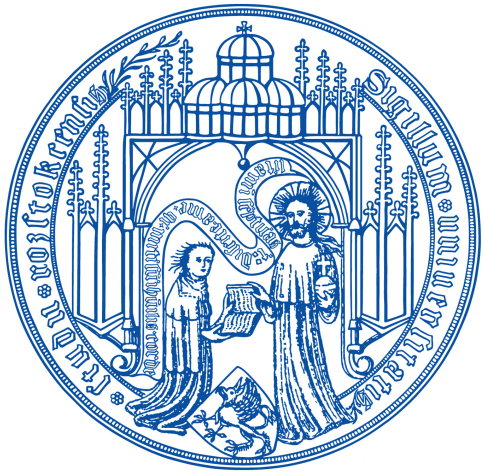
Technical Report

On the Analysis and Simulation of Converged Data Traffic in Time-Sensitive Networks

Julian Oertel, Helge Parzyjegla, and Peter Danielis

University of Rostock
Institute of Computer Science



December 16, 2025

https://doi.org/10.18453/rosdok_id00004769

Analysis and Simulation of Converged Data Traffic in Time-Sensitive Networks

Julian Oertel, Helge Parzyjegla, and Peter Danielis
Institute of Computer Science, University of Rostock, Germany
{julian.oertel, helge.parzyjegla, peter.danielis}@uni-rostock.de

Abstract

Time-Sensitive Networking (TSN) is a key technology for converged industrial communication systems enabling the integration of traffic with diverse Quality of Service (QoS) requirements such as timeliness, throughput, and reliability. To ensure that all data streams meet their requirements, the Network Calculus (NC) allows to analytically derive worst-case performance estimations. In this paper, we model rate-constrained streams alongside time-triggered hard real-time traffic in a line topology. Additionally, we compare the analytical results with simulations conducted in the OMNeT++ network simulator. Our comparison shows that the worst-case delay predicted by the NC is 3 to 4 times higher than the delay observed in simulations. Similarly, the estimated backlog (i.e., the amount of data queued in TSN switches) is 1.25 to 2.8 times larger than the simulation results. A further experiment indicates that NC predictions are more accurate for less complex network configurations.

1 Introduction

The Industrial Internet of Things (IIoT) and Industry 4.0 rely on converged networks to support diverse communication requirements in industrial automation [1]. *Time-Sensitive Networking* (TSN), an extension of the Ethernet standard, plays a crucial role in enabling deterministic communication by providing mechanisms for bounded latency and high reliability [2]. TSN accommodates different traffic classes with varying requirements, including time-triggered traffic using gate control lists (GCLs) and time-aware shapers (TAS), and rate-constrained traffic managed by credit-based shapers (CBS) and used for audio/video bridging (AVB). Ensuring that critical quality-of-service (QoS) requirements are met is essential in such networks. A common approach for worst-case performance evaluation is network calculus (NC), which provides analytical bounds on latency and buffer usage [3,4]. However, the accuracy of these worst-case estimates remains an open question. To assess their validity, a comparative analysis with simulation results is necessary. Despite the importance of this comparison, there is a lack of reproducible and transparent experimental studies in the literature that systematically evaluate NC against simulation. This paper addresses this gap by systematically comparing NC worst-case analysis with simulation results, leveraging the open-source OMNeT++ framework and the INET model suite to provide a reproducible evaluation. By assessing

the accuracy and applicability of NC in estimating worst-case performance metrics, this paper offers new insights into its practical strengths and limitations for TSN performance evaluation.

The remainder of this paper is structured as follows: In Section 2, we provide a brief introduction to NC and the simulation software OMNeT++. Section 3 desribes the modeling approach using NC, while Section 4 outlines the simulation setup in OMNeT++. Our results are presented in Section 5. Section 6 discusses related work. We conclude in Section 7.

2 Background

This section provides an overview of NC as a worst-case analysis framework for TSN and introduces OMNeT++ as a simulation tool for evaluating network performance. Understanding both methods is essential for assessing their applicability and accuracy in TSN analysis.

2.1 Network Calculus

NC is a system theory for communication networks [4]. It is based on min-plus algebra and max-plus algebra and allows to calculate worst-case performance of communication networks [4]. For calculation, data flows are modeled by so called *arrival curves* which provide upper bounds on the burstiness and data rate of each flow arriving at a network device. Additionally, *service curves* represent lower bounds for the performance of network devices. Combined, they can be used to calculate worst-case bounds for the delay that a data frame experiences when passing through a network device and the maximum amount of frames that await sending in a queue of a switch (backlog). This worst-case calculation can be further improved by taking into account a credit-based shaper's influence on the maximum output of a switch which is modelled by *shaper curves*. This output of a switch is modelled by an *output arrival curve* which serves as input for subsequent network devices and ultimately enables to calculate a worst-case performance estimate for a whole network.

