Traffic Pattern-Based Scheduling for Wireless Non-TSN End Nodes

Pablo Avila-Campos*, Jetmir Haxhibeqiri*, Xianjun Jiao*, Ingrid Moerman*, Jeroen Hoebeke*
*IDLab, Ghent University – imec
{pabloesteban.avilacampos, jetmir.haxhibeqiri, xianjun.jiao, ingrid.moerman, jeroen.hoebeke}@ugent.be

***Abstract*—Time-Sensitive Networking (TSN) ensures reliable traffic delivery in industrial automation, multimedia, and automotive systems. While effective in wired networks, wireless TSN (WTSN) faces challenges like delays and interference, complicating traffic scheduling. This paper presents a data-driven approach to improve WTSN management by addressing the residual service time (RST) problem, which increases link latency. Tested in a Wi-Fi TSN-based environment, the proposed WTSN digital twin framework preserves TSN traffic guarantees while significantly reducing RST and hence link latency.**

***Index Terms*—wireless time-sensitive networking, time synchronization, scheduling, openwifi, IEEE802.11.**

I. Introduction

For many emerging technologies, including industrial, vehicular, and professional audio-video systems, a delayed frame can be as problematic as a dropped one. In these applications, a reliable and predictable network service is essential. To achieve this, deterministic networking has been proposed. Due to its characteristics, network determinism is particularly useful for maintaining guarantees at maximum network capacity. This helps reduce network oversizing, energy consumption, and required capacity [1].

Wireless networks, with their inherently lower and variable capacity compared to wired networks, are strong candidates for determinism. The shared channel and changing conditions require continuous adaptation to maintain quality of service, which can not always be achieved. Introducing determinism can enable applications demanding mobility and reliable service guarantees. Wireless Time-Sensitive Networking (WTSN) aims to adapt wired TSN standards to wireless networks [2].

As WTSN evolves, the shift from non-TSN to TSN devices will be gradual, with end devices being the last to adopt TSN. Non-TSN devices may produce unsynchronized data, leading to frames misaligned with the TSN schedule, as shown in Figure 1. This mismatch causes frames to remain in the transmission queue for varying periods, known as Residual Service Time (RST), resulting in higher latency and jitter [3].

For time-triggered, periodic, and highly time-sensitive traffic flows, high latency and jitter are unacceptable, making a solution necessary. Our proposed approach addresses this issue by capturing traffic frame arrivals, analyzing their generation patterns, predicting future arrival times, and using this information to continuously update the WTSN transmission schedule.

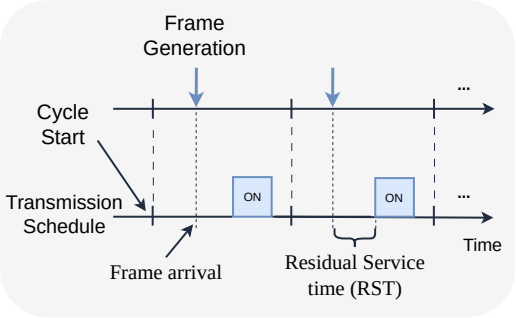

Figure 1. Residual Service Time (RST)

II. Design and Implementation

This work presents a wireless TSN-focused framework comprising traffic monitoring, a pre-processing and modeling stage, and a scheduling and configuration procedure.

A. Traffic Monitoring

The proposed control loop framework starts with the traffic monitoring at the WTSN node. Traffic monitoring has two components. The first, at the node driver level, filters frames from the monitored traffic flow and timestamps their arrivals just before transmission. These timestamps are then pushed to userspace, collected in batches, and encoded as $\mathbf{d} = [t_{\text{arr}_1} - t_{\text{arr}_{M_{\text{prev}}}}, t_{\text{arr}_2} - t_{\text{arr}_1}, \ldots, t_{\text{arr}_M} - t_{\text{arr}_{M-1}}]$, where M represents the total number of captured arrival timestamps. This encoding reduces size and simplifies the modeling process in the WTSN Controller.

B. Pre-processing and Modeling

Once the traffic flow frames timestamps differences $\mathbf{d}$, arrive at the WTSN Controller, they are first pre-processed in order to scale them using the minimum and maximum value in $\mathbf{d}$. Then, assuming the arrival timestamps differences follow a linear relationship as $\hat{\mathbf{d}} = X\beta$, where $\hat{\mathbf{d}}$ represents the predicted frame arrivals, while β are the learning coefficients. and X the design matrix $X = \begin{bmatrix}1 & 2 & 3 & \ldots & M\end{bmatrix}^T$.

To estimate the β coefficients, the well-known least squares estimator is used: $\hat{\beta} = (X^T X)^{-1} X^T \mathbf{d}$. Using this, the next $\hat{\mathbf{d}}$ differences can be predicted, allowing for the estimation of $\hat{t}_{\text{arr}}$.

C. Schedule Calculation and Setting

Predicted arrival timestamps enable optimal placement of transmission slots within the cycle. Their offsets are computed using $\text{mod}(\hat{\mathbf{t}}_{\text{arr}}, cycle)$, and slot positions are chosen based on percentile values reflecting required service levels (SL). The scheduler then programs the WTSN node accordingly. If slots from multiple clients overlap, the scheduler retains the higher-priority slot and shifts the other to the next available position; for equal priorities, a first-come, first-served rule applies.

