Worst-case Delay Bounds in Time-Sensitive Networks with Packet Replication and Elimination

Ludovic Thomas, Ahlem Mifdaoui, Jean-Yves Le Boudec

Abstract—Packet replication and elimination functions are used by time-sensitive networks (as in the context of IEEE TSN and IETF DetNet) to increase the reliability of the network. Packets are replicated onto redundant paths by a replication function. Later the paths merge again and an elimination function removes the duplicates. This redundancy scheme has an effect on the timing behavior of time-sensitive networks and many challenges arise from conducting timing analyses. The replication can induce a burstiness increase along the paths of replicates, as well as packet mis-ordering that could increase the delays in the crossed bridges or routers. The induced packet mis-ordering could also negatively affect the interactions between the redundancy and scheduling mechanisms such as traffic regulators (as with per-flow regulators and interleaved regulators, implemented by TSN *asynchronous traffic shaping*). Using the network calculus framework, we provide a method of worst-case timing analysis for time-sensitive networks that implement redundancy mechanisms in the general use case, *i.e.*, at end-devices and/or intermediate nodes. We first provide a network calculus toolbox for bounding the burstiness increase and the amount of reordering caused by the elimination function of duplicate packets. We then analyze the interactions with traffic regulators and show that their shaping-for-free property does not hold when placed after a packet elimination function. We provide a bound for the delay penalty when using per-flow regulators and prove that the penalty is not bounded with interleaved regulators. Finally, we use an industrial use-case to show the applicability and the benefits of our findings.

Index Terms—Network Calculus, Time-Sensitive Networking (TSN), Deterministic Networking (DetNet), Packet Replication Elimination and Ordering Functions (PREOF), Frame Replication and Elimination for Redundancy (FRER), Asynchronous Traffic Shaping (ATS)

1 Introduction

Time-sensitive networks were specified by the deterministic networking (DetNet) working group of the Internet Engineering Task Force (IETF), as well as the time-sensitive networking (TSN) task group of the Institute of Electrical and Electronics Engineers (IEEE), for supporting safety-critical applications in several domains, such as aerospace [1], automation [2] and automotive [3].

As opposed to the *best-effort* service, safety-critical applications require a *deterministic* service [4, §3.1] [5] with zero congestion loss, high levels of reliability, bounded out-of-order delivery and guarantees on the end-to-end latency of each flow. Time-sensitive networks provide this service by relying on a set of redundancy and scheduling mechanisms. The former reduce the probability of end-to-end losses whereas the later aim to guarantee latency bounds [5].

Verifying these bounds on a network is a known intractable issue for simulators and real-life experiments because worst-case situations are not captured by stochastic metrics [6, §1]. Therefore, both the TSN and the DetNet working groups recommend using analytical tools for conducting worst-case timing analyses and for proving the determinism of the network's service [7, §L.3, §N.2] [8]. Among them, the network-calculus framework [9] computes latency, jitter, and backlog bounds, assuming that the sources [resp., the servers] respect some contract of maximum traffic generation [resp., of minimum service]. It has been used to prove certification requirements in avionics [10]. The worst-case timing performance of time-sensitive networks, when focusing only on scheduling mechanisms, has been widely analyzed in the literature [11]–[16].

The main issue addressed in this paper is the effect of the redundancy mechanisms on the delay guarantees in time-sensitive networks. These redundancy mechanisms, such as *frame replication and elimination for redundancy (FRER)* [18] in TSN and *packet replication, elimination and ordering functions (PREOFs)* [4] in DetNet, decrease the end-to-end packet-loss ratio by distributing *"the contents of [. . .] flows over multiple paths in time and/or space, so that the loss of some of the paths does need not cause the loss of any packets"* [4]. To do so, the DetNet packet-replication function (PRF) replicates each incoming packet into several outgoing packets that can take different paths (Fig. 1). The paths then merge and

TABLE 1
Main Acronyms Used in the Paper and Comparison with the Terms of the Working Groups.

In this paper		Term used in DetNet [4]	Term used in TSN [17]
PREFs	Packet replication and elimination functions	Packet replication and elimination functions	**FRER:** Frame replication and elimination for redundancy [18]
PRF	Packet replication function	Packet replication function	Stream splitting function [18, §7.7]
PEF	Packet elimination function	Packet elimination function	Sequence recovery function [18, §7.4.2]
POF	Packet ordering function	Packet ordering function	Does not exist in TSN (March 2022)
REG	Traffic regulator	Shapers [19, §2.3.3.3]	**ATS:** Asynchronous traffic shaping [20]

• *L. Thomas and A. Mifdaoui are with ISAE-Supaéro, Université de Toulouse. Toulouse, France.*
E-mail: ludovic.thomas@isae-supaero.fr
• *J-Y. Le Boudec is with the I&C Departement, EPFL. Lausanne, Switzerland.*

Manuscript received someday; revised some other day

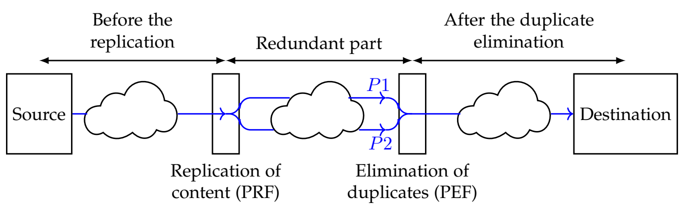
Before the replication
Redundant part
After the duplicate elimination
Source
P1
P2
Destination
Replication of content (PRF)
Elimination of duplicates (PEF)

Fig. 1. The three sections of a replicate's path: before, in, and after the redundant part.

multiple copies of the packet (the replicates) reach a packet-elimination function (PEF) that forwards only the first replicate and eliminates the subsequent ones (the duplicates). The PEF generally relies on a sequence number in the packet header to identify the replicates [4]. In Table 1, we compare the terms used in IETF DetNet and IEEE TSN. The main acronyms used thorough the paper are also listed in Table 1.

The packet replication and elimination functions (PREFs) could increase the worst-case end-to-end latency of the flows: [4, §3.1] recalls that their use is "*constrained by the need to meet the users' latency requirements*". Therefore, understanding how PREFs affect the worst-case latency guarantees is fundamental to: (i) determine the applicability of PREFs in industrial networks; (ii) perform trade-offs between latency and loss-ratio requirements; and (iii) design networks with stringent requirements on both aspects.