2.2 OMNeT++

OMNeT++[1] is an open-source event-based simulation framework used for analyzing communication networks and protocols. Due to its modularization and extendability, OMNeT++ allows users to define custom modules interacting alongside a library of predefined components. This library can be further extended by using INET[2], which includes additional components for the internet stack. Since it also provides TSN features such as periodic gates and credit-based shaping, we use INET for our simulation.

[1] https://omnetpp.org/
[2] https://inet.omnetpp.org/

3 Modeling Time-triggered and Rate-constrained Traffic with NC

In this section, we present the modeling of time-triggered and rate-constrained traffic using network calculus. The modeling was done by Zhao et al. [5].

3.1 Notation

We use the following notation for modeling the needed functions based on [6] and [5]:

T_{GCL}	Hyperperiod, defined as the least common multiple of all GCL periods of every port and class.
C	Maximum link rate.
I^X	Idle slope of credit-based shaper of class x.
S^X	Send slope of credit-based shaper of class x. $S^X = I^X - C$
$L_i^{X,h}$	Time slot in which a queue of class x at port h has guaranteed service.
$o_i^{X,h}$	Starting time of $L_i^{X,h}$.
$o_{i,j}^{X,h}$	Duration of time between two time slots of guaranteed service: $o_{j,i}^X = o_j^X - o_i^X$
$[f(t)]_\uparrow^+$	Denotes the non-negative non-decreasing closure of a function: $[f(t)]_\uparrow^+ = \max_{0 \leq s \leq t} \{f(s), 0\}$

We omit the port h for idle slopes and send slopes as we keep both parameters at the same value for each port.

3.2 Arrival Curve

Depending on the specific properties of a given stream, there are different possibilities for modeling data streams as arrival curves in network calculus. In this paper, we model a stream k of traffic class X at an outgoing port h and time $t > 0$ as

$$\alpha_k^{X,h}(t) = l_{max} + rt \tag{1}$$

Therein, l_{max} refers to the maximum length of a data frame and r is the required data rate. An example of an arrival curve can be seen in Figure 1. If multiple streams arrive at the same outgoing port h of the same class X, they need to be aggregated accordingly. For this, we calculate the sum of their respective arrival curves

$$\alpha^{X,h}(t) = \sum_{\forall k} \alpha_k^{X,h}(t) \tag{2}$$

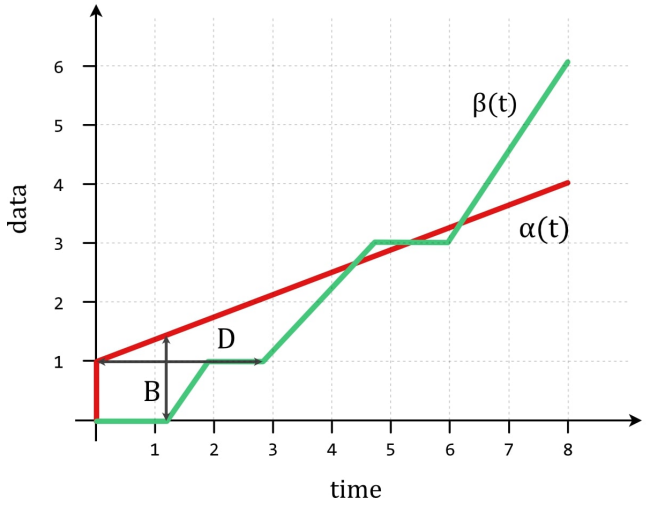


Figure 1: Examples for an arrival curve $\alpha(t)$, a service curve $\beta(t)$, and the resulting delay D and backlog B.

3.3 Credit Bounds

A credit-based shaper accumulates credit based on an idle slope when waiting for service and depletes credit based on a send slope when receiving service. A frame can be transmitted when the credit is at least 0 which leads to a minimum credit of

$$c_{min}^{X} = \frac{l_{max}^{X}}{C} \cdot S^{X} \tag{3}$$

where $\frac{l_{max}^{X}}{C}$ represents the time it takes to transmit a frame of maximum length [5]. The maximum credit depends on all queues of higher prioritized traffic classes and the maximum length of a frame of all lower prioritized classes. In the worst case, service for a queue is delayed due to the transmission of a frame with maximum length of a lower prioritized queue and the subsequent transmissions of each higher prioritized queue. According to [5] the maximum credit is then calculated as

$$c_{max}^{X} = I^{X} \cdot \frac{\sum_{i=1}^{X-1} c_{min}^{i} - l_{max}^{>X}}{\sum_{i=1}^{X-1} I^{i} - C} \tag{4}$$

