Scheduling Rate Constrained traffic in End Systems of Time-Aware Networks

Oana Hotescu
ISAE-SUPAERO, University of Toulouse
Toulouse, France
Email: oana.hotescu@isae-supaero.fr

Anaïs Finzi
TTTech Computertechnik AG
Vienna, Austria
Email: anais.finzi@tttech.com

***Abstract*—Nowadays, most of cyber-physical systems in avionics, automotive or recent Industry 4.0 domains require networked communication for mixed-critical applications. Ethernet-based networks such as AFDX, TTEthernet or TSN are capable to support transmission of both safety-critical and non-critical flows. This paper focuses on the TTEthernet network compliant with the avionics ARINC 664-P7 standard supporting time-triggered communication (TT) together with rate-constrained (RC) and best-effort (BE) traffic. Due to a global synchronization, TT communication with low latency and minimal jitter is ensured with static schedules computed offline. For event-triggered RC flows, bounded jitter at the source and end-to-end latency are guaranteed with worst-case analysis methods. With the increasing demands of applications, flows with Quality of Service (QoS) requirements such as video or audio may be transmitted as BE flows. However, on current configurations, no guarantees are offered to BE flows. In this paper, we aim at increasing the maximum RC utilization and improving the QoS of BE flows to allow the transmission of video or audio traffic with low jitter and end-to-end delay requirements. For this, we focus on the scheduling mechanisms and propose a scheduling approach based on a static slotted table that is applied at end systems. This table integrates the TT schedules usually obtained with Satisfiability Modulo Theories (SMT) approaches and establishes offsets of RC flows that reduce the end-to-end delay of BE flows. Several strategies for offset computations are proposed based on the distribution of flows locally at end system or globally at switch. We show that local strategies perform better than the global ones to reduce end-to-end delay of BE flows.**

I. Introduction

In domains such as avionic, automotive, or emerging industrial automation, safety-critical operations are executed by distributed cyber-physical systems interconnected by specific networks and buses guaranteeing bounded communication latency and quality of control. In the last two decades, a particular interest has been shown for the design of real-time networks based on switched Ethernet to ensure communication for this kind of systems. Technologies like the Avionics Full-duplex switched Ethernet (AFDX) following the ARINC 664-7 standard [1], TTEthernet (SAE AS6802) [2] and more recently Time-Sensitive Networking (TSN), specified by the IEEE 802.1 standard [3], seek to provide deterministic and real-time transmission for time-critical flows.

With the increasing demands in terms of applications, the high critical flows may need to share the network with flows of lower criticality or even with traditional best-effort flows for which the network has to provide some Quality of Service (QoS) guarantees. For instance, in the avionic context, it is envisioned that the transmission of flight control and command flows, as well as video flows from surveillance cameras or maintenance flows to be supported by the same shared network. Thus, the support of this converged transmission becomes a very challenging aspect in the design of time-aware networks such as AFDX, TTEthernet and TSN. The classification, isolation and scheduling of different types of flows become the key to address this challenge.

The deterministic guarantees required by the high critical applications in terms of bounded communication latency is a very important aspect. To satisfy this requirement and achieve certification of the system, a proof of correctness is needed to verify the temporal behaviour of flows. This verification can be performed by means of analysis methods such as Network Calculus [4]–[6], Trajectory Approach [7], Compositional Performance Analysis [8] or Forward End-to-End Analysis [9]. This methods consider worst-case scenarios to determine upper bounds on the end-to-end delays.

Scheduling mechanisms can be introduced in the emitting devices and switches to handle different types of traffic. The choice of the scheduling strategy depends on the reference of time available in the network, on how many types of flows has to be scheduled and on the timing guarantees required by each type of traffic. In synchronous networks such as TTEthernet or TSN, time-triggered schedules can be defined based on a network-wide notion of time, while the asynchronous AFDX network allows only event-triggered policies.

AFDX has been originally designed to carry avionics data and consequently AFDX switches offer on the egress ports only two FIFO queues of high and low priority treated in a strict priority manner. However, recent research has focused on other event-triggered scheduling strategies such as Deficit Round Robin (DRR) [10] or Weighted Round Robin (WRR) [11] to support the introduction of several types of traffic in AFDX. At transmitter level, the scheduling policy has to ensure upper bounded jitter for the avionics flows according to the ARINC 664-7 standard. A promising solution is the one presented in [12] based on a local time-triggered table scheduling relying on the internal clock of end devices. This solution ensures bounded jitter for avionics flows and reduces the pessimism of end-to-end delay bounds while keeping the asynchronous profile of flows if combined with an event-triggered scheduling at switches.

TTEthernet is a standard designed to offer strict deterministic guarantees to real-time traffic through the synchronous Time-Triggered (TT) traffic and two traffic classes of asynchronous Rate-Constrained (RC) traffic inherited from the AFDX standard. In addition, TTEthernet enables the transmission of non-time-sensitive Best-Effort (BE) traffic. The determinism of TT traffic is ensured via offline communication schedules based on the global clock synchronization, ensuring a contention-free and precise delivery of critical frames across a switched multi-hop network. For RC traffic, determinism is ensured via a strict shaping and policing of the traffic in the devices of the network such as for the AFDX network.

In this paper, we will consider the avionic context. The main goal is to adapt the table scheduling proposed for AFDX at transmitter level to the case of TTEthernet in order to guarantee bounded end system (ES) jitter for full compliance with the ARINC 664-P7 standard. Considering TT schedules derived with the SMT method, we propose several table strategies to schedule RC flows that i) guarantee bounded ES jitter, ii) reduce end-to-end delay bounds and iii) enhance QoS of BE flows. Each scheduling strategy is evaluated by worst-case analysis with Network Calculus for RC and by simulation for BE on representative scenarios. We show that with task-synchronized table strategies, the worst-case delay of RC flows is improved in average by 30% and the maximum number of flows in an ES can be multiplied by up to 4. For a low end-to-end delay of BE flows which is a mandatory requirement if these flows are video or audio, we show that table strategies considering local distribution of RC flows at end systems performs better than strategies considering distribution of RC flows in switches.