Three main challenges arise from conducting worst-case timing analyses of PREFs. First, the replication of packets through the network can induce a burstiness increase along the paths of replicate packets, which leads to increasing delay and backlog bounds in the crossed nodes. Second, the traffic exiting the PEF can exhibit both an increased burstiness and a mis-ordering of the packets. This can lead to increased delay bounds in the nodes placed after the PEF. Third, the coexistence of the packet mis-ordering with the burstiness increase could negatively affect the behavior of the devices that have been designed for tackling each issue individually. For example, packet-ordering functions (POFs) have been specified in IETF DetNet for removing only the packet mis-ordering. Similarly, traffic regulators (also called *shapers*) are scheduling mechanisms designed for removing only the burstiness increase. If a traffic regulator (as in TSN ATS, *asynchronous traffic shaping*) is placed after the PEF for removing the burstiness increase caused by the redundancy, then the packet mis-ordering that coexists with this burstiness increase could negatively affect the behavior of the traffic regulator.

The existing worst-case timing analyses of redundancy mechanisms in time-sensitive networks [21], [22] are limited to the assumption of using redundancy mechanisms at the end-systems as with *Avionics Full-dupleX switched Ethernet (AFDX)* [10] and *Parallel Redundancy Protocol (PRP)* [23]. This assumption discards the main challenges detailed above. More recent works [21, §4.3.2] [24] based on simulation consider redundancy mechanisms at intermediate nodes. However, firm conclusions are difficult to draw from simulations as they not cover the worst-case behavior.

Therefore, our primary goal in this paper is to bridge these gaps and to provide a method of worst-case timing analysis for time-sensitive networks that implement redundancy mechanisms in the general use-case, i.e., at end-systems and/or intermediate nodes. Specifically:

- We provide a network-calculus toolbox that enables the computation of upper bounds on the burstiness increase (Theorem 1) and on the amount of reordering (Theorem 2) due to the elimination of duplicate packets. Theorem 1 is useful for computing delay bounds in the nodes located after the elimination of duplicates, whereas the bound from Theorem 2 can be compared to the application's requirements to decide if the packets should be reordered prior to their delivery.
- We analyze the interactions between redundancy mechanisms and traffic regulators. We show that the packet mis-ordering due to the elimination of duplicates leads to a bounded increase of the worst-case delay with per-flow regulators (PFRs) (Theorem 3) and to unbounded delays with interleaved regulators (IRs) (*e.g.*, TSN ATS) (Theorem 4). The problem goes away if the packets are re-ordered after the elimination and before the regulator (Theorem 5).
- We conduct performance analyses for an industrial use-case that highlight the interest of our introduced approach to tighten the delay bounds in comparison to intuitive computation approaches.

In Section 2, we illustrate the issues posed by PREFs using a toy example. In Section 3, we relate our proposed approach to the state of the art, and we describe the system model in Section 4. Our main theoretical contributions are detailed in Sections 5 and 6, that cover the network-calculus toolbox for redundancy mechanisms and the analysis of the interaction between such mechanisms and traffic regulators. Finally, we validate our approach on an industrial use-case in Section 7.

2 Illustration of the Issues Posed by Packet Replication and Elimination

In this section, we illustrate the issues posed by packet replication and elimination functions (PREFs) in time-sensitive networks, as identified in the Introduction. We first detail the burstiness increase and the mis-ordering introduced by PREFs. Afterwards, we focus on the problems arising from the interactions between PREFs and traffic regulators.

2.1 Burstiness and Misordering Introduced by PREFs

To highlight the effect of PREFs on the burstiness and packet order, we consider the toy example in Fig. 2: a periodic flow with a rate r_0 of one packet per every time unit is replicated at the output of the vertex B and sent over two paths: C [resp., D], with a minimum delay of zero time units [resp., six time units] and a maximum delay of one time unit [resp., seven time units]. A possible trace of packets for the toy example is given in Fig. 3. Here, the path through C drops

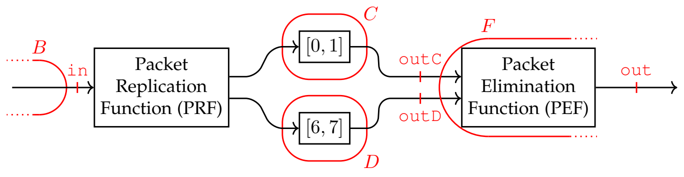
B
in
Packet Replication Function (PRF)
C
[0, 1]
D
[6, 7]
outC
outD
F
Packet Elimination Function (PEF)
out

Fig. 2. Toy example used thorough the paper. A flow is replicated on two paths, C and D, with different delay bounds. The paths then merge into F, that removes the duplicates.

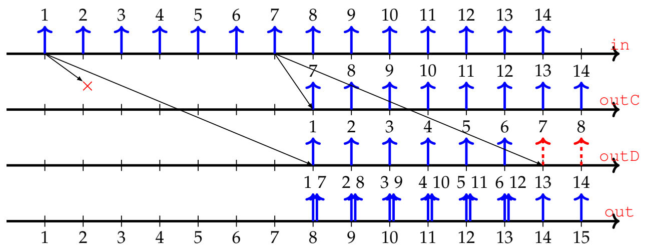

Fig. 3. An example trajectory on the toy example of Fig. 2 causing a double-rate output of the PEF.

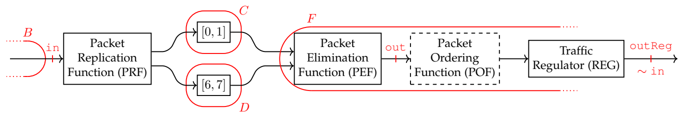
B
in
Packet Replication Function (PRF)
[0, 1]
C
[6, 7]
D
F
Packet Elimination Function (PEF)
out
Packet Ordering Function (POF)
Traffic Regulator (REG)
outReg
∼ in

Fig. 4. The toy example of Fig. 2 extended with POF and REG to deal with the mis-ordering and burstiness increase issues due to PEF.

all Data Units 1 to 6: they are only received through D with a latency of seven time units (7 t.u.). After 7 t.u., the link through C is available again and the Data Units 7 to 14 are received through both C and D, with a latency of 1 [resp., 7] t.u. The PEF receives the sum of "outC" and "outD". It drops the duplicates but forwards the packets that contain not-already-seen data units. Its output is on the Line "out".