3.4 Service Curve

The service curve for rate-constrained traffic in our model is shaped by the time slots that are reserved for time-triggered traffic and time slots reserved for traffic of higher prioritized queues. According to Zhao et al. [5], the service curve is calculated by

$$\beta_{CBS}^{X,h}(t) = I^{X} \cdot \left[t - z_{TT+GB}^{X,h}(t) - \frac{c_{max}^{X}}{I^{X}} \right]_{\uparrow}^{+} \tag{5}$$

with

$$z^{X,h}_{TT+GB}(t) = \max_{0 \leq i \leq N^{TT,h}-1} \left\{ \sum_{j=i}^{i+N^{TT,h}-1} (L^{TT,h}_j + L^{X,h}_{j,GB}) \cdot \left\lceil \frac{t - o^h_{j,i} + L^{X,h}_{j,GB} - L^{X,h}_{i,GB}}{T_{GCL}} \right\rceil \right\} \tag{6}$$

$\frac{c^X_{max}}{I^X}$ Equation (5) denotes the amount of time that is needed to reach the maximum credit. Similarly, $z^{X,h}_{TT+GB}(t)$ denotes the duration of time that the rate-constrained queue does not receive service due to time-triggered traffic and associated guard bands. It is derived from the notion, that the minimum service provided up to a time t can be determined by a cyclic shift of time slots with guaranteed service in order to find the maximum duration without service. It therefore assumes the worst possible starting point in the periodic opening and closing of gates. An example of a service curve is provided in Figure 1.

3.5 Shaper Curve

To improve the worst-case estimation, we utilize shaper curves to act as an upper bound for the output arrival curves. According to Zhao et al. [5], the shaper curve of a credit-based shaper is calculated with

$$\sigma^{X,h}(t) = \left[t - \frac{\beta_{TT}(t)}{C} \right]^+_{\uparrow} \cdot I_x + c^X_{max} - c^X_{min} \tag{7}$$

$\frac{\beta_{TT}(t)}{C}$ is the duration in which time-triggered traffic is serviced exclusively and $c^X_{max} - c^X_{min}$ is the maximum amount of data that can be sent at a time. The service curve for time-triggered traffic is then calculated with

$$\beta^h_{TT}(t) = \min_{0 \leq i \leq N^h_{TT}-1} \left\{ \sum_{j=i}^{i+N^h_{TT}-1} \beta^h_{TDMA}\left(t + t_0, L^h_{TT,j}\right) \right\} \tag{8}$$

where

$$\beta^h_{TDMA}\left(\Delta t, L^h_{TT}\right) = C \cdot \max \left\{ \left\lfloor \frac{\Delta t}{T^h_{GCL}} \right\rfloor L^h_{TT}, \Delta t - \left\lceil \frac{\Delta t}{T^h_{GCL}} \right\rceil \cdot \left(T^h_{GCL} - L^h_{TT}\right) \right\}$$

and

$$t_0 = T^h_{GCL} - L^h_{TT_j} - o^h_{0,i} - o^h_{j,i}$$

More details on the calculation of the shaper curve are provided by Zhao et al. [5], and Wandeler and Thiele [7]. In Figure 2, we show an example of a shaper curve.

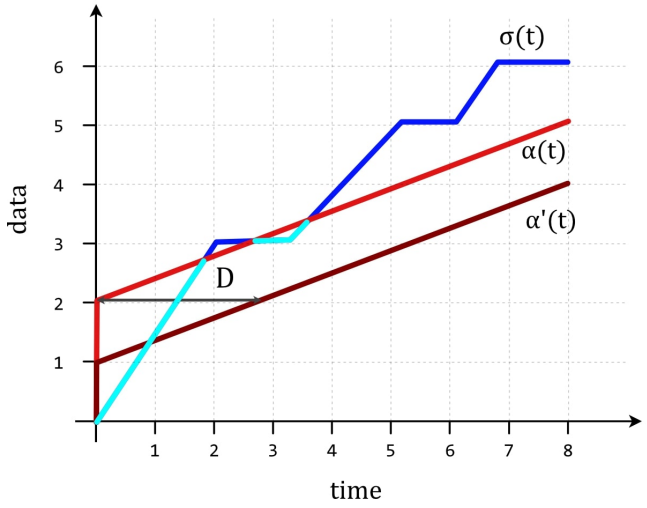


Figure 2: Image showing the relationship of input arrival curve $\alpha'(t)$, output arrival curve $\alpha(t)$, delay D, and shaper curve $\sigma(t)$. The final output arrival curve is the minimum of $\sigma(t)$ and output arrival curve $\alpha(t)$.

