# Asynchronous Traffic Shaping with Jitter Control

Alexej Grigorjew, Florian Metzger, Tobias Hoßfeld
*University of Würzburg, Germany*
{alexej.grigorjew | florian.metzger | tobias.hossfeld}@uni-wuerzburg.de

Johannes Specht
*University of Duisburg-Essen, Germany*
johannes.specht@uni-due.de

Franz-Josef Götz, Feng Chen, Jürgen Schmitt
*Siemens AG, Germany*
{franz-josef.goetz | chen.feng | juergen.jues.schmitt}@siemens.com

***Abstract*—Asynchronous Traffic Shaping enabled bounded latency with low complexity for time sensitive networking without the need for time synchronization. However, its main focus is the guaranteed maximum delay. Jitter-sensitive applications may still be forced towards synchronization. This work proposes traffic damping to reduce end-to-end delay jitter. It discusses its application and shows that both the prerequisites and the guaranteed delay of traffic damping and ATS are very similar. Finally, it presents a brief evaluation of delay jitter in an example topology by means of a simulation and worst case estimation.**

## I. Introduction

Deterministic and time sensitive networking is gaining importance in various fields of technology, most notably in industrial and automotive use cases. In the field of guaranteed latency, the efforts can be split into synchronized approaches such as timed gates (IEEE Std. 802.1Qbv [1], 802.1Qch [2]), and asynchronous approaches such as the Credit-Based Shaper (802.1Qav [3]) and Asynchronous Traffic Shaping (ATS, 802.1Qcr [4][5]).

While the latter is able to provide per-hop latency guarantees to reserved streams without the need for time synchronization and complex network planning, some end devices require low jitter which is not considered by current asynchronous shaping mechanisms. In addition, the network itself could benefit from lower jitter when using mechanisms such as Frame Replication and Elimination for Reliability (FRER, 802.1CB [6]). If the delay variance on different redundant paths of the same stream is bounded, the required memory for the elimination step of FRER can be reduced significantly.

In this document, the advantages of Asynchronous Traffic Shaping are combined with jitter control to support these applications with low complexity overhead. Therefore, traffic damping is proposed as a new mechanism. This work presents the requirements and assumptions for the application of traffic dampers in an asynchronous environment, and it shows that the guaranteed latency of the proposed damping approach is equal to the upper bound of ATS. In addition, it shows that the traffic damper's queuing complexity for bridge manufacturers is comparable to ATS as the same number of queues is required for the same latency bound. Finally, a short evaluation is presented that compares the delay variance of traffic damping and ATS from a per-hop bridge-local perspective.

Table I
Used variables, symbols, and abbreviations in this document

| Variable | Description |
|---|---|
| TQ | Transmission Queue |
| Prop | Refers to **Prop**agation delay |
| SF | Refers to Store and Forward delay (transmission delay) |
| Proc | Refers to **Proc**essing delay (switch fabric) |
| Shaper | Refers to Shaping delay (damper delay) |
| $d^{abc}$ | Delay of the process *abc* (being one of the above) |
| $d^{abc,def}$ | Combined delay of *abc* and *def* (e.g., $d^{TQ,SF,Shaper}$) |
| $d_i^{abc}$ | Delay component *abc* as perceived by stream $i$ |
| $f_i$ | A particular frame of stream $i$ |
| $\mathcal{H}_i$ | Set of all streams with a higher priority than $i$ |
| $\mathcal{E}_i$ | Set of all streams with the same priority as $i$ |
| $\mathcal{L}_i$ | Set of all streams with a lower priority than $i$ |
| $\mathcal{Q}_i$ | Set of all streams with the same priority as $i$ that share the same shaping queue (i.e., same ingress port as $i$) |
| $p$ | A certain, fixed priority level |
| $p_i$ | The priority level of frames $f_i$ from stream $i$ |
| $\delta_p$ | Per-hop delay guarantee of priority $p$ at the current link |
| $\delta_p(h)$ | Per-hop delay guarantee of priority $p$ at the link $h$ |
| $c$ | A latency class $c$. Each priority $p$ can be split into multiple classes $c$. |
| $c_i$ | The latency class of frames $f_i$ from stream $i$ |
| $b_i$ | Burst size of stream $i$, upper bound on the short-term amount of data |
| $r_i$ | Leak rate of stream $i$, upper bound on the long-term data rate of $i$ |
| $\hat{\ell}_i$ | Maximum length of frames from stream $i$ |
| $\check{\ell}_i$ | Minimum length of frames from stream $i$ |
| $r$ | Link speed (e.g., $1\,\mathrm{Gbit/s}$) |
| $t_1$ .. $t_7$ | Local bridge time at certain points in the bridge pipeline |
| $\Delta t_{a,b}$ | Time interval between $t_a$ and $t_b$: $\Delta t_{a,b} = t_b - t_a$. |
| $w_i(\Delta t)$ | Cumulative amount of packet data (number of bits) that was transmitted by stream $i$ in the time interval $\Delta t$ |
| $t_1(f_i)$ | Local bridge time $t_1$ as perceived by the frame $f_i$, i.e., the local time at which the frame $f_i$ is enqueued into the transmission queue |