We observe that the traffic after the PEF is much more bursty than before the replication function: between t.u. 8 and t.u. 13, the PEF simultaneously outputs the "older" packets 1-6 received through the "long" path and the "newer" packets 7-12 received through the now-active "short" path. This increases the load on the downstream link with a doubled rate, $2r_0$, for a duration of 6 t.u.

The toy example hence suggests that packet replication and elimination functions (PREFs) can significantly increase the flows' burstiness, which could further worsen the congestion and the worst-case delay in the downstream nodes. Obtaining a bound on this burstiness increase is important for computing the end-to-end latency of the flow. Indeed, a delay bound for the flow in the third section of Fig. 1 (after the PEF) can be obtained from such an upper bound on the flow's worst-case traffic at the output of the PEF and from a lower bound on the minimum service provided by the nodes located after the PEF.

A first approach for bounding the traffic of the flow after the PEF, which we denote as *intuitive*, consists in doing as if the PEF would never drop a packet (i.e., even the duplicates are forwarded). This approach requires the network engineer to dimension all the downstream nodes in order to support a sustained double rate. In Theorem 1, we provide a better bound for the traffic at the output of the PEF. It leads to better end-to-end latency bounds, as we show in Sec. 7.

We also observe that PREFs can create a mis-ordering: In the toy example, Data Unit 6 exits the PEF five time units after Data Unit 7. Obtaining an upper bound for this mis-ordering is important for comparing it to the application's requirements. We provide such bound in Theorem 2.

2.2 Interactions Between PREF and Other Devices

If either the end-to-end latency bound or the mis-ordering bound does not meet the system requirements, then we can use one of the devices specified by the working groups for tackling the corresponding issue.

For example, if the receiving application does not tolerate any mis-ordering, then the DetNet packet-ordering function (POF) [4, §3.2.2.2] can be used after the PEF to correct the mis-ordering introduced by PREFs. Similarly, if the end-to-end latency of a flow does not meet its requirements due to a high worst-case delay in the third section of Figure 1, then using traffic regulators [20] just after the PEF appears as a natural choice: Traffic regulators (REGs) have been designed for removing the burstiness increase [11], [12] thus for reducing the worst-case delay in downstream nodes. They come in two flavors: per-flow regulators (PFRs) process each flow individually whereas interleaved regulators (IRs) process flow aggregates.

To the best of our knowledge, the interactions between PREFs and other devices such as POFs and regulators have not yet been analyzed. For instance, many properties of the regulators rely on the assumption that the upstream system is first in, first out (FIFO) [12]. As observed on the toy example, this assumption does not hold with PREFs.

Assume, for example, that the traffic regulator in Fig. 4 shapes the traffic back to the profile it had at the input "in". In terms of burstiness, this makes the middle section in Fig. 1 transparent to the third section. The regulator processes the traffic from the "out" line of Fig. 3 and forces the packets to be as spaced as in the "in" line by delaying and storing the packets if required. Clearly, the upstream system between "in" and "out" in Fig. 3 is not FIFO, because the packets exit the PEF out of order. Thus the properties of the regulators that depend on this assumption might not hold and the cohabitation of the PEF and the REG could negatively affect the latency bounds. A packet-ordering function (POF) (dashed box in Fig. 4) can be used after the PEF and before the regulator to force the upstream system to be FIFO. If such POF is placed, then we would expect to retrieve all the properties of regulators.

In Sec. 6, we analyze the interactions between PREFs and regulators. We observe that the conclusions depend on the type of the regulator: either PFR or IR (as with TSN ATS).

3 Related Work

The most relevant timing analyses of redundancy mechanisms in time-sensitive networks can mainly be categorized according to the assumption of where to enable the packet replication and elimination functions.

The existing approaches in this area considering the packet replication and elimination only at the end-devices concern mainly *High-availability Seamless Redundancy (HSR)* and *Parallel Redundancy Protocol (PRP)* [23, §4]. Both mechanisms eliminate the duplicates only at the destination; thus their analysis does not require to bound the traffic at the output of the PEF and discards the mis-ordering issue. In [22], worst-case delay bounds are computed in HSR-based networks by using network calculus. The idea consists in taking the maximum of the delay bounds along each of the redundant paths. In [25], model checking is used to analyze how well PEF algorithms based on sequence numbers can detect duplicates in AFDX, a PRP-based network.

On the other hand, there exist only few seminal works in the literature considering the packet replication and elimination anywhere in the network. These works mainly concern

TABLE 2
Notations

Term	Definition
$\mathcal{G}$	The graph of the network for the class of interest.
f	A flow.
$\mathcal{G}(f)$	The graph of flow f.
EP-vertex in $\mathcal{G}(f)$	A vertex at which the duplicates of f have not been eliminated yet.
Diamond ancestor of n in $\mathcal{G}(f)$	A vertex that is not an EP-vertex of $\mathcal{G}(f)$ and that is contained in any paths of f between its source and n.
$d_f^{a\to n}$ [resp., $D_f^{a\to n}$]	Lower [resp., upper] delay bound for f between a and n, along any possible path $a \to n$ within $\mathcal{G}(f)$.
$\texttt{PEF}_n(f)$	Packet-elimination function at output-port n that eliminates the duplicates of flow f.
$\texttt{POF}_n(\mathcal{F}, o)$	Packet-ordering function at n that uses reference o to force the order of the data units of the aggregate $\mathcal{F}$.
$\texttt{REG}_n(\mathcal{F}, o)$	Regulator (either interleaved or per-flow) that shapes the flows within $\mathcal{F}$ in a FIFO manner.
$\sigma_{n,f}$	Shaping curve for f at the regulator within n.
α_{f,n^*} [resp., $\alpha_{f,\texttt{FUN}^*}$]	For n a vertex in $\mathcal{G}$ [resp., FUN a function], the arrival curve of f at the output of n [resp., of the function FUN].
$\gamma_{r,b} : t \mapsto rt + b$	Leaky-bucket arrival curve with rate r and burst b
$\delta_D : t \mapsto \infty, t > D$	Service-curve of a D-bounded-Delay element
$\lvert x\rvert^+$	$= \max(0, x)$
t.u.	Time unit (arbitrary unit used in the examples)
d.u.	Data unit (arbitrary unit used in the examples)

FRER [18], which is the first mechanism enabling such an assumption. As mentioned in [18, §C.9] and further illustrated in Sec. 2, the elimination of duplicates within the network raises issues in computing the end-to-end delay bounds. In [26], further concerns about FRER have been discussed. In [27], a simulation framework based on OMNeT++ has been developed for TSN mechanisms, including FRER [21, §4.3.2]. However there is no specific experiment for assessing the effect of FRER on latency bounds. Furthermore, obtaining the worst-case delay bounds with simulators is a known intractable problem [6, §I].

