Work-in-Progress: Queue Assignment and Parameter Selection in TSN Credit-Based Shapers

Tamim Ahmed and Monowar Hasan
School of EECS, Washington State University, Pullman, WA, USA
Email: tamim.ahmed@wsu.edu, monowar.hasan@wsu.edu

***Abstract*—Time-Sensitive Networking (TSN) provides deterministic guarantees and is suitable for real-time packet transmissions. TSN switches maintain a set of queues to process application flows. Many existing designs assume the assignments of traffic flows to switch queues are known. Our research explores cases where the flow-to-queue assignments cannot be known a priori. Specifically, we consider a TSN switch compliant with the IEEE 802.1Qav standard, which introduces the Credit-Based Shaper (CBS) for flow control. We develop an optimization-based technique to jointly determine flow-to-queue mapping and related CBS parameters. As we demonstrate in our work-in-progress, without considering TSN-specific properties, commonly used priority assignments, such as the Rate Monotonic scheme, can reduce flow admissibility.**

I. Introduction

Time-Sensitive Networking (TSN) [1] is a foundational communication technology for real-time applications that provides deterministic latency guarantees over Ethernet. Among TSN traffic-shaping mechanisms, the Credit-Based Shaper (CBS), as defined in IEEE 802.1Qav [2], is one of the most commonly used techniques. CBS regulates transmission through per-queue credit accumulation and depletion and enforces strict priority across queues. In TSN switches, each flow is assigned to one of a limited number of hardware (priority) queues. In contrast to classical fixed-priority scheduling on CPUs, where each task receives a unique priority level, TSN queues enable flow multiplexing, allowing multiple flows to share a single queue. The challenge of flow-to-queue assignment in TSN switches, particularly those managed by CBS, has received limited attention in the literature.

Conventional priority assignment strategies, such as Rate Monotonic (RM) [3] and Audsley's algorithm [4], were originally developed for processor scheduling. They do not consider TSN-specific characteristics, including non-preemptive transmission, flow multiplexing, and CBS parameters. Direct application of these schemes to TSN may result in suboptimal or infeasible configurations, especially at high utilizations. As illustrated in Fig. 1 and discussed in Section III, RM-based assignments can result in deadline violations even when alternative flow-to-queue mappings exist that satisfy all timing constraints.

This research is partly supported by the U.S. National Science Foundation Award 2345653. Any findings, opinions, recommendations, or conclusions expressed in the paper are those of the authors and do not necessarily reflect the sponsor's views.

To address this gap, our work-in-progress explores the problem of assigning traffic flows to switch queues. Specifically, we present an *optimization-based approach to jointly determine flow-to-queue assignments and corresponding CBS parameters*. Our preliminary evaluation indicates that jointly optimizing queue assignments while considering CBS parameters significantly improves schedulability compared to conventional priority assignment strategies. The proposed method outperforms period-based assignments such as RM, particularly in moderately to highly utilized cases. Our findings demonstrate the need for additional exploration for queue assignment and parameter selection in TSN networks.

II. System Model

A. Flow Model

We consider a set of N periodic flows, denoted by $\mathcal{F}$ where the flows are traversing through a TSN switch π_s. Each flow $F_i \in \mathcal{F}$ is characterized as follows: $F_i = \{e_i, O_i, D_i, T_i, pri_i\}$, where e_i is the worst-case packet processing delay at the egress of π_s, O_i denotes the release offset, D_i is the relative deadline, T_i represents the period (inter-arrival time), and pri_i is the (unknown) priority. We assume deadline is equal to its period in our flow model. Let p_{ij} denote the j^{th} packet instance of flow F_i where the packets are generated periodically. We denote the arrival time of packet p_{ij} as $a_{ij} = (j-1)T_i + O_i,\ \forall j \in \mathbb{N}^+$, and the packet must complete its transmission before its absolute deadline, $d_{ij} = a_{ij} + D_i$. The payload size of the packet is denoted by $\hat{l}_{ij}$. Given the transmission rate R_s of the egress port of switch π_s, the corresponding packet processing delay is $e_i = e_{ij} = \frac{\hat{l}_{ij}}{R_s}$. We denote the set of all packets as $\mathcal{P} = \{p_{ij} | F_i \in \mathcal{F}\}$. Packet transmissions are *non-preemptive*. The packet schedule repeats after a fixed time window, also known as a hyperperiod, defined as $H = LCM(T_1, T_2, \ldots, T_N)$. Each flow F_i will release exactly $\frac{H}{T_i}$ packets over the interval of $[0, H)$.

