Received 16 September 2022; accepted 1 October 2022. Date of publication 10 October 2022; date of current version 24 October 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3212939

Delay-Guaranteeing Admission Control for Time-Sensitive Networking Using the Credit-Based Shaper

LISA MAILE (Graduate Student Member, IEEE), KAI-STEFFEN J. HIELSCHER, AND REINHARD GERMAN

Computer Networks and Communication Systems, Friedrich-Alexander-Universität Erlangen-Nürnberg, 91058 Erlangen, Germany

CORRESPONDING AUTHOR: L. MAILE (e-mail: lisa.maile@fau.de)

ABSTRACT With real-time communication being a key part of the fourth industrial revolution, the need for Quality of Service (QoS) in industrial networks is gaining increasing importance. Time-Sensitive Networking (TSN) faces this need, for example, by introducing new scheduling mechanisms. The Credit-Based Shaper (CBS) has been introduced to TSN to offer low delays for multiple traffic classes by applying rate limitations. Currently, flows are reserved decentrally in CBS networks using a Stream Reservation Protocol (SRP). In contrast, the new TSN standard IEEE 802.1Qcc allows for a centralized architecture to favor short reconfiguration latencies. However, no online admission control scheme which offers safe delay bounds has been proposed for this central architecture. To close this gap, we propose two models for admission control in TSN networks using CBS. Both models offer deadline-guaranteeing flow allocation, including routing and prioritization of flows, and configure forwarding devices while eliminating packet loss. Our models utilize the mathematical framework of Network Calculus to calculate worst-case flow delays and buffer sizes. We show how our models allow for more reservations than the decentralized standard approach by improved resource utilization. We validate our models both in synthetic and industrial network scenarios. Additionally, we compare the effects and parameters of our two models, providing guidance on when to choose them.

INDEX TERMS Admission control, computer network management, credit-based shaper, network calculus, quality of service, routing, time-sensitive networking.

I. INTRODUCTION

TIME-SENSITIVE Networking (TSN) has been developed to provide Quality of Service (QoS) over Ethernet networks, allowing for higher data rates, lower costs, and easier integration with existing IT systems than traditional fieldbuses [1]. It is used for various industrial communications [2] and is part of the 3GPP standard since Release 16 [3] to serve as wired transmission medium for 5G communication [4]. The goal is to guarantee strict QoS requirements such as maximum end-to-end delay and the elimination of packet loss for reliable communication. For this, the realization of a deadline-aware admission control scheme is a highly interesting field of research, with great benefits for practical applications.

A main advantage of TSN is the possibility to include mixed traffic types on the same medium. Traffic criticality can be classified as hard real-time (HRT), soft real-time (SRT), and non-real-time (NRT) traffic [5], [6]. To reserve HRT traffic in TSN, previous works propose the use of time-triggered (TT) gates which allow for explicit time-slot reservations of flows using a globally synchronized clock [7]. While being the only solution for ultra-low latency and zero-jitter requirements, TT gates introduce a significant waste of bandwidth and static configurations [8], [9]. Thus, it should be considered only for dedicated scenarios with highly jitter-sensitive and static networks.

As a consequence, other schedulers should be considered for HRT and SRT traffic when the latency and jitter requirements are less tight. The most discussed are called Strict Priority (SP) [10], Credit-Based Shaper (CBS) [11], Deficit Round Robin (DRR) [12], and Asynchronous Traffic Shaper (ATS) [13]. All are able to utilize the complete available

bandwidth and do not rely on globally synchronized clocks. In contrast to SP and DRR, CBS introduces a maximum bandwidth per class, which reduces buffer sizes and packet loss, and allows for better performance of low-priority traffic [14], [15]. When comparing CBS and ATS, results - e.g., by Zhao et al. [16] - show the tendency of CBS to outperform ATS in terms of delay, jitter, and backlog for low and medium network utilization.

However, for CBS, scheduler configuration and flow allocation with deadline guarantees has rarely been covered in research so far. Up to now, CBS networks are either used with a "rule-of-thumb" configuration - without keeping deadline guarantees at all - or flows are allocated decentrally using a Stream Reservation Protocol (SRP) [17]. For the decentral reservation, worst-case latencies are calculated at every bridge, and the information is forwarded over a flow's path. This requires strict sending behavior and the standard only provides the worst-case delay for a single priority up to now. With limited information available at every hop and the strict requirements, this decentral configuration is overly pessimistic [18].

In contrast, TSN now offers an architecture to centrally configure the network instead, by introducing the Central Network Controller (CNC) [19]. Thereby, IEEE 802.1Qcc standard defines the communication interface and network architecture, but it does not include a central reservation algorithm yet. To the best of our knowledge, no central admission control scheme for CBS has been proposed so far. Our solution allows the successive allocation, also called online configuration, so flows can be registered and de-registered from the network while the network is operating using the CNC. Thereby, for each flow, we assign routes and priorities for an arbitrary number of CBS queues and both, periodic and aperiodic, traffic. With this, we can allocate all three different traffic types - HRT, SRT, and NRT - in one network. The proposed solution offers flexibility for flow requirements, sending interval, rate, and deadline, which can be chosen freely. Additionally, we offer flexibility for the network, as reservations can be changed or updated without violating existing reservations. This flexibility faces the need for continuous network reconfiguration, which has emerged for the Industry 4.0 and, e.g., mobile devices in the context of 5G.

We propose two mathematical models for deadline-guaranteeing admission control, which rely on Network Calculus (NC). Our first model allows for a simple configuration of networking devices by fixing the IdleSlope values for each queue. In the context of CBS, IdleSlope denotes the reserved bandwidth for CBS queues. We call this model *Fixed Slope Model*. The second model is called *Maximum Delay Model*. It dynamically calculates the required IdleSlope depending on the characteristics of the reserved flows. Thus, this model offers tight IdleSlopes which reduce the burstiness of traffic and the delay for lower priority queues in the network.

The outline of the remaining paper is as follows. We discuss related work in Section II and explain the used notation, all variables, and the network representation in Section III. Section IV introduces the principle of CBS scheduling, the decentral admission control, as well as the mathematical framework NC which we will use for the central admission control. Section V first starts with an overview of the general procedure of our two models for routing, device configuration, and flow allocation in CBS networks. Our *Fixed Slope Model* is explained in Section V-B, while the *Maximum Delay Model* follows in Section V-C. We compare our two models in Section VI and illustrate their dependencies on input parameters in a theoretical network. Besides, we conducted extensive simulations in industrial networks to show the scalability and applicability of our models and compared our results to the decentral admission control scheme of the standard. Finally, Section VII concludes this paper.

II. RELATED WORK

The configuration of real-time communications has received a lot of attention in the last few years. Several researchers have investigated differences between decentral and central configuration in TSN. Migge et al. [18] demonstrate various drawbacks of the existing decentral configuration and illustrate how it does not allow schedulability in realistic industrial networks. At the same time, they also show that CBS would be able to schedule all required flows with an alternative configuration. However, they do not discuss how this configuration can be determined. Álvarez et al. [6] compare the decentral approach of TSN with a central solution. Their central solution, however, does not use TSN, but Flexible-Time-Triggered-enabled Ethernet. They show that, due to the communication overhead, the decentral approach has a significantly higher latency when requesting new flow reservations than a central approach can offer. Similarly, Thiele and Ernst [20] use the compositional performance analysis (CPA) to model access on the same resource, in this case, a Software Defined Networking (SDN) server. Their goal is to identify the scalability of central approaches. Thereby, they have not implemented a schedulability analysis but instead assume fixed ranges for the duration of each reservation task.

We present an overview of the existing work dealing with the configuration of CBS networks in Table 1. Several research papers have investigated the impact of TT gates, which restrict the sending times of queues, on CBS flows. Laursen et al. [21] and Pop et al. [22] propose a heuristic routing approach to search for paths that minimize the delay for a previously known set of CBS flows. They also assume given priorities and IdleSlopes, thus, resulting in an offline configuration. Offline configuration refers to a static configuration to initially set up the network, without considering any existing reservations. As the standard only offers delay analysis for the highest priority class, they limit their approach to this priority as well. Besides, as the latency calculation of the standard does not provide safe guarantees [28], we denote these deadlines as "soft" in Table 1. Gavriluţ et al. [23], [24] define an optimization

TABLE 1. Overview over CBS configuration approaches. Only handling of CBS flows is illustrated in this table.

								Input				
	Year	Scheduler	# prio.	deadlines	online	routing	fixed sending	topology	flow	prio	paths	slopes
Laursen et al. [21] & Pop et al. [22]	2016	TT&CBS	1	soft		✓	●	●	●	●		●
Gavrilut et al. [23], [24]	2018	TT&CBS	∞	hard				●	●	●	●	●
Li et al. [25]	2019	TT&CBS	2	hard				●	●	●	●	
Chuang et al. [26]	2020	TT&CBS	2	–	✓	✓		●	●	●		●
Gavrilut et al. [5]	2020	TT&CBS	∞	hard				●	●		●	
Berisa et al. [27]	2022	TT&CBS	∞	hard				●	●	●	●	●
IEEE 802.1Qat [17]	2010	CBS	1	soft	✓		●	●	●	●	●	●
Our Solution	2022	CBS	∞	hard	✓	✓		●	●			(●)

problem which they solve by a greedy metaheuristic with an adaptive search procedure. Their goal is to find timings for the TT gates with which TT flows and CBS flows are scheduable, with given paths, priorities, and IdleSlopes. Li et al. [25] use particle swarm optimization with NC to optimize the IdleSlopes for CBS queues while considering one TT gate. As NC derives guaranteed bounds on maximum network latency, we denote these deadlines as "hard". The optimization aims at minimizing the IdleSlopes for a given set of flows, with known priorities, paths, and TT traffic schedules. Instead of successively updating the network for each new reservation, they optimize the network configuration for a complete set of flows. Chuang et al. [26] propose an online routing approach to reserve two CBS queues, as well as TT queues, in parallel. In their solution, the IdleSlopes were known before and not targeted for optimization, and the flow delays are not considered in the implementation. Gavrilut and Pop [5] use Tabu Search on a given set of flows with known paths and a previously defined number of CBS classes. They derive the required number of TT queues, the priority assignment of CBS flows, and the IdleSlopes per queue in an offline approach. Berisa et al. [27] extended the work of Raagaard et al. [29] and Gavriluţ et al. [24] with an iterative approach for TT flows, assuming that all CBS flows are previously configured. Different from the previous solutions, the TSN standard IEEE 802.1Qat [17] offers a decentral reservation protocol with soft latency analysis as proposed in [30]. While this allows for a Plug&Play usage and does not require an additional central entity in the network, it has the disadvantages as described above: limited scheduablity, overly pessimistic and non-safe delays, and high communication overhead.

To the best of our knowledge, no previous work has presented a central online admission control scheme which allows for deadline-guaranteeing flow reservations in CBS networks. Besides, for CBS networks without TT gates, only the decentral approach of the standard has been proposed so far. Analyzing and configuring these CBS networks without TT gates is of high importance, as TT gates require synchronized devices and reduce the usable bandwidth significantly.

Compared to existing research, we do not assume previously known paths. The priority of flows is assigned depending on their deadline requirements and the network state for each hop individually. We, thus, do not need an a-priori mapping to the traffic classes. We also offer one model which automatically derives and configures the required IdleSlopes per queue. Further adding to existing approaches, we consider the maximum buffer sizes at each queue, thus, preventing buffer overflows. We do not require a strict sending behavior, even allowing for legacy end-stations without CBS scheduling capabilities. Besides, we only update bridges on the path of each new flow and do not require a reconfiguration of the complete network to support the online reconfiguration of the network.

The concept of our models is an extension of the framework developed by Guck et al. [31] who have proposed two mechanisms to reserve time-critical flows in networks using Strict Priority (SP). While for SP, the service of each queue is defined by the leftover rate from higher-priority queues [31], the service of CBS is defined by the IdleSlope parameter. Thus, our models utilize the flexibility of this parameter by dynamically assigning the required IdleSlope to each queue. While the output of SP queues is only limited by the link rate, CBS enforces additional rate limits on the output of each queue, and this shaping effect also allows for lower backlogs in the network. As this shaping effect depends on preceding nodes on the flow's path, neither [31] nor the decentral configuration showed how to utilize this behavior effectively, which is another contribution of our work.