## II. Backgrounds

All used symbols in each figure and equation in this document are explained in Table I. Thereby, sets are always uppercase calligraphic letters ($\mathcal{H}_i$, $\mathcal{E}_i$, $\mathcal{L}_i$ and $\mathcal{Q}_i$). Small letters represent an element of the set ($x \in \mathcal{Q}_i$) or individual values (link speed $r$, burst size $b_i$). Any properties of the directly *observed* stream are referred to with index $i$, while any other

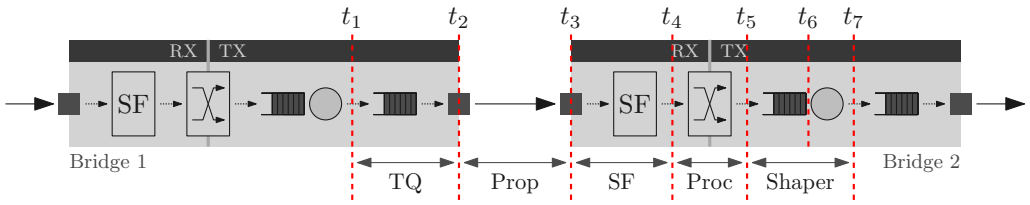


Figure 1. Illustration of per-hop delay components.

*interfering* streams are referred to with $x$.

### A. Priority Levels and Queues

Each frame belongs to a certain priority level, which can be internally mapped to a specific traffic class at the switches. This document assumes strict priority scheduling as in IEEE standard 802.1Q [7], therefore up to eight priority levels are supported at the egress, and each has its own transmission queue. Each individual queue is assumed to be processed first-in first-out (FIFO).

This document introduces traffic dampers as an additional shaper in the switch pipeline. In this model, the shaper is located before the transmission queue and each shaper has its own shaper queue. The number of individual shapers is addressed in the queue allocation rules in Section III-C. In general, for each priority level and each ingress port, an individual shaper queue is required at each egress port.

Note that this two-level queuing is used to describe the behavior of the switch. It is possible to achieve the same behavior with a single-level queue implementation by applying more sophisticated transmission selection algorithms.

### B. Switch Delay Model

Figure 1 presents an abstract model of the transmission process from one bridge to another. In this work, the latency of a transmission is regarded from shaper to shaper, as opposed to measuring the time from in-port to in-port.

On the sending bridge, the latency is measured right after the frame $f_1$ is considered eligible by the shaper at the moment $t_1$. It is enqueued in the transmission queue (TQ) of its priority level at the respective egress port.

At the moment $t_2$, the transmission selection algorithm selects $f_i$. The time $f_i$ spends in the transmission queue can be expressed by $d^{TQ} = \Delta t_{1,2} = t_2 - t_1$. The frame is transmitted towards the next hop. At the moment $t_3$, the first bit is received, and $t_4$ marks the reception of the last bit. The transmission delay (store and forward, SF) of the frame can be expressed by $d^{SF} = \Delta t_{3,4} = t_4 - t_3$. The internal processing time of the switch fabric (ASIC/CPU, selection of output port) is marked by $d^{Proc} = \Delta t_{4,5} = t_5 - t_4$.

Finally, the frame is enqueued into its respective shaper queue. This queue operates in FIFO order, the number of individual queues is addressed later. Only the head of the shaper queue is processed by the shaper (in this case, a damper), the remaining frames must wait for their turn. The moment when a frame hits the head of the shaper queue is marked by $t_6$. The moment when the shaper flags the head frame as eligible and hands it to the transmission queue (TQ) is marked as $t_7$. The entire shaping delay can be written as $d^{Shaper} = \Delta t_{5,7} = t_7 - t_5$. The moment $t_7$ equals the start $t_1$ of the transmission towards the next hop (bridge 3).

Note that the moments $t_{\{1..7\}}$ refer to local times in their respective bridge. Clocks are not synchronized in this model. In particular, the propagation delay $d^{Prop}$ cannot be derived by $t_3 - t_2$, as these times are measured based on different reference systems.

An alternative model may be considered in which transmission $d^{SF}$ and processing $d^{Proc}$ are interleaved instead of consecutive. For the application of the proposed damper, it is sufficient if $d^{SF,Proc} = \Delta t_{3,5} = t_5 - t_3$ holds.

### C. Related Concepts

This paper applies traffic dampers to reduce delay jitter by examining the queuing delay of the previous hop and adjusting the eligibility time of each frame in the current hop accordingly. A similar methodology has been proposed earlier by Zhang and Ferrari [8] under the name of *delay-jitter controlling regulator*. However, they do not formally address the problem of head of line blocking when using interleaved regulators.

Queuing delay guarantees with interleaved queues are discussed for the Urgency-Based Scheduler [5]. The paper describes Queue Allocation Rules (QARs) for the Asynchronous Traffic Shaper (ATS), a work-conserving[1] shaping mechanism designed in the context of Time-Sensitive Networking (TSN) [9].

In this work, the idea of using interleaved regulators for non-work-conserving shaping is addressed to control delay jitter of real-time transmissions in Ethernet networks with strict priority queuing. It formally derives the amount of required queues to avoid head of line blocking in interleaved queues. It further extends this idea beyond the use of a single delay guarantee for each supported priority level to support novel redundancy mechanisms developed by the TSN working group, such as Frame Replication and Elimination for Reliability (FRER) [6].

Several standards developed under the TSN working group are also aiming towards eliminating jitter and providing low delay guarantees. Timed gate control lists (802.1Qbv [1]) allow full isolation of streams and deterministic delays, but they

[1]Non-work-conserving schedulers may leave the server idle even when there are packets waiting for transmission [8].