This paper is organized as follows. First, we present a background on TTEthernet in Section II, followed by the related work in Section III. Then, we introduce the network model in Section IV. Table scheduling algorithms are defined in Section V for which a worst-case analysis is formulated with network calculus in Section VI. The performance evaluation of different table schedules is shown in Section VII before concluding the paper by Section VIII.

II. TTEthernet overview

TTEthernet [2] is a deterministic, synchronized and congestion-free network protocol based on the IEEE 802.3 Ethernet standard and compliant with the AFDX ARINC 664-P7 specification [1] coming from the avionics area.

TTEthernet networks are defined as multi-hop networks composed by end systems interconnected by switches and bidirectional physical communication links. Figure 1 is an example of a TTEthernet network with 4 switches, 12 end systems and 18 communication links. Unidirectional data flows can be sent between a sending end system and multiple receiving end systems through predefined data flow paths with the implementation of virtual links (VL) inherited from ARINC 664-P7. Data between the sender and receivers is communicated through VLs by means of frames.

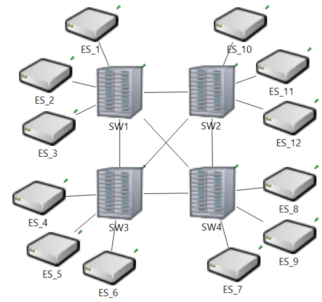

Fig. 1: TTEthernet network architecture

TTEthernet distinguishes between two traffic categories: the time-triggered (TT) and the event-triggered (ET) traffic.

Time-triggered traffic is especially suitable for applications with high criticality requirements in both temporal and safety domains. Offets are assigned statically for every TT messages in every device in their flow path to avoid any message conflicts among TT messages and to guarantee their strictly deterministic behavior. These schedules rely on a global synchronization time base established and maintained by Protocol Control Frames (PCF) that are exchanged between end systems and switches.

Event-triggered traffic includes rate-constrained (RC) and best-effort (BE) messages. RC messages follow the asynchronous communication paradigm inherited from AFDX. These messages are assimilated to sporadic messages with unknown arrival times, but minimum time intervals between consecutive instances defined at design as the Bandwidth Allocation Gap (BAG). Each RC virtual link has a BAG value in the set of powers of 2: {1, 2, 4, 8, 16, 32, 64, 128} milliseconds that is enforced by a traffic shaping at emitting end system in order to guarantee that there is enough bandwidth allocated for the transmission of VLs sharing the same physical link. Inside the network, each switch realizes a traffic policing function that checks whether, indeed, the end systems produce the well-shaped sequence of messages for each VL according to the associated BAG and a maximum frame size (MFS) defined between 64 and 1518 bytes. The switch can drop messages that are sent too early, hence, violating the minimal inter-frame gap. Based on this mastered traffic shaping, determinism of RC messages within bounded end-to-end latency can be guaranteed.

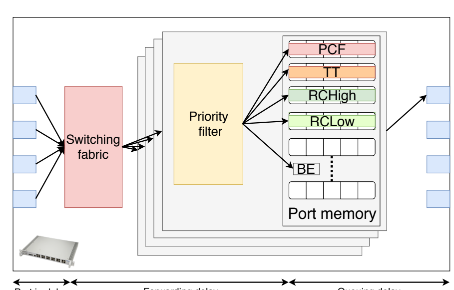

Fig. 2: Considered switch architecture [13]

Finally, BE messages are treated based on classical Ethernet BE principle. These messages are transmitted during idle intervals between TT and RC messages. Thus, it is not possible to guarantee whether or when these messages can be transmitted.

TTEthernet switches are designed to offer 8 levels of priority as illustrated in Figure 2. The synchronization PCF frames have the highest priority in the TTEthernet network, the next priority is used by the TT flows, the two next priorities are used by the AFDX RC_{HIGH} and RC_{LOW} traffic, while the remaining 4 lowest priorities are available for BE flows.

III. Related work

In the last years, significant research has been conducted in the effort to design efficient schedules to address timing constraints of mixed-critical flows with Ethernet-based real-time networks. First, for the synchronous TT traffic in TTEthernet, *Steiner* [14] proposes a scheduling method based on constraint problem formulation with a Satisfiability Modulo Theory (SMT) solver. This approach is adapted in [15] to allow the transmission of asynchronous event-triggered RC traffic by introducing empty slots for RC frames between TT slots. An alternative method for integration of RC trafc into TT scheduling based on a Tabu-search metaheuristic is proposed by *Tamas-Selicean et al.* [16] providing optimal TT schedules that maximize bandwidth allocation for RC flows.

The work of *Craciunas et al.* [17] aims to extend the end-to-end guarantees of TT messages to the application layers of end systems in order to reduce the end-to-end communication latency. The proposed solution consists in simultaneously generating the scheduling of TT messages together with the tasks executed by the processors on the end systems. Two formulations of the scheduling optimization problem are given: one based on the SMT approach and a second one derived as a Mixed-Integer Linear Programming (MILP) problem.

In the case of TSN, similar approaches have been proposed to synthesize offline schedules for the IEEE 802.1Qbv Time-aware Scheduler (TAS). Static offline schedules guaranteeing low bounded jitter and end-to-end latency are defined for TT flows with Satisfiability/Optimization Modulo Theory (OMT) based methods [18] or metaheuristics [19].