Details of the delay analysis of CBS using NC can be found in [32], [33]. A complete overview of NC results for TSN has been presented in [34]. We use NC to calculate end-to-end delays and buffer sizes in the network and, thus, configure and guarantee network characteristics.

III. NOTATION

Our models consider two main QoS parameters: a) the end-to-end delay from sending to receiving for each flow and b) the backlog at each queue. We name the amount of data in the buffer of a forwarding queue *backlog*. When we refer to the physical capacity of a queue, we use *queue size*. With the

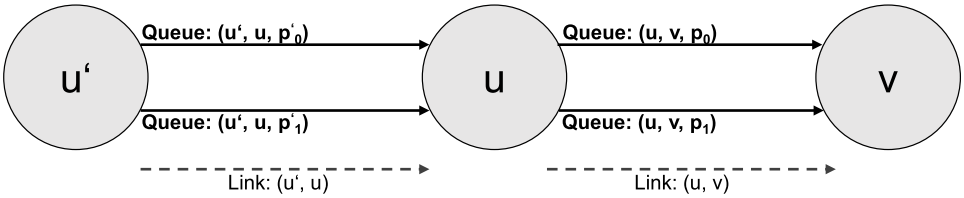


FIGURE 1. Queue-dependent network graph representation: Each CBS output queue results in one edge.

backlog and delay being individual for each output queue, we model the network as a directed multigraph $G = (V, E)$ where each vertex V (hereafter called *node*) represents either an end-station (ES) or a forwarding device. Each edge E represents one CBS output queue. Table 2 provides a list of all variables for our models.

We identify each CBS output queue with priority p uniquely with the tuple $(u, v, p) \in E$ with $u, v \in V$ and $0 \leq p < Q$. We thus have Q edges between the nodes u and v. Physical links are denoted by (u, v) instead. Be aware that more important priorities get lower numbers, e.g., priority 0 has the highest priority. Fig. 1 illustrates our graph model for a simple network. In this example, we have three nodes with two CBS queues per output port.

We introduce the superscript $\bullet^{u,v,p}$ to refer to queue-dependent parameters in the following. Analogously, link-dependent parameters are denoted as $\bullet^{u,v}$. Finally, if a parameter is specific for an output queue (u, v, p) in combination with its preceding output queue (u', u, p'), the superscript of this parameter is $\bullet^{(u',u,p'),(u,v,p)}$. Reservation independent variables are highlighted by the superscript $\widehat{\bullet}$. To reduce the notation complexity, we assume - without loss of generality - the same link capacity C and number of priorities Q at each output port.

IV. FUNDAMENTALS

This section first presents all delay components in a network and introduces the CBS scheduler. Due to the non-trivial queuing delay in CBS queues, our models use NC to derive guaranteed boundaries for delay and backlogs in the network. Therefore, we present the corresponding NC models for CBS. The section concludes with the state-of-the-art decentral configuration approach.

A. END-TO-END DELAY MODELING

We consider the delay from sending a packet at the so-called TSN *talker* device to receiving the same at the TSN *listener* device. The end-to-end delay for a flow f is the sum of all *processing*, *queuing*, *transmission*, and *propagation* delays over its whole path. For a path Φ, we name the set of links $\mathcal{L}_\Phi$ and all scheduled output queues Q_Φ. Then, the complete end-to-end delay for any flow f traversing path Φ is:

$$D_f = \sum_{(u,v)\in\mathcal{L}_\Phi} \left(D^{u,v}_{prop} + D^{u,v}_{t}\right) + \sum_{(u,v,p)\in Q_\Phi} \left(D^{u,v,p}_{Q} + D^{u}_{proc}\right) \tag{1}$$

TABLE 2. Variables and definitions.

Variable	Definition
Topology / Routes	
C	link capacity
Q	number of CBS priorities
B_{phy}	physical queue size
Φ	network path, consisting of links and queues
$\mathcal{L}_\Phi$	set of links on path Φ
Q_Φ	set of queues on path Φ
$\mathcal{L}^-$	set of preceding links
Q^-	set of preceding queues
Flows	
F	set of flows currently reserved in the network
F_{max}	max. set of reservable time-critical flows
CMI	sending interval / sliding window of a flow
MFS	max. packet size of a flow
MIF	max. packets per interval for a flow
Frame Sizes	
L	max. packet size
l	max. packet size of lower priorities
L^{max}	max. packet size in the network
Delays	
D_f	worst-case end-to-end delay of flow f
D_f^{max}	end-to-end deadline constraint of flow f
D_{prop}	propagation delay
D_{proc}	processing delay
D_t	transmission delay
D_q	queuing delay
CBS Scheduling	
$idSl$	IdleSlope
$sdSl$	SendSlope
$idSl_{max}$	max. sum of IdleSlopes over all priorities
c_{max}	max. reachable CBS credit
c_{min}	min. reachable CBS credit
r_α	total reserved rate
Network Calculus	
α	arrival curve
b, r	max. arriving burst and rate
$\beta_{R,T}$	service curve with latency T and rate R
σ_l	shaping curve of link
σ_{CBS}	shaping curve introduced by CBS queue
B	worst-case max. backlog
D	worst-case max. delay (queuing and transmission)

The propagation delay $D^{u,v}_{prop}$ is the time each bit needs to be transmitted over the link which depends on the physical length of the Ethernet cable. The processing delay D^{u}_{proc} is the time from the full reception of a packet to its arrival at the output queue of the same device. This delay accounts for the

time needed to process the packet information, e.g., looking up the routing table and applying policing rules. We refer to $D^{u,v}_{prop}$ and D^{u}_{proc} as hardware delays and consider their upper bounds as constant per link or queue. Estimations for these delays can be found, e.g., in [35].

The queuing delay $D^{u,v,p}_{Q}$ for a flow depends on the number of packets in the queue, its analysis is done using NC. It is introduced in each forwarding device, e.g., all intermediate TSN bridges and scheduled talker devices. Worst-case queuing delay is hard to evaluate and, thus, previous studies consider average queuing delays instead [36]. In contrast, we use NC as this offers the mathematical means to provide hard guarantees for queuing delays. By upper bounding an arrival curve $\alpha(t)$, which represents the aggregate of all incoming flows, and lower bounding a service curve $\beta(t)$, which represents the minimum service per queue, we can define the maximum queuing delay at each queue for all flows [37]:

$$D^{u,v,p}_{q} = h(\alpha(t), \beta(t)) \tag{2}$$

Thereby, h denotes the maximum horizontal distance between two functions, see also Section IV-C-6. Finally, the transmission delay $D^{u,v}_{t}$ is the serialization time for packets over Ethernet links. We account for its effect with an NC model called packetizer [37]. Therefore, we add a burst to so-called shaper curves, as will be explained in Section IV-C-4.

The queuing model of our admission control scheme uses the presented NC curves - arrival, service, and shaper - and finds reservation-independent bounds for them. The resulting hard delay bounds are then used to reserve flows and, thus, offer a delay-guaranteeing system. We consider the maximum worst-case delay, meaning that all experienced delays will be lower or equal.

In the following, we assume - without loss of generality - that the output queues of talker devices are not CBS-scheduled. Be aware that, in the decentral approach, ES have to be CBS-scheduled [17], as the decentral approach relies on the CBS shaping effect to derive the worst-case latency. This will be explained in Section IV-D. In contrast, in a central approach, for each queue, we know whether preceding flows are shaped or not and can include the actual worst-case arrival in our calculations. As a result, the usage of CBS in ES is optional in our approach. This can reduce the implementation complexity of ES and allows for the integration of legacy devices. However, our models can easily be adapted to consider shaped ES if required.

B. CREDIT-BASED SHAPER

CBS was originally proposed for Audio and Video traffic and is defined in the IEEE 802.1Qav standard [11]. It has been designed to reduce backlogs in nodes and ES by introducing a similar behavior as token-bucket shaping.

Fig. 2 shows the structure of an output port implementing CBS. Each output port consists of up to eight FIFO queues and each queue has a priority. Incoming packets are put in the queue with their priority, thus, multiple flows share the same queue. The transmission of each queue is defined by its Transmission Selection Algorithm (TSA) and an optional TT gate. TSA refers to the scheduler, in Fig. 2, either SP or CBS. A packet that is waiting in a queue is only eligible for transmission if the queue's TSA allows its forwarding and the gate is open. If multiple packets are eligible at the same time, they are selected according to their priority.

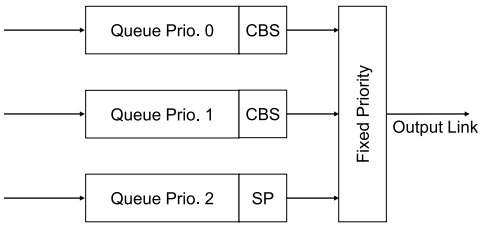


FIGURE 2. Example output port with CBS scheduling.

We assume all TT gates to be open constantly, thus, not affecting the transmission. In our models, the TSA of the highest priority queues is CBS. For each output port, a maximum of seven CBS queues are recommended. Each CBS queue has a credit $c^{u,v,p}(t)$ and packets are only allowed for transmission if $c^{u,v,p}(t) \geq 0$.

We explain the functionality of CBS with the example shown in Fig. 3, assuming two CBS queues. The phases are illustrated by letters at the bottom of Fig. 3.

- Phase (a): In this example, for priority 0, three packets are waiting for transmission at time $t = 0$ while in priority 1 only one packet is queued. For illustration, we assume that a best-effort (BE) packet is under transmission when the four CBS packets arrive.
- The credit $c^{u,v,p}(t)$ for each CBS queue increases with the rate called IdleSlope $idSl^{u,v,p}$. The credit of a queue increases when scheduled traffic cannot be sent due to an ongoing transmission of other queues or because the credit is negative. E.g., in Fig. 3 both queues need to wait for the transmission of the BE packet at $t = 0$ and, thus, their credit increases.
- Phase (b): After the BE packet is finished, both CBS queues have a positive credit and eligible packets, so the priority is used to decide on transmission. Thus, priority 0 can send first.
- During the transmission of packets, the credit of the sending queue decreases with the rate called SendSlope $sdSl^{u,v,p}$ with $sdSl^{u,v,p} = idSl^{u,v,p} - C$.
- Phase (c): In the worst-case, priority 0 will start a maximum packet transmission at the moment its credit reaches 0. This delays the transmission of priority 1 the most.
- Phase (d): After sending three packets of priority 0, the credit of this queue is negative, and priority 1 is allowed to send.
- As the credit of priority 0 is negative, it increases with $idSl^{u,v,0}$. New packets of priority 0 would be delayed until the credit $c^{u,v,0}(t)$ reaches at least 0 again.

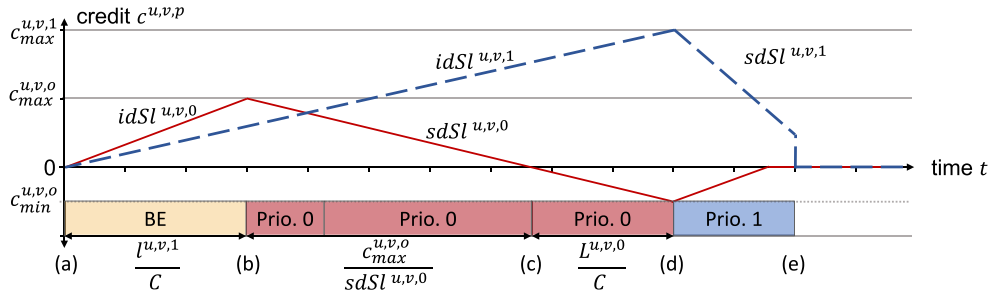


FIGURE 3. Transmission behavior of priority 0 and 1, illustrating the credit evolution. At the bottom, packet transmissions and their worst-case duration to maximize the credit of priority 1 are illustrated.

- Phase (e): If a queue is empty, but the credit is positive, the credit of this queue is set to zero as shown for priority 1 at the end of Fig. 3.

With this behavior, $idSl^{u,v,p}$ of a queue defines both, the maximum and minimum guaranteed bandwidth. Defining a maximum bandwidth is different when compared to, e.g., SP or DRR scheduling. As a result, CBS queues introduce a per-queue shaping behavior and, thereby, improve the overall delays for lower priority traffic.