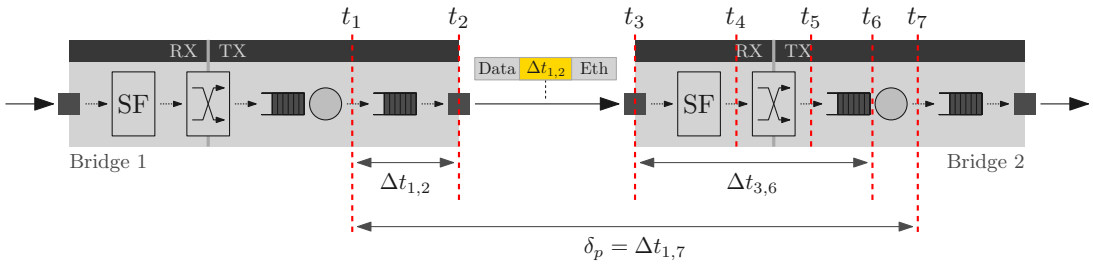


Figure 2. Transmission of a frame's experienced transmission queue delay $\Delta t_{1,2}$ to the subsequent bridge.

require network-wide time synchronization [10] and a central control unit with network-wide stream information for full jitter control. The proposed mechanism in this work does not rely on synchronization and can be implemented in a distributed control plane.

Finally, the proposed traffic damper relies on delay information that is carried over by the previous bridge towards the current bridge. This is achieved by injecting the experienced queuing delay as an additional header field into the payload of the Ethernet frame. The subsequent bridge can extract this information and apply it to derive the frame's eligibility time. The general idea to embed information of the network's state into data plane frames without requiring intervention of the control plane is recently being discussed as *In-band Network Telemetry* [11]. Under this term, it is primarily being used in the context of software-defined networks and mostly for monitoring and diagnosis purposes. In this paper, the carried information is used by the data plane for the shaper's operation.

### D. Asynchronous Traffic Shaping

This work makes use of the specification and delay analysis of Asynchronous Traffic Shaping (ATS) [5][4]. In particular, the traffic model, the queue allocation rules, and the final latency bound are of importance.

*1) Traffic model:* There is no requirement for time synchronization, each talker may emit its frames asynchronously. However, each flow $f_i$ must satisfy a leaky bucket constraint [12][13] characterized by a burstiness $b_i$ and a leak rate $r_i$. For any arbitrary time interval of duration $\Delta t$, the leaky bucket constraint limits the cumulative packet data $w_i(\Delta t)$ of flow $f_i$ to

$$w_i(\Delta t) \leq b_i + \Delta t \cdot r_i. \tag{1}$$

Many more specific traffic models can be mapped to this general model. For example, periodic traffic with a fixed packet inter-arrival time $T_i$ and a fixed packet length $\ell_i$ could be mapped to $r_i = \frac{\ell_i}{T_i}$ and $b_i = \ell_i$.

The applied traffic shaper re-enforces the constraint in Equation 1 at every bridge such that the requirements for the ATS delay bound are satisfied at each hop on a frame's path.

*2) Queue allocation rules:* In order to prevent head of line blocking in the shaper queue that could otherwise violate the latency bound, ATS applies three queue allocation rules (QARs) that regulate which frames may share the same shaper queue [5].

QAR 1: Frames $f_i$ and $f_x$ are not allowed to share a shaper queue if both are received from different servers, i.e., if both are received by different in-ports.

QAR 2: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are sent in different priority levels to the current bridge.

QAR 3: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are sent in different priority levels from the current bridge.

In particular, if both $f_i$ and $f_x$ are received by the same ingress port and are sent in the same priority level, they are allowed to share the same shaper queue.

*3) ATS latency bound:* If the above conditions are met and Asynchronous Traffic Shaping is used, an upper bound for the queuing (TQ), transmission (SF), and shaper latency can be derived. The maximum per-hop latency $d_i^{TQ,SF,Shaper}$ of all frames $f_i$ from a flow $i$ is bounded by [5, Eq. 21]

$$d_i^{TQ,SF,Shaper} \leq \max_{x \in \mathcal{Q}_i} \left( \frac{b_{\mathcal{H}_i} + b_{\mathcal{E}_i} - \check{\ell}_x + \hat{\ell}_{\mathcal{L}_i}}{r - r_{\mathcal{H}_i}} + \frac{\check{\ell}_x}{r} \right). \tag{2}$$

Thereby, $\mathcal{Q}_i$ is a subset of all streams $\mathcal{E}_i$ with the same priority as $i$ that are received on the same ingress port and therefore share a queue with $f_i$. $b_{\mathcal{H}_i}$ is the sum of all bursts of streams $x$ with higher priority $p_x$ than $i$, $b_{\mathcal{E}_i}$ is the sum of all bursts from equal priority streams (including $i$ itself), $\hat{\ell}_{\mathcal{L}_i}$ is the largest maximum frame size $\hat{\ell}_x$ from all lower priority streams $x$, and $r_{\mathcal{H}_i}$ is the sum of all leaky bucket rates from higher priority streams:

$$b_{\mathcal{H}_i} = \sum_{x \in \mathcal{H}_i} b_x \tag{3}$$

$$b_{\mathcal{E}_i} = \sum_{x \in \mathcal{E}_i} b_x \tag{4}$$

$$\hat{\ell}_{\mathcal{L}_i} = \max_{x \in \mathcal{L}_i} (\hat{\ell}_x) \tag{5}$$

$$r_{\mathcal{H}_i} = \sum_{x \in \mathcal{H}_i} r_x \tag{6}$$

Equation 2 can be simplified if all involved streams are assumed to have the same (worst case) minimum frame size $\check{\ell} = 64\,\text{bytes}$:

$$d_i^{TQ,SF,Shaper} \leq \frac{b_{\mathcal{H}_i} + b_{\mathcal{E}_i} - \check{\ell} + \hat{\ell}_{\mathcal{L}_i}}{r - r_{\mathcal{H}_i}} + \frac{\check{\ell}}{r}. \tag{7}$$

Equation 7 is equal for all streams of the same priority level, the computation of the maximum and all individual differences are vanished by this simplification. The impact on the accuracy is minimal as the formula is quickly dominated by the sum of bursts $b_{\mathcal{H}_i} + b_{\mathcal{E}_i}$ and rates $r_{\mathcal{H}_i}$ [5].

## III. Combining ATS with Jitter Control

The original idea of Asynchronous Traffic Shaping (ATS) is to re-shape the traffic at every hop in order to recover the leaky bucket constraint from Equation 1. Thereby, only the inter-frame timings are relevant, consecutive frames are spaced out in order to prevent accumulated bursts. The individual frame's delay is not directly considered in the shaping process. This leads to high variance in the end-to-end delays of individual frames, i.e., high jitter, especially under low load conditions when the leaky buckets can recover frequently. The minimum possible delay and the worst case situation with full interference and shaping may be far apart.

Therefore, the application of *traffic dampers* [14] is suggested in order to re-shape every frame individually to its maximum delay, effectively minimizing jitter. The remainder of this section covers their application in more detail, along with its prerequisites and resulting guarantees.

### A. Assumptions and prerequisites

The application of traffic dampers comes with three major conditions.

1) The local time intervals $\Delta t_{1,2}$ and $\Delta t_{3,6}$ can be measured and are available to the data plane (i.e., the shaper) of the bridge.

2) The bridge is able to push additional fields to the frame's header at the moment $t_2$, i.e., after it is selected for transmission and dequeued from the transmission queue. In particular, the local value $\Delta t_{1,2}$ of bridge 1 must be transmitted to bridge 2.

3) All steps in the depicted pipeline of Figure 1 preserve the frames' order, i.e., no reordering can occur for two frames that share the same queues.

4) Further, the accuracy of the technique can be improved if the bridge is able to assess the propagation delay $d^{Prop} = \Delta t_{2,3}$ of a particular link. This would improve the accuracy of the end-to-end delay itself, it is not required for jitter control since (i) the propagation delay of a particular link is constant, and (ii) all frames of the same stream traverse the same links. Without loss of generality, the remainder of this work assumes that the propagation delay $d^{Prop} = \Delta t_{2,3}$ is known to the receiving bridge of a transmission.

### B. Specification of shaping delay

Let $\delta_p$ be the per-hop delay guarantee specified for all frames of priority level $p$. If the above prerequisites are met, the eligibility time $t_7$ of the asynchronous traffic damper can be specified by:

$$t_7 = t_1 + \delta_p \tag{8}$$

Therefore, the individual shaping delay $\Delta t_{6,7}$ of the shaper queue's head frame is specified by:

$$\begin{aligned} \Delta t_{6,7} &= t_7 - t_6 \\ &= t_1 + \delta_p - t_6 \\ &= t_1 + \delta_p - (t_1 + \Delta t_{1,2} + \Delta t_{2,3} + \Delta t_{3,6}) \\ &= \delta_p - \Delta t_{1,2} - \Delta t_{2,3} - \Delta t_{3,6} \end{aligned} \tag{9}$$

The per-hop delay $\delta_p$ and the time intervals $\Delta t_{2,3}$ and $\Delta t_{3,6}$ are known to the receiving bridge as per the above assumptions (Section III-A). The propagation delay $\Delta t_{2,3}$ can be omitted with a minor impact on accuracy in local networks. The transmission queue delay $\Delta t_{1,2}$ is transmitted in an extra header field by the previous bridge, as illustrated by Figure 2.

This methodology does not require the bridge to hold any per-stream state beyond the lifetime of a single frame. However, it depends on the information $\Delta t_{1,2}$ received by the previous bridge. This information is subject to transmission errors, just as any other information in the frame. Therefore, the frame's checksum must be updated at every hop to include the respective delay field in order to prevent the spread of malicious frames. For further elaboration, refer to [14].

### C. Avoiding head of line blocking

Assume the frames in the shaper queue are ordered, as indicated by Figure 4. The shaping delay $\Delta t_{6,7}$ is only applied to the head $f_1$ of the shaper queue. Therefore, if the subsequent frames $\{f_2, f_3, ...\}$ require a smaller shaping delay, they are subject to head of line (HoL) blocking, as they exceed their eligibility time in the shaper queue.

Similarly to ATS, HoL blocking can be avoided by employing additional shaper queues and preventing certain frames from sharing the same queue. The number of required queues and the frame groups that must be separated is derived in the following.