3.6 Delay Bound and Backlog Bound

The maximum delay that a stream experiences at one switch is equal to the horizontal distance between the (aggregated) arrival curve of the stream and the service curve of the switch at a port h.

$$D_k^{X,h} = h(\alpha^{X,h}(t), \beta^{X,h}(t)) \tag{9}$$

The overall delay bound for one stream along a route is calculated by summing individual delays at each switch

$$D^X = \sum_{h \in dr_k} D^{X,h} + (|dr_k| - 1) \cdot d_{tech} \tag{10}$$

where dr_k represents an ordered set of all nodes of a route for stream k, $(|dr_k|-1)$ denotes the number of hops and d_{tech} represents a constant delay that is identical for each switch. Similarly, the backlog for one queue of traffic class X at port h is equal to the vertical distance between arrival curve and service curve.

$$B^{X,h} = v(\alpha^{X,h}(t), \beta^{X,h}(t)) \tag{11}$$

3.7 Output Arrival Curve

The output arrival curve is the maximum output of a switch and is used as input for subsequent network devices. According to Zhao et al. [5] it is given by

$$\alpha_k^{X,h}(t) = \alpha_k^{X,h'}(t + D_k^{X,h'}) \tag{12}$$

This represents a left shift of the initial arrival curve at port h' which can be seen as every bit being delayed by the maximum delay. This shift is illustrated

```
network TsnLinearNetwork extends TsnNetworkBase
{
    parameters:
        @display("bgb=858,816");
    submodules:
        client: TsnDevice
        switch: TsnSwitch
        server: TsnDevice
    connections:
        client.ethg++ <--> Eth1G <--> switch.ethg++;
        switch.ethg++ <--> Eth1G <--> server.ethg++;
```

Listing 1: Simple network description in NED.

in Figure 2. The shaper curve of the credit-based shaper and the shaper curve of the physical link provide additional bounds on the output arrival curve. Additionally, arrival curves of streams that cross the same ports h' and h have to be aggregated at port h. Therefore,

$$\alpha_{k,h'}^{X,h}(t) = \left(\sum_{\forall dr_k : h', h \in dr_k} \alpha_k^{X,h}(t) \right) \wedge \left(\sigma_{link}(t) + l_{max,k}^{X,h} \right) \wedge \left(\sigma^{X,h'}(t) + l_{max,k}^{X,h} \right) \tag{13}$$

For a given set of streams, we can now calculate all service curves and shaper curves at each port of each switch. With the aggregation of arrival curves and the calculation of output arrival curves, we are able to traverse the network along the streams and estimate delay bounds and backlog bounds.

4 Simulation Setup in OMNeT++

This section describes the simulation setup in OMNeT++, focusing on the implementation of GCLs for time-triggered traffic, the CBS for rate-constrained traffic, and the configuration of different traffic streams in the network.

4.1 Gate Control Lists

GCLs can be defined in INET at each `transmissionGate` of a switch. The `durations` parameter specifies a list of times during which the gate is closed or open. In addition, the parameter `offset` can be used to shift the start of the durations and `initiallyOpen` can be used to specify whether the first duration is to be considered open or closed. The sum of all times in a list in `durations` then forms the period of the GCL.

In principle, there is no limit to the number of queues, so any number of queues can be specified using the `numTrafficClasses` parameter. The priorities for the queues are assigned by the index of the queue. Note that in INET a lower index means a higher priority.

For the simulation, time-triggered traffic should alternate exclusively with rate-constrained traffic, i.e., either the gates of the TAS or those of the CBS should be open at all times. This can be achieved by using the same GCL for all gates of a port and setting the `initiallyOpen` parameter for the CBS queues to `false`.

```
1  *.client.numApps = 1
2  *.client.app[0].typename = "UdpSourceApp"
3  *.client.app[0].display-name = "video"
4  *.client.app[0].io.destAddress = "server"
5  *.client.app[0].io.destPort = 1000
6  *.client.app[0].source.packetLength = 1500B - 58B
7  *.client.app[0].source.productionInterval = 5ms
8
9  *.server.numApps = 1
10 *.server.app[0].typename = "UdpSinkApp"
11 *.server.app[0].display-name = "video"
12 *.n15.server[0].io.localPort = 1000
13
14 *.switch.bridging.streamCoder.encoder.mapping =
15                     [{stream: "video", pcp: 2}]
```

Listing 2: Configuration of a stream in a network in INET

4.2 Credit-Based Shaper

We configure the CBS by assigning the type `Ieee8021qCreditBasedShaper` to a gate and then setting the parameter `idleSlope`. It is also possible to set the behaviour in guard bands using the value of `accumulateCreditInGuardBand` - an extension to INET version 4.5 that was developed with the INET community as part of our work.

4.3 Streams

Streams can be created in INET as `UdpSourceApp` on a terminal. Listing 2 shows how a stream named "video" can be placed through the topology shown in Listing 1

First, the number of applications to run, their names and types are specified for a client. The `destAddress` and `destPort` specify the destination of the stream. The `packetLength` can be used to specify the packet length of the packets as generated by the application. Note that various headers are added to the packet as it travels up the protocol stack to the bit transmission layer, so the packet on the line 58 B will be larger than previously specified. When calculating the credit in the CBS, the actual length of the packet on the line is used plus the length of the interpacket gap of 12 B, which specifies the minimum distance between two packets [8, p. 282]. The interval between packets generated by the application is specified by the parameter `productionInterval`, which together with the packet length defines the initial arrival curve of this stream. The parameter `initialProductionOffset` specifies when the periodic production of packets should start.