C. NETWORK CALCULUS MODELS

NC derives guarantees on queuing and transmission delays and backlogs by defining maximum and minimum curves to describe traffic entering and leaving a system, in our case, a TSN output queue. We will present the corresponding NC definitions and guarantees for CBS in this section. The basic curves are illustrated in Fig. 4. A detailed introduction to NC can be found in [37], [38]. For the definition of all variables, please refer to Table 2.

1) FLOW ARRIVAL CURVE

An arrival curve $\alpha(t)$ represents the maximum number of bits that arrive at an output queue in a time interval of length t [37]. To describe this, we model the maximum sending behavior for flows. The IEEE 802.1Qcc standard [19] defines that each talker informs the CNC about the characteristics of its flow for reservation. The flow characteristics are defined by the maximum number of packets, called *MaxIntervalFrame* (*MIF*), with a maximum size of *MaxFrameSize* (*MFS*) bytes that are sent within any interval of length *ClassMeasurementInterval* (*CMI*). From this, the maximum amount of data in bits m that flow f can send during an interval of length *CMI* can be derived as $m = MIF \cdot MFS \cdot 8$. For consistency, we assume packet sizes and *MFS* to include all headers, preambles, and inter-packet gaps (IPG).

Using these values, each flow can be described at the talker by an arrival curve of the following form [39], [40], [41]:

$$\alpha(t) = b + r \cdot t \tag{3}$$

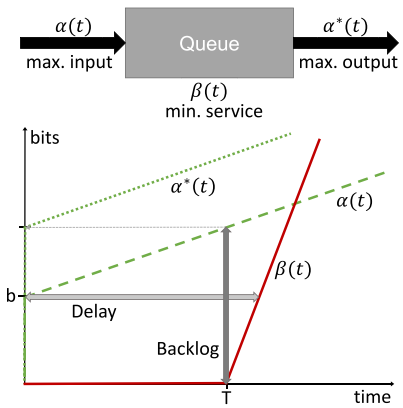


FIGURE 4. Simple NC curves and guarantees for a single queue.

with $b = m \cdot (1 - \frac{r}{C})$ and $r = \frac{m}{CMI}$ for periodic sending. Thereby, it is assumed that all traffic m is transmitted in a burst. Due to the serialization of packets, this burst can be described by b. For aperiodic flows, m is doubled. Thus, we can derive the complete arrival curve from each flow's reservation request at the CNC.

2) SERVICE CURVE

Service curves model the minimum service of a flow during each period of length t at one output queue [37]. The service curve for CBS is a rate-latency curve with the following form [32], [39], [40]:

$$\beta_{R,T}(t) = R \cdot [t - T]^+ \tag{4}$$

For queue (u, v, p) the rate R and latency T are defined as:

$$\begin{aligned} R^{u,v,p} &= idSl^{u,v,p} \\ T^{u,v,p} &= \frac{c^{u,v,p}_{max}}{idSl^{u,v,p}} \end{aligned} \tag{5}$$

Since the IdleSlope is the allocated bandwidth for each queue, it is also the rate parameter $R^{u,v,p}$ in this formula. $c^{u,v,p}_{max}$ models the highest credit that is possible for this queue. Thus, flows of priority p are guaranteed to send after this worst-case value is reached, which is modeled as the latency $T^{u,v,p}$. This is also illustrated with phase (b) and (d) in

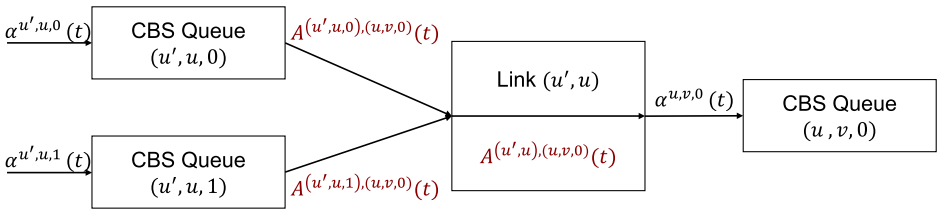


FIGURE 5. Arrival curve for queue $(u, v, 0)$. We illustrate the effect of the preceding link and CBS shaping. For simplicity, we assume that queue $(u, v, 0)$ has only one preceding input link u', u and two preceding CBS queues.

Fig. 3. The maximum and minimum CBS credits at (u, v, p) are defined as [16], [39]:

$$c_{max}^{u,v,p} = idSl^{u,v,p} \cdot \frac{\sum_{i=0}^{p-1} c_{min}^{u,v,i} - l^{u,v,p}}{\sum_{i=0}^{p-1} idSl^{u,v,i} - C}$$
$$\text{and} \quad c_{min}^{u,v,p} = sdSl^{u,v,p} \cdot \frac{L^{u,v,p}}{C} \tag{6}$$

3) OUTPUT ARRIVAL CURVE

After a flow traverses an output queue, its burstiness can increase due to the experienced delay. A flow's maximum output arrival curve describes this changed transmission behavior of a flow after being served by a queue with the service curve β. This output arrival curve then serves as input to the next node on the flow's path. We calculate the maximum output arrival curve for each flow after each hop with [37]:

$$\alpha^*(t) = \sup_{0 \le s} \{\alpha(t+s) - \beta(s)\} \tag{7}$$

With a token-bucket arrival and a rate-latency service curve $\beta_{R,T}(t)$, this equation can be simplified to $\alpha^*(t) = \alpha(t+T)$, as illustrated in Fig. 4. To analyze the end-to-end delay of flows over the network, we iteratively calculate the output arrival curve after each queue and use it as input at the next queue.

4) SHAPER CURVES

In NC, shaper curves can be used to apply upper bounds on the arrival of queues, if the system offers this behavior. Thereby, they reduce the arriving traffic and, thus, can improve the worst-case guarantees on delay and backlog. Precisely, CBS networks allow for two shaper curves: link shaping and CBS queue shaping.

Without shaping, all flows scheduled by one queue are considered as an aggregate and summed up. For multiple flows, this models an instantaneous arrival of packets from all flows. As they are transmitted over physical links, the arrival of all packets cannot happen instantaneously but is shaped by the link rate. Assuming an input link rate of C, the link shaper curve is defined by $\sigma_{l'}^{u,v}(t) = C \cdot t$. However, as we put complete packets in the output queues, we have to account for the transmission delay per link. Therefore, we add the concept of a packetizer to this shaping behavior, leading to [16], [33], [37]:

$$\sigma_l^{u,v}(t) = L^{u,v} + Ct \tag{8}$$

For cut-through networks, this shaper curve would not be increased by a whole packet $L^{u,v}$ but only by the size of the header required to read before forwarding the packet.

As $idSl^{u,v,p}$ also denotes the maximum output rate for queue (u, v, p), it defines an upper bound on the maximum output and, thus, to the arrival curve of subsequent network elements. The concept of CBS shaping to introduce an upper bound on the output arrival curve was first proved in [39]. In contrast to the worst-case credit, as in (4), CBS shaper curve models the optimal credit evolution, resulting in a non-greedy maximum shaper curve of [16], [39], [42]:

$$\sigma_{CBS}^{u,v,p}(t) = idSl^{u,v,p} \cdot \left(t + \frac{c_{max}^{u,v,p} - c_{min}^{u,v,p}}{idSl^{u,v,p}} \right) \tag{9}$$

5) QUEUE ARRIVAL CURVE

Let $A^{(u',u,p'),(u,v,p)}(t)$ denote the aggregated arrival of all flows at queue (u, v, p) after queue (u', u, p'), see also Fig. 5. We apply the CBS shaping effect for each preceding queue:

$$A^{(u',u),(u,v,p)}(t) = \sum_{p'=0}^{Q-1} \min\left(A^{(u',u,p'),(u,v,p)}(t),\ \sigma_{CBS}^{u',u,p'}(t) \right) \tag{10}$$

Simply spoken, $A^{(u',u),(u,v,p)}$ denotes the aggregated arrival from link (u', u) heading to queue (u, v, p) and upper bounded by the shaping of the CBS queues.

Additionally, the arriving traffic of one link is shaped according to the link shaping curve. Let $\mathcal{L}_{u,v,p}^-$ be the set of incoming links for queue (u, v, p). Then, the shaped arrival curve for one queue can be described as:

$$\alpha^{u,v,p}(t) = \sum_{(u',u) \in \mathcal{L}_{u,v,p}^-} \min\left(A^{(u',u),(u,v,p)}(t), \sigma_l^{u',u}(t) \right) \tag{11}$$

For illustration, the order of shaping and the resulting arrival curves are shown in Fig. 5.

Fig. 6 shows all presented NC curves for multiple hops. For illustration, we assume that ES are not CBS-shaped. Thus, the arrival curve at the first hop is the sum of all flow

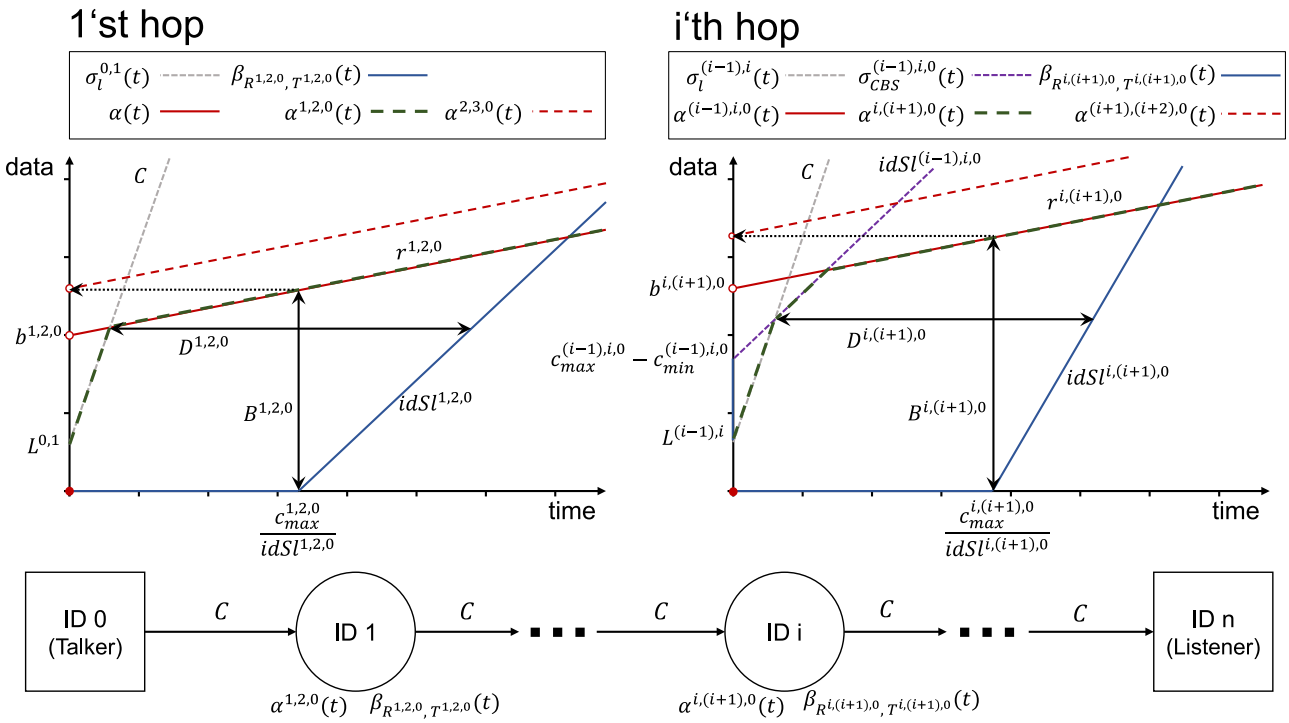


FIGURE 6. NC analysis for a CBS network with a single CBS queue of priority 0. We illustrate the curves for the first hop with ID 1 with no CBS shaping and all further hops with ID *i* with CBS shaping.

arrival curves shaped by the first link. This is visualized by the green dotted line for the first hop. We calculate the maximum output after each queue. All further queues have preceding CBS queues, thus, we apply the CBS shaper curves as well. As a result, for the arrival curve at the *i*'th hop traversing to the *(i+1)*'th hop, we shape all output arrival curves of preceding queues by the link and CBS shaper curves. This is exemplified on the right side of Fig. 6.