Head of line blocking is avoided if the eligibility time $t_7(f_2)$ of a subsequent frame $f_2$ is always later than the eligibility time $t_7(f_1)$ of the current frame $f_1$:

$$\begin{aligned} & t_7(f_2) \geq t_7(f_1) \\ \Leftrightarrow\ & t_1(f_2) + \delta_{p_2} \geq t_1(f_1) + \delta_{p_1} \end{aligned} \tag{10}$$

Equation 10 can be simplified if both frames belong to the same priority, i.e., if $\delta_{p_2} = \delta_{p_1}$. This can be achieved by separating frames with different priority levels into different shaper queues. All frames in the same queue have the same guarantee $\delta_p$ and Equation 10 can be simplified to:

$$t_1(f_2) \geq t_1(f_1) \tag{11}$$

To satisfy the remaining condition, refer to Figure 3. The bridges A and B send frames towards the same bridge C, which will be transmitted further by the same egress port of C. Assume that $t_1^A(f_1) < t_1^B(f_2) = t_1^A(f_1) + \Delta t_{A,B}$, and assume that the transmission queue delay $d^{TQ} = \Delta t_{1,2}$ of bridge A is longer than $d^{TQ}$ of B by $\Delta t_{A,B}$:

$$d_A^{TQ}(f_1) > d_B^{TQ}(f_2) + \Delta t_{A,B} \tag{12}$$

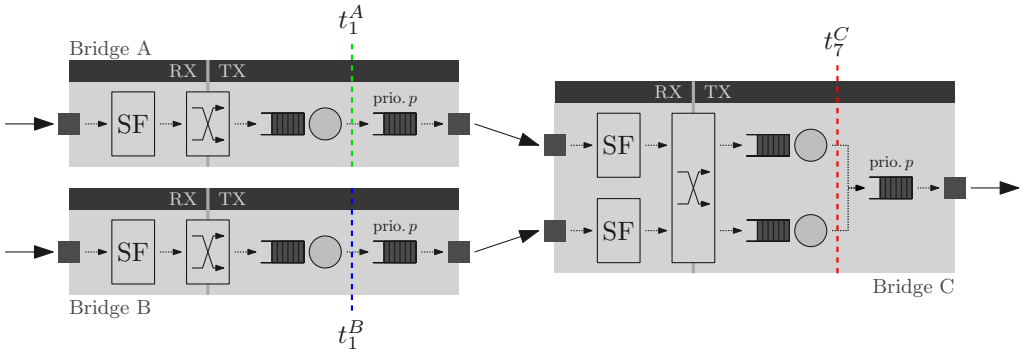


Figure 3. Frame isolation in different shaper queues to avoid HoL blocking.

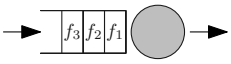


Figure 4. Order of the frames $f_1$, $f_2$, $f_3$ in the shaper queue.

If all other delays (propagation, transmission, processing) are similar and both frames share the same shaper queue, frame $f_2$ arrives in the queue before frame $f_1$ does. However, due to the smaller queuing delay $\Delta t_{1,2}$, the damper assigns a higher shaping delay to $f_2$, effectively blocking $f_1$ which would be eligible for transmission earlier. Therefore, frames arriving from different ingress ports may not share the same shaper queue.

If all frames that share the same shaper queue arrive from the same in-port (bridge A), it always holds

$$t_1^A(f_2) \geq t_1^A(f_1) \;\Leftrightarrow\; t_1^C(f_2) \geq t_1^C(f_1) \tag{13}$$

since the individual steps of the bridge pipeline preserve frame order as per Assumption 3.

In summary, head of line blocking can be prevented by applying the same queue allocation rules (QARs) as with Asynchronous Traffic Shaping (ATS):

QAR 1: Frames $f_i$ and $f_x$ are not allowed to share a shaper queue if both are received from different servers, i.e., if both are received by different in-ports in the current bridge (due to Equation 11).

QAR 2: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are *received* in different priority levels ($p_i \neq p_x$) by the current bridge (due to Equation 11, as this would also imply that they came from different transmission queues).

QAR 3: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are *sent* in different priority levels ($p_i \neq p_x$) by the current bridge (due to Equation 10).

### D. Possible per-hop delay guarantees

Naturally, Equation 8 only holds if the per-hop delay guarantee is always larger than the delay itself:

$$\delta_p \geq \Delta t_{1,7} \tag{14}$$

The remaining question is, how low may the per-hop delay guarantee be specified without violating Equation 14. In other words, how can the bridges verify at any point in time whether the per-hop delay guarantee can still be satisfied.

As shown in Section III-C, the dampers are subject to the same queue allocation rules as ATS. If the traffic model in Section II-D1 can also be satisfied at every bridge on a frame's path, the ATS latency bound from Equation 2 and 7 can be applied.