For RC asynchronous traffic, a worst-case analysis is needed in order to upper bound end-to-end delays and provide the guarantees required by applications. Various research work in the literature have been dedicated to worst-case analysis methods for successfully bounding the communication delays in AFDX [4], [7], [10]. Among these methods, an important investment have been shown in the industry for the Network Calculus (NC), which is a well-established theory [20] that has been used to successfully certify the ARINC 667-P7 standard for the Airbus A380. In TTEthernet, the previous studies on AFDX could not be directly applied for the RC traffic because of the static TT schedules. The timing analysis for RC messages considering TT flows is studied with NC in [5] for the three integration policies. In [13], NC analysis of RC traffic is integrated into SMT scheduling to reduce the impact of TT messages on the RC ones, while in [21] the analysis is extended to consider the impact of synchronisation PCF messages. For TSN, the timing analysis of AVB traffic is addressed with NC as well [6], [22].

No particular interest has been shown to the lower-priority BE traffic since it is a no guarantee traffic class. But with the increasing needs of applications, it can be interesting to achieve a good QoS for BE traffic in order to allow the transmission of soft-real time flows as BE flows for a better use of the network bandwidth. To this end, a recent study [12], [23] successfully addresses the problem of introduction of additional traffic in AFDX by designing optimal offline table schedules at emitting end systems that minimize the end-to-end latency of BE flows and improve the worst-case bounds for critical avionic flows. As for TTEthernet, to the best of our knowledge, no previous study has been dedicated to the QoS of BE traffic. A single work in [24] intends to maximize the bandwidth allocation for BE flows, but without providing guarantees on the end-to-end delay. In TSN, a recent work [25] has focused on the QoS of BE flows proposing a SMT/OMT-based solution that synthesize feasible schedules for TT traffic which increase the QoS level for the BE traffic.

In this paper, we aim at improving the BE traffic guarantees with TTEthernet and increasing the number of RC flow in the ES. Our solution extends the table scheduling method presented in [12] for AFDX end systems by integrating TT schedules designed with the optimized SMT approach [13] and combines it with the existing TTEthernet scheduling mechanism at switch. The result is RC flows with offsets only in the source end systems, contrary to the TT flows which are scheduled from source to destinations.

The main goal is to achieve a full ARINC 664-P7 compliant TTEthernet network while guaranteeing QoS to BE flows. We propose several heuristics for RC flows allocation in the table scheduling meant to reduce the end-to-end delay of BE flows and the pessimism of RC flows bounds. The end-to-end analysis of RC flows with NC in [21] will be also extended to consider the traffic behaviour imposed by tables schedules at source end systems.

IV. Network model

In this paper, we consider a TTEthernet network similar to the one depicted in Figure 1 composed of a set of end systems, switches and physical links. Each end system emits flows of data on predefined VLs. All types of traffic are considered:

- A set of PCF flows Φ^{PCF} emitting synchronization frames according to a predefined Integration Cycle IC;
- A set of TT flows Φ^{TT}: each TT flow i in the set is defined by an initial offset o_i, a period P_i a maximum frame size MFS_i and a Sending Window sw_i during which the transmission must start and end;
- A set of RC flows Φ^{RC}: each RC flow i in the set is defined by an initial offset o_i, a Bandwidth Allocation Gap BAG_i, a maximum frame size MFS_i and a maximum initial ES jitter at sender output j_i;
- A set of BE flows Φ^{BE}: each BE flow i in the set is defined by an initial offset o_i, a period P_i, and a maximum frame size MFS_i;

Given this network model, we consider global static schedules for TT flows synthesized with the improved SMT approach from [13] establishing the initial offsets for these flows. Based on these schedules, additional scheduling strategies need to be defined in order to guarantee QoS of asynchronous RC and BE flows. RC flows compliant with ARINC 664-P7, have requirements in term of bounded jitter at emitting end system output (i.e. $j_i \leq J_{max} = 500\mu$s) and of end-to-end delays that meet the deadline of avionic applications.

To ensure the ES jitter constraint of RC flows, the interval between two consecutive transmission dates, denoted Δt, has to be greater than the difference between the BAG and J_{max} as stated in Eq. (1) and depicted in Figure 3. This condition guarantees that frames do not violate the minimum inter-frame interval, avoiding them to be dropped by the switch.

$$\Delta t \geq BAG - J_{max}, \quad \Delta t = t_{i+1} - t_i \tag{1}$$

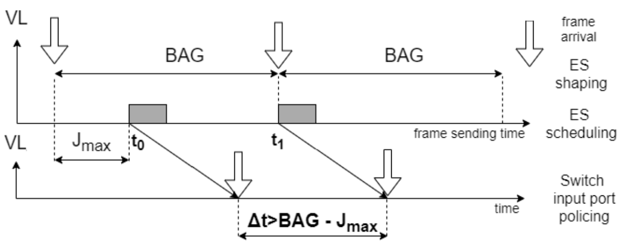

Fig. 3: Jitter at end system output port

As a result, the number of RC flows is only limited by their period and the available bandwidth. For instance, on a 1 Gbps network, in an end system without TT flows, up to 10403 (resp. 162) RC flows with periods of 128 ms (resp. 2 ms) can be scheduled as time-triggered, but only 40 RC flows can be defined as event-triggered, and guarantee the jitter constraint.

The end-to-end delay of a frame is the time required for its transmission from source to destination. It is given by the emission lag due to the queuing delay at end system scheduler and the network crossing delay. Worst-case analysis tools do not include the emission lag in the computation of the end-to-end delay considering only the network performance from source end system output to destination end system input.

For RC flows, upper bounds on the end-to-end delay are computed with worst-case analysis methods such as NC to formally verify that end-to-end delay is lower than the application deadline. For BE flows, a better QoS is achievable if the end-to-end delays are maintained as low as possible. Since these flows have a very low priority in the network, their end-to-end delay is directly related to the scheduling of flows of higher priority, namely TT and RC flows. To ensure these requirements, we focus next in this paper on the definition of schedules for the RC flows that integrated with the pre-computed TT schedules are able to reduce the end-to-end delay of BE flows. Our scheduling approach based on the table scheduling strategy will be introduced in Section V.

