR). Based on these parameters, the worst-case end-to-end delay (WCD) bounds are determined for a targeted TT queue, providing a complete view of WCD performance under every expected OR. Thus, the critical contributions of the paper are summarized as follows:

- We propose a flexible window-overlapping scheduling (FWOS) algorithm that allows TT windows to overlap each other and considers the relative positional relationships between window offsets for consecutive nodes. The network calculus (NC) approach is used to formulate the worst-case latency bounds. The FWOS algorithm is evaluated under a realistic vehicle use case considering three overlapping metrics: the priority of overlapping, the position of overlapping, and the overlapping ratio (OR). A critical discussion is introduced based on GCL design parameters.
- Based on the latency performance evaluation under flexible overlapping scenarios, we create an additional TSN scheduling constraint, leading to more relaxed GCL implementations. For each particular latency deadline, the FWOS algorithm defines the maximum allowable OR that maximizes the solution space for unscheduled traffic, while guaranteeing TT latency requirements at the same time. Additionally, the FWOS algorithm presents an improvement in WCD boundaries, even in a complete isolation scenario among TT windows compared with the latest related works.

The remainder of the paper is organized as follows. Section 2 describes related works, and Section 3 introduces a general background about the basic structure of the TSN system and a brief overview of the IEEE 802.1Qbv protocol. Our system model and the basics of the network calculus approach are described in Sections 4 and 5, respectively. Section 6 presents the worst-case latency analysis for the FWOS algorithm. Then, we evaluate the performance of the FWOS model, with a relevant discussion in Section 7. Finally, the paper is concluded in Section 7.

2. Related Work

The IEEE TSN standards define two different shaping approaches, namely time-aware shaping (TAS) in IEEE 802.1Qbv [5] for scheduled streams and credit-based shaping (CBS) for unscheduled flows in IEEE 802.1Qav [4]. Although other shapers can implement the traffic schedule in TT-Ethernet, such as the burst-limiting shaper (BLS) [14,15] and the peristaltic shaper (PS) [16], the TSN task group has standardized the TAS to shape TT streams. The TAS mechanism is more applicable than the others to ensure a bounded latency for control traffic [17].

As an alternative to the TAS technique, some researchers have proposed asynchronous time shaping (ATS) as it does not require the selected nodes to be fully synchronized. The ATS mechanism is presented in the ongoing IEEE P802.1Qcr project. ATS-aware nodes can temporally reshape incoming frames in a different queuing system using an urgency-based scheduler (UBS). Specht et al. [18,19] and Zhou et al. [20,21] proved that the ATS mechanism provides low and bounded worst-case latencies with high link use, low complexity, temporal reshaping, and complete synchronization independence. In [22], the authors concluded that the flow control strategy in the ATS method provides a better real-time performance guarantee for aperiodic traffics than the CBS in a heavy network load. However, after comparing TAS and ATS performances, Nasrallah et al. [23] confirmed that the TAS method with tight configurations achieves the related latency targets for both sporadic and periodic data streams. In contrast, the preferred use of the ATS approach is only in the case of light network loads.

Several scheduling models have been implemented based on the TAS mechanism and have examined the latency performance for TT traffic under the complete synchronization assumption. The targeted TT latency deadlines have been guaranteed using simulation-based algorithms, e.g., in [24,25], and analytically, e.g., in [16,26]. Besides, many researchers have suggested improving the GCL design to increase the scalability and adaptability that fit critical time applications.

Fixed-timing algorithms result in a fluctuated quality of service (QoS) for all traffic types, depending on the instantaneous loading condition and network size. Therefore, Nasrallah et al. [23] proposed an adaptive duration for scheduled and unscheduled transmission intervals in the GCL design. The window duration is adjusted based on each traffic class's latency threshold and the incoming flows' specifications. OMNet++ results confirmed that queuing delay increases with decreasing window duration for related traffic and vice versa. Kim et al. [27] added a protection interval to transmit the emergency event flows in each transmission cycle rapidly. The simulation results ensured the required services for emergency traffic, with minimal impact on the scheduled flows. Oliver et al. [28] and Hellmanns et al. [29] addressed the essential design compromises between scheduling efficiency and physical resources affecting the required scalability, adaptability, and self-configuration of the Industrial Internet of Things (IIoT) system. The authors also suggested adapting the hyper-period length and raster size according to the incoming traffic. In [30], Farzaneh et al. proposed adjusting the GCL according to the network topology and loading conditions' availability. Yu et al. [31] presented a scheduling design with online updating to suitably integrate incoming data. Several other pieces of the research have proposed joint routing and scheduling algorithms, as in [32–34]. Other interesting works have suggested using the software-defined network (SDN) protocol to manage TSN scheduling, as in [1,35,36]. However, all the above algorithms are simulation

based, which cannot express all pessimistic transmission cases and, then, leads to unsafe latency evaluations [37].

As known, critical time applications have hard and soft real-time traffic. Implementing GCLs with strict timing constraints reduces the overall solution space, minimizing the bandwidth available for unscheduled traffic and increasing the possibility of mislaying the required determinism for AVB flows. Moreover, such strict timing scheduling methods complicate the GCL synthesis in large-scale use cases. For these reasons, several proposals have been introduced to support soft time traffic, while simultaneously ensuring the QoS requirements for hard real-time flows.

Gavrilit et al. [9,38] proposed a software algorithm based on the greedy randomized adaptive search procedure (GRASP), including the AVB traffic to be scheduled with the TT traffic. The results confirmed that feasible AVB scheduling was performed using the proposed method under a simple use case. However, AVB schedulability cannot be ensured under heavy loading or more complicated networks. Zhang et al. [6] proposed combining hard and soft real-time flows simultaneously, leading to reduced computation complexity. The presented approach chooses a proper transmission cycle and scheduling unit to improve the selection process and reduce the higher-priority impact on the lower-priority flows. Others in [7,39,40] have introduced formal latency analysis for AVB traffic based on predefined GCLs. However, all the above algorithms were implemented under the assumption of complete isolation between scheduled traffic windows, resulting in less overall solution space.

Many scheduling proposals have addressed traffic and window isolations in the GCL design. Here, we discuss studies most related to our work and compare their limitations, as briefly presented in Table 1. Craciunas et al. [41] presented a scheduling scheme under the assumption that scheduled and unscheduled flows are entirely isolated. Further, the proposed GCL synthesis separates scheduled flows against one another by implementing fully isolated transmission windows and fully synchronized nodes. In [42], Craciunas et al. used the targeted bounds of end-to-end latencies and the duration of assigned windows for the scheduled traffic to measure the scalability and schedulability of the proposed algorithm. Oliver et al. [43] presented relaxed GCL synthesis by allowing scheduled frames to interfere at the same-priority queue. However, the associated queue gates were still in a mutually exclusive fashion.

Table 1. Design-based differentiation between relevant studies with associated achievements.

Reference	Gate Control List (GCL) Design	Synchronization Need	Traffic Isolation	Windows	Relative Offsets between Nodes	Key Achievement
[42]	Off-line	All nodes	Isolated	Isolated	Not considered	Deriving scheduling constraints
[43]	Off-line	All nodes	Isolated	Isolated	Not considered	Improved scheduling constraints
[44]	Off-line	All nodes	Interfered	Isolated	Not considered	Improved scheduling constraints
[10,45]	Off-line	All switches	Isolated	Isolated	Not considered	More relaxed schedules
[11]	Off-line	All switches	Isolated	Overlapped without optimization	Not considered	Improved solution space
[12]	Off-line	All switches	Isolated	Overlapped without optimization	Not considered	Preemption improvements
[13]	Off-line	All switches	Isolated	Isolated	Considered	Less pessimistic latencies

More relaxed schedules have been presented, in [10–12,44], under window-based implementation for each priority class. In [10,44], the algorithms have been designed without the need for end systems to be fully synchronized. However, synchronization is still required for all selected switches in the path. Zhao et al. [11] proved that complete isolation between TT windows would reduce the available solution space for soft real-time streams. They analytically formulated the worst-case end-to-end latency for TT flows, resulting in an enhancement of the overall solution space compared with that in the full isolation scenario, while satisfying TT flow requirements. However, the authors just made the improvement based on a particular overlapping case without considering how to optimize the overlapping for every networking scenario and every targeted latency deadline. Zhang et al. [12] examined the preemption technique's effect on the latency bounds based on the algorithm [11]. However, the proposed scheduling designs, in [10–12,44], were implemented under a single node design and did not consider the rational timing difference between the offsets for the same-priority queues in consecutive nodes [13]. These models resulted in high-pessimistic end-to-end latency bounds.

Based on the above, Zhao et al. [13] improved the worst-case latency bounds for TT traffic by considering the relative positional relationships between the same-priority windows. The introduced results realized in a dramatic enhancement of the latency bounds and reduced the degree of pessimism in [11,12]. However, the presented analysis did not include the overlapping situation among TT windows. Moreover, we observe that more latency correctness can be done on that model as the technical switching delay has been considered twice in the overall latency derivations. The first is included indirectly in the maximum waiting time calculation, and the second is added directly in the final end-to-end latency expression. Although this delay is considerably slight compared with other latency components, its impact is notable in large-scale network topologies. A summarized comparison between the above algorithms is illustrated in Table 1.