6) MAXIMUM OUTPUT, DELAY, AND BACKLOG

Using the previously defined shaped arrival and service curves, we can derive an upper bound on the backlog and the delay introduced by each queue for all packets. To find the maximum delay, we calculate the horizontal distance at each kink point of both curves and use the maximum. The definition for the maximum horizontal distance is [37]:

$$D^{u,v,p} \leq \sup_{0 \leq s} \left\{ \inf \left\{ \tau \geq 0 \mid \alpha^{u,v,p}(s) \leq \beta_{R^{u,v,p},T^{u,v,p}}(s+\tau) \right\} \right\} \tag{12}$$

Analogously, the maximum vertical distance between the arrival and service curve is the worst-case backlog, defined as [37]:

$$B^{u,v,p} \leq \sup_{0 \leq s} \left\{ \alpha^{u,v,p}(s) - \beta_{R^{u,v,p},T^{u,v,p}}(s) \right\} \tag{13}$$

D. STATE-OF-THE-ART CONFIGURATION

The IEEE 802.1Qat [17] and 802.1BA-2021 [30] standards define the decentral configuration for CBS queues. Thereby, flows' requirements are propagated in the network using SRP, which also configures the TSN bridges. The bridges keep track of the existing reservations and decline new reservations if not enough resources are available, or the flows' deadline cannot be guaranteed. Thereby, the delay is calculated in each bridge using locally available information. The formula for priority 0 is provided by the standard [30] as follows:

$$D^{u,v,0} = D^{u,v}_{proc} + D'^{u,v}_{t} + \frac{L^{max}}{C} + \left(\underbrace{\frac{idSl^{u,v,0}}{C}}_{\text{perc. of link rate}} \cdot CMI - D^{u,v}_{t} \right) \cdot \frac{C}{idSl^{u,v,0}} \tag{14}$$

The first line of (14) represents *processing* and *transmission* delays, with $D'^{u,v}_{t}$ being $D^{u,v}_{t}$ without IPG. As we are considering worst-case queuing delay, they also add the non-preemptive interference of a maximum size packet L^{max}.

The second line is an estimation of the maximum queuing delay. The percentage of the reserved link rate is used to model the arrival during one *CMI*. Traffic is received by the link rate C and forwarded by the maximum rate $idSl^{u,v,0}$, which is represented by the right side of (14).

For this estimation, the standard defines *CMI* as a global constant for all flows of priority 0. As we can see in (14),

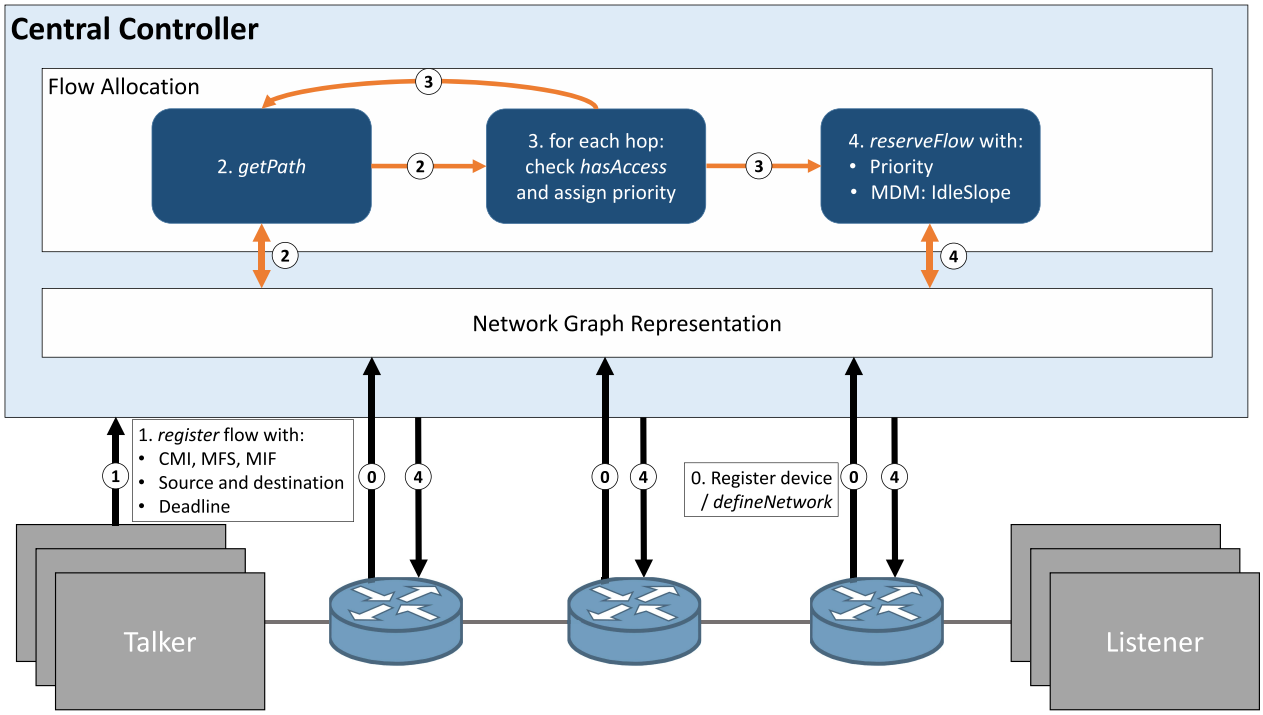


FIGURE 7. The overall process of our central admission control scheme.

higher values for *CMI* increase the estimated queuing delay. Therefore, *CMI* is set to a small value of 125μs.

Equation (14) models arriving traffic up to the queue's IdleSlope and does not consider preceding queues with potentially lower IdleSlopes. However, this can lead to overly pessimistic estimations. Besides, the small and fixed value for the sending interval *CMI* leads to overheads when reserving flows, as will be illustrated in Section VI-C. Additionally, the multitude of reservation messages in the decentral approach can exceed the processing capacity of bridges [43]. With a central architecture, the size and frequency of reservation messages can be reduced. In the following, we will refer to the decentral approach using (14) as *standard approach*.

V. ADMISSION CONTROL

In this section, we will first explain the common basis of our two models and then present the check for reservation by each model in detail.

Following the IEEE 802.1Qcc standard, our models are implemented in the central TSN instance called CNC. The CNC is a separate instance responsible for the reservation and configuration of all flows and nodes in the network. It is aware of the complete network topology and available hardware, similar to an SDN controller. To communicate with the TSN network devices, network management protocols are used such as the Network Configuration Protocol (NETCONF) defined in RFC 6241.[1]

1. https://datatracker.ietf.org/doc/html/rfc6241

A. COMMON BASIS OF BOTH MODELS

Our two models allow reservations for arbitrary networks (e.g., star, ring, tree,. . . ,) and an arbitrary number of CBS queues, assuming that the CBS queues have the highest priority at each hop. Be aware that the TSN standard defines that there must be at least 2 and a maximum of 7 CBS queues.

Fig. 7 shows the general architecture and logic of our central admission control scheme. As input, our models need the network topology and a flow f with the following information:

- *CMI*, *MFS* and *MIF*, as described in Section IV-C-1
- source and destination node
- maximum deadline requirement

This information is forwarded by the ES to the CNC. Besides, each new network device registers itself at the CNC, as defined in [19]. With this information, we represent the topology internally as a graph, as will be described later. No further input is required for the proposed algorithm.

We address three problems in our admission control scheme:

- *Routing and Prioritization:* For each flow f, we find a path through the network and the flow's priority at each hop.
- *Device Configuration:* For each CBS queue in the network, we define individual IdleSlopes.
- *Flow Reservation:* We reserve available resources in the network for flow f.

Each of these problems is solved with three goals:

Algorithm 1: General Flow Allocation

Input: network ← MultiDiGraph with queue sizes and linkrate
Input: method ← cost-function for routing
Input: **MDM:** param ← delay per queue
Input: **FSM:** param ← IdleSlope (optional: reduced queue sizes)

```
1  Network nw = defineNetwork (network, param);
2  Function register (flow f):
3      P ← nw.getPath(flow=f, deadline=D_f);
4      foreach path p ∈ P do
5          success ← True;
6          foreach hop h ∈ p do
7              success ← nw.hasAccess(flow=f, hop=h);
8              if not success then
9                  break;
10             end
11             f ← nw.outputCurve(flow=f, hop=h);
12         end
13         if success then
14             p_f ← p;
15             break;
16         end
17     end
18     if success then
19         nw.reserveFlow(flow=f, path=p_f);
20         break;
21     end
22 end
```

- all newly registered and already reserved flows $f \in F$ keep their end-to-end deadline guarantees
- no CBS queue experiences buffer overflow
- the number of reservable flows, as well as the chance of successful future reservations, is maximized

We can formulate these goals by the following constraints:

$$D_f \leq D_f^{max}, \forall f \in F \tag{15}$$

$$B^{u,v,p} \leq B_{phy}^{u,v,p}, \forall (u,v,p) \in E \tag{16}$$

and with F_{max} being the maximum number of reservable flows, the objective function is:

$$maximize\ (|F_{max}|) \tag{17}$$

As our proposed solution is an online algorithm, we cannot apply an optimization function to a set of pre-defined flows. Instead, we apply an iterative approach, which aims at maximizing the number of reservable flows, as expressed in (17). While the constraints are checked after each iteration and never violated, the objective is only approached by considering variable routing weights. The general algorithm will be explained in the next section.

In CBS networks, additional traffic has the potential to increase the delay for already reserved flows, as well as the buffer sizes in queues. However, for scalability and efficiency, we ensure that adding, changing, or removing reservations only affects nodes on the flow's path, not the whole network. Besides, existing reservations are not violated, when new flows are reserved. To ensure this, our models offer reservation-independent guarantees on the queuing delay for flows. This is solved differently in the two models and will be explained in detail in the corresponding sections.

Of course, we can also run our models offline for a complete set of flows. In this case, we can introduce additional benefits by sorting the flows according to their relevance, as - if not all flows can be reserved - the least relevant flows remain, and their scheduling can be retried with less strict deadline constraints or smaller bandwidth requirements.

1) GENERAL ALGORITHM

Algorithm 1 shows the program structure that our two models have in common: Before flows can be allocated, the network topology is built as a directed multi-graph, representing the network characteristics (line 1). For each new registration of a flow of interest (FoI), we obtain potential candidate routes, with the routing algorithm introduced in Section V-A-2 (line 2-3). Then, the candidate paths are successively checked at each hop whether they allow for a reservation of the FoI in the output queue (line 4-10). After each hop, the output arrival curve is calculated for the next hop on the path (line 11). The first path for which all checks are successful will be chosen for the allocation of the FoI (line 13-end). The CNC will then inform each device on the path about the new flow, its priority at this hop, and the necessary IdleSlopes for each CBS queue.

The main difference between our two models is the check for a reservation on a path, which is represented by the *hasAccess* function (line 7) of Algorithm 1. In this function, the *Fixed Slope Model* (FSM) defines the maximum possible IdleSlope at each hop and, thus, up to which maximum arrival we will reserve flows at this queue. The second model, *Maximum Delay Model* (MDM), assigns a maximum delay to each queue and later calculates the corresponding IdleSlopes to ensure that this delay is guaranteed. The two models also differ in the *outputCurve* function (line 11). Both will be explained in Sections V-B and V-C correspondingly.

If the reservation for a FoI is not possible, e.g., because no path can keep the end-to-end deadline constraint or necessary queues are already fully reserved, our framework either informs about the bottleneck queues or reports which delay could be guaranteed instead. So, the registration for this flow can be retried with the increased deadline constraint or the user can decide whether other flows need to be de-registered to allow the FoI's reservation. Be aware that, depending on the routing choice, flows preventing the FoI from registration might be able to choose a different path. In the future, we want to test this automatically to improve the user's information.

2) ROUTING

A main advantage of our solution is that we do not require an a-priori mapping of flows to priorities. Constant priorities of flows might result in a lower utilization of the network, as high priority queues might be occupied without necessity. In TSN forwarding devices, each flow (called TSN stream) is handled by a stream filter. With this filter, it can get an internal priority value specification assigned which is then used to determine the priority for this hop [44]. However, allowing variable priorities per hop increases the solution space for the configuration. The number of choices for the reservation increases for each physical route between two nodes from Q to $Q^{|\mathcal{L}_\Phi|}$. Note that - if this flexibility is not required - it is possible to only allow constant priorities with our models by including only paths with the same priority in the solution space.