Assume that the talkers emit their frames in accordance to the leaky bucket constraint (cf. Section II-D1), i.e., they limit their cumulative packet data $w_i(\Delta t_{a,b})$ during any arbitrary duration $\Delta t_{a,b} = t_b - t_a$ to

$$w_i(\Delta t_{a,b}) \leq b_i + r_i \cdot \Delta t_{a,b}. \tag{15}$$

Further, assume that the constraint is violated at the $n$-th hop along the stream's path. Traffic damping enforces a constant per-hop delay $d^{const} = \sum_{k=1}^{n} \delta_p(h_k)$ to all frames of a stream, i.e., if $f_1$ is transmitted at the moment $t_a$ from the talker, it is marked eligible at the moment $t_a + d^{const}$ by the shaper at the $n$-th hop. If the constraint is violated, the shaper received more than $b_i + r_i \cdot \Delta t_{a,b}$ packet data in the time interval $\Delta t_{a,b}$. This implies that the talker transmitted more than $b_i + r_i \cdot \Delta t_{a-d^{const},b-d^{const}} = b_i + r_i \cdot \Delta t_{a,b}$ in the time interval $\Delta t_{a-d^{const},b-d^{const}} = \Delta t_{a,b}$. This is a contradiction to the assumption, therefore it must be false.

This means that the ATS traffic model can be applied here. The possible per-hop delay guarantee $\delta_{p_i}$ is constrained by the guarantee of the ATS latency bound:

$$\delta_{p_i} \geq \frac{b_{\mathcal{H}_i} + b_{\mathcal{E}_i} - \check{\ell} + \hat{\ell}_{\mathcal{L}_i}}{r - r_{\mathcal{H}_i}} + \frac{\check{\ell}}{r} \geq d_i^{TQ,SF,Shaper}. \tag{16}$$

In summary, the traffic damping mechanism presented in this document is (i) subject to the same queue allocation rules as ATS, (ii) keeps the leaky bucket condition enforced if it is maintained by the talkers, and therefore, (iii) it can provide the same per-hop latency guarantees as ATS. It does not require to maintain per-stream state in the data plane of the bridges and keeps the end-to-end delay jitter at a minimum due to constant delays.

### E. Increasing the number of per-hop delays

In general, each priority level could support more than one per-hop delay. This could be useful in FRER environments

to equalize the cumulative delays of two redundant paths. On both paths, the frame would belong to the same priority level. In order to minimize the jitter at the replicate-eliminating bridge at the end of the redundant paths, the cumulative delay on both paths could be equalized by the traffic damper. This cannot be achieved by the same configuration if both paths have different numbers of hops. Therefore, supporting different per-hop delays for the same priority level can be desirable to keep additional queuing resources at FRER bridges low.

To support multiple per-hop delay configurations, each priority level $p$ could be split into multiple *latency classes* $c$. Each frame's delay of stream $i$ would then be shaped to the value of $\delta_{c_i}$ instead of the global guarantee $\delta_{p_i}$. For each such per-hop delay $\delta_{c_i}$, a separate shaper queue is required. An alternative set of queue allocation rules could be formulated as follows:

QAR 1: Frames $f_i$ and $f_x$ are not allowed to share a shaper queue if both are received from different servers, i.e., if both are received by different in-ports in the current bridge.

QAR 2**b**: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are received in different *latency classes* ($c_i \neq c_x$) by the current bridge.

QAR 3**b**: Frames $f_i$ and $f_x$ are not allowed to share a queue if both are sent in different *latency classes* ($c_i \neq c_x$) by the current bridge.

Without loss of generality, the remainder of this document assumes a single latency class for each priority level and works with $\delta_p$.

## IV. Evaluation

In order to evaluate the effectiveness of traffic dampers with respect to jitter control, a simulation of both ATS and the damper shaping is conducted. The Discrete Event Simulation tool SimPy[2] has been used to implement the delay model from Section II-B for individual frames, and to chain multiple frame transmission processes for a stream within a network topology. Figure 5 shows the topologies (A and B) of the test setup. The talker on the very left is communicating with the listener on the very right. In topology A, each of the small talkers is communicating with the same listener to create some interference. In topology B, the observed (red) stream and the interfering (blue) streams share the intermediate links, but they address different listeners at the last hop.

In total, the simulation was conducted with 7 bridges and 14 additional interfering talkers per bridge, leading to 99 streams sharing the same resources. Each stream has a fixed frame size of 250 bytes (270 bytes with preamble and inter-frame gap). The frames' inter-arrival times are uniformly distributed between 240 µs and 260 µs, representing periodic streams with slight deviation. This leads to a 85% bandwidth utilization at the last link. The processing delays of all bridges are uniformly distributed between 1 µs and 5 µs. The propagation delay was omitted. The ATS per-hop latency bound of the main talker's stream is roughly 215 µs at the last bridge in this scenario (Equation 7).

[2]https://simpy.readthedocs.io/

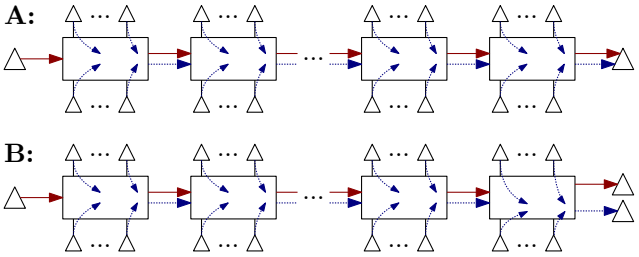


Figure 5. Linear topologies (A and B) used for jitter evaluation. 7 bridges with 14 additional talkers per bridge (blue streams) are deployed.