B. Traffic Queues and Parameters of CBS

Each switch consists of L priority queues defined as $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_L\}$. Within each queue, packets are served in FIFO order. All the queues are handled by a shaper (CBS). The priority pri_i of a flow F_i determines which queue it will be assigned to. All packets of F_i are enqueued in Q_{pri_i} where $pri_i \in \{1, 2, \ldots, L\}$. For each $Q_l \in \mathcal{Q}$, we need to define the following parameters: *(a)* $IdleSlope$ (I_l)—the rate at which credit increases when the queue is backlogged but

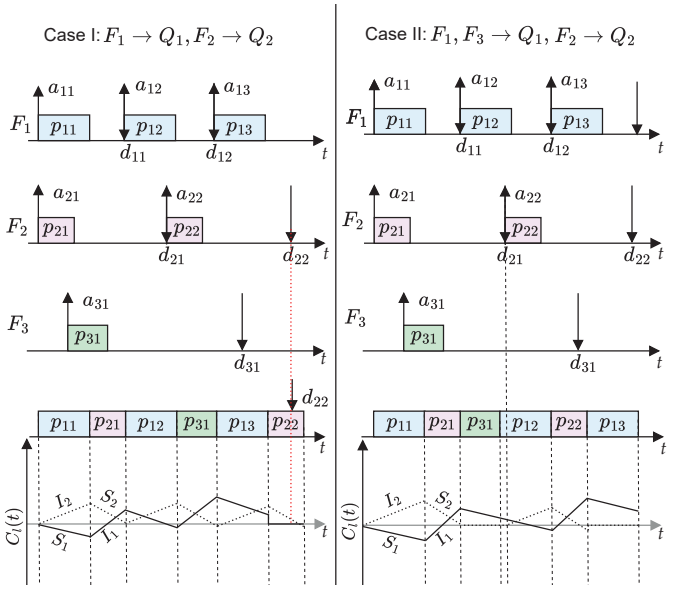

Fig. 1. Impact of priority assignment on CBS scheduling. The period-based assignment leads to a deadline violation (red line) due to interference from higher-priority flows (Case I, left), whereas another assignment exists in which both flows are schedulable (Case II, right).

not transmitting, *(b) SendSlope* (S_l)—the rate at which credit decreases while the queue is transmitting, and *(c)* $C_l(t)$—the credit of Q_l at time t. The CBS must satisfy the following condition:

$$S_l = I_l - R_s, \text{ with } I_l > 0 \text{ and } S_l < 0. \tag{1}$$

The slopes normalized by link rate are represented as follows: $\bar{I}_l = \frac{I_l}{R_s}$ and $\bar{S}_l = \bar{I}_l - 1$.

As defined in IEEE 802.1Qav [2], each queue evaluates the credit variable $C_l(t)$ for three scenarios.

- *Transmission:* When a queue Q_l is transmitting packets the credit becomes $\frac{dC_l}{dt} = \bar{S}_l = \bar{I}_l - 1$,
- *Backlog:* When the queue is not empty and blocked by a higher-priority queue, its credit is $\frac{dC_l}{dt} = \bar{I}_l$, and
- *Idle:* When the queue is empty the credit is zero, i.e., $\frac{dC_l}{dt} = 0$.

When a queue becomes empty, its credit is clipped to:

$$C_l(t) \leftarrow \max(C_l(t), 0). \tag{2}$$

A queue Q_l is eligible to transmit if $C_l(t) \geq 0$ (and the queue is not empty). The queues follow a priority-ordering: if Q_i has higher priority than Q_k and C_i, $C_k \geq 0$, then the packets from Q_i will always be transmitted first.