We use a delay-constrained least-cost (DCLC) routing algorithm, based on the weighted K-shortest-paths routing algorithm, as proposed in [45]. We decided on a loopless routing algorithm, as its complexity increases only linearly with K. The cost-function either represents the path length or the remaining rate of each link. Be aware that this does not ensure optimality for the total of reservations. However, by considering the remaining rate, the reservations can be balanced in the network which increases the chance for future successful reservation, as will be discussed in Section VI-B-1

The routing algorithm returns K candidate paths, i.e., physical routes. For multiple priorities at each link, we use the lowest delay value as constraint in the routing algorithm. Then, we check the priority queues in each candidate path successively. Thereby, the algorithm can be configured to prefer high or low priority queues. Due to our reservation-independent models, the network topology can be changed, especially, new ES can be added during runtime. However, to make sure that no existing reservation is violated, the newly added topology may not change existing paths.

3) UPPER BOUND ON IDLESLOPES

The values for the IdleSlopes in a CBS network are not defined by the standard [11]. However, in practical use cases, the sum of IdleSlopes per output port should not exceed 75% of the outgoing link rate, to ensure that 25% of the link rate is left for non-AVB traffic:

$$\sum_{p=0}^{Q-1} idSl^{u,v,p} \leq idSl^{u,v}_{max} \leq 0.75 \cdot C \tag{18}$$

Thereby, $idSl^{u,v}_{max}$ denotes the user-defined maximum allowed sum of IdleSlopes for each output port.

Our configuration models define the IdleSlope values $idSl^{u,v,p}$ for each CBS queue in the network and reserve flows while guaranteeing the inequality of (18), as well as maximum delay and backlogs.

Additionally, we can use only a subset of CBS queues for our reservations, leaving other CBS queues for traditional or decentral configuration without deadline guarantees. For this scenario, HRT traffic can be reserved using our configuration models to offer strict guarantees, while SRT traffic can still use the remaining CBS queues without the need for reservation, and NRT traffic remains for the BE queue. A typical example of SRT traffic in this context are audio and video flows where the packets should arrive within a certain time but the application can tolerate some packet losses. To assure this, our algorithm needs to lower $idSl^{u,v}_{max}$ as defined in (18), leaving some rate for the non-reserved queues.

B. FIXED SLOPE MODEL

In the following, we will explain how we derive reservation-independent but hard guarantees for delay and backlogs in each of our two models and check for the access of a new FoI. Our first model is called *Fixed Slope Model* (FSM), as it requires the IdleSlope of each CBS queue statically as input parameter. We represent flows using the arrival curves defined by NC and reserve flows dynamically in the queues until an upper bound on the aggregated arrival is reached.

Initialization: As main part of FSM, we define a worst-case service curve for each queue, regardless of the reservation state of the network. Therefore, we use maximum values for the CBS service curve as defined in (4). With the service curves depending on the packet sizes in the individual queues, we need to define L^{max} to be the upper bound on all packet sizes in FSM. As a result, the service curves for all the queues are known before the reservations start.

Given these service curves, the FSM algorithm then computes a maximum allowed arrival curve for each queue:

$$\hat{\alpha}^{u,v,p}(t) = \hat{b}^{u,v,p} + \hat{r}^{u,v,p} \cdot t \tag{19}$$

The long-term arrival rate $\hat{r}^{u,v,p}$ must not exceed the service rate $idSl^{u,v,p}$, otherwise the delay and backlog may increase infinitely:

$$\hat{r}^{u,v,p} \leq idSl^{u,v,p} \tag{20}$$

We then use the maximum physical queue size $B^{u,v,p}_{phy}$ and the maximum rate $\hat{r}^{u,v,p} = idSl^{u,v,p}$ to define the maximum allowed arrival curve. Using $\hat{r}^{u,v,p}$ and $B^{u,v,p}_{phy}$, we can define the initial burst $\hat{b}^{u,v,p}$, as illustrated in Fig. 8.

Given these maximum arrival and service curves, FSM ensures that new reservations do not violate existing reservations, and neither do they influence the reservations at other parts of the network than the flow's path.

The maximum arrival curve can be improved by applying link and CBS shaping. As already discussed in [31], the effect of shaping this maximum allowed arrival curve is limited. For a queue (u, v, p), the set of all input links is $\mathcal{L}^-_{u,v,p}$ and the set of all input edges is $Q^-_{u,v,p}$. The maximum allowed arrival curve can be shaped by considering all input links $n = |\mathcal{L}^-_{u,v,p}|$ and the maximum packet size L^{max}. This results in the following reservation-independent link shaper curve:

$$\hat{\sigma}^{u,v}_l(t) = n \cdot \left(L^{max} + Ct\right) \tag{21}$$

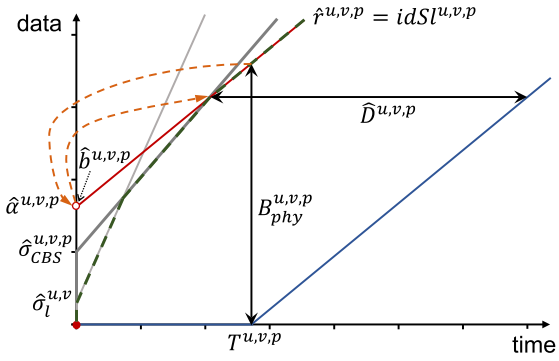


FIGURE 8. Maximum allowed arrival curve and upper bound on delay and backlog for FSM.

We can also define upper bounds for the maximum and minimum credit for each preceding CBS queue. With this, we can model reservation-independent CBS shaping behavior of preceding queues as:

$$\hat{\sigma}^{u,v,p}_{CBS}(t) = \sum_{(u',u,p')\in Q^{-}_{u,v,p}} \sigma^{u',u,p'}_{CBS} \tag{22}$$

The green dotted line in Fig. 8 shows the shaped maximum allowed arrival for a CBS queue which guarantees that the queue size is not surpassed and results in a maximum possible queuing and transmission delay $\hat{D}^{u,v,p}$ at this queue.

We use the derived $\hat{D}^{u,v,p}$ for our DCLC routing algorithm. Be aware that flows experience smaller delays as long as the aggregated arrival curve of the reserved flows is smaller than the maximum allowed arrival curve at this queue.

hasAccess: The *hasAccess* function in Algorithm 1 uses $\hat{\alpha}^{u,v,p}(t)$ for each queue to check for the reservation of a FoI. As long as the sum of arrivals of reserved flows does not exceed $\hat{r}^{u,v,p}$ and $B^{u,v,p}_{phy}$, the reservations are accepted.

Additionally, we shape the aggregate arrival curves at each queue for the actual reserved traffic. In contrast to the previously described shaping of the maximum allowed arrival curve $\hat{\alpha}^{u,v,p}$, shaping the actual queue arrival curves has a significant effect on reducing the backlogs in the network as shown in Section VI. Thereby, the reservations in each queue can be shaped to the link shaping curve using the actual packet sizes $L^{u,v,p}$ instead of L^{max} which we needed for the maximum allowed arrival curve in (21).

While we also use the actual IdleSlopes for CBS shaping in the queues, we omit using the actual packet sizes for CBS shaping. An update in the maximum packet size of a queue will change the CBS shaping behavior of equal and lower priority queues. This requires reshaping in all succeeding queues of the same and lower priorities for each updated packet size, triggering an update of a large part of the network. With our model operating at runtime, we omit this reshaping for performance reasons and use L^{max}. However, when our solution is used as an offline configuration, actual packet sizes for CBS shaping can be used as additional improvement.

outputCurve: The arrival output in Algorithm 1 is calculated for each flow at each hop individually in the *outputCurve* function. With the shaping at each queue, we can spare individual per-flow shaping, increasing the performance of the output calculation. In fact, the output calculation of (7) for each flow can be simplified to $\alpha^*(t) = \alpha(t + T^{u,v,p})$.

reserveFlow: If it is possible to reserve the FoI in all queues on the path, the reservation in the network devices can be triggered. Otherwise, the next candidate path will be checked until the reservation is possible. If no path allows for the FoI's reservation, the user can be informed for further actions.

C. MAXIMUM DELAY MODEL

Instead of fixing the IdleSlopes, our *Maximum Delay Model* (MDM) uses a maximum delay $\hat{D}^{u,v,p}$ for each queue as input. It adapts the IdleSlopes depending on the reservations to guarantee that this delay is never surpassed. In contrast to the first model, we do not define an upper bound on the arrival curves. Instead, we define the IdleSlopes and, thus, the service curves, after each reservation.

hasAccess: When checking for access of a new FoI, we add its arrival to each queue successively on a candidate path, shape the aggregated arrival at each queue, and calculate the required service curves to meet the predefined delay. As the service curves change with each registration and de-registration of a FoI, the output arrival and shaper curves also change. To ensure that new registrations will not lead to violations of existing reservations, we need to define reservation-independent bounds for these two curves.

Again, we use the actual packet sizes for link shaping at each queue. As the IdleSlopes are potentially changed with every reservation, the worst-case IdleSlope for each queue is $idSl^{u,v}_{max}$. However, if we assume $idSl^{u,v}_{max}$ for the CBS shaping of each queue, this worst-case shaping would only lead to an insignificant effect, as only the sum of queues could actually reach this value. While we shaped each queue to the real preceding IdleSlopes $idSl^{u,v,p}$ in FSM, we can only use $idSl^{u,v}_{max}$ for each queue in MDM. Thus, as an improvement, we aggregate the CBS shaper curves per preceding link instead of shaping each preceding queue individually. This is reflected by changing (10) to:

$$\min\left(\sum_{p'=0}^{Q-1} A^{(u',u,p'),(u,v,p)}(t),\ \sum_{p'=0}^{Q-1} \sigma^{u',u,p'}_{CBS}(t)\right) \tag{23}$$

A proof can be obtained by simple case differentiation. For the aggregated CBS shaping curves, we can then replace the sum of IdleSlopes with $idSl^{u,v}_{max}$ as:

$$\sum_{p=0}^{Q-1} \sigma^{u',u,p'}_{CBS}(t) \le \sum_{p=0}^{Q-1} \hat{b}^{u,v,p}_{CBS} + idSl^{u,v}_{max} \tag{24}$$

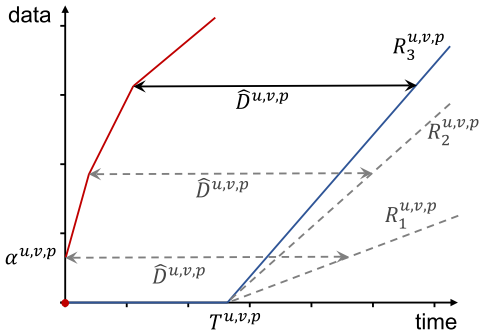


FIGURE 9. Illustration of IdleSlope calculation for MDM at each queue. The resulting IdleSlope at this queue is $R_3^{u,v,p}$.

The maximum burst for the CBS shaper curve is derived by assuming worst-case IdleSlopes for each queue:

$$\hat{b}_{CBS}^{u,v,p} = \max_{idSl^{u,v,i}, i \leq p} \left(c_{max}^{u,v,p} - c_{min}^{u,v,p} \mid \sum_{i=0}^{p} idSl^{u,v,i} \leq idSl_{max}^{u,v} \right) \tag{25}$$

We replace the packet size for all shaping curves with L^{max}, for the same reasons as in FSM.

Using the shaped arrival curve, we calculate service curves for each queue that actually guarantee $\hat{D}^{u,v,p}$ after each new reservation of a FoI. To derive $idSl^{u,v,p}$, we check every kink point i of the shaped arrival curve and calculate the rate $R_i^{u,v,p}$ which results in a delay of $\hat{D}^{u,v,p}$ at this point. This is illustrated in Fig. 9.