III. Demo Setup

The proposed WTSN traffic pattern modeling scheduling method was tested using our in-house WTSN Evaluation Kit (EK), built on the openwifi IEEE 802.11/Wi-Fi baseband chip and FPGA design [4]. As shown in Figure 2, the setup includes a Xilinx Zedboard with an AD-FMCOMMS4-EBZ transceiver as the STA and a Xilinx ZC706 board with an AD-FMCOMMS4-EBZ transceiver as the AP. Both devices connect via Ethernet to a service switch for control and configuration.

Two Intel NUC nodes are connected to the service switch, one managing CNC Centralized Network Controller (CNC) and Centralized User Configuration (CUC) functions and the other serving as a monitoring dashboard. The EK also includes TSN features like Precision Time Protocol-based synchronization and a TSN gating system within the wireless nodes [5].

The network control and configuration links use a ZeroMQ PUB/SUB model with agents in all WTSN nodes' userspace. This setup serves three functions: i) transmitting time-sensitive traffic arrival data to the WTSN controller, ii) applying the schedule defined by the WTSN scheduler, and iii) measuring wireless link latency in traffic flows. Furthermore, retransmissions are disabled across all nodes to prevent interference with latency measurements.

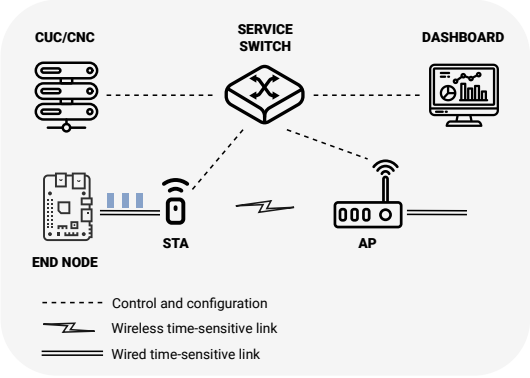

Figure 2. Hardware Setup

IV. Measuring Procedure and Results

The testing traffic flow is generated at the End Node by an unsynchronized traffic generator using the $time.perf\_counter()$ Python function with a generation cycle of 8192 μs. To evaluate the effectiveness of the proposed framework, the real-time end-to-end link latency is recorded and displayed on a dashboard, as shown in Figure 3.

Since the TSN schedule is also set to 8192 μs, given the described RST, the latency fluctuates between 0 and 8192 μs when the controller is inactive. However, when the controller is active, a 96.7% reduction in end-to-end latency is achieved at the 90th percentile.

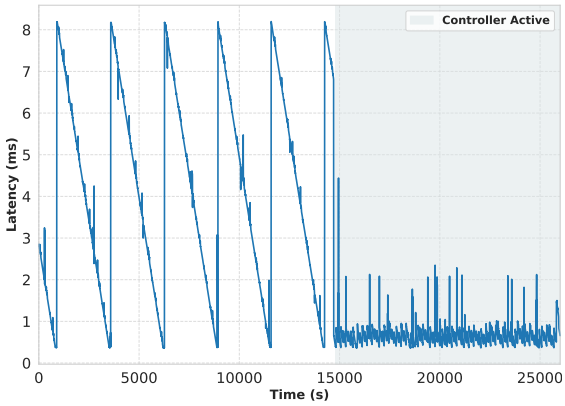

Figure 3. Dashboard end to end latency real time plotting

V. Conclusion

This work demonstrates how a traffic pattern learning framework based on linear regression can predict the optimal wireless TSN schedule for non-synchronized traffic flows. The framework was developed and tested using our openwifi setup, where a central CUC/CNC unit collects traffic behavior data, models it, and makes scheduling decisions. Simultaneously, the wireless end-to-end latency is measured and displayed in real-time on a dashboard, allowing a clear comparison between scenarios with and without the mechanism active.

Acknowledgment

This research was partially funded by the Flemish Government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen" program.

References

[1] K. Anil, "Adoption rate of wireless tsn is expected to ramp up quickly for industrial iot applications," 3 2023. [Online]. Available: https://networkbuilders.intel.com/blog/adoption-rate-of-wireless-tsn-is-expected-to-ramp-up-quickly-for-industrial-iot-applications

[2] D. Cavalcanti, C. Cordeiro, M. Smith, and A. Regev, "Wifi tsn: Enabling deterministic wireless connectivity over 802.11," *IEEE Communications Standards Magazine*, vol. 6, pp. 22–29, 12 2022.

[3] P. Avila-Campos, J. Haxhibeqiri, M. Girmay, I. Moerman, and J. Hoebeke, "Residual service time optimization for legacy wireless-tsn end nodes," in *WiMob 2023 - The 19th International Conference on Wireless and Mobile Computing, Networking and Communications.*, 2023. [Online]. Available: http://www.ieee802.org/1/tsn

[4] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman, "Openwifi: A free and open-source IEEE802.11 SDR implementation on SoC," *IEEE Vehicular Technology Conference*, vol. 2020-May, 5 2020.

[5] M. Aslam, W. Liu, X. Jiao, J. Haxhibeqiri, G. Miranda, J. Hoebeke, J. Marquez-Barja, and I. Moerman, "Hardware efficient clock synchronization across wi-fi and ethernet-based network using ptp," *IEEE Transactions on Industrial Informatics*, vol. 18, pp. 3808–3819, 6 2022.