V. Scheduling RC flows with tables

In this section, we present a scheduling method based on the table scheduling approach aiming to decide the dates of transmission of RC flows at end system level. This approach has been proposed in [12] for AFDX flows as an efficient solution to guarantee null jitter for avionics flows and reduced end-to-end delay for both avionics and BE flows. Different levels of QoS for BE flows can be achieved, depending on the strategy used to distribute avionics flows in the table. In this paper, we aim to leverage the table scheduling strategy based on a uniform allocation heuristic that performs very close to a optimal allocation for the BE flows, according to [12].

A. Table scheduling definition

A table scheduling is designed offline to statically reserve slots for flows at emitting end system. This table is formalized in [12] as a grid composed of L lines $\times$ C columns offering time slots for the transmission of frames. The schedule is defined for a duration given by the hyper-period of flows and is cyclically repeated at run-time. This table duration is then divided in time slots of a fixed duration allowing the transmission of any frame length in the configuration. In this paper, we consider a slot duration of 15.625 μs that allows a frame of maximum length 1518 bytes to be completely transmitted at 1 Gbits/s. For a lower data rate, several slots can be reserved for a frame. Each line in the table has a duration set to 1 ms which is the greatest common divisor of flows periods, resulting in lines of 64 slots. The number of lines L will be given by the hyper-period of flows. In this table, slots are statically reserved for PCF, TT and RC flows depending on their type. The remaining slots are given to BE flows.

B. Assumptions and objectives

In order to adapt the table scheduling to the TTEthernet context, we need to consider the set of TT flows for which offsets are generated with the SMT approach based on a global synchronization established by the exchange of PCF frames. The next step consists of deciding where to place the RC flows in the table scheduling by keeping in mind the following requirements:

- Ensure bounded jitter for RC flows ($j < J_{max}$) at source end system. The table scheduling algorithm needs to be adapted in order to model the case of non-null jitter.
- Reduce end-to-end delay of RC and BE flows. The previous version of table scheduling considers a strategy based on uniform allocation of avionics flows at source end system only. In this paper, we aim at extending this approach by considering the case of a uniform allocation of RC flows at switch egress ports as well. The table schedules at source end systems will be defined based on a uniform distribution of flows at switch, thus reducing the possibility of contention and the queuing delays for both RC and BE flows. This strategy is possible on TTEthernet thanks to the global synchronization.

C. Allocation strategies

When building the table scheduling, one should take into consideration two aspects: 1) the slot reservation strategy per flow deciding the interval between frames and 2) the distribution of flows in the table depending on their initial offset setting the first slot for the reservation.

1) Slot reservation: For TT flows, the offsets computed with the SMT approach decide their first slot in the table. Starting with this slot, each frame of a TT flow receives a slot every period. For PCF frames, one slot is reserved at each integration cycle. For RC flows, a slot can be reserved every BAG. Given the fact that the table may not be synchronized with the task level, an important emission lag can be experienced by RC frames with this BAG reservation. If the RC frame arrives at the end of the reserved slot, it will have to wait for the next reserved slot for at most one BAG duration. An efficient way to reduce this emission lag and improve system reactivity is to over-reserve slots at intervals smaller than the BAG, i.e every 1 ms. Due to the RC traffic shaping preceding the scheduling at end system, RC flows will never use more than one slot per BAG. Considering table lines of a duration of 1 ms, slots assigned to a RC flow are located on a single column with these two reservations in the table such as in Figure 4. Moreover, both reservations ensure null jitter for RC flows since the interval between consecutive frames of the same RC flow is constant and equal to BAG. For more flexibility, it is also possible to allow jitter for RC flows within the limit of the maximum authorized value, J_{max}. This jitter has an impact on the disposition of slots in the same column. For frames with jitter, slots need to be shifted left or right by the main column such as in Figure 4 for $RC2$ and $RC3$.

PCF		RC1		TT1			RC2			TT2			RC3		
PCF		RC1					RC2						RC3		
PCF		RC1		TT1				RC2				RC3			
PCF		RC1					RC2						RC3		
PCF		RC1		TT1			RC2			TT2			RC3		
PCF		RC1						RC2				RC3			
PCF		RC1		TT1			RC2						RC3		
PCF		RC1					RC2						RC3		

Fig. 4: Table scheduling for 1 PCF flow, 2 TT flows and 3 RC flows (column reservation)

2) Algorithms for RC offsets computation: The computation of initial offsets for the RC flows depends on the table allocation strategy. In this paper, we use the uniform allocation strategy that distributes RC flows at regular intervals. To compute these intervals, we propose two criteria: $i)$ a constant interval is maintained between two consecutive RC flows column assignments for a uniform distribution of RC flows only and $ii)$ a constant interval is maintained between any RC flow column assignment and another critical PCF/TT/RC flow column assignment next to it. In this way, constant free intervals are created for the transmission of BE flows reducing their waiting delay for transmission.