Thus, as stated above, there are no formal analyses of delay bounds of redundancy mechanisms, such as TSN FRER or DetNet PREOF, when the packet replication and elimination is performed anywhere in the network, or on the interactions between redundancy and scheduling mechanisms.

4 System Model

Our system model is divided into three abstraction levels. It results from an analysis of the TSN and DetNet documents and Appendix A details its applicability for these standards. Notations used thorough the paper are listed in Table 2.

4.1 Network and Flow Model

Type of network: We consider an asynchronous packet-switching full-duplex store-and-forward network that transports *data units* between applications. We assume that there is one or several classes of traffic and that flows are statically assigned to a class. We focus on one class and denote by $\mathcal{G}$ the underlying graph for this class [6, Chap. 12]. $\mathcal{G}$ contains one vertex per output port in the network (see Sec. 4.2 for the exact mapping between the two notions) and (a, b) is a directed edge of $\mathcal{G}$ if at least one flow crosses b just after a. The network does not need to be feed-forward, it can contain cyclic dependencies (*i.e.*, $\mathcal{G}$ can contain cycles) [13].

Data unit versus *packet*: At any time, a *data unit* can be transported by several *packets* located at several locations. A *flow* f is a coherent sequence of data units that originate from a unique source and that follow a directed acyclic subgraph of $\mathcal{G}$ to reach one or several destinations. An example of such a flow graph, noted $\mathcal{G}(f)$, is shown in Fig. 5.

Flow constraints: We assume that each flow is constrained by a network-calculus arrival curve α_f^0 at the output of its

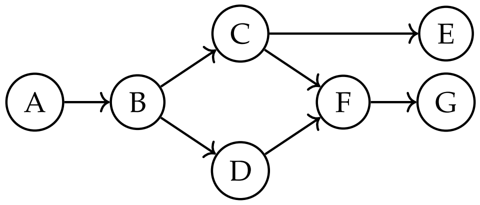

Fig. 5. Example of a flow graph $\mathcal{G}(f)$ (Sec. 4.1) of flow f with a source A, destinations E and G and redundant paths to reach G.

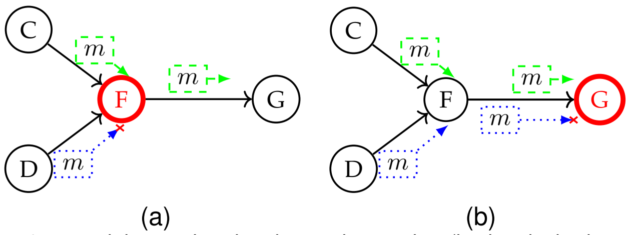

Fig. 6. F might receive the data unit m twice (in the dashed green and the dotted blue packets). (a) F contains a PEF, it drops the dotted blue packet that contains the already-seen data unit m. (b) Only the destination G contains a PEF, it receives the data unit twice and drops the dotted blue packet.

source application. For an observation point M (that can be a vertex or a function), we note $\alpha_{f,M}$ the arrival curve of f at M. For n a vertex of $\mathcal{G}(f)$ [resp., for FUN a function], we note α_{f,n^*} [resp., $\alpha_{f,\texttt{FUN}^*}$] the arrival curve of f at the output of vertex n [resp., at the output of the function FUN].

Position of PRF, PEF in a flow graph: When a vertex, such as B in Fig. 5, has several children, we consider that an implicit PRF has been installed on B for the flow f: it sends a copy of each incoming data unit to each child. When a vertex has several parents, such as F in Fig. 5, this means that it can receive the same data unit several times, within different packets. However, this does not necessarily mean that it implements a PEF. If a PEF is present on such a vertex (case of F in Fig. 6a), then it forwards only the first received packet that contains the data unit. If the vertex does not contain a PEF (case of F in Fig. 6b), then it forwards all the packets, and might consequently forward the same data unit several times. Packets that transport already-seen data units at a given location are called *duplicates*.

Assumption on the elimination of duplicates: When several paths of a flow merge, we assume that the duplicates are eliminated before the path can split again. We believe that this assumption does not restrict the analysis of industrial systems. Indeed, the main use-case for having a PEF a few hops after the merge point (as in Fig. 6b) is when the edge router does not support PEF. The edge router then forwards all the received packets to the end-system, that is responsible for removing the duplicates.

EP-vertex: *elimination-pending (EP)* vertices of $\mathcal{G}(f)$ are the only vertices that can observe a data unit of f more than once. Formally, if a vertex, that does not contain a PEF for f, has several parents in $\mathcal{G}(f)$ (vertex F in Fig. 6b), then we qualify it as an EP-vertex of $\mathcal{G}(f)$. An EP-vertex can have at most one child in $\mathcal{G}(f)$. Additionally, a vertex that does not contain a PEF for f and is a child of an EP-vertex is also an EP-vertex of $\mathcal{G}(f)$.

Diamond ancestor: For any two vertices a and n in a flow graph $\mathcal{G}(f)$, we say that a is a *diamond ancestor* of n in $\mathcal{G}(f)$ if a is not an EP-vertex in $\mathcal{G}(f)$ and all paths in $\mathcal{G}(f)$ from the

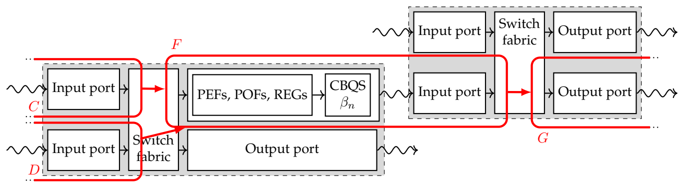

Fig. 7. Model of two devices in the network (in gray dashed boxes) and their relation with the flow graph (vertices in thick red).

source of f to n contain a. In Fig. 5, B is a diamond ancestor of F because B is not an EP-vertex of $\mathcal{G}(f)$ and any paths from A (source of f) to F contain B.