The service rate must be greater or equal to the long-term reserved rate $r_\alpha^{u,v,p}$ at each queue. Thus, the resulting service rate is:

$$R^{u,v,p} = idSl^{u,v,i} = \max\left(r_\alpha^{u,v,p}, \max_i \left(R_i^{u,v,p} \right) \right) \tag{26}$$

For $\beta_{R,T}(t)$, $T^{u,v,p}$ has been introduced in Section IV-C-2 as:

$$T^{u,v,p} = \frac{\sum_{i=0}^{p-1} c_{min}^{u,v,i} - l^{u,v,p}}{\sum_{i=0}^{p-1} idSl^{u,v,i} - C} \tag{27}$$

Since $l^{u,v,p}$ depends on BE traffic, we replace it with L^{max}. Besides, we see that the service curve depends on the IdleSlopes and maximum packet sizes (due to c_{min}) of higher priority queues. Therefore, after each reservation, we need to successively update the service curves of all lower priority queues (u, v, i) with $p < i < Q$ and, thereby, recalculate all $idSl^{u,v,i}$.

outputCurve: MDM changes $T^{u,v,p}$ after each reservation. Therefore, in the worst-case, the output arrival curve thus would be increased by $\alpha^*(t) = \alpha(t + \hat{D}^{u,v,p})$ (remember that we used $T^{u,v,p}$ for FSM). This pessimism is a significant limiting factor for MDM.

However, we can improve this as we know that an output arrival curve is only relevant if at least one reservation is made. With the smallest Ethernet packet having a size of 64 B and the maximum rate being $idSl_{max}^{u,v}$, we know for a given delay $\hat{D}^{u,v,p}$, the largest possible value for $T^{u,v,p}$ is

$$T^{u,v,p} = \hat{D}^{u,v,p} - \frac{64\ B}{idSl_{max}^{u,v}} \tag{28}$$

leading to

$$\alpha^*(t) = \alpha\left(t + \hat{D}^{u,v,p} - \frac{64\ B}{idSl_{max}^{u,v}} \right) \tag{29}$$

reserveFlow: If we find valid IdleSlope that can guarantee $\hat{D}^{u,v,p}$ in each queue, the flow is reserved in the network devices. If instead the reservation violates constraint (16) or (18), other priorities and paths are checked successively. If all paths fail, the FoI cannot be reserved.

For MDM, prioritizing lower priority queues in routing can lead to a blocking situation. Low priority queues may get assigned all flows and, thus, set the IdleSlope to the maximum value. This would result in higher priority queues having no IdleSlope for their reservation left and, thus, being unused. A potential solution to this shortcoming is by choosing either variable cost-functions [31], prefer high priority queues if possible, or set minimum values for the IdleSlopes of each queue that need to be available for them.

Redefining the IdleSlopes for every flow reservation is both advantageous and disadvantageous. On the one hand, it ensures that the chosen slopes are always at the minimum necessary rate and, thus, reduces the burstiness of lower priority traffic. On the other hand, devices which need to reserve a new flow also need to reconfigure their IdleSlopes. Reconfiguring queues during runtime is also defined in the TSN standard [19]. However, if bridges do not allow safe reconfiguration at runtime, then MDM may only be used offline before the network setup.

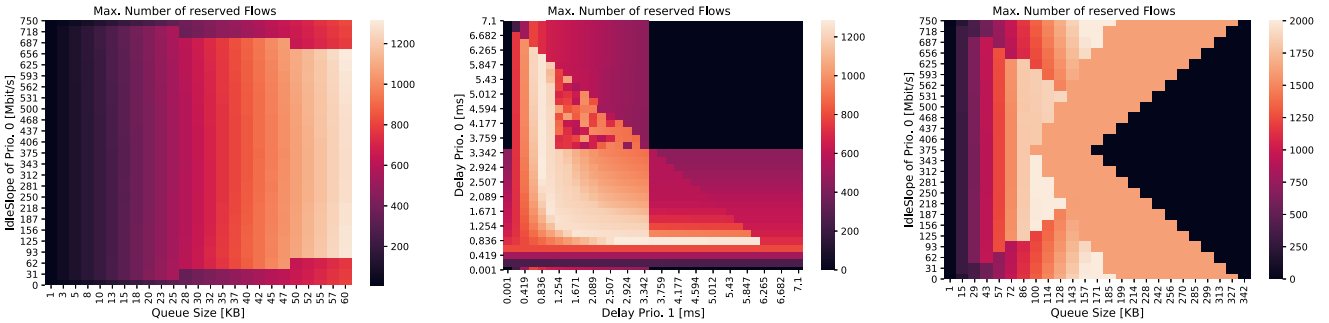
VI. EVALUATION

We categorized our evaluations into three main parts, each investigating separate influences. First, we illustrate the effects of each model in individual queues of a small network. Afterwards, we apply both models to industrial networks to study network-wide effects. Finally, we compare the benefits of our central approaches to the decentral solution of the TSN IEEE 802.1Qat standard.

If not stated differently, all of our simulations allow a reservation of up to 75% of the available link capacity, we assume two CBS queues, the priority of each flow is assigned dynamically at each hop, and the maximum BE packet size is assumed to be 1522 B. The default routing algorithm will prioritize high priority queues with $K = 2$ and the duration of flow reservations is exponentially distributed with a mean of 100 s.

A. INDIVIDUAL QUEUES STUDIES

In the following, we assume a two-hop line-topology with 1 Gbit/s links.



(a) Parameter variation for FSM. (b) Parameter variation for MDM. (c) Effect of increased queue size in FSM.

FIGURE 10. Effects of input parameters on the number of reserved flows.

1) INPUT PARAMETER EFFECT

To illustrate the effect of different input parameters in each model, we observe the number of flows that can be reserved. On average, 250 new flows are registered per second. The flow parameters are derived from [31] with a data rate uniformly distributed between 50kB/s and 150kB/s, a burst size between 70B and 150 B, and a maximum packet size ranging from 64 B to the burst of the flow. This leads to a situation in which the resources cannot serve all flows and, thus, we can show the limits of both models. We assign a deadline of 4 ms to 20% of all flows, while the others are set to 7 ms. Fig. 10(a) varies the IdleSlope for priority 0 in FSM, and priority 1 receives the remaining slope percentage up to 75% of the link bandwidth. We can see the effect of varying IdleSlopes and queue sizes in FSM in Fig. 10(a). Fig. 10(b) shows the result for MDM with a fixed queue size of 60kB and varying delay values for the two priority queues.

As we can see, for the optimal parameters, the maximum number of reserved flows becomes similar in both models. In FSM, assigning large IdleSlopes values to one of the two queues prevents reservations in the other queue - due to the high delay for small IdleSlopes. In MDM, assigning high delays leads to the rejection of flows - due to buffer overflow - as we use the delay to increase the burstiness at each hop. On the other hand, small delays require MDM to assign higher IdleSlope values, leading to a rejection of flows as these delays cannot be guaranteed. The dappled area of Section V-C has been explained before: It occurs when low priority queues set the IdleSlopes close to the maximum capacity. Then, reservations in higher priority queues are prevented as no IdleSlope is left, resulting in a too pessimistic reservation. Thus, choosing fitting input values for MDM is of significant importance. With Fig. 10(b), we could show that it is advantageous to distribute delays over both queues with higher priorities having a similar or lower priority than low priority queues.

To show the effect of large queue sizes on FSM, we also conducted the simulation for larger queue sizes in Fig. 10(c). Be aware that this simulation now offers more resources than Fig. 10(a) and Fig. 10(b) and, thus, prevents a direct comparison. However, we can show that with increased queue size, the maximum allowed arrival curve will increase, leading to a higher per-hop delay. Thus, FSM is not able to reserve flows with low deadline constraints anymore. In such cases, it is thus beneficial to artificially reduce the queue size. For MDM, higher queue sizes are always beneficial.

Note that the delays for both MDM and FSM are not entitled to tightness, as - to enable online reservation - we applied worst-case estimations, e.g., on the arrival and preceding shaping behavior, as explained in Sections V-B and V-C.

2) SHAPING EFFECT

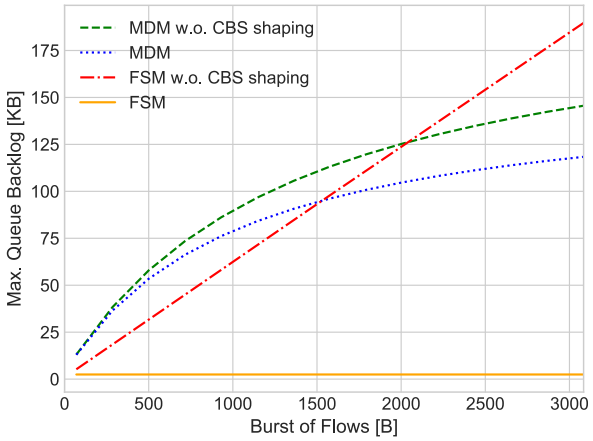
The shaping effect of CBS when reserving flows has a significant impact on the backlogs and assigned IdleSlopes in the network. To compare both models, we do not assign deadline constraints for the flows to allow that both models reserve a total of 150 flows in parallel. We use the optimal values derived for both models from a parameter variation. Fig. 11(a) shows the reserved backlog in the second hop. As we can see for FSM, without CBS shaping, the backlog increases linearly as the burst size of the flows increases linearly too. Introducing CBS shaping fixes this backlog to a maximum value for FSM. For MDM, the backlog in the second queue does not increase linearly, as the IdleSlopes in the queues are increased with larger arrivals. Therefore, we see in Fig. 11(b) how the IdleSlopes are constant for FSM but increasing for MDM. However, the assigned IdleSlopes are improved by CBS shaping for MDM leaving more rate to lower priority queues.

B. INDUSTRIAL NETWORK STUDIES

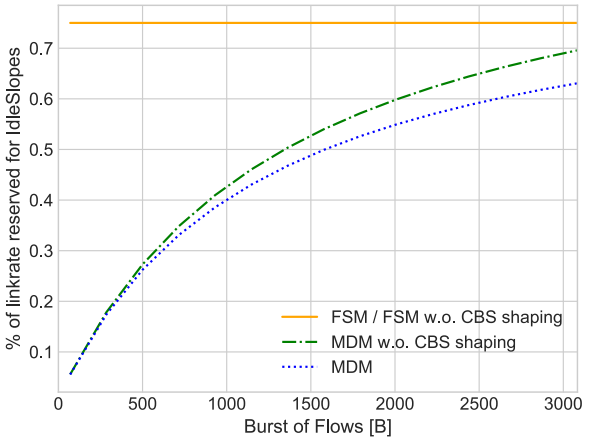
To study network-wide effects, we use ring topologies with a size of m nodes, with $4 \leq m \leq 8$, as typical structure for industrial networks, as well as an automotive network defined by Renault. All topologies are illustrated in Fig. 12.

1) ROUTING EFFECT

We want to study the effects of the different routing methods as proposed in Section V-A-2. We differentiate between preferring priorities and weighting paths with the remaining link rate. In balanced networks, where the traffic load is equally

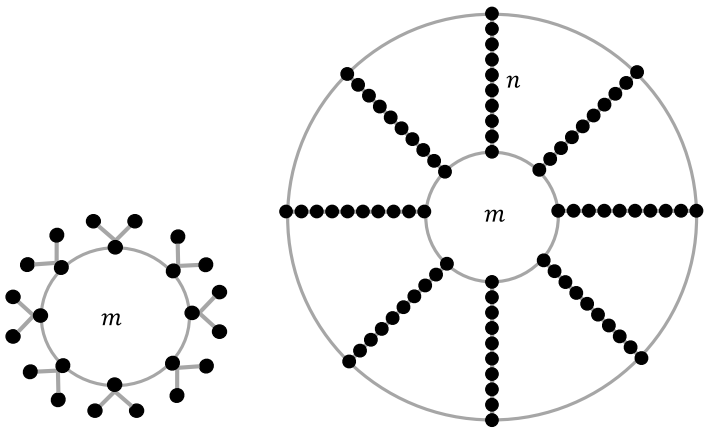


(a) Evolution of reserved backlog with and without CBS shaping.



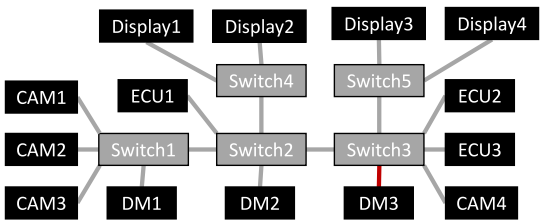
(b) Sum of reserved IdleSlopes with and without CBS shaping.

FIGURE 11. **Shaping effect on CBS queues.**



(a) Ring topology for routing simulation.

(b) Ring and line topology for runtime simulation.