III. Problem Overview

We want to determine which queues the flows will be mapped to and the CBS parameters required for the priority assignment to work. Let us start with a simple example with three flows, F_1, F_2, and F_3, and their periods $T_1 < T_2 < T_3$. Consider the following packet instances for F_1, F_2, and F_3, respectively: $\{p_{11}, p_{12}, p_{13}\}$, $\{p_{21}, p_{22}\}$, and $\{p_{31}\}$, as illustrated in Fig. 1. In Case I (left of Fig 1), the flows are assigned using the RM policy. As $T_1 < T_2 < T_3$, F_1 is assigned to Q_1 and F_2, F_3 are assigned to Q_2 where Q_1 has higher priority than Q_2. Due to priority-driven, non-preemptive transmission in TSN, at $t = 0$, p_{11} from Q_1 will start its transmission, and its credit ($C_1(t)$) will start decreasing. After the transmission of p_{11}, $C_2(t) > 0$ and p_{21} from Q_2 is scheduled to transmit. Subsequently, p_{12}, p_{31}, p_{13}, and p_{22} are transmitted (because the packets in each queue are dispatched in a FIFO order). However, for this transmission order, packet p_{22} missed the deadline. In contrast, an alternative flow-to-queue mapping exists that ensures that all packets meet their deadlines, as shown in Case II (right side of Fig. 1). For instance, if we assign F_1 and F_3 to Q_1 and F_2 to Q_2, p_{22} experiences less interference, and there is no deadline miss. This simple illustration shows that arbitrary priority assignment can lead to deadline violations and reduced flow admissibility; therefore, a systematic exploration is necessary.

IV. Parameter Selection Using ILP

We formulate an optimization problem to determine flow priorities (i.e., the mapping of flows to queues). Our goal is to jointly determine: flow-to-queue assignments (pri_i), flow release offsets (O_i), and initial credit values of each queue ($\bar{I}_l$). We need to obtain the *(a)* flow-queue assignments (discrete choices), and *(b)* the amount of credit each queue receives as well as the timing of packet transmission (continuous choices). We obtain these by formulating an Integer Linear Program (ILP). We now introduce the optimization constraints (Section IV-A) and objective function (Section IV-B).

A. Optimization Formulation: Constraints

We consider four types of constraints as presented below.

1) Queue Assignment and CBS Configuration: We need to decide how flows are mapped to each queue and how their credits are allocated. Let x_{il} be a binary decision variable, where $x_{il} = 1$ represents that flow F_i is assigned to queue Q_l. Since each flow can be assigned to exactly one queue, we have the following constraint:

$$\sum_{Q_l \in Q} x_{il} = 1, \quad \forall F_i \in \mathcal{F}. \tag{3}$$

We also need to determine the $IdleSlope$ for each flow. The $IdleSlope$ for queue Q_l, denoted as $\bar{I}_l$, is a decision variable in our formulation. We store this credit information using an auxiliary variable, β_i where $\beta_i = \sum_{Q_l \in Q} x_{il} \bar{I}_l$. Note that multiple flows can be assigned to the same queue, as queues support multiplexing. When a queue serves multiple flows, the total utilization of these flows can not exceed the overall credit of the queue. Hence, we have the following constraints that ensure total credit allocation does not exceed the link capacity:

$$\sum_{F_i \in \mathcal{F}} x_{il} \frac{e_i}{T_i} \leq \bar{I}_l, \quad \forall Q_l \in Q \text{ and} \tag{4}$$

$$\sum_{Q_l \in Q} \bar{I}_l \leq 1. \tag{5}$$

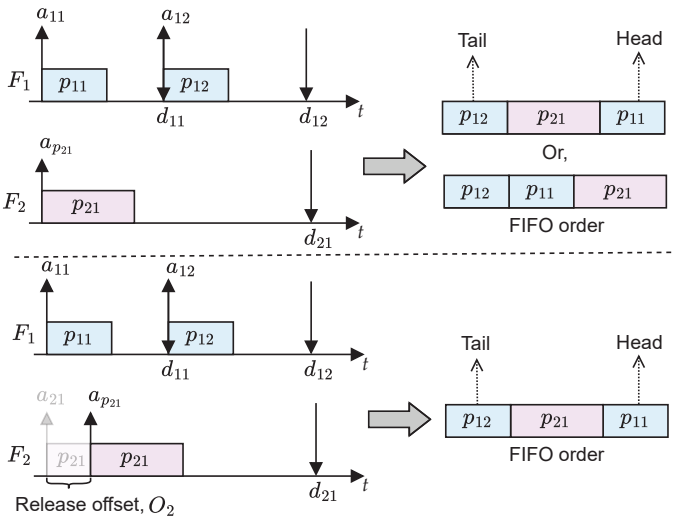

Fig. 2. Simultaneous flow arrival can introduce non-determinism (top figure), which can be prevented by introducing release offsets (bottom figure).