These strategies can be applied $I)$ locally at end system or $II)$ at switch egress port. In the second scenario, a global table based on PCF synchronization is designed for flows sharing the same first switch egress port in their path. In this partially global table, the distribution of RC flows follows the strategies

Algorithm 1: Heuristic for uniform table allocation of RC flows with PCF and TT flows

Input: $\{pcfFlows\{o, ic\}\}$, $\{ttFlows\{o, P\}\}$, $\{rcFlows\{BAG\}\}$, L, C, J_{max}, $colReservation$, $uniformRC$
Output: $\{rcFlows\{o\}\}$, $table[L][C]$
foreach $l \in [1..L]$, $c \in [1..C]$ **do**
  $table[l][c] \leftarrow 0$; //init empty table
end
//place PCF frames in table
foreach $pcf \in pcfFlows$ **do**
  $l \leftarrow compute\_initial\_line(pcf.o)$;
  $c \leftarrow compute_initial_column(pcf.o)$;
  while $l< L$ **do**
    $table[l][c] \leftarrow 1$; //reserve slot (l, c) for PCF flow
    $l \leftarrow l + pcf.ic$;
  end
end
//place TT frames in table
foreach $tt \in ttFlows$ **do**
  $l \leftarrow compute\_initial\_line(tt.o)$;
  $c \leftarrow compute\_initial\_column(tt.o)$;
  while $l< L$ **do**
    $table[l][c] \leftarrow 2$; //reserve slot (l, c) for TT flow
    $l \leftarrow l + tt.P$;
  end
end
//compute offsets and place RC flows in table
$startCol \leftarrow 0$;
while *offsetNotPlaced* **do**
  //compute offsets starting from $startCol$ column
  if *uniformRC* **then**
    $compute\_unif\_RC\_offsets(table[L][C], rcFlows, startCol)$;
  else
    $compute\_unif\_intervals(table[L][C], rcFlows, startCol)$;
  end
  foreach $rc \in rcFlows$ **do**
    if *colReservation* **then**
      $intervalReservation \leftarrow 1$;
    else
      $intervalReservation \leftarrow rc.BAG$;
    end
    $l \leftarrow compute\_initial\_line(rc.o)$;
    $c \leftarrow compute\_initial\_column(rc.o)$;
    while $l < L$ **do**
      if $table[l][c] != 0$ **then**
        if $J_{max} == 0$ **then**
          $offsetNotPlaced \leftarrow true$;
        else
          $newC \leftarrow c$; // keep shifted column
          $offsetNotPlaced \leftarrow$
          $check\_slot\_left\_right(table[L][C], l, c, newC, J_{max})$
        end
      else
        $offsetNotPlaced \leftarrow false$;
      end
      if $offsetNotPlaced$ **then**
        $startCol \leftarrow startCol + 1$;
      else
        $table[l][c] \leftarrow 3$; //reserve slot (l, c) for RC flow $l \leftarrow l + intervalReservation$;
      end
    end
  end
end

in i) and ii). The offsets so decided are then transposed to the local table of the RC flow corresponding end system.

Scheduling strategies can be implemented according to Algorithm 1 and can be adapted to any type of slot reservation with null and non-null jitter. In Algorithm 1, the function $compute\_unif\_intervals$ computes uniform intervals between any PCF/TT/RC flows by dividing the total number of columns to the total number of flows in the table, while the function $compute\_unif\_RC\_offsets$ computes uniform intervals between RC flows only by dividing the total number of columns to the number of RC flows. For non-null jitter, if a slot is not available for the computed offset, left and right slots are also checked before a new offset computation.

VI. Integrating offsets into worst-case analysis

In this section, we present the framework used to compute worst-case delays for RC traffic and how we adapt it to consider the RC offsets.

A. Background on Network Calculus

The timing analysis done in this paper to compute worst-case delays is based on the Network Calculus [20]. This framework is well recognized and has been successfully used for the certification of AFDX networks [4]. It is used to compute upper bounds of delay and backlog. These bounds depend on i) the traffic arrival described by the so-called *arrival curve* $\alpha(t)$, i.e. the maximum amount of data that can arrive in any time interval, and on ii) the availability of the crossed node described by the so-called minimum *service curve* $\beta(t)$, i.e. the minimum amount of data that can be sent in any time interval. The worst-case delay is the maximum horizontal distance between $\alpha(t)$ and $\beta(t)$.

In this paper, we base our analysis on the TTEthernet model and analysis proposed in [21], Sections 5 and 8 respectively.

B. Application to table scheduling

Applying [21] to table scheduling requires two modifications: 1) on the delay in the output port of the source end system and 2) on the RC arrival curves in the first output port of the first switch in each path.

Firstly, in the source end systems, the delays in the output ports are given by: i) the flow transmission time in case of a task-synchronized RC flow or ii) the maximum interval between two slots reserved for the flow in the case of asynchronized RC flow.

Secondly, knowing the slots reserved for each RC flows, the arrival curve must be computed for each flow at the first switch egress port in the path. In the case of a BAG reservation, it is easy to use the offsets to directly compute the arrival times of the RC frames in the output port by summing the transmission time in the end system output port, the link delay, the switch ingress port delay and forwarding delay. Then the arrival curve can be computed by applying Th.11 from [21] to the RC traffic. In the case of over-reservation however, it is more complicated because not all the slots can be used within the same period. So first, we compute all the combinations of slots permutations as detailed in Section VI-C. Secondly, we apply Th.11 from [21] to all these combinations and keep the maximum of the computed values.

C. Computing RC offset permutations

For each VL in the source end system, we compute the earliest and latest arrival times such as a benchmark slot bm is used by the first frame, and then compute the earliest and latest arrival times of the following frames within the hyperperiod, as illustrated in Figure 5. We call these windows the possible arrival time windows, denoted Ψ_l, with l the number of windows after the window corresponding to bm. We consider that the slot s_n^{bm} has been assigned to vl and corresponds to the arrival window Ψ_l. A slot s_k^{bm} can be used by the vl if: i) the slot s_k^{bm} is either 1) within or 2) the first slot after Ψ_{l+1} and ii) the interval between the start of the slots s_n^{bm} and s_k^{bm} is greater or equal to the BAG of the VL.