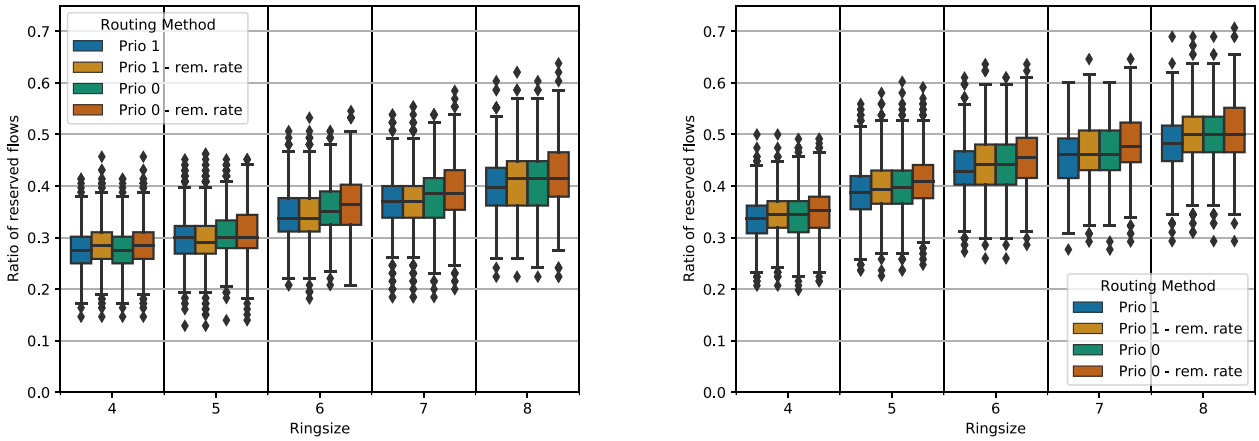
(c) Renault's automotive network. DM = data manager; CAM = camera; ECU = electronic control unit

FIGURE 12. **Industrial network topologies.**

distributed over the network, weighting the paths with the remaining link rate does not provide any benefit. Therefore, we conduct this experiment in a ring with a single high load link. All links have 100Mbit/s and each node is connected to two ES, the network topology is illustrated in Fig. 12(a). We simulate half the flows to have strict real-time requirements of 2.5ms deadlines, the others with a relaxed deadline of 5ms. The average number of flows in the simulation is 533, with 75% having neighboring source and destination device (shortest path over only two hops) and 25% having randomly distributed sources and destinations. Each flow has an exponentially distributed amount of data per interval with a mean of 188 B and a sending interval of 250μs. This leads to a situation in which the short path would be fully occupied with strict and relaxed flows. However, relaxed flows could also choose the longer route instead. For MDM, we set the input delays uniformly distributed over each hop, thereby, enabling the strict flows to take the short and the relaxed flows the longer route. For FSM, we simply assume IdleSlopes of 50% for the high and 25% for the low priority queue. The simulation is repeated 250 times per routing algorithm and ring size. As we can see in Fig. 13(a) for MDM and in 13(b) for FSM, prioritizing high priority queues and balancing the flows by weighting the path with the remaining available rate results in superior performance. However, we can also show that the chosen cost-function for the routing algorithm does not affect the reservations significantly.

2) RUNTIME

To show the scalability on a larger network scale, we add a line topology to the previous ring. Each node in the ring is connected to a line of n nodes ($4 \leq n \leq 8$). After the n nodes, a second ring is added to connect all line topologies. This results in a maximum of 80 networking devices, as illustrated in Fig 12(b). A randomly chosen device is working as an industrial programmable logic controller (PLC). All flows either originate or end at the PLC and random devices located in the line topologies. The definitions of flows are



(a) Reserved flows in MDM.

(b) Reserved flows in FSM.

FIGURE 13. Routing effects.

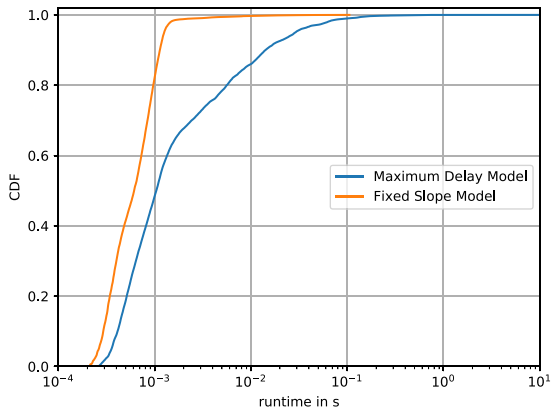


FIGURE 14. Runtime for flow reservation, including routing, priority assignment, and admission control.

the same as in Section VI-A-1. The other parameters are randomly chosen within the following intervals: The queue size is uniformly randomized in [7.71 kB - 380.5 kB]. Each link's capacity is either 1Gbit/s or 10Gbit/s. This ensures that not always the shortest path reservation is possible. For MDM, the per-hop delay for the two priority queues are uniformly distributed in [5μs - 5ms, 5μs - 50ms], for FSM, the slope of the first queue ranges from $x \in [0.1 - 0.65]$ with the second IdleSlope being $0.75-x$. The routing algorithm is chosen randomly for each run. We conducted the simulations on an Intel Core i9-10920X using Python3. We ensured a Single Core use to increase comparability. Fig. 14 shows the runtime distribution function after 130 simulations. As we can see, for FSM, all flows can be reserved in less than 1s with an average of 759.6μs, the maximum at 105.5ms, and an average of 2487 flows reserved in parallel. For MDM, 99.991% flows are below 1s, with an average of 7.14ms and the maximum at 9.84s. On average, 1795 flows are reserved in parallel. Since these results are for single-thread calculations, we can conclude that both our models are applicable in Industry 4.0 scenarios.

C. COMPARISON WITH IEEE 802.1QAT

In the following, we will compare our centralized approach with the decentral reservation of the standard approach (see Section IV-D). As presented in Section II, the standard approach is the only alternative state-of-the-art solution for deadline-guaranteeing online admission control. With the standard covering priority 0 only, we also only consider one queue for the following simulations. In these comparisons, both of our models profit in the same way from the increased configuration flexibility and allow for the same number of reserved flows. Thus, the choice of the model can solely depend on the preferred model characteristics.

With the following network taken from an automotive case study done by Renault [18], we illustrate the effects of the central and decentral configuration when reserving realistic traffic. Table 3 shows the flow settings, adapted from the Renault network in [18]. For this simulation, the flows do not de-register after their reservation. The network combines all three traffic types, HRT, SRT, and NRT. Fig. 12(c) illustrates the topology with 100Mbit/s links. Only the link between *Switch3* and *DM3* has 1Gbit/s.

Thereby, the main limitations of the standard approach are due to its strict sending requirements. The sending interval is always fixed on 125 µs. In contrast, our models allow for arbitrary sending intervals. The small sending interval leads to a significant increase in data traffic, due to the following reasons.

When splitting flow demands over multiple intervals, the standard requires a multitude of headers for each flow when compared to our models. Besides, for flows that require less than a minimum packet size per 125 µs, the standard still requires a reservation of one packet. With a minimum Ethernet frame size of 64B and a sending interval of 125 µs, the minimum reservable flow rate in the standard approach is $>$ 4Mbit/s.

Besides, the number of packets needs to be distributed within the flow's deadline, as already shown in [18]. For illustration, assume an *ADAS Video* stream from Table 3. In

TABLE 3. Renault network flows, adapted from [18].

Type	#flows	#packets	Data	Source	Destination	Interval	Deadline
Audio	8	1	[128 or 256] B / interval	CAM (random)	DM (random)	10ms	10ms
Video	6	30	1446 B / interval	CAM1	DM1	16ms	30ms
				CAM2	DM1		
				CAM3	DM2		
				CAM4	DM2		
				CAM4	Display1		
				CAM4	Display3		
ADAS Video	2	30	1446 B / interval	CAM4	DM3	16ms	10ms
Command&Control	11	1	[256 - 1024] B / interval	ECU (random)	DM (random)	10ms	10ms
File Transfer	14	variable	[1 - 20] MB / s	DM (random)	Display (random)	0.2ms	-

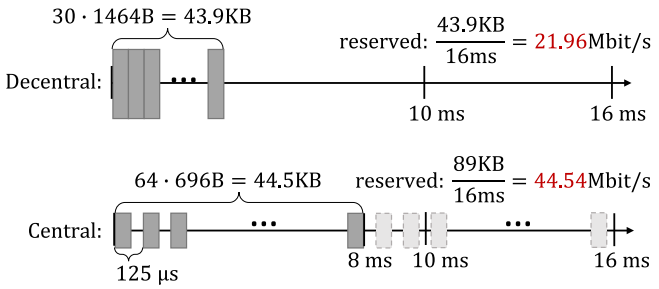


FIGURE 15. Central versus decentral sending scheme. Central: Only the requested data is reserved. Decentral: Data has to be reserved every 125 µs.

the central approach, we reserve the data per flow interval of 16 ms. The shaping will be done on the path due to the CBS shaping queues and our reservations ensure that all buffer sizes are limited. Instead, the decentral approach needs to split the flow demand over 125 µs intervals. When we assume a path delay of 2 ms, the standard approach needs to finish the data transmission after 8 ms, to keep the deadline constraints of the flow. However, for the same video data, this leads to an increased reservation rate, as illustrated in Fig. 15: While only 44.5 kB of actual video data are sent (over 8 ms, including headers), we have to reserve packets in every CMI and, thus, we reserve twice the rate in total.

For the automotive network of Renault, we illustrate the average rates, including all headers, for flows which ask for reservation in Fig. 16(a). In the standard approach, small flows, such as *Audio* and *Comm. & Control* have to reserve a minimum packet size in every CMI, which increases the overall rate significantly. The rates for the *Video* traffic differ in the standard approach, as we have to consider the deadline in their reservation.

To demonstrate the effect on schedulability, Fig. 16(b) shows the number of successful reservations for our central models and the decentral standard approach. We repeated all simulations 100 times and randomized the request order and time of the flows. As we can see, the standard approach is not able to serve the requirements defined by Renault even for this small network due to bandwidth constraints, whereas our models both reliably succeeded in reserving all flows.

Be aware that central approaches can only be compared up to certain limits with decentral approaches, as a decentral approach has different benefits, such as no single point of failure.

VII. SUMMARY

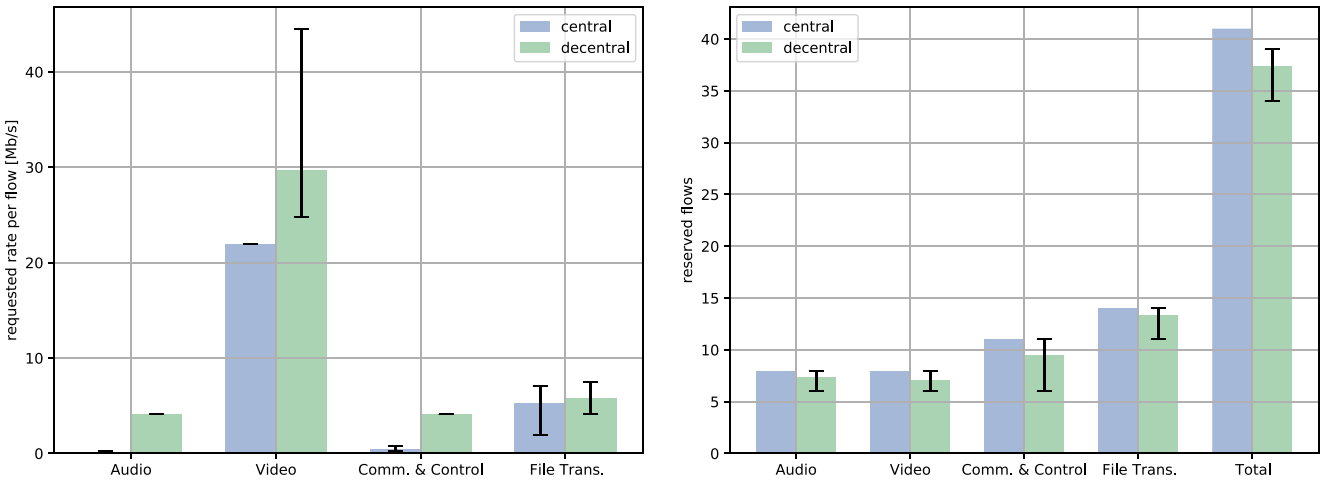
In this paper, we presented admission control schemes for the Central Network Controller in Time-Sensitive Networks, which implement Credit-Based Shaper queues. We combine the priority assignment and routing of flows with finding deadline-guaranteeing configurations for the network.

