# TSN and ATS: The Influence of Shaping on the System

Bruna Giovana Machado, Florian Wiedner*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: brunagiovana.machado@tum.de, wiedner@net.in.tum.de

***Abstract*—Rather automotive, aerospace or transportation, multiple industries depend on deterministic communication with strict timing requirements. In order to fulfill the necessary requirements, delays caused by traffic interferences must be minimized. The success of managing this traffic is critical to these industries and one possible solution is traffic shaping. This paper analyzes traffic shaping within Time-Sensitive Networking (TSN) and its impact on network performance and costs , focusing on Asynchronous Traffic Shaping (ATS) and comparing it with a synchronous shaper Time-Aware Shaping (TAS). We examine three ATS algorithms: Urgency-Based Scheduler, the Paternoster mechanism, and the ATS standard draft. Through evaluation of different simulations, we compare ATS with unshaped traffic, various scheduling mechanisms for ATS and TAS, highlighting the benefits and costs associated with each approach.**

***Index Terms*—time-sensitive networking, asynchronous traffic shaping, traffic shaping**

## 1. Introduction

Time-Sensitive Networking (TSN) is a set of IEEE 802 standards that ensure deterministic communication over standard Ethernet. One central mechanism of TSN is the traffic shaping techniques, managing traffic and providing bounded latency and reduced frame loss [1].

Time-Aware Shaping (TAS) is a TSN shaper, providing deterministic transmissions through synchronization among all network participants, however, in dynamic networks this characteristic is limiting [2]. Asynchronous Traffic Shaping (ATS) offers a flexible alternative, adapting to diverse network conditions [3]. We focus on the shaping mechanisms of ATS, their influence and costs.

In Section 2, we explore the theoretical background of TSN, the role of shaping and the functionality of TAS. In Section 3, we introduce three ATS's algorithms. In Section 4, we evaluate ATS, comparing it with unshaped traffic, different scheduling mechanisms, and TAS. In Section 5, we concludes and suggest ideas for future work.

## 2. Theoretical Background of TSN

From TSN's perspective, there are only two types of devices: bridges and end stations. End stations are further divided into talkers (sources) and listeners (targets) [1].

### 2.1. Shaping

In the case of data with strict time constraints, which is present in multiple industries, managing the delay caused through interferences by other participants of the network traffic is critical. TSN shapers introduce controlled delay aiming at bounded low latency and zero congestion loss by controlling the traffic flow at every hop, thus avoiding long bursts [1]. We focus on two TSN shapers: TAS and ATS.

### 2.2. Time-Aware Shaper

TAS requires the scheduling of traffic classes to be synchronized across all bridges from the talker to the listener(s), depicted in Figure 1. TAS schedules traffic streams in two reserved time-triggered windows: (i) for low-priority traffic, such as best effort (BE) and (ii) for scheduled traffic (ST) [4]. TAS employs a gate driver mechanism that opens and closes according to a time schedule for each port in a bridge. The Gate Control List (GCL) contains Gate Control Entries that define the transmission eligibility of a queue. A frame is allowed to be transmitted if (i) the queue has a frame ready to transmit, (ii) higher priority queues with an open gate have no frames to transmit, and (iii) the frame transmission can be completed before the gate of the queue closes [2], [4].

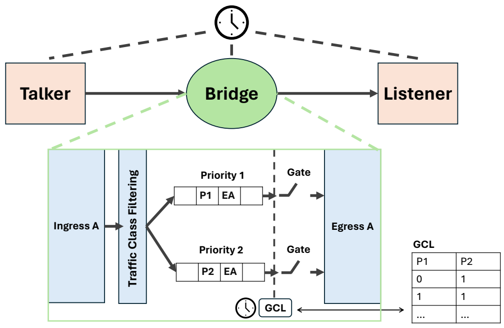


Figure 1: Visual representation of TAS adapted from [5]

## 3. Asynchronous Traffic Shaping

To avoid the critical failure of a timing misalignment, ATS is introduced as an alternative. It imposes similar traffic determinism without strict timing synchronization by introducing an independent clock at every bridge and end station [4].

The original concept of ATS [6] occurs at every hop and is depicted in Figure 2. First, individual frames are queued at a shaped queue of the desired egress port according to the flow state. The separation process of per flow state queues is called interleaved shaping. These queues follow three Queuing Admission Rate (QAR) schemes:

- **QAR1:** Frames from different sources are stored separately.
- **QAR2:** Frames from the same source with different priorities must be kept apart.
- **QAR3:** Frames from the same source with the same sender's priority but different receiver's priorities are separated [6].