Lost data unit: We say that a data-unit m of a flow f is *lost for* a vertex n [resp., for a function FUN] if the vertex n in $\mathcal{G}(f)$ [resp., the function FUN] never observes the data unit m in any packet. In Fig. 5, if the link $B \rightarrow C$ fails, then a data unit m is lost for E but not necessarily for G. The main purpose of PREFs is to reduce the probability of losing a data unit for any destinations of the flow.

Worst-case latency: Let f be a flow and d one of the destinations of f; the end-to-end (ETE) upper [resp., lower] latency bound of f for d is an upper bound [resp., lower bound] on the maximum [resp., minimum] delay that each data unit m of f takes to reach d, assuming that m is not lost for d.

4.2 Device Model

Device: The model for each *device* in the network is illustrated in Fig. 7: it consists of *input ports*, *output ports*, and a *switching fabric*. The vertices in the network's graph $\mathcal{G}$, such as vertex F in thick red in Fig. 7, are made of the output port on one device, followed by the input port on the subsequent device. The devices are connected through *transmission links* that can lose packets.

Input port: We assume that each *input port* contains a store-and-forward step that we model as a network-calculus packetizer [9, §1.7.2], [13, Thm. 1]. Any additional processing delay (*e.g.*, decryption, CRC check, etc.) is assumed to be bounded between known values and is modeled using the network-calculus bounded-delay element [9, Prop. 1.3.3].

Switching fabric: As illustrated in Fig. 7, the *switching fabric* between vertices C and F forwards packets of flow f from the input port within C to the output port within F if and only if $C \rightarrow F$ is an edge in $\mathcal{G}(f)$. The switching fabric implements the PRF. When a packet is forwarded from one input port to two or more output ports, we say that the data unit contained in the incoming packet is replicated and transported by several new packets (one per recipient output port). Any delay within the switching fabric is assumed to be bounded and is modeled by using the network-calculus bounded-delay element [9, Prop. 1.3.3].

Output port: We model each *output port* as in Fig. 7. It contains a FIFO-per-class class-based queuing subsystem (CBQS). We assume that, for each vertex n, we know a network-caclulus service curve β_n that the CBQS offers in a FIFO manner to the class of interest. The service curve can be obtained through an analysis of the scheduling policy [15] and includes any additional technological latency. The CBQS can be preceded by a set of optional functions.

Packetized streams: Within a device, between the output of the input port (that contains the packetizer) and the input of the CBQS, the stream of bits for each flow is packetized.

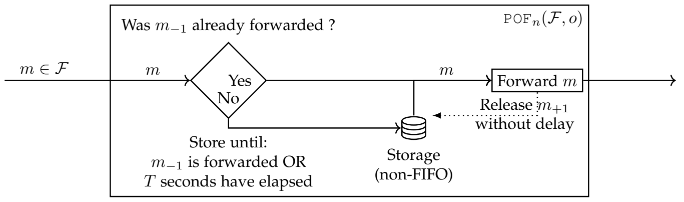
Was m_{-1} already forwarded ?
$\mathtt{POF}_n(\mathcal{F}, o)$
$m \in \mathcal{F}$
m
Yes
No
m
Forward m
Release m_{+1} without delay
Store until: m_{-1} is forwarded OR T seconds have elapsed
Storage (non-FIFO)

Fig. 8. Functional model of the packet-ordering function $\mathtt{POF}_n(\mathcal{F}, o)$. For a data unit m, m_{-1} [resp., m_{+1}] refers to the data unit of the aggregate $\mathcal{F}$ that exited the reference o just before [resp., just after] m.

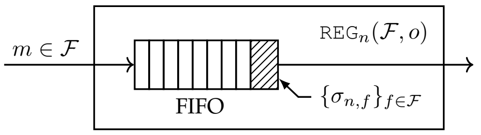
$m \in \mathcal{F}$
$\mathtt{REG}_n(\mathcal{F}, o)$
FIFO
$\{\sigma_{n,f}\}_{f \in \mathcal{F}}$

Fig. 9. Model of a regulator $\mathtt{REG}_n(\mathcal{F}, o)$, with shaping curves $\{\sigma_{n,f}\}_f$.

4.3 Model for the Functions

PEF: For a flow f crossing n, the output port in n can contain a *packet-elimination function (PEF)* for flow f, noted $\mathtt{PEF}_n(f)$. For each incoming packet of f, we assume that $\mathtt{PEF}_n(f)$ determines without any delay if the data unit contained in the packet has already been observed by $\mathtt{PEF}_n(f)$. If so, the packet is identified as a duplicate and is discarded. For the stream of packets that contains never-seen data units of f, the $\mathtt{PEF}_n(f)$ is transparent: FIFO and without any delay.

POF: Consider a set of flows $\mathcal{F}$ crossing n such that for each flow $f \in \mathcal{F}$, o is a diamond ancestor of n in $\mathcal{G}(f)$. The output port in n can contain a *packet-ordering function (POF)* for the aggregate $\mathcal{F}$ with reference o, noted $\mathtt{POF}_n(\mathcal{F}, o)$. We assume that $\mathtt{POF}_n(\mathcal{F}, o)$ has the knowledge of the order in which the data units belonging to the aggregate $\mathcal{F}$ exited the reference o. $\mathtt{POF}_n(\mathcal{F}, o)$ then enforces the same order at its own output, by delaying the packets that are out of order.

However, a data unit m cannot be delayed by $\mathtt{POF}_n(\mathcal{F}, o)$ for a duration longer than the POF's timeout parameter T: After being stored for a duration T, m is immediately released, even if the previously-expected data unit has not been received so far. The timeout allows the POF to recover from losses without blocking the following data units forever [28], [29]. We assume that the timeout value of every POF conforms with the recommendations of [28, §IV.B]. As a consequence, the timeout cannot only be triggered when one of the data units m of $\mathcal{F}$ is *lost for* the POF.

The model of POF is illustrated in Fig. 8. A possible implementation is given in [28, §3.4] and [29]. A POF cannot be placed at an EP-vertex: we always assume that the duplicates are eliminated before the flow is handed to the POF, which is consistent with the assumptions in [29, §4.1].