For latency evaluation, analytical models are statically designed to cover all system corner cases, which are incredibly significant for real-time systems [26], especially in strict automated designs and automotive driving systems. Although there are several analytical approaches used to implement latency models, network calculus (NC) theory [45] is preferred in most studies as it induces less pessimistic and safe worst-case latencies compared to other approaches [46,47]. Accordingly, we used the NC approach to calculate WCD bounds.

3. Relevant Background

This section introduces the basic architecture of a TSN and the mechanism of the IEEE 802.1Qbv protocol briefly.

3.1. Basic Architecture of a TSN

As mentioned previously, the TSN is a new extension of the conventional Ethernet network, although both systems' physical components are the same. Precisely, the TSN communication system consists of several end systems (ESs) (information sources and destinations), switches (SWs), and physical links that connect the ESs and SWs through the full-duplex communication technique, as depicted in Figure 1. The network model is an undirected graph $G(\mathcal{P}, \mathcal{V})$, where $\mathcal{P} = \{\downarrow_1, \dots, \downarrow_k\}$ represents the data links set, with $\mathcal{V} = ES \cup SW$, $ES = \{ES_1, \dots, ES_n\}$, and $SW = \{SW_1, \dots, SW_m\}$. Each dataflow link i corresponds to $\downarrow_i = [\mathcal{V}_a, \mathcal{V}_b]$ where $\mathcal{V}_a$ and $\mathcal{V}_b \in \mathcal{V}$. Each flow must pass through one or more links to move between source and destination, which correspond to the route or path $r_i \in \mathcal{R}$, where $\mathcal{R}$ is the set of an ordered sequence of data stream links that connect a single ES to one (unicast) or more (multicast) ESs. Here, we consider the rate of the egressed flows from a node h is denoted as C_h .

Figure 1. Example of a simple time-sensitive network (TSN) architecture: TSN components, unicast and multicast transmission, and alternative routes.

3.2. IEEE 802.1Qbv Protocol

The IEEE 802.1Qbv protocol defines the time-aware shaping (TAS) technique to control traffic transmissions between TSN elements using a gating mechanism [17]. Each switching egress port has eight priority queues, as depicted in Figure 2, each of which has a gate with either an open or a closed states. Changing the gate state is controlled by the GCL under guaranteed synchronization for all selected switches in the transmission path. Only one priority frame from a specific queue can be transmitted during a related open interval by the first-in first-out (FIFO) mechanism. TT gates are prioritized over other gates to send the scheduled frames if they are open simultaneously. However, if the open gates correspond to unscheduled flows, i.e., AVB and BE traffic, the credit-based shaper (CBS) is implemented between the gates.

Figure 2. An IEEE 802.1Qbv capable switch with multiple input and output ports, showing delay components between adjacent nodes.

The capable switch described in IEEE 802.1Qbv must simultaneously realize fabric switching, filtering, and traffic policy selection for flows arriving from the ingress ports [19]. Fabric switching and filtering processes redirect the arrival of traffic to the appropriate priority queue, as depicted in Figure 2. Using the GCL timing schedule, the queued frames are selected to the egress port and then to the physical link. The frames experience related delay components during the transmission process to achieve these switching functions, as differentiated in Figure 2.

4. FWOS System Model and Design Decisions

TSN switching considers two dependent traffic isolation mechanisms, i.e., spatial and temporal isolations, to differentiate between ingress flows. Spatial isolation is applied in each switch by selecting incoming frames to one of eight priority queues served by various forwarding constraints at the egress port. Temporal isolation is provided in the gate control list (GCL) schedules. The GCL is an off-line timetable specifying the open and close events for all associated gates. To protect TT flows from unscheduled traffic transmissions, the preemption technique is applied by allowing the TT flows to interrupt the transmission of non-expressed flows. Against other TT transmissions, complete isolation between TT windows in the GCL timings undoubtedly provides targeted deterministic latency behavior for scheduled (TT) traffic, as proposed in [6]. Nevertheless, complete window isolation wastes the available bandwidth for unscheduled traffic and may not guarantee the determinism for soft critical time traffic (AVB).

There are two ways to process scheduled and unscheduled traffic in a TSN, back-to-back and porosity configurations, as depicted in Figure 3, respectively. In both structures, flexible overlapping between scheduled open windows saves more available bandwidth for unscheduled flows. The overlapping can occur from either one edge or two window edges (opening and closing). The more the TT windows overlap, the more the available time intervals for unscheduled traffic increase. It cannot be neglected that flexible overlapping schedules negatively affect the performance of scheduled traffic. Thus, essential evaluations of timing are necessary to cover all overlapping situations.

Figure 3. Configurations of the time-triggered (TT) schedule under both full isolated windows and flexible overlapping windows: (a) back-to-back configuration and (b) porosity configuration.

In this work, we are interested in evaluating the performance of the time-aware shaping (TAS) technique under the flexible window-overlapping scheduling (FWOS) algorithm. This model allows TT windows to interfere with one another without missing the TT queues' latency deadlines. Hence, only the hard real-time flows (TT flows) are considered here for latency calculations, assuming that the set of all TT flows is denoted as $\mathcal{S}$. Each TT flow ($s_m \in \mathcal{S}$), where m is the priority of the flow, consists of a set of frames ($\mathcal{F}_m$). Each node h is designed to select the ingress frame to one of N_q^h TT priority queues ($Q_1^h, \dots, Q_{N_q^h}^h$), where Q_1^h and $Q_{N_q^h}^h$ represent the highest- and lowest-priority queues, respectively. Each Q_m^h frame is selected to the egress port by its gate (G_m^h) with the open-window length W_m^h and the period T_m^h . Note that the presented scheduling model is proposed assuming guaranteed synchronization for all selected switches.

For performance evaluation, we consider Q_k^h as a targeted queue with an open-window duration of W_k^h , where $1 \leq k \leq N_q$. The Q_k^h window may overlap with any Q_m^h window, as shown in Figure 4, and the overlapping ratio (OR) between them in the i -th cycle represents

the length of the time interval ($L_{k,m}^{h,i}$), when both windows are simultaneously open, divided by the total length of the Q_k^h window, $W_k^{h,i}$, as

$$OR_{k,m}^{h,i} = \frac{L_{k,m}^{h,i}}{W_k^{h,i}} \quad (1)$$

Figure 4. Overlapping configuration between two TT windows with different priority queues.

As presented in Figure 4, the OR varies from 0 to 1, covering all the expected overlapping conditions. Note that zero overlapping means that when the gate of the targeted TT queue (G_k^h) is open, all other TT gates are closed (complete isolation), and one means that in each moment when G_k^h is open, there is at least another TT gate open (complete overlapping), as clearly shown in Figure 4.

Based on the above overlapping formula and the overlapping pattern in Figure 4, we implement here the GCL schedule for all selected nodes and then calculate the worst-case end-to-end latency for the Q_k^h frames using the network calculus (NC) approach. Note that our GCL implementation is off-line based for worst-case analysis and the dynamism is not considered. Related details for the NC basis and the worst-case analysis are presented below in Sections 5 and 6, respectively.

5. Network Calculus Basics

The network calculus (NC) [48] approach is a powerful tool to represent the flows' timing properties in the domain of communication networks deterministically. These timing properties include strict upper- and lower-bound computations used for performance evaluations, such as end-to-end latencies, network use, and buffer requirements. In this paper, we use NC to determine the worst-case end-to-end latency for TT flows. Two appropriate curves have to be implemented, the arrival curve and the service curve, to describe the characteristics of flows and related nodes' availability. The analysis is mainly provided based on the min-plus, which uses convolution operation, as defined in the following formulas [48].

$$(f \otimes g)(t) = \inf_{0 \leq \lambda \leq t} \{f(t - \lambda) + g(\lambda)\}$$

$$(f \oslash g)(t) = \sup_{\lambda \geq 0} \{f(t + \lambda) - g(\lambda)\}$$

where $\inf$ and $\sup$ mean infimum and supremum, i.e., maximal lower bound and minimal upper bound, respectively. The notations $\otimes$ and $\oslash$ represent the convolution and deconvolution of min-plus operations, respectively.

The arrival curve $\alpha(t)$ describes the arrival process $R(t)$ of a stream. This process represents the cumulative function, at the ingress switching port, by counting the bits that reach the node until t , as expressed in this inequality [48]:

$$R(t) \leq \inf_{0 \leq \lambda \leq t} \{R(\lambda) + \alpha(t - \lambda)\} = (R \otimes \alpha)(t) \quad \forall \lambda < t$$

The arrival curve is modeled using the input traces and the traffic configuration, as combined in the triple p, j, d , where p represents the period of incoming streams, j denotes the jitter, and d is the lowest inter-arrival distance of traffic in the specified flow. Accordingly, the lower and upper arrival curves are formed as follows [48]:

$$\alpha^l(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor \quad (2)$$

$$\alpha^u(\Delta) = \min \left\{ \left\lceil \frac{\Delta + j}{p} \right\rceil, \left\lceil \frac{\Delta}{d} \right\rceil \right\} \quad (3)$$

The above bounds mean that in any time interval (Δ) , there arrive at least $\alpha^l(\Delta)$ and at most $\alpha^u(\Delta)$ stream events that are modeled by $\alpha(t)$. The arrival curve $\alpha(t)$ can be represented in a simple schematic diagram presented in Figure 5a. For the worst-case latency computations, the upper bound of the arrival curve is considered.

Figure 5. Examples of upper and lower bounds: (a) the arrival curve and (b) the service curve for a periodic time division multiple access (TDMA) system.