One challenge is to ensure that not many (or all) flows are mapped to one queue while others remain unutilized. The following constraint ensures that each queue must serve at least one flow and at most $\lceil \frac{N}{L} \rceil$ flows to prevent queue flow starvation or queue under-utilization.

$$1 \leq \sum_{F_i \in \mathcal{F}} x_{il} \leq \left\lceil \frac{N}{L} \right\rceil, \quad \forall Q_l \in Q. \tag{6}$$

2) Temporal Requirements: When multiple flows are mapped to the same queue (multiplex), they are served in a FIFO order. But for simultaneous packet arrivals, as can occur with periodic flows, this introduces nondeterminism in our formulation. Figure 2 (top part) shows a case in which simultaneous packet arrivals result in two different enqueuing orders. We can prevent this by adding a small offset and delaying the flow release as shown in the bottom part of Fig. 2. For instance, an offset O_2 for flow F_2 results in a deterministic FIFO enqueueing order.

For any packet p_{ij}, its arrival is given by: $a_{ij} = (j-1)T_i + O_i$. Here, O_i, another decision variable in our formulation, represents the release offset. A packet can only start transmission after its arrival, and the processing is assumed to be less than its deadline. We formally represent these two constraints as follows:

$$s_{ij} \geq a_{ij}, \quad \forall p_{ij} \in \mathcal{P} \quad \text{and} \tag{7}$$
$$s_{ij} + e_{ij} \leq d_{ij}, \quad \forall p_{ij} \in \mathcal{P}, \tag{8}$$

where s_{ij} denotes the transmission start time of packet p_{ij}.

3) Flow Ordering: The egress port can process only one packet at a time, since the transmission is non-preemptive. Let us introduce a binary variable y_{ijkl} where $y_{ijkl} = 1$ presents p_{ij} transmits before p_{kl}. Hence, we have the following constraints:

$$s_{ij} + e_{ij} \leq s_{kl}, \text{ when } y_{ij,kl} = 1 \quad \text{and} \tag{9}$$
$$s_{kl} + e_{kl} \leq s_{ij}, \text{ when } y_{ij,kl} = 0 \tag{10}$$

that satisfies the non-preemption requirements. In addition, the following condition enforces that the j^{th} instance of flow F_i is always scheduled before the $(j+1)^{th}$ instance.

$$s_{ij} + e_{ij} \leq s_{i(j+1)}, \quad \forall F_i \in \mathcal{F}. \tag{11}$$

As noted before, flows in the same queue use FIFO, but as the switch has priority queues, flows in different queues are served based on their priority. We define a binary variable σ_{ik} where $\sigma_{ik} = 1$ implies F_i and F_k share a queue. Since queue assignments (x_{il}) are decision variables, we can detect the co-assignment in the same queue using $\sigma_{ik} = \sum_{Q_l \in Q} w_{ikl}$, where $w_{ikl} \leq x_{il}$, $w_{ikl} \leq x_{kl}$, and $w_{ikl} \geq x_{il} + x_{kl} - 1$. The (unknown) variable O_i is the release offset that prevents non-determinism when flows arrive simultaneously. In particular, the offset for a pair of flows must differ by at least 1. We introduce a binary variable δ_{ik} to decide the direction of F_i and F_k (i.e., either F_i has a smaller offset than F_k or vice versa). The following constraints ensure that when both flows are in the same queue ($\sigma_{ik} = 1$), their offsets differ.