REG: Consider a set of flows $\mathcal{F}$ crossing n such that, for each flow $f \in \mathcal{F}$, o is a diamond ancestor of n in $\mathcal{G}(f)$. The output port in n can contain a *regulator (REG)* for the aggregate $\mathcal{F}$ with reference o, noted $\mathtt{REG}_n(\mathcal{F}, o)$. The regulator is configured with a set of shaping curves, one per flow f of the aggregate $\mathcal{F}$, which we note $\{\sigma_{n,f}\}_{f \in \mathcal{F}}$. For each $f \in \mathcal{F}$, $\sigma_{n,f}$ must be concave and must be an arrival curve of f at the output of the reference vertex o. The regulator then puts all the packets of the aggregate $\mathcal{F}$ in a FIFO queue (Fig. 9) and examines only the head-of-line packet. It releases the head-of-line packet as soon as doing so does not violate the shaping curve $\sigma_{n,f}$, where f is the flow of the head-of-line packet. When the regulator processes a single flow, $\mathcal{F} = \{f\}$, we model it as a *per-flow regulator (PFR)* [9, §1.7.4]. When $\mathcal{F}$ contains two or more flows, we model it as an *interleaved regulator (IR)* [12].

We consider that each output port contains a forwarding pipeline before the CBQS with the following optional functions, in this order: PEFs → POFs → REGs.

Example: Consider two flows f, g, both with the same flow graph of Fig. 5 and a PEF at F. The output port F processes streams of packets coming from both parents C and D. A first possible example of the organization of the functions before the CBQS within vertex F is shown in Fig. 10. Each flow is first processed by its respective PEF, then both duplicate-free flows are reordered as an aggregate by using $\texttt{POF}_F(\{f,g\},B)$. This function enforces the same order for the aggregate as the one at the output of B, *i.e.*, before the redundant section. Last, they are both processed by the same interleaved regulator that enforces two different contracts for f and for g, but that keeps the aggregate $\{f,g\}$ FIFO. A variant of this situation is shown in Fig. 11. After elimination, each flow is now independent from the other one, where the POFs enforce per-flow order and the two REGs are per-flow regulators (PFRs). This situation is different from Fig. 10 because a packet of f cannot be delayed by a packet of g. In addition, this configuration could have a higher hardware cost than in Fig. 10.

FIFO assumptions: With the exception of POF, each network element is assumed to be FIFO for the class of interest.
Assumptions on losses: With the exception of PEF, each function, each CBQS, each switching fabric, each input port and each internal connection within a device is assumed to be lossless (does not lose any packets). Packets can be lost on the transmission links between devices. This model covers various failures, including random media losses, the shutdown of an output port (equivalent to its out-going link losing all packets) and the shutdown of an input port (equivalent to its in-going link losing all packets).

As packets can be lost on transmission links, the network is not assumed to be lossless. Of course, the latency bounds computed in this paper are only valid for the non-lost data units (the data units for which at least one replicate reaches the destination), but these bounds remain valid even if some other data units are lost in the network.

5 Toolbox for the Deterministic Analysis of Packet Replication and Elimination

Network calculus [9] is a mathematical framework for computing deterministic latency bounds. It relies on the concepts of arrival and service curves. An arrival curve $\alpha_{f,M}$ at a specific observation point M and for a specific flow f is a constraint on the maximum amount of traffic of flow f that can cross M over any periods of time $[s,t]$, which is equivalent to: $\forall s \leq t, R(t) - R(s) \leq \alpha(t-s)$, with $R(t)$ the amount of data of flow f crossing M between 0 and t. Also, a service curve β_S of a specific network element S is a constraint on the minimum amount of traffic that the network element must serve. Network calculus gives delay and backlog bounds in network elements given the arrival-curve and service-curve constraints [9], [30].

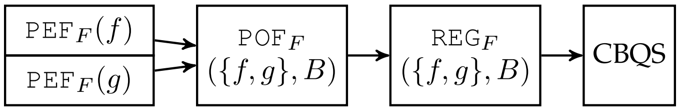
$\texttt{PEF}_F(f)$
$\texttt{PEF}_F(g)$
$\texttt{POF}_F$ $(\{f,g\},B)$
$\texttt{REG}_F$ $(\{f,g\},B)$
CBQS

Fig. 10. Example of an organization of the optional functions within an output port. After their respective PEF, the two flows share the same POF and the same regulator (REG).

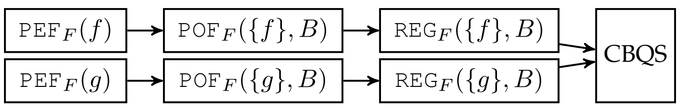
$\texttt{PEF}_F(f)$
$\texttt{POF}_F(\{f\},B)$
$\texttt{REG}_F(\{f\},B)$
$\texttt{PEF}_F(g)$
$\texttt{POF}_F(\{g\},B)$
$\texttt{REG}_F(\{g\},B)$
CBQS

Fig. 11. Example of an the organization of the optional functions within an output port with one POF and one REG per flow.

In this section, we compute an upper bound of the burstiness increase caused by PREFs by computing an arrival curve α_{f,PEF^*} for the flow f at the output of the packet-elimination function. The arrival curve α_{f,PEF^*} can then be combined with the service curves of the downstream elements (that can be found in [15], [31]) to compute a delay bound in the last section of Fig. 1. This delay bound is useful for validating the system's latency requirements.

We also quantify the amount of mis-ordering introduced by the redundancy. This bound can be compared to the application's requirement to decide if reordering is required before delivering the data to the application. If so, the same bound can be combined with the results of [28] to configure the packet-ordering function (POF) that corrects this mis-ordering. The consequences of such reordering on the flow's delay and burstiness are also analyzed.

5.1 Output Arrival Curve of a PEF

Theorem 1 (Output arrival curve of a PEF)**.** Let $\texttt{PEF}_n(f)$ be a packet-elimination function for flow f at the output port of vertex $n \in vertices(\mathcal{G}(f))$. Assume that $\alpha_{f,\text{PEF}^{\text{in}}}$ is an arrival curve of f at the input of $\texttt{PEF}_n(f)$. Then

1/ $\alpha_{f,\text{PEF}^{\text{in}}}$ is an arrival curve for the flow at the output of the PEF.