The service curve $\beta(t)$ describes the transmission availability of the network resources, as modeled in the departure process $R^*(t)$ of a stream. This process represents the cumulative function, at the egress switch port, by counting the bits that can be served in the related node up to time t , as expressed in the following inequality [48]:

$$R^*(t) \leq \inf_{0 \leq \lambda \leq t} \{R(\lambda) + \beta(t - \lambda)\} = (R \otimes \beta)(t) \quad \forall \lambda < t$$

In a given time interval ω , where the switch can serve the arrived streams by a rate C , the service curve's lower and upper bounds are specified depending on network resource availability. Figure 5b shows an example of service curve bounds for a periodic time division multiple access (TDMA) resource.

In each node, the traffic is expressed by the arrival curve at the ingress port and served by the egress port's service curve. The worst-case latency boundaries are obtained by

considering the maximum distance (D_{max}) between the upper-bound arrival curve ($\alpha^u(t)$) and lower-bound service curve ($\beta^l(t)$) [48], as represented in Figure 6.

$$D_{max}Del(\alpha^u, \beta^l) \quad (4)$$

$$D_{max} = \sup_{\lambda \geq 0} \left\{ \inf \left\{ \tau \geq 0 : \alpha^u(\lambda) \leq \beta^l(\lambda + \tau) \right\} \right\}$$

Figure 6. The worst-case latency bound using the network calculus method.

6. Worst-Case Latency Analysis of the FWOS Model

To analyze the worst-case latency performance for a TT frame with a given priority class, the arrival and service curves must be defined. The arrival curve depends on the limitations described in the previous section, and the service curve has to be determined based on the predefined GCL schedule. For the worst-case analysis, the service curve requires one to find the duration of the guaranteed service (contention-free) intervals for the corresponding queue and the maximum waiting time needed to serve that frame using these intervals. This section presents these required calculations considering the proposed scheduling algorithm, which includes all overlapping scenarios between TT open windows. First, we compute the length of the contention-free interval based on all pessimistic cases. Then, the maximum waiting time is determined for every selected node. After that, these calculations are used to define the service curve and, then, analytically implement the worst-case end-to-end latency.

6.1. Length of the Contention-Free Interval

Servicing the arrived traffic in each node is based on the open-window interval for the associated priority queue. Under flexible overlapping between the open windows for TT queues, each open window serves the arrived traffic according to more stringent scheduling constraints. In particular, these windows are divided into three different intervals: block, contention-based, and contention-free intervals. (i) Block intervals represent guard band intervals, where the remaining time is not enough to transmit the whole frame. (ii) Contention-based intervals represent time slots' aggregation, where the related window is overlapped with another TT window and the priority or the non-preemption technique may prevent the frame from being served. Although the targeted traffic may obtain the required service during contention-based intervals, safe latencies are formulated based only on contention-free intervals. (iii) Contention-free intervals represent the guaranteed service interval that has to be specified to determine the worst-case latency bounds. To define contention-free intervals, we have to specify all block and contention-based intervals. Block intervals are located at the end of each open window, but contention-based intervals depend on overlapping with other TT windows. Thus, the overlap must be comprehensively analyzed to specify these intervals.

As mentioned, three overlapping aspects have to be considered: priority, position, and the OR. Accordingly, the latency of TT traffic is evaluated and discussed below,

under all these aspects to give a clearer view of system performance with a variety of relevant parameters.

6.1.1. Opening-Edge Overlapping

As mentioned, we consider Q_k^h as a targeted TT queue for latency calculations in any node h . Then, to include the overlapping at the opening edge of the Q_k^h window, we assume that when its gate (G_k^h) changes from closed to open in the i -th gating cycle (at $t_k^{h,o,i}$), there is another TT gate open ($G_m^h(t_k^{h,o,i}) = 1$), as depicted in Figure 7. The length of the overlapping interval between Q_k^h and any other TT queue (Q_m^h) open windows can be found easily as

$$I_{k,m}^{h,B,i} = \begin{cases} t_m^{h,c,i} - t_k^{h,o,i} & , t_m^{h,o,i} < t_k^{h,o,i} < t_m^{h,c,i} \\ 0 & , \text{Otherwise} \end{cases} \quad (5)$$

Figure 7. Opening-edge overlapping with higher- and lower-priority windows.

The associated overlapping ratio is determined as

$$OR_{k,m}^{h,B,i} = \frac{L_{k,m}^{h,B,i}}{W_k^{h,i}} \quad (6)$$

This overlapping may occur from higher- or/and lower-priority queues; the related latency effects are discussed separately.

Higher-priority overlapping at the opening edge: The overlapping ratio between the Q_k^h window and the higher-priority $Q_{k^-}^h$ window ($k^- \in \{1, \dots, k-1\}$) at the opening edge can be determined by using Equation (6) as $OR_{k,k^-}^{h,B,i}$. Thus, the most decisive influence from the higher-priority overlapping at the opening edge, as shown in Figure 7, is included by considering only the queue that has the largest overlapping ratio with the Q_k^h window, as follows:

$$OR_{k,H}^{h,B,i} = \max_{1 \leq k^- < k} \{OR_{k,k^-}^{h,B,i}\} = \frac{\max\{L_{k,1}^{h,B,i}, \dots, L_{k,k-1}^{h,B,i}\}}{W_k^{h,i}} = \frac{L_{k,H}^{h,B,i}}{W_k^{h,i}} \quad (7)$$

In the worst case, servicing Q_k^h frames is not guaranteed in a time slot if there is any higher-priority window open at the same time slot. It contributes to reducing the contention-free interval, which starts at the closing time of the higher-priority window that has the largest overlapping. Thus, the starting time of the guaranteed service window can be given by

$$t_H^{h,B,i} = t_k^{h,o,i} + L_{k,H}^{h,B,i} = t_k^{h,o,i} + OR_{k,H}^{h,B,i} \cdot W_k^{h,i} \quad (8)$$

Lower-priority overlapping at the opening edge: Similarly, based on the window overlap, as shown in Figure 7, the largest overlapping ratio between Q_k^h and $Q_{k^+}^h$ windows ($k+1 \leq k^+ \leq N_q$) at the opening edge is defined as

$$OR_{k,L}^{h,B,i} = \frac{\max_{k+1 \leq k^+ \leq N_q} \{L_{k,k^+}^{h,B,i}\}}{W_k^{h,i}} = \frac{L_{k,L}^{h,B,i}}{W_k^{h,i}} \quad (9)$$

Although the Q_k^h frame has the priority to be transmitted before $Q_{k^+}^h$ frames, it has to wait until completing the transmission of $Q_{k^+}^h$ frames at the transmission status if the non-preemption technique is considered between TT transmissions. Accordingly, the Q_k^h frame cannot be transmitted until passing the maximum size of lower-priority frames ($f_{k^+}^{h,max}$). Thus, the i -th contention-free interval is reduced at the opening edge by the value ($d_{k^+}^{h,np,i}$), which can be given as

$$d_{k^+}^{h,np,i} = \begin{cases} \min\{f_{k^+}^{h,max}, L_{k,k^+}^{h,B,i}\} & , t_{k^+}^{h,o,i} < t_k^{h,o,i} < t_{k^+}^{h,c,i} \\ 0 & , \text{Otherwise} \end{cases} \quad (10)$$

Under the worst case, the maximum delay is considered as

$$d_L^{h,np,i} = \max_{k+1 \leq k^+ \leq N_q} \{d_{k^+}^{h,np,i}\} \quad (11)$$

By just considering the maximum influence of lower-priority overlapping on the i -th Q_k^h open window $W_k^{h,i}$, we can define the starting time of the i -th contention-free interval ($\bar{W}_k^{h,i}$) as

$$t_{k,L}^{h,B,i} = d_L^{h,np,i} + t_k^{h,o,i} \quad (12)$$

6.1.2. Closing-Edge Overlapping

Similar to the opening-edge overlapping case and as depicted in Figure 8, the length of the overlapping interval between Q_k^h and any other TT queue (Q_m^h), at the closing edge, can be found easily as

$$L_{k,m}^{h,E,i} = \begin{cases} t_k^{h,c,i} - t_m^{h,o,i} & , t_m^{h,o,i} < t_k^{h,c,i} < t_m^{h,c,i} \\ 0 & , \text{otherwise} \end{cases} \quad (13)$$

Figure 8. Closing-edge overlapping with higher- and lower-priority windows.