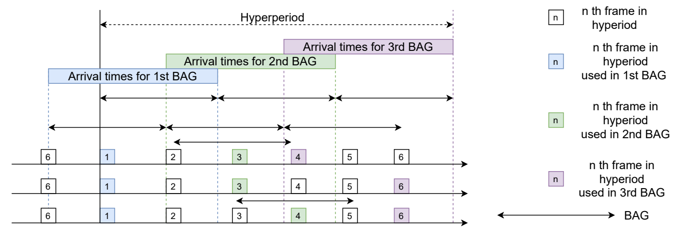

Fig. 5: Three examples of RC offset permutations

VII. Performance evaluation

In this section, we evaluate the performance of the table scheduling on the transmission of RC and BE flows. In a first time, the bounds on the end-to-end delay of RC flows are computed with the NC framework. Afterwards, the simulation approach is used to measure delays of RC and BE flows.

A. Evaluation setup

We consider for this evaluation study a realistic adaptation to 1 Gbps data rate of the Orion CEV use case described in [5], [21] and illustrated in Figure 1. The network configuration consists of 12 end systems and 4 switches transmitting 20 TT flows and 26 RC flows. The transmission of one BE flow is enabled on one of the end systems (ES_12). This BE flow follows the specification of a XGA (768 × 1024) avionics uncompressed video flow described in [12] requiring a bandwidth of 672 Mbps. The complete set of flows emitted by end system ES_12 is described in Table I.

TABLE I: Flows emitted by ES_12 under investigation

Flow	PCF12	RC9	RC11	RC13	RC16	TT3	TT4	BE
P/BAG (μs)	10000	16000	2000	2000	8000	2000	2000	18.076
MFS (Bytes)	64	424	1096	964	1379	411	64	1518

At end system level, the following table strategies are considered: local (L) and partially global (PG) allocations with uniform intervals between RC flows or between any critical PCF/TT/RC flows, with BAG reservation (BR) or column reservation (CR), with null jitter ($J0$) or non-null bounded jitter ($Jmax$).

TABLE II: Worst-case delays of RC flows on ES_12

	Worst-case end-to-end delay (μs)							
Flow	**RC9**		**RC11**		**RC13**		**RC16**	
Allocation	**sync**	**async**	**sync**	**async**	**sync**	**async**	**sync**	**async**
Table-free	-	149.30	-	170.65	-	260.46	-	227.06
L_RC_BR_J0	97.12	1697.12	113.44	313.44	200.36	400.36	171.45	971.45
L_RC_CR_J0	97.52	197.52	113.44	213.44	200.36	300.36	173.19	273.19
L_Any_BR_J0	97.16	1697.16	114.29	314.29	200.44	400.44	171.27	971.27
L_Any_CR_J0	97.54	197.54	114.29	214.29	200.44	300.44	172.98	272.98
L_RC_BR_Jmax	97.12	1700.25	113.44	313.44	200.36	400.36	171.64	971.64
L_RC_CR_Jmax	97.58	202.27	113.44	213.44	200.38	300.38	174.28	274.28
L_Any_BR_Jmax	97.16	1697.16	113.44	313.44	200.67	400.67	171.47	971.47
L_Any_CR_Jmax	100.80	202.36	118.40	219.96	204.38	305.94	176.30	277.86
PG_RC_BR_J0	97.54	1697.54	113.44	313.44	200.34	400.34	170.92	970.92
PG_RC_CR_J0	97.54	198.50	113.44	214.48	200.34	293.73	172.24	270.46
PG_Any_BR_J0	97.78	1697.78	113.44	313.44	200.65	400.65	171.35	971.35
PG_Any_CR_J0	97.78	197.78	113.44	213.44	200.65	300.65	172.68	272.63
PG_RC_BR_Jmax	96.76	1697.51	117.51	313.44	200.42	401.95	168.02	971.59
PG_RC_CR_Jmax	100.28	200.28	113.46	213.46	204.02	304.02	174.77	274.77
PG_Any_BR_Jmax	97.83	1697.83	113.44	313.44	200.71	402.27	172.11	972.11
PG_Any_CR_Jmax	100.60	200.60	113.46	213.46	204.24	304.24	175.29	275.29
Max gain sync table	34.94%	-	33.52%	-	23.07%	-	26%	-
Max gain sync vs. async (BR)	94.30%	-	63.80%	-	50.10%	-	82.70%	-
Max gain sync vs. async (CR)	50.18%	-	46.17%	-	33.19%	-	36.55%	-
Max gain CR vs. BR (async)	-	88.38%	-	32.08%	-	26.98%	-	72.18%

B. Results

First, we show results obtained with the worst-case analysis in Table II, followed by simulation results in Table III.

The worst-case analysis consider the scenario of table schedule synchronized with the application task level as well as the asynchronous case. Table II shows the worst-case end-to-end delays of the RC flows emitted by ES_12 for all table allocations. Compared to a table-free priority-based scheduling policy, delays are considerably reduced with task-synchronized table scheduling. If task synchronization is not enabled, the performance of the table scheduling depends on the reservation. On the studied configuration, the column reservation performs on average closer to the table-free policy than the BAG reservation. Despite larger worst-case delays, the ES jitter of RC flows is completely controlled by the table scheduling strategy. More generally, for the entire network configuration, compared to a table-free policy, an average gain of 30% is achieved with task-synchronized table scheduling. The use of task-synchronized tables leads to an average gain of 77% with BAG reservation and 45% with column reservation compared to the use of task-asynchronous tables. In this last case, an average gain of 60% is obtained with the column reservation in comparison to the BAG reservation. Partially global scheduling tends to perform slightly better than local scheduling as it also prevents collisions between RC frames in the first switch but the difference would be more visible on a more loaded network. No significant difference is noticed on worst-case delays between scenarios considering null jitter compared to the non-null jitter ones.

In addition to the worst-case analysis of RC flows, a simulation evaluation has been conducted for the studied network configuration. Simulation results has been obtained with an in-house OMNeT++ network simulator integrating TTEthernet features in extention to the AFDX simulator used in [12].