$$O_i - O_k \geq 1, \text{ when } \sigma_{ik}{=}1 \wedge \delta_{ik}{=}1, \quad \text{and} \tag{12}$$
$$O_k - O_i \geq 1, \text{ when } \sigma_{ik}{=}1 \wedge \delta_{ik}{=}0. \tag{13}$$

4) Credit Requirements: Recall that a queue can only transmit when it has positive credit, i.e., $C_l(t) > 0$. The credit decreases during transmission. A queue can accumulate credits over time in order to transmit a packet. For packets p_{ij} and p_{kl} that share a queue, we need to ensure that the difference in their start times is long enough for the queue to recover its transmit credits. We formally present this requirement using the following constraints:

$$\beta_i\,(s_{kl} - s_{ij}) \geq e_i - M\,(2 - y_{ij,kl} - \sigma_{ik}), \quad \text{and} \tag{14}$$
$$\beta_k\,(s_{ij} - s_{kl}) \geq e_k - M\,(1 + y_{ij,kl} - \sigma_{ik}), \tag{15}$$

where M denotes a sufficiently large number necessary to relax the constraints in the ILP formulation. When p_{ij} transmits before p_{kl} (i.e., $y_{ijkl} = 1$ and $\sigma_{ik} = 1$), Eq. (14) becomes effective and if the transmission order is reversed, then Eq. (15) will be applied. In any other cases, the constraints are "relaxed" due to the presence of a large number M.

B. Optimization Formulation: Objective

The objective of our ILP formulation is the following:

$$\min \quad -w_s \sum_{p_{ij} \in \mathcal{P}} sl_{ij} - w_{\bar{l}} \sum_{Q_l \in Q} \bar{l}_l + w_o \sum_{F_i \in \mathcal{F}} O_i. \tag{16}$$

The first term in Eq. (16) maximizes the slack interval $sl_{ij} = d_{ij} - s_{ij} - e_{ij}$. A larger slack means every packet finishes its transmission well before its deadline (hence, have enough safety margin before credit expiration). The second term maximizes the total $IdleSlope$ allocation. A higher value

Algorithm 1 Flow-to-Queue Mapping & Parameter Selection

Input: The set of all packets $p_{ij} \in \mathcal{P}$
Output: x_{il}, O_i, and $\bar{I}_l$ for each flow $F_i \in \mathcal{F}$ and each queue $Q_l \in \mathcal{Q}$

1: /* *Solve the ILP following the requirements presented in Section IV-A and Section IV-B* */
2: **if** `ILP.Solve()` = INFEASIBLE **then**
3: **return** UNSCHEDULABLE /* *Cannot obtain feasible solution* */
4: **else**
5: **return** x_{il}, O_i, $\bar{I}_l$ $\forall F_i \in \mathcal{F}$ and $Q_l \in \mathcal{Q}$
6: **end if**
7: /* *IEEE 802.1Qav describes schedule generation and transmission rules (e.g., start time) for given parameters* */
8: Generate schedule using (x_{il}, O_i, $\bar{I}_l$)
9: Obtain start times s_{ij} $\forall$ $p_{ij} \in \mathcal{P}$
10: Calculate response time: $R_{ij} = s_{ij} + e_{ij} - a_{ij}$, $\forall p_{ij} \in \mathcal{P}$
11: **if** $R_{ij} \leq D_i, \forall$ $p_{ij} \in \mathcal{P}$ **then**
12: **return** SCHEDULABLE /* *All packets meet deadlines* */
13: **else**
14: **return** UNSCHEDULABLE
15: **end if**

of $\bar{I}_l$ means Q_l accumulates its credit at a faster rate. The last term minimizes offsets for each flow, since large offsets delay the flow start, which in turn can cause deadline misses. The parameters w_s, $w_{\bar{I}}$, and w_o are weights that represent the relative importance of the three dimensions (the slack maximization, $IdleSlope$ allocation, and offset minimization).

C. Parameter Selection

Algorithm 1 presents the overall workflow. We first solve the ILP problem and obtain the flow-to-queue assignments, release offsets, and CBS idle-slope parameters; or report infeasibility (Lines 2–6). Based on these parameters, we determine packet schedule and start times as listed in IEEE 802.1Qav (Lines 8–9). While we developed an algorithm to formally generate schedules and start times, we omit the details due to space constraints. Based on packet start times and their schedules, we calculate packet response times and determine their schedulability (Lines 10–15).

