Automatic Closed-Loop Control Applied to TSN Scheduling

Alitzel Torres[1,2], Antonio Ramírez[2], José Luis Briz[1], Juan Segarra[1], Álex Gracia[1], Héctor Blanco[3]

[1] Affiliation: Grupo de Investigación en Arquitectura de Computadores (gaZ)
Instituto de Investigación en Ingeniería de Aragón (I3A)
Universidad de Zaragoza, Mariano Esquillor s/n, 50018, Zaragoza, Spain.
Tel. +34-976762707, e-mail: 923983@unizar.es
[2]Affiliation: CINVESTAV Unidad Guadalajara.
[3]Affiliation: Intel Deutschland GmbH.

Abstract

A Time-Aware Shaper (TAS) ensures ultra-low jitter and bounded delays in Time-Sensitive Networks (TSN), assigning streams to queues and opening/closing gates through a Gate Control List (GCL). We propose a novel TAS implementation with a closed-loop controller replacing the GCL, enhancing robustness and network utilization while meeting critical requirements.

Introduction

Time-Sensitive Networks (TSN) follow standards that enable deterministic communication for real-time applications over Ethernet or WiFi, allowing traffic with different priorities to coexist. We distinguish between non-critical (*Best-Effort*, BE) and critical traffic. A TSN system consists of *end-stations* (sources and destinations of frames) and *bridges* (data link layer interconnection devices with time synchronization and traffic shaping capabilities). An IEEE 802.1Qbv Time-Aware Shaper (TAS) assigns incoming traffic to queues sharing a common egress port. Transmission windows for each queue are regulated by per-queue gates that open and close following a Gate Control List (GCL) synthesized from a schedule solution.

Typically, the systems under consideration schedule time windows offline to transmit critical streams and use the remaining time to accommodate (unscheduled) BE traffic. Still, offline-computed schedules tend to be rather conservative, as they assume the worst-case scenario for clock skew, frame size, frame-processing time in *bridges*, to name a few, causing the windows to be larger than the actual size of the frames (Figure 1). Therefore, our principal objective is to reclaim for BE traffic the slack inside the conservatively scheduled windows.

Proposed Approach

We present a novel proposal for implementing a TAS based on a closed-loop controller that opens and closes the gates, adjusting the windows to the actual critical frames size, and replacing the TAS automaton that opens and closes according to the GCLs. We first model the TSN using Timed Continuous Petri Nets (TCPN), extending the bottom-up methodology in [1] and focusing on representing TSN elements by TCPN modules. Afterward, these modules are merged, following the network topology, to obtain a monolithic model representing the TSN system behavior.

The equations describing the dynamic behavior of the TCPN allow different control techniques to be applied for opening and closing the TAS gates. In addition, these techniques increase the robustness of the system against manageable disturbances and parametric variations. We propose the use of *backstepping* control [2], as this technique is well-suited to the system given its characteristics, although others could be considered. Figure 2 outlines our control scheme.

Starting from the conservative windows w, given by the offline-computed schedule, we adjust their starting and ending times to open gates only when critical frames are ready for transmission. Updated windows ω are translated into control references r, which specify how many bits in the scheduled frames must pass through each gate at each instant. The controller compares these references to the actual frames each gate is attending and uses the equations from the TCPN model to adjust any discrepancies, activating the gates so that the correct number of bits passes at each instant.

Case Study

We consider an in-line network topology consisting of 2 *bridges* with 2 *end-stations* each. Six critical streams are scheduled using the method in [3], although others could be employed, considering a maximum clock skew and a processing time of frames in *bridges* of 2 µs. Figure 3 shows some results obtained during the simulation. Note that

Figures 3a and 3b are practically the same, meaning the reference obtained is tracked correctly, so the system output is the expected one.

Conclusions

We propose a controller to replace the IEEE 802.1Qbv TAS automaton of a TSN in opening and closing gates, which tracks the error between the expected and actual critical traffic. It enables the implementation of dynamic schedulers, recovering bandwidth for non-critical traffic (BE) by tightening the windows of critical traffic and adjusting the schedule without stopping the TSN system to deploy the new TAS GCLs.

Acknowledgements

This work was supported by MCIN/AEI/10.13039/501100011033 (grant PID2022-136454NB-C22), and by Dept. of Science, University and Knowledge Society, Government of Aragon (grant to reference research group T58_23R), Spain. A. Torres was supported by SECIHTI (grant 1141966), Mexico.

REFERENCES

[1]. TORRES-MACÍAS, A.G. et al. Modeling Time-Sensitive Networking Using Timed Continuous Petri Nets. *IFAC-PapersOnLine.* 2024, 58(1), 300-305.

[2]. VAIDYANATHAN, S. and AZAR, A.T. *Backstepping Control of Nonlinear Dynamical Systems*. Elsevier Science, 2020.