Table III presents the maximum measured values of jitter, emission lag and end-to-end delay as well as the offsets computed for the evaluated RC flows with all the proposed schedules. Only the case of task-asynchronous tables is shown here. To capture delay distribution of a wide range of possible scenarios, Monte Carlo simulations are performed considering random offsets between the application and the starting date of the schedule. It can be noticed that the measured end-to-end delay of any RC flow is much lower than the worst-case one and is not impacted by the allocation. However, a significant impact of the table is visible on the maximum emission lag that can reach several milliseconds for the BAG reservation. With the column reservation, the maximum emission lag is 0 most of the time or can reach at most 1 ms. As for the jitter, a bound of 50 μs has been considered in these simulations. In the non-null jitter allocations, the maximum jitter value is of 15.625μs which is within the fixed bound. With the table-free scheduling, the jitter is non-null and it is not controlled.

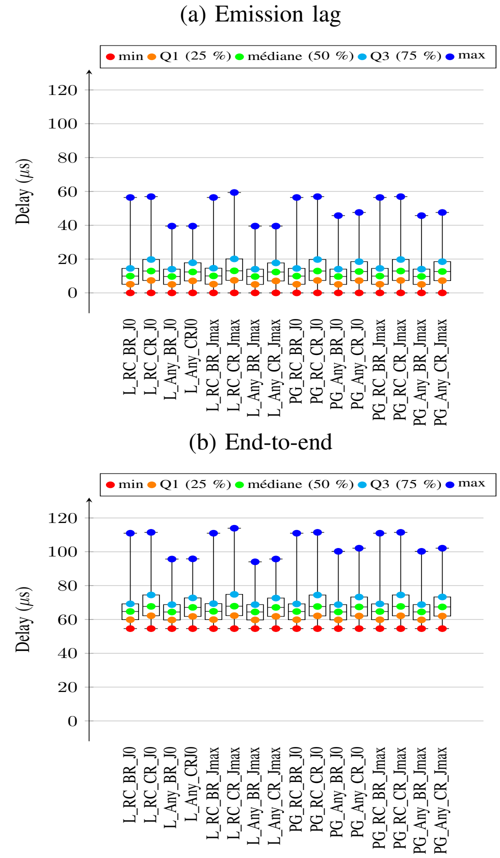

Fig. 6: Delay statistics for a BE flow

The results for the BE flow are shown in Figure 6. The statistics on both the emission lag and the end-to-end delay show a better performance with table allocations considering uniform interval between any critical flows rather than between RC flows only. Local allocations seem to perform better than the partially global ones highlighting the impact of the scheduling at emitters on the QoS of BE flows in reception.

VIII. Conclusion

In this paper, we investigated the approach of table scheduling in TTEthernet for improving the QoS of BE flows. Table

TABLE III: Simulation delays (jitter, lag, end-to-end) for asynchronous RC flows of ES_12

	Delay (μs)															
Flow	**RC9**				**RC11**				**RC13**				**RC16**			
Allocation	**Max J**	**Offset**	**Max Lag**	**Max E2E**	**Max J**	**Offset**	**Max Lag**	**Max E2E**	**Max J**	**Offset**	**Max Lag**	**Max E2E**	**Max J**	**Offset**	**Max Lag**	**Max E2E**
Table free	12.594	-	12.594	41.696	15.986	-	15.986	37.829	24.754	-	24.754	51.791	29.105	-	29.105	62.080
L_RC_BR_J0	0	62.5	15000	19.447	0	296.875	1000	30.304	0	531.25	1000	36.848	0	765.625	7000	50.128
L_RC_CR_J0	0	62.5	0	19.447	0	296.875	0	30.304	0	531.25	0	36.848	0	765.625	0	50.128
L_Any_BR_J0	0	437.5	15000	19.447	0	593.75	1000	30.304	0	750	1000	36.848	0	906.25	7000	50.128
L_Any_CR_J0	0	437.5	0	19.447	0	593.75	0	30.304	0	750		36.848	0	906.25	0	50.128
L_RC_BR_Jmax	0	46.875	15000	19.447	15.625	281.25	1015.625	30.304	0	515.625	1000	36.848	0	750	7000	50.128
L_RC_CR_Jmax	0	46.875	0	19.447	15.625	281.25	15.625	30.304	0	515.625	0	36.848	0	750	0	50.128
L_Any_BR_Jmax	0	421.875	15000	19.447	15.625	578.125	1015.625	30.304	15.625	734.375	1015.625	36.848	0	890.625	7000	50.128
L_Any_CR_Jmax	15.625	421.875	15.625	19.447	15.625	578.125	15.625	30.304	15.625	734.375	15.625	36.848	15.625	890.625	15.625	50.128
PG_RC_BR_J0	0	62.5	15000	19.447	0	296.875	1000	30.304	0	531.25	1000	36.848	0	765.625	7000	50.128
PG_RC_CR_J0	0	62.5	0	19.447	0	296.875	0	30.304	0	531.25	0	36.848	0	765.625	0	50.128
PG_Any_BR_J0	0	78.125	13000	19.447	0	468.75	0	30.304	0	781.25	1000	36.848	0	890.625	0	50.128
PG_Any_CR_J0	0	78.125	0	19.447	0	468.75	0	30.304	0	781.25	0	36.848	0	890.625	0	50.128
PG_RC_BR_Jmax	0	62.5	15000	19.447	0	296.875	1000	30.304	0	531.25	1000	36.848	0	765.625	7000	50.128
PG_RC_CR_Jmax	0	62.5	0	19.447	0	296.875	0	30.304	0	531.25	0	36.848	0	765.625	0	50.128
PG_Any_BR_Jmax	0	78.125	15000	19.447	0	468.75	1000	30.304	0	781.25	1000	36.848	0	890.625	7000	50.128
PG_Any_CR_Jmax	0	78.125	0	19.447	0	468.75	0	30.304	0	781.25	0	36.848	0	890.625	0	50.128