In addition to the client application, there is also an application on the server that receives the packets (line 10). The `UdpSinkApp` (line 11) is also given a name (line 12) and a port (line 13) where the packets are expected. In the switch, `streamCoder.encoder.mapping` in line 15 must also be used to specify which stream is to be assigned which priority. In the example, the "video" stream is given the priority 2, where a higher value of priority code point (PCP) implies a higher priority.

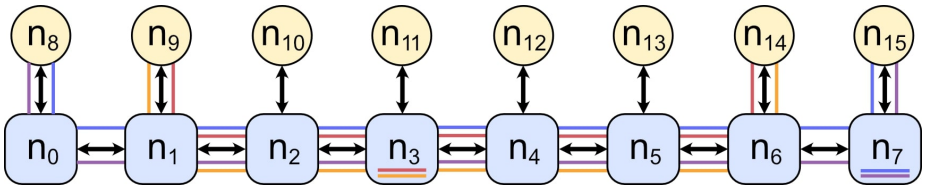


Figure 3: Topology and used stream set as well as marked switches where the backlog is calculated/measured.

4.4 Recorded Measurements

OMNeT++ offers to record various values of a simulation that have been marked for this purpose in the source code. The values required for comparison between simulation and NC are already recorded in INET and can therefore be retrieved directly from the result files. The `meanBitLifeTimePerPacket` parameter is used for the delay that occurs for a stream, which is the total time a packet takes to travel from source to destination. This includes any delay caused by queuing, the switch, or the transmission itself. As the value is recorded for each packet and the delay varies depending on the gate states, only the maximum value should be included for comparison with the NC. The result of the value `queueBitLength` is used as a measure of the size of the backlog created per queue at the port of a switch. This indicates how much data remains in a queue at different times. Again, only the maximum backlog is analyzed.

5 Results

In this section, we first describe experimental setup before presenting the results of the experiments. With this report, we also provide a software artifact containing the implementation of the network calculus and the simulation setup [9].

5.1 Experimental Setup

For our experiment, we chose to evaluate a simple line topology depicted in Figure 3 with four data streams representing rate-constrained traffic. In OMNeT++, we use a simulation time of 20 ms. The setup consists of eight switches (n_0 to n_7) and eight end stations (n_8 to n_{15}). We further define a link rate of 1 Gbit The streams vary by payload size, source/destination, and priority. We define two longer streams, passing through all switches where one has a high priority and a payload size of 786 B (blue), and one has a lower priority and a payload size of 1442 B (purple). Additionally, we define two shorter streams routed from n_9 to n_{14} where one has a high priority and payload size of 1442 B (yellow), and one has a lower priority and a payload size of 786 B.

To represent time-triggered traffic, we require a schedule representing the time slots that are reserved for this traffic. For this, we use schedules calculated by Schweissguth et al. [10] for the same line topology. The chosen schedule specifies a hyper period of $T_{GCL} = 610\,\mu s$. In each period, 150 µs are reserved for time-triggered traffic which equates to 24.6 % of the time.

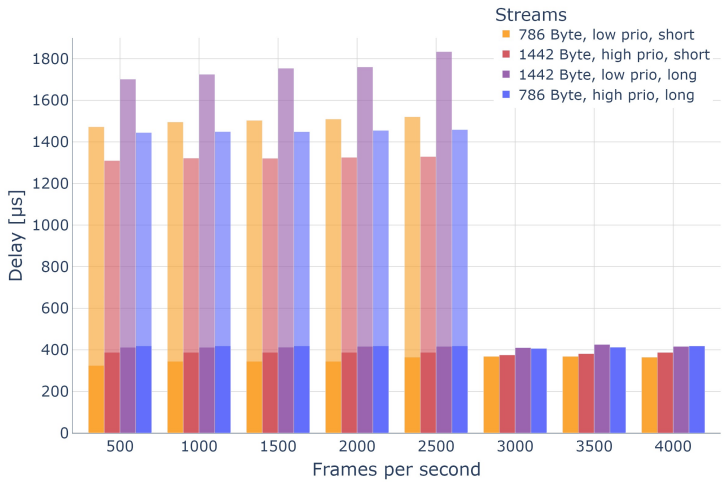


Figure 4: Calculated maximum delays of streams (transparent) in comparison to simulation results (opaque).

For our experiment, we want to observe, how the amount of data traffic influences the worst-case estimates of the network calculus in relation to the results of the simulation. We therefore subsequently increase the number of frames that are sent by each stream per second, starting from 500 FPS stepwise increased by 500 FPS up to 4000 FPS.