Following these queueing schemes ensures traffic isolation [7]. Additionally, prioritizing high-priority traffic reduces their queuing time by allowing them to bypass lower-priority traffic [8].

Afterwards, the shaper merges the shaped queues conforming to the receiver's priority traffic class. The frames in the shared queue are then regulated by the transmission selection algorithm based on eligibility time [6]. In figure 2, the chosen transmission selection algorithm is strict priority FIFO.

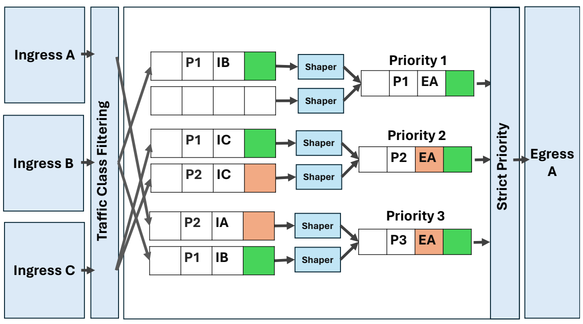

Figure 2: Visual representation of ATS adapted from [6]

Figure 2 depicts a switch with three ingress ports and one egress port with three priorities and therefore three shared queues. All of them follow the queuing schemes. The priorities from the ingress ports are depicted with "P1" or "P2" followed by the ingress port, e.g. "IA".

### 3.1. UBS algorithms

ATS, formally known as Urgency Based Scheduler (UBS), was created by Specht and Samii [6]. UBS has two different algorithms that utilize a per flow state:

- **Length Rate Quotient (LRQ)**: a frame-by-frame leaky bucket algorithm
- **Token Bucket Emulation (TBE)**: a token-based leaky bucket algorithm

$LRQ$: aims at a consistent transmission rate even for unpredictable flow patterns. The state of each flow $f_i$ contains a timestamp $t_i$ with an eligibility time for the current frame, based on the size of the previous frame $l$ and the permitted burst rate of a flow $\hat{r}_i$. The frame is delayed at least until the local time of the device $t_{now}$ reaches $t_i$. The eligibility time for the next frame of $f_i$ is $t_i = t_{now} + l/\hat{r}_i$ [6].

$TBE$: focuses on achieving a transmission with an average rate. The state of each flow $f_i$ contains a timestamp $t_i$ as well as a bucket level $b_i$. The frame is delayed until the token count $T$ is greater than or equal the frame size $l$. The tokens are measured as $T = b_i + (t_{now} - t_i) \cdot \hat{r}_i$. The eligibility time for the next frame is the current device time $t_i = t_{now}$ and the bucket level $b_i = \min\{\hat{b}_i, (t_{now} - t_i) \cdot \hat{r}_i\} - l$. This means that the delay between packets from the same flow is removed if enough "tokens" are available, possibly causing bursts [6].

TABLE 1: Variable definitions for equations (1) and (2)

| Parameter | Definition |
|---|---|
| $d$ | Upper bound on per hop delay |
| $I$ | Set with all flow indices |
| $b$ | Burst size |
| $l$ | Frame size |
| $r$ | Burst rate |
| $C(i)$ | Flows with the same priority as $i$ |
| $H$ | Flows with higher priority |
| $L$ | Flows with lower priority |
| $\hat{a}$ | Maximum |
| $\check{a}$ | Minimum |

The mathematical evaluation of the worst-case delay of a single hop is given by [6]:

$$d_{LRQ} \leq \max_{i \in I} \left( \frac{\hat{b}_H + \hat{b}_{C(i)} + \hat{l}_L}{r - \hat{r}_H} + \frac{\hat{l}_i}{r} \right) \quad (1)$$

$$d_{TBE} \leq \max_{i \in I} \left( \frac{\hat{b}_H + \hat{b}_{C(i)} + \hat{b}_i - \check{l}_i + \hat{l}_L}{r - \hat{r}_H} + \frac{\check{l}_i}{r} \right) \quad (2)$$

The variables of equations (1) and (2) are defined in table 1.

### 3.2. Paternoster queuing and scheduling

The Paternoster algorithm [9] is an improvement over the peristaltic shaper (802.1Qh Cyclic Queue and Forwarding (CQF)) [9]. It operates in a four-phase cycle: prior, current, next, and last. Packets are first added to the current queue. If they exceed the reservation's bandwidth for an epoch, they are moved to the next and then to the last queue. Once the three queues are full, the incoming frames are discarded. These phases rotate left after each epoch duration $\tau$, with current becoming prior, prior becoming last, etc. Each queue has a reserved bandwidth allocation for an epoch. Unlike CQF, Paternoster works asynchronously, reduces average delay, and handles multiple epoch reservations within a single epoch.