allocations considering uniform distribution of flows at end systems or switches were proposed. While the first distribution is computed locally at end system, the second one requires global synchronization. These allocations consider BAG or column reservations guaranteeing bounded jitter for RC flows as specified by the ARINC664-P7. Using task-synchronised slots improves the worst-case delays of RC flows by an average of 30%. However, using asynchronous slots largely increases the delays, with on average 46%. In the later case, we showed that column reservation performed with in average 60% better than the BAG reservation. Additionally, with the slots, we are able to multiply the maximum number of RC flows by up to 260 while fulfilling the ES jitter constraint. Evaluation by simulation has shown that local allocations perform better than partially global allocations for well dimensioned network configurations. Thus, if TT flows are not required, local strategies can be easily applied on non synchronized networks.

As a future work, we plan to investigate the gain of table scheduling designed according to a network-wide optimal flow allocation that can be more appropriate for larger industrial configurations in comparison to the heuristic-based approach.

REFERENCES

[1] Aeronautical Radio Inc. ARINC specification 664 P7-1, "Aircraft Data Network, Part 7: Avionics Full Duplex Switched Ethernet (AFDX) Network," 2009.

[2] SAE (Society of Automotive Engineers), "SAE AS6802: Time-Triggered Ethernet," 2016.

[3] IEEE, "802.1Q - IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks," *" https://standards.ieee.org/standard/802 1Q-2018.html*, 2018.

[4] F. Frances, C. Fraboul, and J. Grieu, "Using network calculus to optimize the AFDX network," 2006.

[5] L. Zhao, P. Pop, Q. Li, J. Chen, and H. Xiong, "Timing analysis of rate-constrained traffic in TTEthernet using network calculus," *Real-Time Systems*, vol. 53, no. 2, pp. 254–287, 2017.

[6] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of AVB traffic in TSN networks using network calculus," in *RTAS*. IEEE, 2018, pp. 25–36.

[7] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Applying and optimizing trajectory approach for performance evaluation of AFDX avionics network," in *2009 IEEE Conference on Emerging Technologies & Factory Automation*. IEEE, 2009, pp. 1–8.

[8] D. Thiele, P. Axer, and R. Ernst, "Improving formal timing analysis of switched Ethernet by exploiting FIFO scheduling," in *Proceedings of the 52nd Annual Design Automation Conference*, 2015, pp. 1–6.

[9] N. Benammar, F. Ridouard, H. Bauer, and P. Richard, "Forward end-to-end delay for AFDX networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 858–865, 2017.

[10] A. Soni, J.-L. Scharbarg, and J. Ermont, "Quantum assignment for QoS-aware AFDX network with deficit round robin," in *Proceedings of the 27th International Conference on Real-Time Networks and Systems*, 2019, pp. 70–79.

[11] M. Boyer, N. Navet, M. Fumey, J. Migge, L. Havet, and T. Avionics, "Combining static priority and weighted round-robin like packet scheduling in AFDX for incremental certification and mixed-criticality support," in *5TH European Conference for Aeronautics and Space Sciences (EUCASS)*, 2013.

[12] O. Hotescu, K. Jaffrès-Runser, J.-L. Scharbarg, and C. Fraboul, "Multiplexing Avionics and additional flows on a QoS-aware AFDX network," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 282–289.

[13] A. Finzi and S. S. Craciunas, "Integration of SMT-based scheduling with RC network calculus analysis in TTEthernet networks," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 192–199.

[14] W. Steiner, "An evaluation of smt-based schedule synthesis for time-triggered multi-hop networks," in *2010 31st IEEE Real-Time Systems Symposium*. IEEE, 2010, pp. 375–384.

[15] ——, "Synthesis of static communication schedules for mixed-criticality systems," in *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*. IEEE, 2011, pp. 11–18.

[16] D. Tămaş-Selicean, P. Pop, and W. Steiner, "Design optimization of TTEthernet-based distributed real-time systems," *Real-Time Systems*, vol. 51, no. 1, pp. 1–35, 2015.

[17] S. S. Craciunas and R. S. Oliver, "Combined task-and network-level scheduling for distributed time-triggered systems," *Real-Time Systems*, vol. 52, no. 2, pp. 161–200, 2016.

[18] R. S. Oliver, S. S. Craciunas, and W. Steiner, "IEEE 802.1 Qbv gate control list synthesis using array theory encoding," in *RTAS*. IEEE, 2018, pp. 13–24.

[19] F. Dürr and N. G. Nayak, "No-wait packet scheduling for IEEE time-sensitive networks (TSN)," in *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, 2016, pp. 203–212.

[20] J. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the Internet*. Springer-Verlag, 2001.

[21] A. Finzi and L. Zhao, "Impact of as6802 synchronization protocol on time-triggered and rate-constrained traffic," in *32nd ECRTS*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[22] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, and M. Boyer, "Latency Analysis of Multiple Classes of AVB Traffic in TSN with Standard Credit Behavior using Network Calculus," *IEEE Transactions on Industrial Electronics*, 2020.

[23] O. Hotescu, K. Jaffrès-Runser, J.-L. Scharbarg, and C. Fraboul, "Impact of source scheduling on end-to-end latencies in a QoS-aware avionics network," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. ACM, 2019, pp. 643–646.

[24] D. Tămaş-Selicean and P. Pop, "Optimization of TTEthernet networks to support best-effort traffic," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. IEEE, 2014, pp. 1–4.

[25] B. Houtan, M. Ashjaei, M. Daneshtalab, M. Sjödin, and S. Mubeen, "Synthesising Schedules to Improve QoS of Best-effort Traffic in TSN Networks," 2021.