The relevant overlapping ratio between Q_m^h and Q_k^h windows is

$$OR_{k,m}^{h,E,i} = \frac{L_{k,m}^{h,E,i}}{W_k^{h,i}} \quad (14)$$

Higher-priority overlapping at the closing edge: The strongest influence from higher-priority overlapping ($OR_{k,k^-}^{h,E,i}$) ($1 \leq k^- < k$) at the closing edge, as shown in Figure 8, is included by considering the maximum overlapping ratio, as follows:

$$OR_{k,H}^{h,E,i} = \max_{1 \leq k^- < k} \{OR_{k,k^-}^{h,E,i}\} = \frac{\max_{1 \leq k^- < k} \{L_{k,k^-}^{h,E,i}\}}{W_k^{h,i}} = \frac{L_{k,H}^{h,E,i}}{W_k^{h,i}} \quad (15)$$

In the same manner as in the opening-edge consideration, as the $Q_{k^-}^h$ frame has the priority to be transmitted before the Q_k^h frame, during the overlapping interval, the service is not guaranteed for Q_k^h frames. Thus, the ending time of the i -th Q_k^h contention-free interval under the higher-priority overlapping at the closing edge can be given by

$$t_{k,H}^{h,E,i} = t_k^{h,c,i} - L_{k,H}^{h,E,i} = t_k^{h,c,i} - OR_{k,H}^{h,E,i} \cdot W_k^{h,i} \quad (16)$$

Lower-priority overlapping at the closing edge: Similarly, we can determine the largest OR between Q_k^h and $Q_{k^+}^h$ windows ($k < k^+ \leq N_q$) at the closing edge as

$$OR_{k,L}^{h,E,i} = \max_{k < k^+ \leq N_q} \{OR_{k,k^+}^{h,E,i}\} = \frac{L_{k,L}^{h,E,i}}{W_k^{h,i}} \quad (17)$$

As Q_k^h has a higher priority than k^+ queues, the overlapping at the closing edge does not affect the contention-free interval ($\bar{W}_k^{h,i}$). However, in a particular case of frame arrival, the lower-priority overlapping can affect the maximum waiting time of the associated frame, as we discuss next.

6.1.3. Block Intervals

Servicing the Q_k^h frame at the corresponding window cannot be guaranteed when the remaining time is insufficient to transmit the entire frame. Accordingly, at the end of each open window, there is a guard band interval ($d_k^{h,gb}$), named here as block interval. This interval equals the time required to transmit the largest size of the Q_k^h frame, $f_k^{h,max}$ [12], i.e.,

$$d_k^{h,gb} = \frac{f_k^{h,max}}{C_h} \quad (18)$$

Then, under the effect of this guard band interval, the contention-free window ends with

$$t_k^{h,gb,i} = t_k^{h,c,i} - d_k^{h,gb} \quad (19)$$

6.1.4. Overall Effects

Based on the overlapping considerations discussed above, the starting and ending times of the contention-free Q_k^h window are influenced by several parameters accordingly. By accumulating all these aspects together in worst-case computations, the starting and ending times are constrained by

$$t_k^{h,B,i} = \max \{ t_L^{h,B,i}, t_H^{h,B,i} \} \quad (20)$$

$$t_k^{h,E,i} = \min \{ t_k^{h,gb,i}, t_H^{h,E,i} \} \quad (21)$$

Then, the length of the Q_k^h contention-free window in the i -th cycle is

$$\bar{W}_k^{h,i} = \begin{cases} \max \left\{ t_k^{h,E,i} - t_k^{h,B,i}, \frac{f_k^{h,max}}{C_h} \right\} & , t_k^{h,B,i} < t_k^{h,E,i} \\ 0 & , t_k^{h,B,i} \geq t_k^{h,E,i} \end{cases} \quad (22)$$

By considering the i -th open window as a benchmark, as shown in Figure 9, the relative offset of the i -th contention-free window from the opening edge can be given by

$$R_k^{h,i} = \begin{cases} t_k^{h,B,i} - t_k^{h,o,i} & , \bar{W}_k^{h,i} \neq 0 \\ 0 & , \bar{W}_k^{h,i} = 0 \end{cases} \quad (23)$$

Figure 9. Relative Offsets of contention-free windows with the benchmark in the hyper-period.

In addition, the relative offset of the j -th contention-free window, considering the i -th window as a benchmark, is

$$R_k^{h,j,i} = (j - i) \cdot T_k^h - R_k^{h,i} + R_k^{h,j} \quad (24)$$

6.2. Maximum Waiting Time Consideration

The waiting time represents the time delay that the arrived frame experiences from the arrival instance to the first contention-free window's starting time. The maximum waiting time for the Q_k^h frame before getting the service considers the most pessimistic arrival instance. In general, the maximum waiting time until the chance to serve the frame is bounded by the earliest instance of the frame arrival in the selected node ($t_k^{h, arrive} \Big|_{earliest}$) and the starting time of the nearest contention-free interval for the associated queue ($t_k^{h, B}$), as follows:

$$WT_k^h = t_k^{h, B} - t_k^{h, arrive} \Big|_{earliest} \quad (25)$$

As discussed in [13], the earliest arrival instance depends on the selected node's order. The arrival instance in the first node (source) $t_k^{FN, arrive}$ is arbitrary and can be at any moment during the T_k^{FN} interval. On the other hand, in the non-first node (selected switch in the route), the frame arrival is bounded by the open-window interval for the corresponding queue in the previous node.

In the first node, the worst arrival instance happens when the first frame arrives at the end of the associated contention-free interval ($\bar{W}_k^{FN, i}$) and the remaining time is not enough to transmit it, as depicted in Figure 10. Thus, this frame waits for the next contention-free window to be transmitted. A more pessimistic arrival case happens when the Q_k^{FN} frame arrives at the end of the contention-free interval, and there is a lower-priority frame in the transmission status. Even if there is enough time to transmit the Q_k^{FN} frame, it has to wait for that lower-priority frame, as shown in Figure 10. The non-preemption technique prevents the Q_k^{FN} frame to interrupt that lower-priority frame. After the transmission of the Q_{k+}^{FN} frame finishes, the remainder of the time is not enough to transmit Q_k^{FN} frame. Accordingly, the overlapping between Q_k^{FN} and Q_{k+}^{FN} open windows maximizes the waiting time if the overlapping interval in the previous open window ($L_L^{FN, E, i-1}$) is larger than ($f_k^{FN, max} / C_{FN}$). The waiting time is maximized by the term $D_L^{FN, np, 0}$, which is limited by the length of the highest lower-priority overlapping interval in the previous open window ($L_L^{FN, E, i-1}$) and the time required to transmit the largest size of the lower-priority frame ($f_{k+}^{FN, max} / C_{FN}$), as follows:

$$D_L^{FN, np, 0} = \begin{cases} \max_{k < k^+ \leq N_q} \left\{ \min \left\{ \frac{f_{k^+}^{FN, max}}{C_{FN}}, L_L^{FN, E, i-1} \right\} \right\} & , t_m^{FN, o, i-1} < t_k^{FN, c, i-1} < t_m^{FN, c, i-1} \\ 0 & , otherwise \end{cases} \quad (26)$$

$$\text{where } L_L^{FN, E, i-1} = W_k^{FN, i-1} \cdot \max_{k < k^+ \leq N_q} \{ OR_{k, k^+}^{FN, E, i-1} \}.$$

Figure 10. The maximum waiting time at the first node.

Then, by taking the contention-free window $\bar{W}_k^{FN,i}$ as a benchmark, the maximum waiting time $WT_k^{FN,i}$ at the beginning of the backlog time interval can be given by

$$WT_k^{FN,i} = t_k^{FN,B,i} - \left(t_k^{FN,E,i-1} - D_L^{FN,np,0} \right) \quad (27)$$

For the non-first node h , the arrival instance is bounded by the starting time of the contention-free intervals in h and all preceding nodes connected to h input ports $((h-1)^1, \dots, (h-1)^{IN^h})$, as shown in Figure 11, where IN^h represents the number of input ports in h . In any case, the frame cannot arrive at h out of these boundaries. As depicted in Figure 11, the earliest frame arrives at $(t_k^{h,arrive}|_{earliest})$, which equals the starting time of the first Q_k^{h-1} contention-free window in the preceding node $(h-1)$ $t_k^{h-1,B,i}$ scheduled after the previous Q_k^h contention-free interval added to the selection and propagation delays. The frame selection delay is the time required to select the largest size of the Q_k^{h-1} frame to the physical link ($D_{k,select}^{h-1,max} = f_k^{h-1,max} / C_{h-1}$). The propagation delay ($D_{prop}^{h,h-1}$) is the time required for a Q_k^{h-1} frame to transfer from $h-1$ to h . Note that when the frame arrives at the node, a switch processing delay (D_{proc}^h) is experienced through the switch input buffer, switching fabric, and priority filtering at node h . This delay is needed even if the frame arrives through the contention-free interval in h . Thus, it is worth noting that the lowest bound of the maximum waiting time should be limited by the propagation and switch processing delays, as follows:

$$WT_k^{h,i} = \begin{cases} D_k^{h,h-1} & , D_k^{h,h-1} \geq D_{prop}^{h,h-1} + D_{proc}^h \\ D_{prop}^{h,h-1} + D_{proc}^h & , D_k^{h,h-1} < D_{prop}^{h,h-1} + D_{proc}^h \end{cases} \quad (28)$$

where $D_k^{h,h-1} = t_k^{h,B,i} - (t_k^{h-1,B,i} + D_{k,select}^{h-1,max})$.

Figure 11. The maximum waiting time at the non-first node in the selected path.

Note that the switch processing delay is counted even if the maximum waiting time is greater. Thus, this delay should be removed from the overall latency expression. Although the authors in [13] considered the frame arrival limitations in the non-first nodes, their latency calculation counted the switch processing delay twice, indirectly in the maximum waiting time calculations and overall latency expression. Even if this delay is a little small

and constant, it can result in more pessimistic latencies when the maximum waiting time is larger.