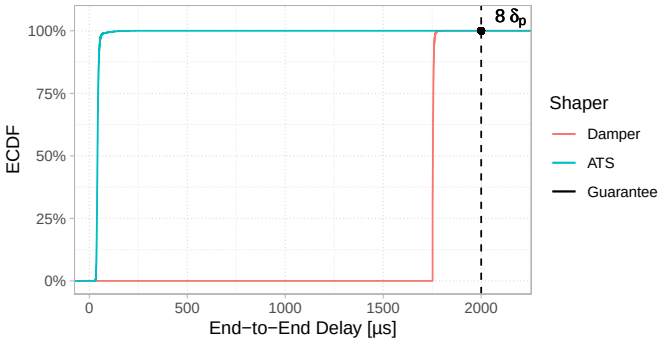


Figure 6. Comparison of delay distributions for ATS and traffic damping. Only topology A is shown, as both scenarios behave similarly.

In order to increase the observed jitter artificially, the talkers are scheduled to send their frames for maximum interference at their respective bridge. In addition, all talkers skip every 5-th frame in their periodic schedule. This is done to (i) have the main stream recover all of its shaper tokens, and (ii) prevent all interference on the path afterwards to include the minimum latency in the measurement for a high overall jitter. Missing periodic frames may occur commonly in a dynamic environment, for example due to reboots, short outages, changes in the topology, or reconfigurations on the bridges.

The results of the simulated comparison are presented in Figure 6. The x-axis shows the observed end-to-end delay values for both shapers. The y-axis shows the empirical cumulative distribution function of a three seconds simulation. In short, it shows $P(delay \leq x) = y$ for the observed values. The colors indicate the shaping mechanism. Thereby, the vertical dashed black line indicates the end-to-end delay guarantee $8 \cdot 250\,\mu s = 2\,ms$. The most apparent difference between the two mechanisms is the much higher end-to-end delay of the traffic damper (mean $1.75\,ms$) compared to ATS (mean $43\,\mu s$). All delays of the damper are equalized to the maximum value, which is determined by the configured per-hop latency guarantee $\delta_p = 250\,\mu s$. However, the theoretical bridge-local delay guarantees are the same for both shapers. In addition, ATS shows a slightly larger variance, especially at the last 5%.

In order to evaluate the variance in more detail, Figure 7 shows both delay distributions of topology A (same listener) normalized to their minimum value, i.e., each observed end-to-

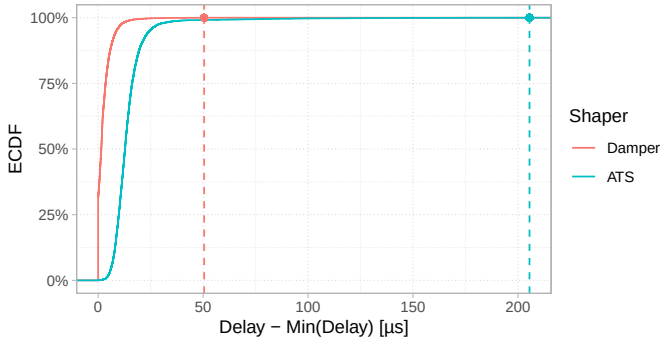


Figure 7. Comparison of jitter distributions for ATS and traffic damping in a 3 seconds simulation (topology A).

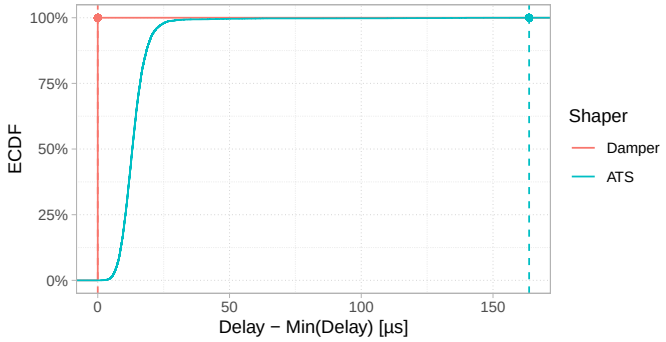


Figure 8. Comparison of jitter distributions for ATS and traffic damping in a 3 seconds simulation (topology B).

end delay is reduced by the minimum observed value for that shaper, so both curves start at 0. Effectively, Figure 7 shows the empirical jitter distribution functions of both mechanisms. In addition, the maximum observed jitter (100% intersection) is marked by a vertical dashed line of the same color.

The traffic damper shows a much steeper increase than ATS with more than 25% of all frames observed having the same minimum delay ($1752\,\mu s$). The maximum observed jitter is also much lower, slightly above $50\,\mu s$ compared to the $205\,\mu s$ of ATS. However, the measured jitter is not zero for topology A. The damper equalizes the delays of all frames to the deterministic worst case delay guarantee, but traffic damping only works from shaper to shaper. In this scenario, the listener does not reshape traffic and is still subject to delay jitter in the transmission queue of the last hop. This is observed in the red curve of Figure 7 and caused by an alternating order of frames due to the small variations in talker inter-arrival times (up to $20\,\mu s$).

The remaining jitter is predictable with bridge-local information and only depends on the streams that share the last link. The maximum jitter with traffic damping is limited by the maximum transmission queue delay of the last used link towards the listener (and the maximum deviation of transmission delay due to variable frame sizes). In topology A, all $7 \cdot 14 = 98$ streams share the last link, and in the worst case, their respective shaper marks all of them eligible at the same time, leading to a maximum $d^{TQ} = 98 \cdot \frac{270\,\text{byte}}{1\,\text{Gbit/s}} \approx 212\,\mu s$. For comparison, the worst case end-to-end jitter of ATS is given by the difference between the minimum transmission delay ($8 \cdot \frac{258\,\text{byte}}{1\,\text{Gbit/s}} = 16.5\,\mu s$[3]) and the maximum delay ($864\,\mu s$, sum of all per-hop delay bounds from Equation 7). Without current network-wide information from all streams, the latter must be estimated with the accumulated maximum latency, i.e., the sum of all per-hop delay guarantees $\delta_p$ on all links of the observed stream. In this example, it equals $8 \cdot 250\,\mu s = 2\,ms$.