According to [9], the algorithm's per hop worst-case delay is defined as

$$d_{Paternoster} \leq 3 \cdot \tau, \quad (3)$$

This delay occurs when both the current and next queues are full, forcing the frame to wait in the last queue for up to three cycles before transmission.

### 3.3. ATS algorithm

The ATS standard algorithm [3, Sec. 8.6.11.3] is a derivation of TBE.

According to [3], each bridge in a network has a set of tables for different purposes, which include parameters necessary for the traffic regulation. These tables include the ATS Shaper Instance Table [3, Sec. 12.31.5] with parameters and variables for independent instances of ATS shapers, the ATS Shaper Group Instance Table [3, Sec. 12.31.6], catering to group instances of ATS shapers, and the ATS Port Parameter Table [3, Sec. 12.31.7], which contains parameters shared by all ATS shaper instances connected to a reception port.

The final eligibility time is determined by taking the maximum of three values [3, 8.6.11.3]: the frame's arrival time, the group eligibility time (the most recent eligibility time processed by any ATS shaper in the group), and the scheduler eligibility time (the earliest moment when a frame has accumulated enough tokens to be considered for transmission). For a frame to be considered valid, its eligibility time must be less than or equal to the arrival time plus the $MaxResidenceTime$ parameter, which limits how long a frame can reside in the bridge [3, Sec. 8.6.11.3]. This eligibility time is then used by the ATS transmission selection algorithm [3, Sec. 8.6.8.5].

Due to the worst-case delay equation for the ATS standard algorithm [3, Annex V] being an extension of the equation 2, it is not covered in this paper.

## 4. Evaluations of ATS

To determine the influence and costs of shaping through ATS, it is crucial to evaluate and compare its algorithms from different perspectives. Given the diversity of the simulations compared in this paper, their setups will be explained.

### 4.1. Comparison of ATS and unshaped traffic

**Setup.** The evaluation in [6] simulates two different scenarios to evaluate UBS algorithms by delay. The first scenario features four talkers connected to one switch (S0), which is then connected to another switch (S1), leading to the only listener. In the first scenario, switch S0 is equipped with four queues, and S1 with one, meaning one queue per ingress port. All four talkers transmit four flows each, totaling 16 flows. The second scenario involves one talker (T0) and one listener (L0) connected through five switches, dealing with interfering flows and increased link utilization. In the second scenario, only three flows are transmitted from T0 to L0, with eight additional flows introduced along the path to simulate a more realistic multi-hop environment. Each scenario includes two series: (i) with a single priority level and (ii) with dual priority levels. Both scenarios utilize the equations 1 and 2 to predict the expected worst-case delays for LRQ and TBE, which are anticipated to be identical in the first series. Specht and Samii [6] compare LRQ, TBE, per-flow queues shaped with LRQ (LRQ-F), and strict priority FIFO scheduling (SPO) through trajectory analysis. Here, we focus on comparing LRQ and TBE with SPO.

**First scenario analysis.** In the first series, each of the four flows occupies one queue at S0 and then compete for the single queue at S1. The simulation results indicate equal delays caused by both LRQ and TBE algorithms, with a high discrepancy between expected and simulated results upon entering the second switch, and a low discrepancy at the first switch. With only one priority level, the delay induced by the shapers is higher than that of SPO. This effect is especially clear at the second hop, where the delay is considerably higher due to the single queue scheme [6].

In the second series, flows are assigned different priorities at each hop. Discrepancies between expected and actual delays increase at each hop, similar to the first series. High-priority flows experience lower delays than low-priority ones. However, at S1, the delay for low-priority flows under SPO is notably higher compared to UBS algorithms. This occurs because SPO suppresses low-priority flows (last eight flows at S1) due to the buildup of high-priority flows (first eight flows at S1) [6]. LRQ maintains a consistent transmission rate, while TBE averages rates with minimal bursts, thus avoiding this issue.

**Second scenario analysis.** At the first hop in the first series, delays for the three flows are identical across all methods. However, at the following hops, SPO exhibits significantly higher delays than the expected worst-case delays for all UBS algorithms. This likely stems from SPO's inability to manage traffic bursts, leading to congestion under heavy traffic loads [6].

In the second series, low-priority flows face higher delays compared to high-priority flows. With priorities changing at each hop, delays from UBS algorithms closely align with the expected worst-case scenarios 1 and 2. Nevertheless, SPO's delay is nearly double that of UBS algorithms for low-priority flows.