6.3. Service Curve Determination

As discussed before, the targeted traffic obtains the required service according to its contention-free intervals in each node. The length of the contention-free interval varies from an open window to another, depending on their related overlapping situations, as shown in Figure 10. However, their configurations are repeated after the hyper-period (T_{GCL}^h), which represents the least common multiple (LCM) of cycles for all TT queues, as follows [13]:

$$T_{GCL}^h = LCM_{1 \leq i \leq N_q} (T_i^h)$$

Therefore, the number of Q_k^h contention-free intervals in the hyper-period are given by $M = T_{GCL}^h / T_k^h$. By taking the i -th cycle as a reference, the upper and lower bounds of the service curve for the Q_k traffic can be given by [12]

$$\beta_k^{h,u,i}(t) = \sum_{j=i}^{i+M-1} \beta_k^{h,u,j,i}(t) \quad (29)$$

$$\beta_k^{h,l,i}(t) = \sum_{j=i}^{i+M-1} \beta_k^{h,l,j,i}(t) \quad (30)$$

where

$$\beta_k^{h,u,j,i} = \beta_{T_{GCL}^h, \bar{W}_k^{h,j}} \left(t + T_{GCL}^h - \bar{W}_k^{h,j} - R_k^{h,j,i} \right) \quad (31)$$

$$\beta_k^{h,l,j,i} = \beta_{T_{GCL}^h, \bar{W}_k^{h,j}} \left(t + T_{GCL}^h - \bar{W}_k^{h,j} - WT_k^{h,i} - R_k^{h,j,i} \right) \quad (32)$$

The terms $\bar{W}_k^{h,j}$ and $WT_k^{h,i}$ are determined using Equations (22), (27), and (28). Note that $\beta_{T,L}^h(t)$ is formulated for the TDMA protocol [12] as

$$\beta_{T,L}^h(t) = C_h \cdot \max \left\{ \left\lfloor \frac{t}{T} \right\rfloor L, t - \left\lceil \frac{t}{T} \right\rceil (T - L) \right\} \quad (33)$$

where C_h represents the data rate of the corresponding physical link. Depending on the arrival instance of the earliest frame, the service curve can be determined. As depicted in Figure 12, the best servicing (upper service curve) is obtained if the first frame arrives at the node's ingress port when the associated contention-free interval has instantaneously started. On the other hand, the frame faces the worst resource service (lower service curve) if it arrives at the end of the contention-free interval, i.e., there is no guaranteed service for the frame at that open window.

Figure 12. Upper and lower bounds of the service curve based on the waiting time limits.

However, the aggregated bounds of the service curve obtained in Equations (29) and (30) can be slightly different if we change the reference window in the calculation. Therefore, the upper and lower bounds of the service curve for Q_k^h traffic are the largest and lowest instantaneous values, respectively, from all possible service curves [12], i.e.,

$$\beta_k^{h,u}(t) = \max_{1 \leq i \leq M} \{ \beta_k^{h,u,i}(t) \} \quad (34)$$

$$\beta_k^{h,l}(t) = \min_{1 \leq i \leq M} \{ \beta_k^{h,l,i}(t) \} \quad (35)$$

An example of the upper and lower service curves is depicted in Figure 12.

6.4. Worst-Case End-to-End Latency for Q_k Traffic

The network calculus (NC) approach considers the upper bound of the arrival curve and the service curve's lower bound to compute the upper bound end-to-end latency. For every node h in the selected networking path, the upper arrival curve $\alpha_k^{h,u}(t)$ and the lower service curve $\beta_k^{h,l}(t)$ have to be defined to calculate the maximum experienced latency in each link. The upper arrival curve at the source node can be found using Equation (3). After that, in every node h from the selected path, the aggregate upper arrival curve at the ingress port for Q_k^h traffic can be determined using the ingress arrival curve at the previous node, as follows:

$$\alpha_k^{h,u}(t) = \alpha_k^{h-1,u}(t + WCD_k^{h-1}) \quad (36)$$

where WCD_k^{h-1} is the maximum latency experienced by Q_k^{h-1} traffic in $h-1$. This latency equals the largest horizontal distance between the upper arrival curve at the ingress port of node $(h-1)$ and the lower service curve of the related resource, as follows [13]:

$$WCD_k^{h-1} = H(\alpha_k^{h-1,u}(t), \beta_k^{h-1,l}(t)) \quad (37)$$

Then, we can calculate the upper bound of the end-to-end latency as the summation of the maximum latencies experienced in all nodes through the selected path, as

$$WCD_k^{total} = \sum_{h=1}^N WCD_k^h \quad (38)$$

where N is the number of nodes that the k -th-priority traffic passed through. According to all mathematical analysis presented in Section 6, the FWOS algorithm can be summarized as follows:

Algorithm 1. Flexible Window-Overlapping Scheduling (FWOS) Algorithm**START****Definition:** $N \stackrel{\text{def}}{=}$ Number of the selected nodes, and $h \in \{1, 2, \dots, N\}$, $Q_k \stackrel{\text{def}}{=}$ Targeted TT-queue for latency calculation with priority k , and $k \in \{1, 2, \dots, N_q^h\}$, $OS_k^{h,h-1} \stackrel{\text{def}}{=}$ Relative offsets between k th windows in the adjacent nodes, $e2e_k \stackrel{\text{def}}{=}$ End-to-end latency deadline for k th frames, $OR_H^o \stackrel{\text{def}}{=}$ Overlapping ratio with higher priority windows at the opening-edge, $OR_H^c \stackrel{\text{def}}{=}$ Overlapping ratio with higher priority windows at the closing-edge, $OR_L^o \stackrel{\text{def}}{=}$ Overlapping ratio with lower priority windows at the opening-edge, $OR_L^c \stackrel{\text{def}}{=}$ Overlapping ratio with lower priority windows at the closing-edge, $OR_{H,L}^{o,c} \stackrel{\text{def}}{=}$ Overlapping ratio with higher and lower priority windows, $OR_{L,H}^{o,c} \stackrel{\text{def}}{=}$ Overlapping ratio with lower and higher priority windows,**Input:** $N, OS_k^{h,h-1}, e2e_k, C_h, N_q^h, T_m^h, W_m^h, m \in \{1, 2, \dots, N_q^h\}$,**Let:** $OR \in \{OR_H^o, OR_H^c, OR_L^o, OR_L^c, OR_{H,L}^{o,c}, OR_{L,H}^{o,c}\}$,**For** $OR = 0 : 1$ **For** $h = 1 : N$ **Calculate:** $\bar{W}_k^h$ (the length of contention-free window), $R_k^{h,j,i}$ (The relative offset between contention-free windows), $WT_k^{h,i}$ (The maximum waiting time),**Calculate:** $\alpha_k^{h,u}(t)$ (The upper arrival curve), $\beta_k^{h,l}(t)$ (The lower service curve),**Calculate:** $WCD_k^{h-1} = H(\alpha_k^{h,u}(t), \beta_k^{h,l}(t))$ (The worst-case latency at node h),**End** h **Calculate:** $WCD_k^{total} = \sum_{h=1}^N WCD_k^h$ (The worst-case end-to-end latency)**End** OR **Output:** $WCD_k(OR_H^o), WCD_k(OR_H^c), WCD_k(OR_L^o), WCD_k(OR_L^c), WCD_k(OR_{H,L}^{o,c}), WCD_k(OR_{L,H}^{o,c})$ **Output:** Optimized OR , where $WCD(OR) \leq e2e_k$ **Design** GCL**END****7. Performance Evaluation****7.1. Case Study and Experimental Setup**

In this section, we evaluate our proposed model's performance on a TSN system based on a realistic vehicle use case, as shown in Figure 13a. A simplified representative network model for the vehicle is presented in Figure 13b with two switches and four end systems. The targeted end systems can be sensors, cameras, or actuators used in the connected vehicles to collect the relevant information from the surrounding environment.

Figure 13. Graphical model for the applied use case: (a) networking elements for a realistic vehicle use case and (b) representative networking model for the vehicle.

We assume that physical links connect these network elements with a 1 Gbps data rate in all experiments. As each networking node has to differentiate the traffic to multiple priority queues, the initial GCL for associated queues is listed in Table 2 with full isolation between TT open windows. Note that Q_4^h is considered a targeted queue to evaluate our proposed formulated model under a changeable overlapping ratio with other lower- and higher-priority windows. As a reference, we initiate the largest size of Q_4^h and Q_{4+}^h frames ($f_4^{h,max}, f_{4+}^{h,max}$) by 400 and 300 bytes, respectively. The period of all TT queues is 250 μ s, the duration of Q_4^h open windows is 20 μ s, and the initial offset between every two adjacent nodes is 20 μ s for the fourth-priority queues, as depicted in Table 2.

Table 2. The initial GCL implementation.

Priority	Open and Closed Edges for Window Intervals (μ s)		
	(ES ₁ , SW ₁)	(SW ₁ , SW ₂)	(SW ₂ , ES ₄)
1	(15, 35)	(20, 40)	Not assigned
2	Not assigned	(40, 60)	(45, 65)
3	(65, 85)	(70, 90)	(75, 95)
4	(105, 125)	(125, 145)	(145, 165)
5	(130, 150)	(155, 170)	(190, 210)
6	Not assigned	(175, 190)	Not assigned
7	(185, 200)	(200, 220)	Not assigned
8	(205, 225)	(225, 240)	(235, 250)

ES, end system; SW, switch.

As our model is an NC-based model, we use the Java API of the Real-Time Calculus (RTC) toolbox [49] to construct the GCL schedules and calculate the worst-case end-to-end latencies for the targeted TT queue. The RTC toolbox is an effective tool in real-time systems implemented based on min-plus and max-plus algebra operators for network calculus theory. The algorithm is run on a computer with an Intel Core i7-3770 CPU at 3.40 GHz and 12 GB of RAM.