5.2 Evaluation

In this section, we present the results of the network calculus and the simulation. We evaluate the worst-case estimates and compare the described scenario to a scenario with shorter streams.

5.2.1 Line

The results for the delay are found in Figure 4. The figure shows, that the calculated worst-case delay (transparent) is bigger than the maximum observed delay (opaque). In fact, depending on the specific stream and the data traffic, the calculated delay is three to four times bigger than the observed delay. The calculated worst-case delay ranges from 1171 µs to 1684 µs. The observed delay ranges from 324 µs to 418 µs. The estimate for the lower priority streams is consistently around four times bigger than the observed value, while the estimate for the higher priority streams is only around three times bigger. Moreover, there is a slight increase of this difference between simulation and network calculus for all streams except the yellow stream for more data traffic. Note that it was not possible to calculate delay estimates for 3000 FPS and upward because a service curve at one queue did not catch up to the arrival curve at the same queue, thus resulting in more incoming traffic than outgoing traffic in each period. With this, it is not possible to calculate a delay bound, as it increases with each period. An interesting observation is that the relationship between calculated delay and observed delay is inconsistent for our experiment, i.e. a larger calculated delay

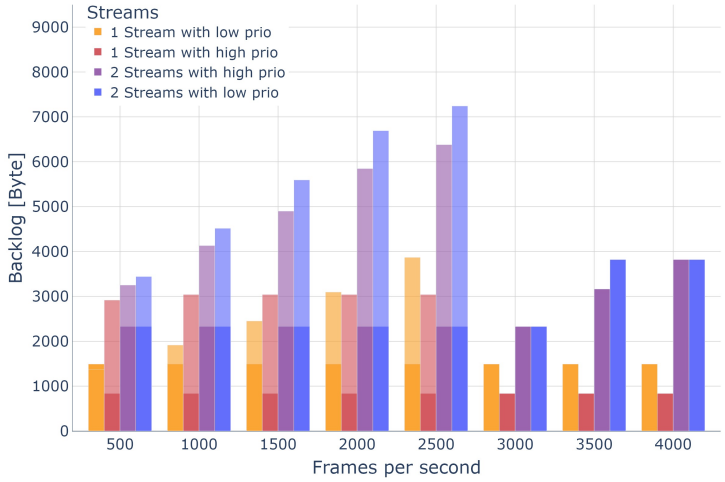


Figure 5: Calculated maximum backlog (transparent) at n_3 and n_7 in comparison to simulation results (opaque). Note that the calculated backlog for one low priority stream is actually smaller than the observed backlog of the simulation. This is the consequence of the fluid model we used in our model, which assumes a continues flow of data in contrast to the packet based measurement of the simulation.

does not necessitate a larger observed delay and vice versa. However, this might also be caused by the limited size of the experiment, and might be different for larger setups or a longer simulation time.

For the backlog the results of network calculus and simulation are more closely aligned which is illustrated in Figure 5. In contrast to the delay, the calculated backlog is mostly two to three times bigger than the simulation and the relationship of a higher calculated backlog corresponding to a higher observed backlog seems to hold true more often. The calculated backlog ranges from 1351 B to 6542 B while the observed backlog ranges from 836 B to 3820 B. Especially for one stream with low priority the calculated backlog is closely aligned to the observed backlog ranging from being 1.25 times to 2.45 times larger (excluding the data from 500 FPS). Again, we were not able to calculate values for the backlog for 3000 FPS and upwards, as the respective service curve and arrival curve diverged. However, it is still visible, that the calculated backlog and observed backlog increase with increasing data traffic.

5.2.2 Short Line

To find out how the worst-case analysis behaves for less complex setups, we replicated the previously described experiment with shorter streams. This change is depicted in Figure 6. Except for the destination of the streams, we keep all other parameters consistent. We now record the backlog at n_1 and n_3. Figure 7 shows the results for the delay. The calculated delay is about 1.7 to 2.6 times larger than the observed delay an thus the difference between the two is smaller in this setup. Moreover, there is almost no increase in the calculated delay with increasing data traffic. This increase is also small for the observed delay.

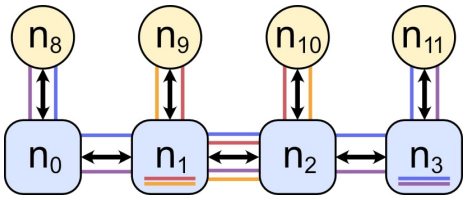


Figure 6: Topology and stream set with shorter streams and marked switches for backlog calculation/measurement.