Note that each port of a bridge and each supported priority level has its own transmission queue in IEEE standard 802.1Q [7]. In topology B (Figure 5), the interfering streams (blue) are physically separated from the observed stream (red) on the last hop, they lead to different listeners. Therefore, the end-to-end delay jitter is entirely eliminated as the transmission queue delay is always zero for the last link. This is illustrated in Figure 8. During the entire 3 seconds of the simulation, each observed frame arrives at the listener with the same deterministic delay of $1752\,\mu s$, which equals $7 \cdot 250\,\mu s$ shaped transmissions and one final $2.064\,\mu s$ transmission towards the listener that does not support the traffic damper.

This is an artificial scenario with no lower priority best effort traffic in mind. In reality, each link could also be used by non-reserved low priority streams of arbitrary frame sizes. In this case, the maximum transmission queue delay is increased by the duration of one MTU frame transmission, i.e., 1542 byte seconds or $12.4\,\mu s$ on a $1\,\text{Gbit/s}$ link. If true zero-jitter transmission is required by the end device, it may also implement the traffic damper and deliberately retrieve the packet at a deterministic time.

## V. Conclusion

This work presented traffic dampers as an asynchronous approach towards guaranteed latency with jitter control. It showed the underlying delay model, assumptions on traffic, prerequisites, and discussed important queue allocation rules and latency bounds for the existing Asynchronous Traffic Shaper (ATS). Afterwards, the specification of eligibility time in the asynchronous traffic damper is presented. The following discussions showed that this traffic damper model is subject to the same queue allocation rules as ATS and can provide the same per-hop latency guarantees.

The evaluation shows an exemplary comparison of both the observed jitter in a simulation, and the theoretical maximum jitter for the given scenario. In this particular scenario, the worst case jitter can be reduced from $847.5\,\mu s$ to $212\,\mu s$, and the worst case expected jitter that is forseen by individual devices without network-wide information is reduced from $2\,ms$ to $250\,\mu s$. If the traffic damper is also applied at the end devices, jitter can be eliminated entirely. Future work may focus on improving frame integrity, as the additional header field required for the asynchronous damping mechanism requires a recomputation of the frame's checksum at each hop [14].

[3] 250 byte frame size plus 8 byte Ethernet preamble = 258 byte.

## REFERENCES


[1] "IEEE Standard for Local and Metropolitan Area Network – Bridges and Bridged Networks – Amendment 25: Enhancements for Scheduled Traffic", *IEEE Std 802.1Qbv-2015*, 2015.

[2] "IEEE Standard for Local and Metropolitan Area Networks — Bridges and Bridged Networks — Amendment: Cyclic Queuing and Forwarding", *IEEE Std 802.1Qch-2017*, 2017.

[3] "IEEE Standard for Local and Metropolitan Area Networks – Virtual Bridged Local Area Networks – Amendment: Forwarding and Queuing Enhancements for Time-Sensitive Streams", *IEEE Std 802.1Qav-2010*, 2010.

[4] "Draft Standard for Local and Metropolitan Area Networks — Bridges and Bridged Networks — Amendment: Asynchronous Traffic Shaping", *IEEE P802.1Qcr/D2.0*, 2019.

[5] J. Specht and S. Samii, "Urgency-based scheduler for time-sensitive switched ethernet networks", in *2016 28th Euromicro Conference on Real-Time Systems (ECRTS)*, Jul. 2016, pp. 75–85.

[6] "Standard for Local and Metropolitan Area Networks – Frame Replication and Elimination for Reliability", *IEEE Std 802.1CB-2017*, 2017.

[7] "IEEE Standard for Local and Metropolitan Area Network – Bridges and Bridged Networks", *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)*, 2018.

[8] H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing", in *IEEE INFOCOM'93 The Conference on Computer Communications, Proceedings*, IEEE, 1993, pp. 227–236.

[9] IEEE, *Time-Sensitive Networking (TSN) Task Group*, 2020. [Online]. Available: https://1.ieee802.org/tsn/ (visited on 06/05/2020).

[10] "IEEE Standard for Local and Metropolitan Area Networks – Timing and Synchronization for Time-Sensitive Applications", *IEEE P802.1AS-Rev*, 2019.

[11] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, "In-band network telemetry via programmable dataplanes", in *ACM SIGCOMM*, 2015.

[12] R. L. Cruz, "A calculus for network delay. i. network elements in isolation", *IEEE Transactions on information theory*, vol. 37, no. 1, pp. 114–131, 1991.

[13] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case", *IEEE/ACM transactions on networking*, vol. 1, no. 3, pp. 344–357, 1993.

[14] J. Specht, *Dampers with forward traffic isolation*, 2019. [Online]. Available: http://www.ieee802.org/1/files/public/docs2019/new-specht-dampers-fti-1119-v01.pdf (visited on 04/15/2020).