7.2. Numerical Results and Discussion

Several experiments were applied to the FWOS model to examine overlapping scenarios with higher- and lower-priority queues. Based on the overlapping experiments, a complete view of our model performance is presented with all overlapping situations together in one chart. Different values of window offsets between nodes are used with varying network scales to compare our results with previous related works in the non-overlapping scenario.

7.2.1. Overlapping vs. WCD Evaluations

In this section, we evaluate the effect of one-sided overlapping of lower- and higher-priority windows with Q_4^h windows. Both opening-edge and closing-edge overlapping are considered here in four different cases, as illustrated in Table 3. For lower priority, the fifth-priority queue is assigned to overlap with the fourth-priority queue with a variable OR, as shown in Table 3, and the third-priority queue for higher priority.

Table 3. GCL implementation in one-sided overlapping scenarios with higher- and lower-priority windows.

Case	Link	Priority	Open–Window Interval ($t_m^{h,o}, t_m^{h,c}$) (μ s)
1 (Lower priority overlapping at the opening-edge)	(ES ₁ , SW ₁)	5	$(85 + W_4^1 \cdot OR_{4,5}^{1,B}, 105 + W_4^1 \cdot OR_{4,5}^{1,B})$
	(SW ₁ , SW ₂)	5	$(105 + W_4^2 \cdot OR_{4,5}^{2,B}, 125 + W_4^2 \cdot OR_{4,5}^{2,B})$
	(SW ₂ , ES ₄)	5	$(125 + W_4^3 \cdot OR_{4,5}^{3,B}, 145 + W_4^3 \cdot OR_{4,5}^{3,B})$
2 (Lower priority overlapping at the closing-edge)	(ES ₁ , SW ₁)	5	$(125 - W_4^1 \cdot OR_{4,5}^{1,E}, 145 - W_4^1 \cdot OR_{4,5}^{1,E})$
	(SW ₁ , SW ₂)	5	$(145 - W_4^2 \cdot OR_{4,5}^{2,E}, 165 - W_4^2 \cdot OR_{4,5}^{2,E})$
	(SW ₂ , ES ₄)	5	$(165 - W_4^3 \cdot OR_{4,5}^{3,E}, 185 - W_4^3 \cdot OR_{4,5}^{3,E})$
3 (Higher priority overlapping at the opening-edge)	(ES ₁ , SW ₁)	3	$(85 + W_4^1 \cdot OR_{4,3}^{1,B}, 105 + W_4^1 \cdot OR_{4,3}^{1,B})$
	(SW ₁ , SW ₂)	3	$(105 + W_4^2 \cdot OR_{4,3}^{2,B}, 125 + W_4^2 \cdot OR_{4,3}^{2,B})$
	(SW ₂ , ES ₄)	3	$(125 + W_4^3 \cdot OR_{4,3}^{3,B}, 145 + W_4^3 \cdot OR_{4,3}^{3,B})$
4 (Higher priority overlapping at the closing-edge)	(ES ₁ , SW ₁)	3	$(125 - W_4^1 \cdot OR_{4,3}^{1,E}, 145 - W_4^1 \cdot OR_{4,3}^{1,E})$
	(SW ₁ , SW ₂)	3	$(145 - W_4^2 \cdot OR_{4,3}^{2,E}, 165 - W_4^2 \cdot OR_{4,3}^{2,E})$
	(SW ₂ , ES ₄)	3	$(165 - W_4^3 \cdot OR_{4,3}^{3,E}, 185 - W_4^3 \cdot OR_{4,3}^{3,E})$

Cases 1 and 2: In these cases, the lower-priority overlapping is evaluated by varying the OR from 0 to 1, with different sizes of Q_{4+}^h frames. Figure 14a,b shows the performance of the worst-case latency for Q_4^h traffic under opening-edge and closing-edge overlapping. As expected, the lowest WCD is obtained under full isolation between Q_4^h and Q_5^h windows ($OR_{4,5}^h = 0$).

Figure 14. The worst-case end-to-end delay (WCD) vs. lower-priority overlapping with different sizes of the lower-priority frame (f_{k+}^{max}): (a) opening-edge overlapping case and (b) closing-edge overlapping case.

In Figure 14a, the WCD increases with an increase in the OR at the opening side because the Q_4^h frame has to wait for the lower-priority frame, which is in the forwarding status. The increase continues until the time required to transmit the largest lower-priority frame is completed, which is assumed to be 200, 300, and 500 bytes, separately. These durations are equivalent to 8%, 12%, and 20% overlapping ratios. If the OR increases more than the largest frame size, the latency is fixed until the guard band interval, equal to 16%

OR at the end of the open-window interval, using Equations (1) and (18). In the guard band interval, large lower-priority frame sizes increase the latency with a percentage greater than smaller sizes, as shown in Figure 14a, because the maximum waiting time increases proportionally, as can be noticed from Equations (26) and (27).

For closing-edge overlapping, as shown in Figure 14b, the effect of lower-priority overlapping appears after passing the guard band interval, which equals 16% OR with ($f_4^{h,max} = 400$ bytes). After the guard band interval, the latency increases until passing the lower-priority frame's length, and then it remains constant, as depicted in Figure 14b.

Cases 3 and 4: In these cases, the influence of the higher-priority overlapping at the opening edge and closing edge is evaluated in Figure 15a,b, respectively, with sizes of Q_4^h frames as 350, 400, and 500 bytes.

Figure 15. WCD vs. higher-priority overlapping with different sizes of the fourth-priority frame (f_4^{max}): (a) opening-edge overlapping and (b) closing-edge overlapping.

Comparing the results in Figure 14a,b with those in Figure 15a,b shows that higher-priority overlapping has a higher impact on latency performance than lower-priority overlapping. In addition, Figure 15a,b proves that the Q_4^h frame experiences higher WCD bounds under opening-edge overlapping than under closing-edge overlapping. This happens because closing-edge overlapping does not affect the guard band interval, where the service is not guaranteed (block interval). As proof for this, it can be observed that both overlapping situations, i.e., opening and closing, have the same behavior with a horizontal shift equal to the guard band duration. For example, with a 500-byte frame size, the latency starts to increase at almost 40% overlapping in the closing-edge case and 20% overlapping (40–20% (guard band duration)) in the opening-edge case. Moreover, the size of the Q_4^h frame plays a significant role in the overall WCD performance, especially at large ORs. For instance, the Q_4^h frame with sizes of 350, 400, and 500 bytes experiences 289.3, 465.5, and 782 μ s WCDs, respectively, under 40% OR at the opening edge.

7.2.2. Implementing Overlapping Constraints Based on Critical Time Deadlines

Based on the apparent fluctuation in TT latency bounds under window overlapping situations, we can add a new constraint for the overall TSN scheduling limitations, which was formulated in [42]. As the critical time traffic requires different WCD deadlines, each TT frame should be forwarded to a priority queue, where it should experience an end-to-end latency no higher than its deadline. We can differentiate between TT priority queues according to different overlapping ratios between corresponding windows. Moreover, the overlapped queue's priority is a significant issue to determine which maximum allowable overlapping ratio guarantees the targeted latency deadline. Based on the results presented in Figure 16, each WCD bound can be met if the related TT overlapping does not exceed a

specific limit. These OR limits are differentiated in Figure 16 concerning the priority of the overlapped queues and the overlapping position.

Figure 16. Maximum allowable overlapping ratio (OR) vs. guaranteed WCD deadlines under one-sided and two-sided overlapping conditions by lower-, higher-, and mixed-priority queues.

It can be noticed that lower-priority overlapping causes a small effect on WCD boundaries, with a trivial difference between the opening, closing, and two-sided overlapping cases, as shown in Figure 16. In particular, slightly after the WCD bound in the entire isolation case (282.7 μ s), the latency deadline can be met under any lower-priority overlapping. At most, the deadline of 287.7 μ s can be met under any lower-priority overlapping. However, large ORs are not acceptable in GCL design as it is considered higher overlapping on the overlapped queue.

In contrast, higher-priority overlapping has the highest impact on the latency, and stricter overlapping constraints should be guaranteed. Opening-edge overlapping has a larger influence than closing-edge overlapping. For example, to meet the 400 μ s latency deadline, the highest allowable OR at the opening and closing edges is 38.62% and 54.62%, respectively. Under two-sided higher overlapping, the performance is almost the same as under closing-edge higher overlapping, and the latency deadline of 400 μ s is met until 54.31% overlapping.

For mixed overlapping cases, the OR limits are between pure lower and pure higher overlapping. For instance, the latency deadline of 400 μ s is guaranteed until 76.84% OR with higher priority at the opening edge and lower priority at the closing edge, and until 85.2% with the opposite.

As noted, the lowest WCD bound that can be achieved is 282.7 μ s. This limit is based on specific framing and timing settings. Adjusting these values increases/decreases the WCD bounds accordingly, as examined in the previous results. Thus, it is worth noting that Figure 16 can be considered a reference chart for window overlapping vs. latency deadlines based on the above networking and data settings. For each specific required setting, we can easily implement related overlapping vs. the latency reference chart.