[3]. TORRES-MACÍAS, A.G. et al. Optimal and Fast IEEE 802.1Qbv Incremental Scheduling. *Manuscript under review*, 2025.

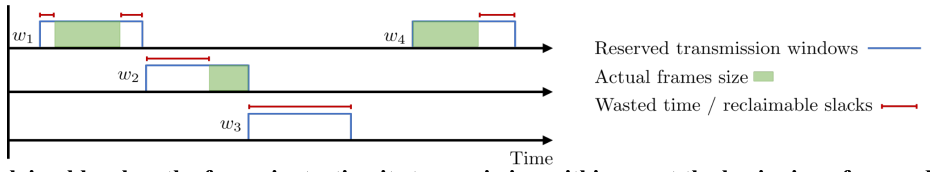

w_1
w_4
w_2
w_3
Time
Reserved transmission windows
Actual frames size
Wasted time / reclaimable slacks

Figure 1: Slack reclaimable when the frame is starting its transmission within w$_1$, at the beginning of w$_4$, and ends it at the end of w$_2$. If the frame is lost, the slack to be reclaimed is the whole window w$_3$.

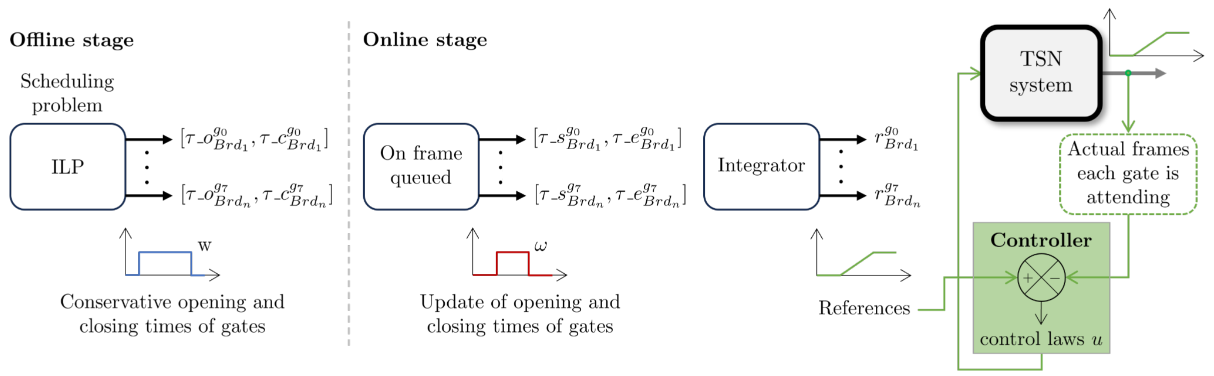
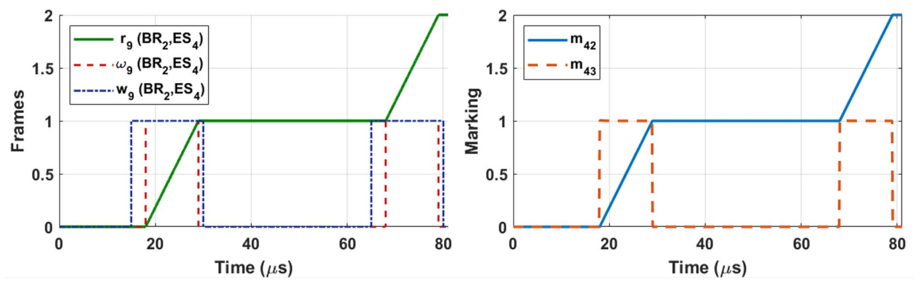

Offline stage
Scheduling problem
ILP
$[\tau\_o^{g_0}_{Brd_1}, \tau\_c^{g_0}_{Brd_1}]$
$[\tau\_o^{g_7}_{Brd_n}, \tau\_c^{g_7}_{Brd_n}]$
w
Conservative opening and closing times of gates
Online stage
On frame queued
$[\tau\_s^{g_0}_{Brd_1}, \tau\_e^{g_0}_{Brd_1}]$
$[\tau\_s^{g_7}_{Brd_n}, \tau\_e^{g_7}_{Brd_n}]$
ω
Update of opening and closing times of gates
Integrator
$r^{g_0}_{Brd_1}$
$r^{g_7}_{Brd_n}$
References
TSN system
Actual frames each gate is attending
Controller
control laws u

Figure 2: Control scheme.



(a) Stream s$_5$ (b) Bridge BR$_2$, port ES$_4$

Figure 3: Some of our simulation results. (a): the updating windows (w→ ω) and references generation (r) processes for stream s_6. (b): marking evolution of the TCPN representing the bridge BR_2 output port to end-station ES_4, traversed by stream s_6.