2/ For every diamond ancestor a of n in $\mathcal{G}(f)$, assume that α_{f,a^*} is an arrival curve for f at the output of a and denote by $d_f^{a\to n}$ [resp., $D_f^{a\to n}$] a minimum [resp., maximum] delay bound for f between the output of a and the input of $\texttt{PEF}_n(f)$, along any possible paths $a \to n$ within the graph $\mathcal{G}(f)$. Then

$$\alpha_f^{a\to n} \triangleq \alpha_{f,a^*} \oslash \delta_{(D_f^{a\to n})-(d_f^{a\to n})} \tag{1}$$

is an arrival curve for f at the output of the PEF.

Furthermore, the min-plus convolution of all above arrival curves

$$\alpha_{f,\text{PEF}^*} = \alpha_{f,\text{PEF}^{\text{in}}} \otimes \alpha_f^{a_1\to n} \otimes \alpha_f^{a_2\to n} \otimes \alpha_f^{a_3\to n} \otimes \dots \tag{2}$$

for any set of diamond ancestors $a_1, a_2, a_3, \dots$ of n in $\mathcal{G}(f)$ is also an arrival curve for f at the output of the PEF, where $\otimes$ denote the min-plus convolution[1].

The result is proved as follows: Item 1/ is a direct consequence of the fact that the PEF has no delay. Item 2/ is obtained by considering the entire system made of the portion of the graph $\mathcal{G}(f)$ between the diamond ancestor a and n. This system is neither lossless nor FIFO, but several classical network-calculus results remain applicable, as we discuss in Appendix B. α_{f,PEF^*} is finally obtained by

1. $f \otimes g : t \mapsto \inf_{s\geq 0}(f(s) + g(t-s))$. The min-plus convolution is associative and commutative [6, §2.1.3].

applying [9, Lemma 1.2.4]. A formal proof of Theorem 1 is given in Appendix C.1.

Application to the Toy Example: An arrival curve α_{f,PEF^*} for f at the output of the PEF within F (Fig. 2) is shown in Fig. 12 with a solid red line.

The first constituent, $\alpha_{f,\text{PEF}^{\text{in}}}$ is the arrival curve at f at the input of the PEF (as per Theorem 1, Item 1). To obtain it, we first observe that the periodic profile of the flow f at the output of B (as on the Line "`in`" of Fig. 3) is constrained by the leaky-bucket arrival curve $\alpha_{f,B^*} = \gamma_{r_0,b_0}$ with a rate of one data unit per unit of time ($r_0 = 1$ d.u./t.u.) and a burst of one data unit ($b_0 = 1$ d.u.). By using the jitter bound within C and D and Proposition 3 in Appendix B, we obtain that the arrival curves of f at the output of C and D, α_{f,C^*} and α_{f,D^*}, equal to the same leaky-bucket arrival curve $\gamma_{r_0,2b_0}$ with a burst $2b_0$ of two units of data. As f enters F from both C and D, we obtain $\alpha_{f,\text{PEF}^{\text{in}}} = \alpha_{f,C^*} + \alpha_{f,D^*} = \gamma_{2r_0,4b_0}$, a leaky-bucket arrival curve with a rate $2r_0$ and a burst $4b_0$.

The second constituent of α_{f,PEF^*} in Fig. 12 is obtained by applying the Equation (1) of Theorem 1, Item 2/ with $a = B$. From Fig. 2, we obtain that a delay lower-bound [resp., an upper-bound] for f from B to F along any possible paths within $\mathcal{G}(f)$ is $d_f^{B\to F} = 0$ t.u. (through C) [resp., $D_f^{B\to F} = 7$ t.u, through D]. We obtain $\alpha_f^{B\to F} = \alpha_{f,B^*} \oslash \delta_{D_f^{B\to F} - d_f^{B\to F}} = \gamma_{r_0,b_0} \oslash \delta_7$, *i.e.*, $\alpha_f^{B\to F} = \gamma_{r_0,8b_0}$.

If we assumes that the PEF does not delete any packet, as in the *intuitive* approach mentioned in Section 2, we only know that f has the arrival curve $\alpha_{f,\text{PEF}^{\text{in}}}$ at the output of the PEF (Item 1 of the Theorem). This arrival curve shows that the traffic can exhibits a burst of $4b_0$ and a rate $2r_0$ twice as big as the normal source rate.

But our theorem goes beyond the intuitive approach: its second item applied with $a = B$ provides a second arrival curve for f: $\alpha_f^{B\to F}$. In the network-calculus framework, we can combine the knowledge of two arrival curves by computing their min-plus convolution: $\alpha_{f,\text{PEF}^*} = \alpha_{f,\text{PEF}^{\text{in}}} \otimes \alpha_f^{B\to F}$ is also an arrival curve for f at the output of the PEF. With the leaky-bucket arrival curves of the toy example, the min-plus convolution is simply the minimum of the two curves, shown with a solid red line on Fig. 12. We observe that Theorem 1 provides a better upper-bound of the traffic than the intuitive approach. For example, α_{f,PEF^*} indicates that the double rate $2r_0$ is only a peak rate that the traffic cannot exhibits forever: flow f keeps a sustained rate r_0, but with a much higher burst $8b_0$. In network calculus, the arrival curves that describe flows with a peek rate ($2r_0$) and a sustained rate (r_0) are called *variable-bit-rate* (VBR) arrival curves. Theorem 1 provides for the toy example the best possible VBR arrival curve, as we prove later.

Remark: Theorem 1 does not require to identify pairs of replication/elimination functions, with one PRF and one PEF in each pair. Therefore, Theorem 1 is suited for complex flow graphs, including graphs with repeated patterns of redundancy, with meshes, as well as graphs where the packet-elimination function is not located at the merge point of the paths. When pairs of PRF/PEF can be identified as in Fig. 1, the following simpler corollary can be used.

Corollary 1 (Application of the theorem to a unique redundant section with parallel systems)**.** Consider a flow f with an arrival curve α_f that is replicated and sent into N systems $\{S_i\}_{i\in[\![1,N]\!]}$ and then processed by a packet-elimination function $\texttt{PEF}(f)$, as in Fig. 13. Note that each S_i is not necessary a single network element but can be any combination of network elements. Assume that the packets forwarded through S_i (*i.e.*, the ones not lost) have a delay through S_i that is bounded within $[d_i, D_i]$. Then,

$$\alpha_f^* = \left(\textstyle\sum_{i\in[\![1,N]\!]} \alpha_f \oslash \delta_{(D_i - d_i)}\right) \otimes \left(\alpha_f \oslash \delta_{\left(\max\limits_{i\in[\![1,N]\!]} D_i - \min\limits_{j\in[\![1,N]\!]} d_j\right)}\right) \tag{3}$$

is an arrival curve for f at the output of $\texttt{PEF}(f)$.