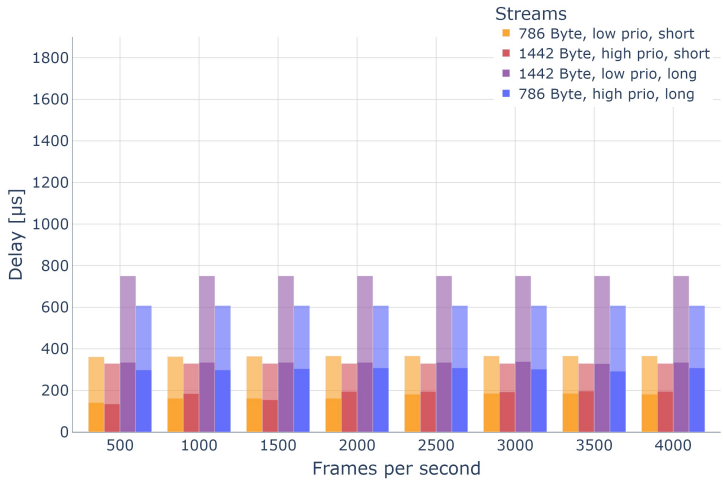


Figure 7: Calculated maximum delays of shorter streams (transparent) in comparison to simulation results (opaque).

For the backlog, the results are shown in Figure 8. Note, that this figure cannot be compared directly to the results of the first setup, as we measure the backlog at different switches. The calculated backlog ranges from being 1.1 times to 2.8 times larger than the observed value. Even though the maximum difference between the two is larger than in the first setup, almost all of the calculated backlog values are only up to 2 times larger than the observed values. Moreover, as we were able to calculate values for all data traffic scenarios, we observe this larger gap due to the notion that the worst-case estimate tends to get worse with more data traffic.

In conclusion, the gap between calculated bounds and observed delay and backlog seems to get smaller for less complex setups.

6 Related Work

Several research groups have investigated the modeling of rate-constrained traffic in TSN. In [11], fundamental considerations for the CBS with two AVB queues are presented, introducing NC for worst-case delay analysis. The study derives equations for minimal and maximal credit, as well as the minimum service curve and arrival curve, and applies them to automotive camera and in-

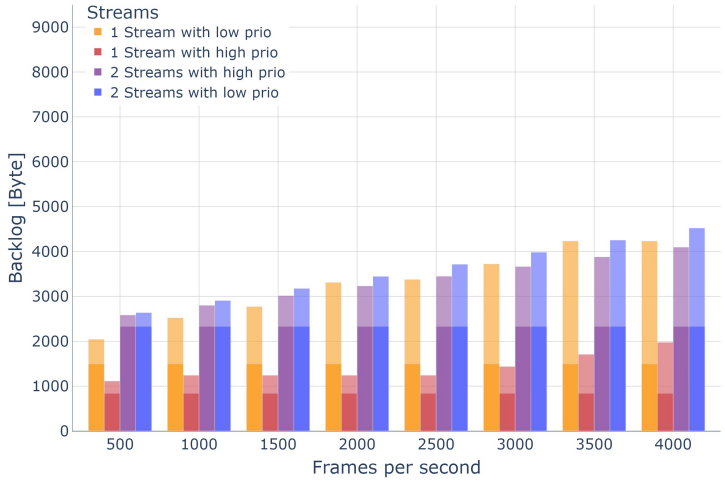


Figure 8: Calculated maximum backlog (transparent) at n_1 and n_3 in comparison to simulation results (opaque) for shorter streams.

fotainment systems. Building on this, [12] extends the equations by introducing the strict service curve, maximum service curve, and shaper curve, leading to tighter delay bounds.

Time-triggered traffic in TSN is examined in [13], where a general NC model is proposed without imposing specific conditions on GCLs or streams. The study derives equations for the minimum service curve, output arrival curve, and delay bound. The integration of AVB and time-triggered traffic is analyzed in [14], considering two AVB queues and deriving service curves while accounting for guard bands and frame preemption. [15] extends these results to multiple AVB queues, introducing new equations for credit bounds and the shaper curve, which refines delay estimates.

The impact of credit behavior in guard bands is highlighted in [16], showing that credit accumulation, as specified in [17], can lead to unfair advantages for higher-priority AVB queues or even unlimited credit growth. While previous studies assumed frozen credit in guard bands, [5] models both approaches. However, only limited work explores the simulation of TSN networks using OMNeT++ for both time-triggered as well as rate-constrained traffic.

Although [13], [15], and [5] mention simulations to validate NC results, they do not provide detailed descriptions. A proof-of-concept extension of OMNeT++ and INET enabling the simulation of time-triggered and rate-constrained traffic, with or without frame preemption, is introduced in [18], followed by a performance analysis across different switch and stream configurations. A TTEthernet simulation model for OMNeT++ and INET is developed in [19], comparing simulated results with mathematical models and real-world measurements, achieving high accuracy. [20] evaluates the potential of Automotive Ethernet with TSN to replace existing in-vehicle network technologies, using CoRE4INET to compare different shaping mechanisms, including Strict Priority, CBS, and Asynchronous Traffic Shaping.