**Evaluation.** Shapers are highly effective for networks with multiple priorities, as they manage traffic efficiently by minimizing bursts. In particular, asynchronous shapers are twice as beneficial in environments with interfering flows. The only downside to a shaper occurs when a network has only one priority and few interfering flows. However, such scenarios are uncommon for many networks, making shapers a valuable solution for such network traffic management issues.

### 4.2. Comparison of scheduling mechanisms for ATS

**Setup.** The UBS algorithms and Paternoster are compared regarding frame loss rate, average number of queued frames and average per-hop delay in the simulations done by Zhou et al. [7], [8]. The topology used in both studies is the same, a talker is connected to a switch, which connects to a listener. Paternoster is simulated with three different epoch durations $\tau$: 0.01s, 0.005s and 0.0025s. The results vary with the bandwidth of the input flow ranging from 4,096 to 20,48 MBit/s in [7] and 32 to 192 MBit/s in [8]. The reserved bandwidth being 5.76 MBit/s and 50 MBit/s accordingly. This generates similar outcomes in both papers.

**Frame loss rate.** With the increase of sent frames, the frame loss rate rises across all algorithms. Lower epoch durations $\tau$ result in higher frame loss rates due to reduced queuing time. This is because the reserved bandwidth is calculated as $3 \cdot \tau \cdot datarate$ [7]. Overall, UBS algorithms typically show a lower or equal frame loss rate compared to Paternoster in both simulations.

**Average number of queued frames.** LRQ stores frames longer than TBE, given the fact that LRQ must wait before transmitting multiple frames from the same flow, unlike TBE, which allows bursts. In comparison to other Paternoster variations, Paternoster A has the least amount of queued frames until the reserved bandwidth is reached. Once it is reached, Paternoster C has the least amount of queued frames [7]. The more frames that are sent, the closer each Paternoster algorithm gets to an equilibrium, which depends on the $\tau$ value. Lower $\tau$ values result in lower equilibrium levels due to higher frame loss. UBS algorithms follow this pattern, losing more frames as the input flow increases, resulting in less frames in each queue.

**Average per-hop delay.** The analysis confirms the worst-case delay of equation 3. All Paternoster variations show increased delay with higher input flow, however, the lower the epoch duration, the smaller frames can be forwarded at faster rates, causing Paternoster C to have the lowest delay of all. In the case of LRQ and TBE, both reach their peak delay at input flows 5.78 MBit/s [7] and 80 MBit/s [8], which is the moment when the traffic is almost overloading. Nevertheless, as soon as the overload is reached, the characteristics from LRQ and TBE of keeping the traffic constant and at an average rate create a sharp decrease [7]. In this environment, the UBS algorithms are focusing on smaller frames, which are not being discarded, clearing out the queue much faster.

**Evaluation.** While the given simulations does not accurately describes a multi-hop network, they successfully show the correlation between frame loss, average number of queued frames and average delay. The average delay and number of queued frames are directly linked to the frame loss rate [7]. Both ATS algorithms exhibit similar frame loss rates, resulting in a similar average amount of queued frames. For networks with many small frames, a low epoch duration $\tau$ yields the best result for Paternoster. Due to the leaky-bucket characteristic of the UBS algorithms, they perform well in overloaded networks, but transmit mostly small frames.

Since the simulation works with one queue and one priority, we can deduced out of Section 4.1 that an unshaped system, would have shown a lower delay, especially for very high input flows. If the simulations included more than one priority, the results would be more insightful. We can, however, deduce that in overloaded networks with priorities, only frames with the highest priority and smallest sizes would be transmitted, as seen in Section 4.1.

### 4.3. Comparison of ATS and TAS

**Setups.** Nasrallah et al. [4] compare the frame loss rate, mean and maximum frame delay of ATS and TAS with a ring network topology. The comparison includes sporadic and periodic scheduled traffic sources. With the knowledge that ATS does not generate extra overhead in a worst-case delay of a FIFO queue system [10] and the network calculus method introduced by Mohammadpour et al. [11], Zhao et al. [5] compare the performance from ATS and TAS using NC for only one priority. The compared aspects are the worst-case backlog (WCB), delay (WCD) and jitter (WCJ). It works with five different topologies, which are variations of ring and tree topologies. Moreover, both simulations use the standard ATS algorithm.

**Frame loss rate.** The results for sporadic ST sources show that ATS has a much lower frame loss rate for ST and a higher frame loss rate for best effort than TAS. The reason for this is that ATS prioritizes ST, causing more congestion in the BE queues, while TAS works with time-scheduled windows, transmitting both ST and BE consistently. Once the ST sources are periodic, the scheduled-windows work more in favor for TAS than ATS [4].