V. Initial Findings

We compare our flow-to-queue mapping with one that assigns flows to queues based on RM order. In our experiment, we randomly generated $N = 16$ periodic flows with a target system utilization, $U = \sum_{i=1}^{N} \frac{e_i}{T_i}$. We generated utilization for each flow using UUnifast algorithm [5]. The overall target utilization ranged from $\{0.4, 0.5, 0.6, \ldots, 1.0\}$. We selected the flow periods from $\{50, 100, 200, 400\}$ ns. For each utilization level, we generated 50 independent flow sets. We assume a TSN switch has 8 queues. For the RM assignment, the flows were sorted by their periods and distributed equally to each queue, and the credit values $\bar{I}_l$ are chosen uniformly across all the queues where $\sum_{l=1}^{8} \bar{I}_l \leq 1$. We used Gurobi [6] to solve the ILP problem. The weight parameters w_s, $w_{\bar{I}}$, and w_o were set to 100, 10, and 1, respectively.

Figure 3 presents the percentage of schedulable flows for RM and our proposed approach. At a lower utilization rate ($U < 0.5$), both methods show similar results. When the utilization increases (i.e., $U > 0.5$), our mapping significantly outperforms RM. Since RM does not account for shaping parameters (e.g., credits), it does not perform well compared to the ILP-based one for highly utilized scenarios that jointly obtain scheduling parameters.

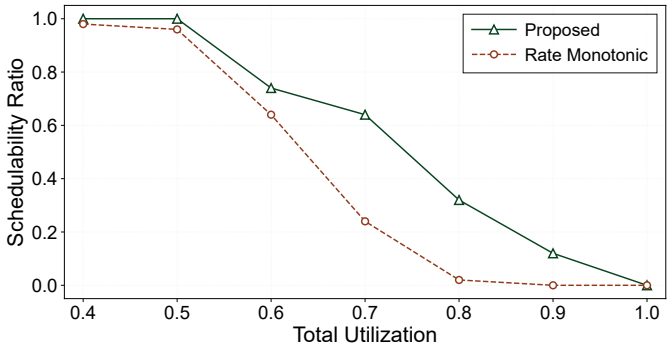

Fig. 3. Flow schedulability comparison for proposed vs RM assignment for 16 flows. The proposed technique can increase flow admissibility when utilization is above 50%.

VI. Conclusion and Ongoing Work

This work explores the problem of priority selection for periodic hard real-time flows in a TSN switch. While our preliminary findings indicate the need to jointly optimize switch parameters, this work is still in its early stages and can be extended in several ways. Our current formulation checks credits for each queue individually. A better solution is possible that can jointly calculate credits for all flows and determine their response times by checking each queue's credit individually from the ILP solver. Improving the optimization formulation is part of our ongoing work.

Our initial formulation considers flow mapping and parameter selection for a single switch. However, because flows are routed through several switches, a global, network-wide mapping is needed to ensure an "end-to-end" guarantee. In this case, a flow may not be mapped to a particular queue at every switch it traverses before reaching the destination. The problem becomes more challenging if the routes are unknown. Finding such a global solution, including optimal route determination, is left for our future work.

References

[1] IEEE 802.1 Working Group, "Time-sensitive networking (TSN) task group." https://1.ieee802.org/tsn/.

[2] "IEEE standard for local and metropolitan area networks—virtual bridged local area networks amendment 12: Forwarding and queuing enhancements for time-sensitive streams," *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pp. 1–72, 2010.

[3] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[4] N. C. Audsley, A. Burns, M. F. Richardson, and A. J. Wellings, "Hard real-time scheduling: The deadline-monotonic approach," *IFAC Proceedings Volumes*, vol. 24, no. 2, pp. 127–132, 1991.

[5] E. Bini and G. C. Buttazzo, "Measuring the performance of schedulability tests," *Real-time systems*, vol. 30, no. 1, pp. 129–154, 2005.

[6] Gurobi Optimization, LLC, "Gurobi optimizer reference manual," 2023. Version 10.0.