Accordingly, we can implement, in the general case, a relaxed window overlapping constraint that will guarantee meeting latency requirements for TT traffic and can be used to improve the overall solution space, as follows:

Constraint definition: By considering Q_k^h as a targeted TT queue, where $k \in \{1, 2, \dots, N_q\}$ and h is a node in the selected path ($h \in \{1, 2, \dots, N\}$), and the lower-priority queue as k^+ , it is assumed that the largest size of Q_k^h and $Q_{k^+}^h$ frames in h is given as $f_k^{h,max}$ and $f_{k^+}^{h,max}$,

respectively; the period of every TT queue (Q_m^h) in h as T_m^h ; the duration of open-window intervals for Q_m^h in h as W_m^h ; data out of h by a rate C_h ; offsets between any adjacent nodes as $OS_k^{h,h-1}$; and end-to-end latency deadline for the Q_k^h frame as $e2e_k$. Then, the window overlapping constraint can be given as

$$\forall WCD \left(OR_{k,P(m)}^{h,edge} \Big|_{max} \right) \leq e2e_k :$$

$$\forall WCD \left(OR_{k,P(m),P(n)}^{h,edge1,edge2} \Big|_{max} \right) \leq e2e_k :$$

$$\left(OR_{k,m}^{h,edge} \leq OR_{k,P(m)}^{h,edge} \Big|_{max} \right) \wedge \left(OR_{k,m,n}^{h,edge1,edge2} \leq OR_{k,P(m),P(n)}^{h,edge1,edge2} \Big|_{max} \right)$$

where m and n represent the priorities of queues that overlap with the Q_k^h window in every node h , $P(m), P(n) \in \{L, H\}$ (L lower priority and H higher priority), and $edge1, edge2 \in \{B, E\}$ (B opening edge and E closing edge).

7.2.3. Comparison with Related Works

As we introduce some modifications to the maximum waiting time expression, as presented in [13], the related comparison has to be considered here. The comparison is performed under complete isolation among windows, as shown in Table 1. In [13], the presented algorithm dramatically reduced the worst-case latency bounds compared with the results in [11]. Hence, our proposed model (FWOS) is compared with that in [13] (STNet) and [11] (STNode) based on two different scenarios, as shown in Figure 17a,b.

Figure 17. Comparison between the proposed flexible window-overlapping scheduling (FWOS) algorithm and related works: (a) FWOS, STNode ([11]), and STNet ([13]) under different relationships between offsets for same-priority windows among three nodes. (b) FWOS and STNet ([13]) with constant relationships between offsets in connections with 10, 20, and 30 hops.

In Figure 17a, the comparison is performed under different window offsets between adjacent nodes ($OS_4^{h,h-1}$). FWOS gives the lowest WCDs compared with others. For a three-hop connection, FWOS reduces the WCD by 55% on average and 60.6% maximum compared with STNode and 3.03% on average and 3.4% maximum compared with STNet. The WCD enhancement of the FWOS algorithm increases under large use cases, as illustrated in Figure 17b. Compared with STNet, FWOS reduces the WCD by 9%, 11.5%, and 12.3% for connection with 10, 20, and 30 hops, respectively. The interpretation of that is our latency improvement depends linearly on the number of selected nodes, as can be noticed from Equations (28) and (38). Our model reduces end-to-end latency bound by a value equal to the propagation and technical switch delays in each link. Note that the

STNode model is excluded in Figure 17b as it produced non-comparable latencies. The STNode model is implemented based on separate latency computations among nodes without considering the relative positional offsets between adjacent nodes. Undoubtedly, it has colossal WCDs compared with others.

8. Conclusions

Several mixed-criticality domains, such as automotive and automated industries, require deterministic latency performances. Missing latency deadlines in such applications may generate dangerous situations for humans or considerable economic waste in highly time-sensitive applications. To support such applications, TSN technology is defined based on several TT-Ethernet amendments. Most of these enhancements have been introduced mainly to serve time-triggered traffic targeting zero jitter delay and bounded end-to-end latency. However, there is a concern that soft real-time traffic could miss their requirements under strict timing constraints in GCL designs.

This paper proposed a flexible window-overlapping scheduling (FWOS) algorithm with complete timing analysis for the WCD using the network calculus approach considering relative window offsets between adjacent nodes. Our model expresses all overlapping situations between TT windows, leading to a more generic timing analysis. Based on a realistic vehicular use case, the numeric results confirm that the impact of higher-priority overlapping on the WCD is larger than that from lower-priority overlapping. Further, overlapping at the opening edge generates bigger WCDs than at the closing edge. Thus, under guaranteed latency deadlines, the overlapping ratio between TT windows must be bounded accordingly. Based on this, the FWOS model can be applied to optimize the maximum-allowable overlapping ratio (OR) that ensures TT latency requirements and maximizes the bandwidth available for unscheduled transmissions. For example, to meet the 400 μ s latency deadline under specific timing assumptions, the highest-allowable OR is 38.62% with higher priority (opening edge), 54.62% with higher priority (closing edge), 54.31% with higher priority (opening and closing edges), 76.84% with higher priority (opening edge) and lower priority (closing edge), 85.2% with lower priority (opening edge) and higher priority (closing edge), and 100% with any case of lower priority. Accordingly, more relaxed GCLs can be implemented to fit the requirements of incoming data from vehicle-related devices, such as sensors, cameras, and actuators. Compared with the previous works, FWOS obtains less pessimistic WCDs, even under fully isolated window scheduling.

Author Contributions: Conceptualization, K.M.S., N.K.N., A.S. and F.H.; methodology, K.M.S., N.K.N. and A.S.; software, K.M.S. and A.S.; validation, K.M.S., N.K.N. and F.H.; formal analysis, K.M.S. and A.S.; investigation, K.M.S. and A.S.; resources, N.K.N.; data curation, K.M.S.; writing—original draft preparation, K.M.S.; writing—review and editing, K.M.S., N.K.N., A.S. and F.H.; visualization, K.M.S. and A.S.; supervision, N.K.N., A.S. and F.H.; project administration, N.K.N.; funding acquisition, N.K.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by UPM PUTRA BERIMPAK under fundamental research grant (9584300).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pang, Z.; Huang, X.; Li, Z.; Zhang, S.; Xu, Y.; Wan, H.; Zhao, X. Flow Scheduling for Conflict-Free Network Updates in Time-Sensitive Software-Defined Networks. *IEEE Trans. Ind. Inform.* **2021**, *17*, 1668–1678. [\[CrossRef\]](#)
2. Finn, N. Introduction to Time-Sensitive Networking. *IEEE Commun. Stand. Mag.* **2018**, *2*, 22–28. [\[CrossRef\]](#)