To the best of our knowledge, no work has systematically compared NC

results with simulation results while considering the different traffic classes of TSN.

7 Conclusions

In this paper, we used the NC to model converged network traffic in a TSN infrastructure including rate-constrained and time-triggered data streams. Additionally, we developed a simulation model for OMNeT++ to replicate these setups in a simulation environment. Together, these models enable comparisons between analytical worst-case estimations and simulation results for various network configurations.

Our experiments show that NC estimations are more pessimistic in complex setups involving longer streams with higher data volumes. In the first setup, the worst-case delay derived by the NC is three to four times higher than the simulation results, while the backlog is 1.25 to 2.45 times larger. In the second setup with shorter streams, the delay is 1.7 to 2.6 times higher and the backlog is 1.1 to 2.8 times larger than the simulation results.

For future work, we aim to explore additional scenarios with varying stream sets and network topologies to further investigate advantages and limitations of our NC modeling approach.

References

[1] D. Z. Lou, J. Holler, C. Whitehead, S. Germanos, M. Hilgner, and W. Qiu, "Industrial networking enabling iiot communication," 2018. [Online]. Available: https://www.iiconsortium.org/pdf/Industrial_Networking_Enabling_IIoT_Communication_2018_08_29.pdf

[2] L. Silva, P. Pedreiras, P. Fonseca, and L. Almeida, "On the adequacy of sdn and tsn for industry 4.0," in *2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*, 2019, pp. 43–51.

[3] J.-Y. L. Boudec and P. Thiran, *Network Calculus*. LNCS, 2001.

[4] A. V. Bemten and W. Kellerer, "Network calculus: A comprehensive guide."

[5] 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 Transactions on Industrial Electronics*, vol. 68, pp. 10 291–10 302, 2021.

[6] L. Maile, K.-S. Hielscher, and R. German, "Network calculus results for tsn: An introduction." IEEE, 2020, pp. 131–140.

[7] E. Wandeler and L. Thiele, "Optimal tdma time slot and cycle length allocation for hard real-time systems." IEEE, 2006, pp. 6 pp.–.

[8] IEEE, "Ieee standard for ethernet," *IEEE Std 802.3-2022 (Revision of IEEE Std 802.3-2018)*, pp. 1–7025, 2022.

[9] J. Oertel, H. Parzyjegla, and P. Danielis, "Software artifact for "analysis and simulation of converged data traffic in time-sensitive networks"," Feb. 2025. [Online]. Available: https://doi.org/10.5281/zenodo.14880055

[10] E. Schweissguth, H. Parzyjegla, P. Danielis, G. Mühl, D. Timmermann, S. Mehner, O. Hohlfeld, D. Hellmanns, and J. Falk, "TSN scheduler benchmarking," in *19th IEEE International Conference on Factory Communication Systems (WFCS 2023)*. IEEE, Apr. 2023.

[11] R. Queck, "Analysis of ethernet avb for automotive networks using network calculus," in *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)*, 2012, pp. 61–67.

[12] J. A. R. D. Azua and M. Boyer, "Complete modelling of avb in network calculus framework," in *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*, ser. RTNS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 55–64.

[13] L. Zhao, P. Pop, and S. S. Craciunas, "Worst-case latency analysis for ieee 802.1qbv time sensitive networks using network calculus," *IEEE Access*, vol. 6, pp. 41 803–41 815, 2018.

[14] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of avb traffic in tsn networks using network calculus," in *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2018, pp. 25–36.

[15] L. Zhao, P. 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," 2018.

[16] M. Boyer and H. Daigmorte, "Impact on credit freeze before gate closing in cbs and gcl integration into tsn," in *Proceedings of the 27th International Conference on Real-Time Networks and Systems*, ser. RTNS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 80–89.

[17] IEEE, "Ieee standard for local and metropolitan area network–bridges and bridged networks - redline," *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014) - Redline*, pp. 1–3654, 2018.

[18] J. Falk, D. Hellmanns, B. Carabelli, N. Nayak, F. Dürr, S. Kehrer, and K. Rothermel, "Nesting: Simulating ieee time-sensitive networking (tsn) in omnet++," in *2019 International Conference on Networked Systems (NetSys)*, 2019, pp. 1–8.

[19] T. Steinbach, H. D. Kenfack, F. Korf, and T. C. Schmidt, "An extension of the omnet++ inet framework for simulating real-time ethernet with high accuracy," in *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '11. Brussels, BEL: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011, p. 375–382.

[20] Z. Zhou, J. Lee, M. S. Berger, S. Park, and Y. Yan, "Simulating tsn traffic scheduling and shaping for future automotive ethernet," *Journal of Communications and Networks*, vol. 23, no. 1, pp. 53–62, 2021.