We introduced a *Fixed Slope Model* and a *Maximum Delay Model* based on Network Calculus. Both models are able to reserve time-critical traffic while assuring safe upper bounds on delay and backlog. While they scale for large industrial networks, their effectiveness depends on the chosen input parameters. We showed that our centralized approaches outperform the existing decentral standard approach. We proposed a priority- and reservation-aware delay-constrained least-cost routing algorithm. Thereby, preferring high priority queues and considering remaining rates is most beneficial for both models. Besides, we are able to utilize the shaping behavior of CBS queues to reduce backlogs in our models, while only updating nodes on a flow's path.

Our *Fixed Slope Model* uses IdleSlopes as input parameters and allows for reservations up to a pre-defined maximum arrival curve. The *Maximum Delay Model* uses per-hop delay values to dynamically adjust the IdleSlopes after each reservation.

In our evaluation, the *Fixed Slope Model* demonstrated lower runtimes, while also being less sensitive to input parameter variations. In comparison, the *Maximum Delay Model* achieves tight IdleSlopes, which improve the service for lower priority queues. These IdleSlopes were further reduced by the introduction of CBS shaping.

In the future, we want to utilize methods such as predictive analysis and machine learning to automatically choose the best set of input parameters. We will also



(a) Average requested rates for all flows from Table 3.

(b) Successful reservations in the network.

FIGURE 16. **Results comparing the decentral and central approach in an internal automotive communication network provided by Renault with minimum and maximum error bars.**

consider probabilistic guarantees, using stochastic Network Calculus, to increase the flexibility of soft real-time traffic. Additionally, we will add more TSN schedulers to the framework to offer higher flexibility for deadline-aware configuration in TSN.

REFERENCES

[1] D. Bruckner et al., "An introduction to OPC UA TSN for industrial communication systems," *Proc. IEEE*, vol. 107, no. 6, pp. 1121–1131, Jun. 2019.

[2] *IEC/IEEE 60802 TSN Profile for Industrial Automation*, IEC/IEEE Standard 60802, Jan. 2019.

[3] "Summary of rel-16 work items (TR21.916)." Dec. 2019. [Online]. Available: https://standards.globalspec.com/std/1283307/ARINC

[4] D. Cavalcanti, "Wireless TSN—Definitions, use cases & standards roadmap," Avnu Alliance, Beaverton, OR, USA, Technische Universität München, Munich, Germany, Rep., Mar. 2020. [Online]. Available: https://avnu.org/download/wireless-tsn-white-paper/

[5] V. Gavriluţ and P. Pop, "Traffic-type assignment for TSN-based mixed-criticality cyber-physical systems," *ACM Trans. Cyber Phys. Syst.*, vol. 4, no. 2, pp. 1–27, Jan. 2020.

[6] I. Álvarez, L. Moutinho, P. Pedreiras, D. Bujosa, J. Proenza, and L. Almeida, "Comparing admission control architectures for real-time Ethernet," *IEEE Access*, vol. 8, pp. 105521–105534, 2020.

[7] *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 25: Enhancements for Scheduled Traffic*, IEEE Standard 802.1Qbv-2015, Mar. 2016.

[8] A. Arestova, K.-S. J. Hielscher, and R. German, "Design of a hybrid genetic algorithm for time-sensitive networking," in *Measurement, Modelling and Evaluation of Computing Systems*, H. Hermanns, Ed. Cham, Switzerland: Springer Int., 2020, pp. 99–117.

[9] S. S. Craciunas, R. S. Oliver, M. Chmelík, and W. Steiner, "Scheduling real-time communication in IEEE 802.1Qbv time sensitive networks," in *Proc. Int. Conf. Real-Time Netw. Syst.*, Oct. 2016, pp. 183–192.

[10] *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks*, IEEE Standard 802.1Q-2014, Dec. 2014.

[11] *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, Jan. 2010.

[12] *IEEE Standard for Local and Metropolitan Area Networks–Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks–Amendment 18: Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Classes*, IEEE Standard 802.1Qaz-2011, Sep. 2011.

[13] *Bridges and Bridged Networks Amendment: Asynchronous Traffic Shaping*, IEEE Standard P802.1Qcr, Sep. 2018.

[14] F. He, L. Zhao, and E. Li, "Impact analysis of flow shaping in Ethernet-AVB/TSN and AFDX from network calculus and simulation perspective," *Sensors*, vol. 17, no. 5, p. 1181, May 2017.

[15] A. Regev and B. Tenea, "Is AVB really much better than classic IEEE 802.1Q queuing?" IXIA, Paris, France, Rep., Sep. 2016.

[16] L. Zhao, P. Pop, and S. Steinhorst, "Quantitative performance comparison of various traffic shapers in time-sensitive networking," *IEEE Trans. Netw. Service Manag.*, early access, Jun. 3, 2022, doi: 10.1109/TNSM.2022.3180160.

[17] *IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)*, IEEE Standard 802.1Qat-2010, Sep. 2010.

[18] J. Migge, J. Villanueva, R. Group, N. Navet, and M. Boyer, "Insights on the performance and configuration of AVB and TSN in automotive Ethernet networks," in *Proc. 9th Eur. Congr. Embedded Real-Time Softw. Syst. (ERTS)*, Tulouse, France, Jan. 2018, p. 10.

[19] *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements*, IEEE Standard 802.1Qcc-2018, Oct. 2018.

[20] D. Thiele and R. Ernst, "Formal analysis based evaluation of software defined networking for time-sensitive Ethernet," in *Proc. Des. Autom. Test Europe Conf. Exhibit. (DATE)*, 2016, pp. 31–36.

[21] S. M. Laursen, P. Pop, and W. Steiner, "Routing optimization of AVB streams in TSN networks," *ACM SIGBED Rev.*, vol. 13, no. 4, pp. 43–48, Nov. 2016.

[22] P. Pop, M. L. Raagaard, S. S. Craciunas, and W. Steiner, "Design optimisation of cyber-physical distributed systems using IEEE time-sensitive networks," *IET Cyber-Phys. Syst. Theory Appl.*, vol. 1, no. 1, pp. 86–94, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/pdf/10.1049/iet-cps.2016.0021

[23] V. Gavriluţ and P. Pop, "Scheduling in time sensitive networks (TSN) for mixed-criticality industrial applications," in *Proc. 14th IEEE Int. Workshop Factory Commun. Syst. (WFCS)*, Jun. 2018, pp. 1–4.

[24] V. Gavriluţ, L. Zhao, M. L. Raagaard, and P. Pop, "AVB-aware routing and scheduling of time-triggered traffic for TSN," *IEEE Access*, vol. 6, pp. 75229–75243, 2018.

[25] E. Li, F. He, L. Zhao, and X. Zhou, "A SDN-based traffic bandwidth allocation method for time sensitive networking in avionics," in *Proc. IEEE/AIAA 38th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2019, pp. 1–7.

[26] C.-C. Chuang, T.-H. Yu, C.-W. Lin, A.-C. Pang, and T.-J. Hsieh, "Online stream-aware routing for TSN-based industrial control systems," in *Proc. 25th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, vol. 1, Sep. 2020, pp. 254–261.
[27] A. Berisa et al., "AVB-aware routing and scheduling for critical traffic in time-sensitive networks with preemption," in *Proc. 30th Int. Conf. Real-Time Netw. Syst.*, Paris France, Jun. 2022, pp. 207–218.
[28] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of AVB traffic in TSN networks using network calculus," in *Proc. Real-Time Embedded Technol. Appl. Symp.*, Apr. 2018, pp. 25–36.
[29] M. L. Raagaard, P. Pop, M. Gutiérrez, and W. Steiner, "Runtime reconfiguration of time-sensitive networking (TSN) schedules for fog computing," in *Proc. IEEE Fog World Congr. (FWC)*, Oct. 2017, pp. 1–6.
[30] *IEEE Standard for Local and Metropolitan Area Networks–Audio Video Bridging (AVB) Systems*, IEEE Standard 802.1BA-2021, Dec. 2021.
[31] J. W. Guck, A. Van Bemten, and W. Kellerer, "DetServ: Network models for real-time QoS provisioning in SDN-based industrial environments," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 4, pp. 1003–1017, Dec. 2017.
[32] R. Queck, "Analysis of Ethernet AVB for automotive networks using network calculus," in *Proc. IEEE Int. Conf. Veh. Electron. Saf. (ICVES)*, Jul. 2012, pp. 61–67.
[33] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, and M. Boyer, "Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus," *IEEE Trans. Ind. Electron.*, vol. 68, no. 10, pp. 10291–10302, Oct. 2021.
[34] L. Maile, K.-S. Hielscher, and R. German, "Network calculus results for TSN: An introduction," in *Proc. Inf. Commun. Technol. Conf. (ICTC)*, Nanjing, China, May 2020, pp. 131–140.
[35] S. M. Rumble, D. Ongaro, R. Stutsman, M. Rosenblum, and J. K. Ousterhout, "It's time for low latency," in *Proc. 13th Workshop Hot Topics Oper. Syst. (HotOS XIII)*, 2011, p. 11.
[36] S. B. H. Said, Q. H. Truong, and M. Boc, "SDN-based configuration solution for IEEE 802.1 time sensitive networking (TSN)," *ACM SIGBED Rev.*, vol. 16, no. 1, pp. 27–32, Feb. 2019.
[37] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Berlin, Germany: Springer-Verlag, 2001.
[38] W. Kellerer and A. Van Bemten, "Network calculus: A comprehensive guide," Dept. Chair Commun. Netw., Technische Universität München, Munich, Germany, Rep. 201603, Oct. 2016.
[39] J. A. R. De Azua and M. Boyer, "Complete modelling of AVB in network calculus framework," in *Proc. 22nd Int. Conf. Real-Time Netw. Syst.*, Versaille, France, Oct. 2014, pp. 55–64.
[40] L. Zhao, P. H. M. Pop, Z. Zheng, H. Daigmorte, and M. Boyer. "Improving worst-case end-to-end delay analysis of multiple classes of AVB traffic in TSN networks using network calculus." 2019. [Online]. Available: https://www.semanticscholar.org/paper/d7bb919d5a525afedf1958cb890918c326a775fd
[41] H. Daigmorte, M. Boyer, and L. Zhao. "Modelling in network calculus a TSN architecture mixing time-triggered, credit based shaper and best-effort queues." Jun. 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01814211
[42] E. Li, F. He, Q. Li, and H. Xiong, "Bandwidth allocation of stream-reservation traffic in TSN," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 1, pp. 741–755, Mar. 2022.
[43] A. Nasrallah et al., "Ultra-low latency (ULL) networks: The IEEE TSN and IETF DetNet standards and related 5G ULL research," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 88–145, 1st Quart., 2019.
[44] *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing*, IEEE Standard 802.1Qci-2017, Sep. 2017.
[45] J. Y. Yen, "Finding the K shortest loopless paths in a network," *Manag. Sci.*, vol. 17, no. 11, pp. 712–716, 1971.

LISA MAILE (Graduate Student Member, IEEE) received the master's degree in computer science from University Ulm in 2018. She is currently pursuing the Doctoral degree with the Department for Computer Networks and Communication Systems, University of Erlangen-Nürnberg, where she is also a Research Associate. Her research is focused on network calculus analysis and simulation of time-sensitive networks. Further research interests include topics of privacy and security in communication networks.

KAI-STEFFEN J. HIELSCHER was born in Münchberg, Germany, in 1972. He received the Ph.D. degree in computer science from the University of Erlangen-Nürnberg in 2008, where he is currently working as a Postdoctoral Researcher and a Research Group Leader with the Department of Computer Science (Computer Networks and Communication Systems). His focus of research includes measurement, modeling, and simulation of distributed systems as well as deterministic network calculus.

REINHARD GERMAN received the master's degree in computer science and the Ph.D. degree from the Computer Science Department, Technical University of Berlin, Germany, in 1991 and 1994, respectively. He is a Full Professor with the Computer Networks Lab, Department of Computer Science, University of Erlangen-Nürnberg, Germany. He is also an Adjunct Professor with the Faculty of Information Technology, Monash University, Melbourne, Australia. His research interests include performance and dependability analysis of interconnected systems based on numerical analysis, network calculus, discrete-event simulation, measurements, testing, vehicular communications, and smart energy constitute major application domains.