3. IEEE Standard Association. *IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 25: Enhancements for Scheduled Traffic*; IEEE Standard 802.1Qbv-2015, 10016-5997; IEEE Standard Association: New York, NY, USA, 2016; pp. 1–57.
4. IEEE Standard Association. *IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks Amendment 12 Forwarding and Queuing Enhancements for Time-Sensitive Streams*; IEEE Standard 802.1Qav-2009, 10016-5997; IEEE Standard Association: New York, NY, USA, 2009; pp. 1–72.
5. IEEE Standard Association. *IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 26: Frame Preemption*; IEEE Standard 802.1Qbu-2016, 10016-5997; IEEE Standard Association: New York, NY, USA, 2016; pp. 1–52.
6. Zhang, J.; Xu, Q.; Lu, X.; Zhang, Y.; Chen, C. Coordinated Data Transmission in Time-Sensitive Networking for Mixed Time-Sensitive Applications. In Proceedings of the IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 18–21 October 2020; pp. 3805–3810.
7. Ren, J.; Yang, D.; Cui, E.; Gong, K. An Analytical Latency Model for AVB Traffic in TSN Considering Time-Triggered Traffic. In Proceedings of the International Conference on Communication Technology Proceedings, ICCT, Tianjin, China, 13–16 October 2021.
8. Nasrallah, A.; Thyagaturu, A.S.; Alharbi, Z.; Wang, C.; Shao, X.; Reisslein, M.; El Bakoury, H. Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research. *IEEE Commun. Surv. Tutorials* **2019**, *21*, 88–145. [[CrossRef](#)]
9. Gavrilut, V.; Zhao, L.; Raagaard, M.L.; Pop, P. AVB-Aware Routing and Scheduling of Time-Triggered Traffic for TSN. *IEEE Access* **2018**, *6*, 75229–75243. [[CrossRef](#)]
10. Reusch, N.; Zhao, L.; Craciunas, S.S.; Pop, P. Window-Based Schedule Synthesis for Industrial IEEE 802.1Qbv TSN Networks. In Proceedings of the 16th IEEE International Conference on Factory Communication Systems (WFCS), Porto, Portugal, 27–29 April 2020.
11. Zhao, L.; Pop, P.; Craciunas, S.S. Worst-Case Latency Analysis for IEEE 802.1Qbv Time Sensitive Networks Using Network Calculus. *IEEE Access* **2018**, *6*, 41803–41815. [[CrossRef](#)]
12. Zhang, P.; Liu, Y.; Shi, J.; Huang, Y.; Zhao, Y. A Feasibility Analysis Framework of Time-Sensitive Networking Using Real-Time Calculus. *IEEE Access* **2019**, *7*, 90069–90081. [[CrossRef](#)]
13. Zhao, L.; Pop, P.; Gong, Z.; Fang, B. Improving Latency Analysis for Flexible Window-Based GCL Scheduling in TSN Networks by Integration of Consecutive Nodes Offsets. *IEEE Internet Things J.* **2020**, *20*, 1. [[CrossRef](#)]
14. Finzi, A.; Mifdaoui, A. Worst-Case Timing Analysis of AFDX Networks With Multiple TSN/BLS Shapers. *IEEE Access* **2020**, *8*, 106765–106784. [[CrossRef](#)]
15. Finzi, A.; Mifdaoui, A.; Frances, F.; Lochin, E. Incorporating TSN/BLS in AFDX for mixed-criticality applications: Model and timing analysis. In Proceedings of the 14th IEEE International Workshop on Factory Communication Systems (WFCS), Imperia, Italy, 28 February 2018.
16. Thiele, D.; Ernst, R.; Diemer, J. Formal worst-case timing analysis of Ethernet TSN's time-aware and peristaltic shapers. In Proceedings of the IEEE Vehicular Networking Conference (VNC), Kyoto, Japan, 16–18 December 2015; pp. 251–258.
17. Maxim, D.; Song, Y.-Q. Delay analysis of AVB traffic in time-sensitive networks (TSN). In Proceedings of the RTNS 2017—International Conference on Real-Time Networks and Systems, Grenoble, France, 4–6 October 2017.
18. Specht, J.; Samii, S. Synthesis of Queue and Priority Assignment for Asynchronous Traffic Shaping in Switched Ethernet. In Proceedings of the 2017 IEEE Real-Time Systems Symposium (RTSS), Paris, France, 5–8 December 2017.
19. Specht, J.; Samii, S. Urgency-Based Scheduler for Time-Sensitive Switched Ethernet Networks. In Proceedings of the 2016 28th Euromicro Conference on Real-Time Systems (ECRTS), Toulouse, France, 5–8 July 2016.
20. Zhou, Z.; Yan, Y.; Berger, M.; Ruepp, S. Analysis and modeling of asynchronous traffic shaping in time sensitive networks. In Proceedings of the IEEE International Workshop on Factory Communication Systems-Proceedings, WFCS, Imperia, Italy, 13–15 June 2018.
21. Zhou, Z.; Berger, M.S.; Ruepp, S.R.; Yan, Y. Insight into the IEEE 802.1 Qcr Asynchronous Traffic Shaping in Time Sensitive Network. *Adv. Sci. Technol. Eng. Syst. J.* **2019**, *4*, 292–301. [[CrossRef](#)]
22. Fang, B.; Li, Q.; Gong, Z.; Xiong, H. Simulative Assessments of Credit-Based Shaping and Asynchronous Traffic Shaping in Time-Sensitive Networking. In Proceedings of the 12th International Conference on Advanced Infocomm Technology (ICAIT), Macau, China, 23–25 November 2020.
23. Nasrallah, A.; Thyagaturu, A.S.; Alharbi, Z.; Wang, C.; Shao, X.; Reisslein, M.; Elbakoury, H. Performance Comparison of IEEE 802.1 TSN Time Aware Shaper (TAS) and Asynchronous Traffic Shaper (ATS). *IEEE Access* **2019**, *7*, 44165–44181. [[CrossRef](#)]
24. Jiang, J. A Time-sensitive Networking (TSN) Simulation Model Based on OMNET++. In Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, 5–8 August 2018.
25. Alderisi, G.; Patti, G.; Lo Bello, L. Introducing support for scheduled traffic over IEEE audio video bridging networks. In Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 17–21 September 2012.
26. Thiele, D.; Ernst, R. Formal Worst-Case Timing Analysis of Ethernet TSN's Burst-Limiting Shaper. In Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 14–18 March 2016.

27. Kim, M.; Min, J.; Hyeon, D.; Paek, J. TAS Scheduling for Real-Time Forwarding of Emergency Event Traffic in TSN. In Proceedings of the International Conference on ICT Convergence, Jeju Island, Korea, 21–23 October 2020.
28. Oliver, R.S.; Ag, T.C. Analysis of Deterministic Ethernet Scheduling for the Industrial Internet of Things. In Proceedings of the 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Athens, Greece, 1–3 December 2014.
29. Hellmanns, D.; Glavackij, A.; Falk, J.; Hummen, R.; Kehrer, S.; Durr, F. Scaling TSN Scheduling for Factory Automation Networks. In Proceedings of the 2020 16th IEEE International Conference on Factory Communication Systems (WFCS), Porto, Portugal, 27–29 April 2020.
30. Farzaneh, M.H.; Kugele, S.; Knoll, A. A Graphical Modeling Tool Supporting Automated Schedule Synthesis for Time-Sensitive Networking. In Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Torino, Italy, 4–7 September 2018.
31. Yu, Q.; Wan, H.; Zhao, X.; Gao, Y.; Gu, M. Online Scheduling for Dynamic VM Migration in Multicast Time-Sensitive Networks. *IEEE Trans. Ind. Inform.* **2019**, *16*, 3778–3788. [\[CrossRef\]](#)
32. Yu, Q.; Gu, M. Adaptive Group Routing and Scheduling in Multicast Time-Sensitive Networks. *IEEE Access* **2020**, *8*, 37855–37865. [\[CrossRef\]](#)
33. Xu, L.; Xu, Q.; Zhang, Y.; Zhang, J.; Chen, C. Co-design Approach of Scheduling and Routing in Time Sensitive Networking. In Proceedings of the 3rd IEEE International Conference on Industrial Cyber-Physical Systems (ICPS), Tampere, Finland, 10–12 June 2020.
34. Atallah, A.A.; Hamad, G.B.; Mohamed, O.A. Routing and Scheduling of Time-Triggered Traffic in Time-Sensitive Networks. *IEEE Trans. Ind. Inform.* **2019**, *16*, 4525–4534. [\[CrossRef\]](#)
35. Nayak, N.G.; Dürr, F.; Rothermel, K. Time-sensitive Software-defined Network (TSSDN) for Real-time Applications. In Proceedings of the 24th International Conference on Real-Time Networks and Systems, Brest, France, 19–21 October 2016.
36. Nayak, N.G.; Durr, F.; Rothermel, K. Incremental Flow Scheduling and Routing in Time-Sensitive Software-Defined Networks. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2066–2075. [\[CrossRef\]](#)
37. Thiele, D.; Ernst, R. Formal worst-case performance analysis of time-sensitive Ethernet with frame preemption. In Proceedings of the IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), Berlin, Germany, 6–9 September 2016.
38. Gavriluț, V.; Pop, P. Scheduling in time sensitive networks (TSN) for mixed-criticality industrial applications. In Proceedings of the 14th IEEE International Workshop on Factory Communication Systems (WFCS), Imperia, Italy, 13–15 June 2018; pp. 1–4.
39. Diaz, N.; Gautam, N. On Latency For Non-Scheduled Traffic in TSN. In Proceedings of the GLOBECOM 2020-2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020.
40. Zhao, L.; Pop, P.; Zheng, Z.; Li, Q. Timing Analysis of AVB Traffic in TSN Networks Using Network Calculus. In Proceedings of the 2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), Porto, Portugal, 11–13 April 2018.
41. Craciunas, S.S.; Oliver, R.S.; Chmelik, M.; Steiner, W. Scheduling Real-Time Communication in IEEE 802.1Qbv Time Sensitive Networks. In Proceedings of the 24th International Conference on World Wide Web, Association for Computing Machinery (ACM), New York, NY, USA, 18–22 May 2015.
42. Craciunas, S.S.; Oliver, R.S. An Overview of Scheduling Mechanisms for Time-sensitive Networks. In Proceedings of the Real-Time Summer School L'École d'Été Temps Réel (ETR), Nantes, France, 14 June–2 July 2021.
43. Oliver, R.S.; Craciunas, S.S.; Steiner, W. IEEE 802.1Qbv Gate Control List Synthesis Using Array Theory Encoding. In Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), Boston, MA, USA, 11–13 April 2018.
44. Hellmanns, D.; Falk, J.; Glavackij, A.; Hummen, R.; Kehrer, S.; Durr, F. On the Performance of Stream-based, Class-based Time-aware Shaping and Frame Preemption in TSN. In Proceedings of the 2020 IEEE International Conference on Industrial Technology (ICIT), Buenos Aires, Argentina, 26–28 February 2020.
45. Le Boudec, P.T. *Network Calculus a Theory of Deterministic Queuing Systems for the Internet*; Springer Science & Business Media: Berlin, Germany, 2001.
46. Maile, L.; Hielscher, K.S.; German, R. Network Calculus Results for TSN: An Introduction. In Proceedings of the 2020 Information Communication Technologies Conference, Nanjing, China, 29–31 May 2020.
47. Finzi, A.; Mifdaoui, A.; Frances, F.; Lochin, E. Network Calculus-based Timing Analysis of AFDX networks with Strict Priority and TSN/BLS Shapers. In Proceedings of the 13th IEEE International Symposium on Industrial Embedded Systems, La Grande Motte, France, 11–13 April 2018.
48. Wandeler, E.; Thiele, L.; Verhoef, M.; Lieverse, P. System architecture evaluation using modular performance analysis: A case study. *Int. J. Softw. Tools Technol. Transf.* **2006**, *8*, 649–667. [\[CrossRef\]](#)
49. Wanderler, E.; Thiele, L. Real-Time Calculus (RTC) Toolbox. 2006. Available online: <http://www.mpa.ethz.ch/Rtctoolbox> (accessed on 25 January 2021).