**Mean frame delay.** For high-priority traffic in the sporadic scenario, ATS performs with lower delays than TAS. As the load increases, ATS keeps a consistent delay for ST, whereas TAS with a 20% gate usage time for ST (TAS 1) increases slightly and with a 30% gate usage time for ST (TAS 2) increases significantly. For low-priority traffic, on the other hand, ATS performs more similarly to TAS 1 and better than TAS 2. The cause for this is the same as for the low frame loss rate. For the periodic scenario, ATS shows similar results [4].

**Worst-case scenarios.** In the case of sporadic ST sources, the WCD for low-priority traffic for ATS is significantly higher than any other traffic from both ATS and TAS. The WCD of ATS for high-priority traffic, however, is the lowest of TAS 1 and TAS 2. On the other hand, for periodic ST sources, the higher the period, the worse the WCD becomes for ATS, whereas TAS stays extremely low [4]. Zhao et al. [5] mention that sporadic flows are not supported by TAS, therefore it shows the same result as previously mentioned. Additionally, TAS has the lowest WCB, WCD and WCJ by far. They compares all three worst-cases for five different topologies. Moreover, they conclude that the increased concentration of transmissions and number of hops increase the traffic transmission determinism [5]. Besides, they hypothesize, that ATS will perform better, once the load increases [5]. The average hop delay discussed in Section 4.2 supports this hypothesis.

**Evaluation.** In scenarios with sporadic transmissions, ATS has a clear advantage over TAS in regards to frame loss rate, mean and maximum frame delay. If the topology of a network creates a high flow transmission concentration or if the transmissions are periodic, the transmissions become more deterministic, which is beneficial for TAS. The hypothesis from Zhao et al. [5], however, introduces the idea that an increased load would generate a better performance for ATS. Therefore, in networks without determinism ATS would certainly perform superior and in the case of high traffic loads, ATS might achieve more advantages.

## 5. Conclusion and future work

Overall, ATS algorithms offer a flexible and efficient traffic management solution, aiming for bounded low latency and zero congestion loss, especially for high-priority traffic. They perform exceptionally well in dynamic and high-load environments and operate independently of the incoming flow pattern and network synchronization.

Future work should prioritize integrations of ATS with other shaping mechanisms and evaluate its performance in more complex network topologies. Some of the mentioned simulations do not attempt to imitate real-life scenarios with complex topologies and multiple hops, which would be a significant area for future exploration. Another potential improvement for ATS algorithms would be the development of methods to recognize and optimize for deterministic traffic.

## References


[1] IEEE 802, "Time-sensitive networking (tsn)," accessed: 2024-05-02. [Online]. Available: https://1.ieee802.org/tsn/

[2] IEEE, *IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 25: Enhancements for Scheduled Traffic*, Std. IEEE 802.1Qbv-2015, March 2016, released: 18.03.2016.

[3] ——. (2020, November) IEEE 802.1Qcr-2020 IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks - Amendment 34: Asynchronous Traffic Shaping. Accessed: 2024-05-27. [Online]. Available: https://1.ieee802.org/tsn/802-1qcr/

[4] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. Elbakoury, "Performance comparison of ieee 802.1 tsn time aware shaper (tas) and asynchronous traffic shaper (ats)," *IEEE Access*, vol. 7, pp. 44 165–44 181, 2019.

[5] L. Zhao, P. Pop, and S. Steinhorst, "Quantitative performance comparison of various traffic shapers in time-sensitive networking," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2899–2928, 2022.

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

[7] Z. Zhou, Y. Yan, M. Berger, and S. Ruepp, "Analysis and modeling of asynchronous traffic shaping in time sensitive networks," in *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, 2018, pp. 1–4.

[8] Z. Zhou, M. S. Berger, S. R. Ruepp, and Y. Yan, "Insight into the ieee 802.1 qcr asynchronous traffic shaping in time sensitive network," *Advances in Science, Technology and Engineering Systems Journal*, vol. 4, no. 1, pp. 292–301, 2019.

[9] M. Seaman, "Paternoster policing and scheduling," March 2017, iEEE 802.1Qcr.

[10] J.-Y. Le Boudec, "A theory of traffic regulators for deterministic networks with application to interleaved regulators," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2721–2733, 2018.

[11] E. Mohammadpour, E. Stai, M. Mohiuddin, and J.-Y. Le Boudec, "Latency and backlog bounds in time-sensitive networking with credit based shapers and asynchronous traffic shaping," in *2018 30th International Teletraffic Congress (ITC 30)*, vol. 2. IEEE, 2018, pp. 1–6.