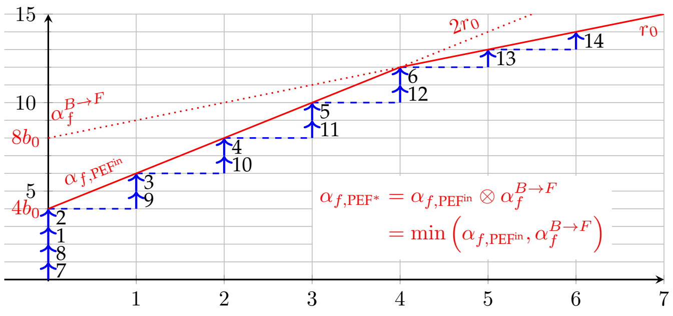

Fig. 12. Solid red: α_{f,PEF^*}, arrival curve of f on the toy example, at the output of the packet-elimination function $\texttt{PEF}_F(f)$, obtained by applying Theorem 1. Dashed blue: Cumulative arrival function obtained with the trajectory of Fig. 14, which shows the tightness of the result.

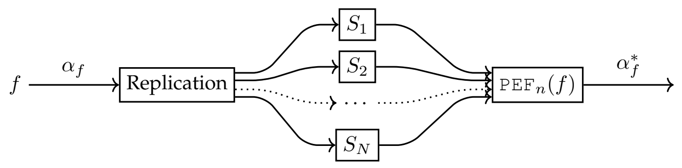

Fig. 13. Notations of Corollary 1. Flow f is replicated and sent to N parallel systems. Corollary 1 gives the arrival curve α_f^* at the output of the packet-elimination function $\texttt{PEF}_n(f)$.

Corollary 1 is a direct application of Theorem 1. A formal proof is given in Appendix C.2. The corollary is of interest for two reasons. First, its simpler notation is likely to cover many industrial applications containing a unique redundant portion with parallel systems. Second, Corollary 1 is tight in the following sense.

Proposition 1 (The result in Corollary 1 is tight with $N = 2$ and leaky-bucket-constrained flows, in the family of variable-bit-rate (VBR) arrival curves.)**. For any** leaky-bucket arrival curve $\gamma_{r,b}$, for any set of values $d_1, D_1, d_2, D_2 \in \mathbb{R}$ such that $d_1 \le D_1$ and $d_2 \le D_2$,
there exists a flow f with arrival-curve $\alpha_f = \gamma_{r,b}$ and no minimum packet length whose content is replicated and sent to two systems S_1 and S_2 in which the packets of f suffer a delay bounded in $[d_1, D_1]$ and $[d_2, D_2]$ respectively; the sum of the outputs of the two systems is then processed by a packet-elimination function $\texttt{PEF}_n(f)$,
such that, the arrival curve α_f^* defined in (3) is the best VBR arrival curve for f at the output of $\texttt{PEF}_n(f)$.

Note that, due to the inherent nature of the PEF processing packets, there could exist staircase arrival-curves that fit the worst-case traffic even better than the arrival curve provided in Corollary 1. However, deterministic computational tools process concave piecewise-linear arrival-curves better

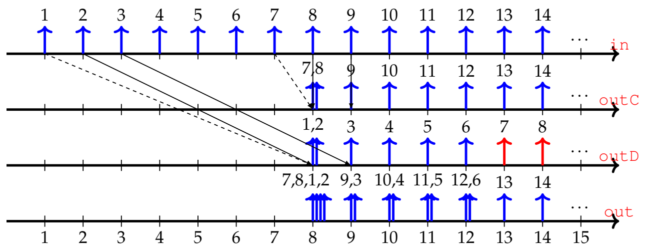

Fig. 14. Trajectory showing that the results of Corollary 1 is tight for the toy example. The cumulative function of f, starting at Time Unit 8 in the above trajectory, is given as a dashed blue line in Fig. 12.

than staircase arrival-curves [32]. Proposition 1 proves that we obtain the best arrival-curve in the family of concave piecewise-linear arrival-curves with two segments or less.

Intuition of the Proof with the Toy Example: We give an intuition of the proof of Proposition 1 by using the toy example of Fig. 2. Our goal is to obtain a cumulative function $R^*(t)$ at the output of PEF such that $t \mapsto R^*(t) - R^*(s)$ "perfectly fits" the arrival curve $\gamma_{2r_0,4b_0} \otimes \gamma_{r_0,8b_0}$, for some observation starting time s (as in Fig. 12). In the scenario of Fig. 3, we already achieved a peak rate of $2r_0$ by using a disconnection of the short link for a duration equal to the delay difference between the two paths. To obtain the worst-case burst, we now simply need to use the jitter within each path and synchronize the moments when the maximum burst on each path reaches the PEF.

This is done by using the trajectory shown in Fig. 14. Here, Packet 1 suffers the maximal delay on the long path and the following packets suffer only 6 t.u. This causes Packets 1 and 2 to exit D at the same time. We do the same with Packets 7 and 8 through C and we synchronize these two events at the same time, so that four packets simultaneously exit the PEF at t.u. 8. In the figure we spread the packets within t.u. 8 for ease of reading, but they exit at the exact same time (t.u. 8). Because of this, we can also put an arbitrary order of arrivals among them (in a real-life system it means that there exists a very small difference in their reception instants).

If we start counting the packets at Time Unit 8, we observe the cumulative arrival function shown in dashed blue in Fig. 12, for which it is clear that the arrival curve in solid red is the best concave piecewise-linear envelope with two segments. The formal proof of Proposition 1 in Appendix C.3 extends the intuition for any choice of values for r, b, d_1, d_2, D_1, D_2 (assuming no minimal packet length).

5.2 Reordering Introduced by the Packet Replication and Elimination Functions

In Sec. 5.1 we provide a characterization of the traffic at the output of a PEF in the form of an arrival curve. The arrival curve can then be used to compute delay and backlog bounds on subsequent vertices, from which we can obtain the ETE delay bounds. However, as we can observe in the toy example (Figures 3 and 14), the data units at the output of the PEF are out-of-order compared to the input. The mis-ordering of the flow's data units cannot be captured by arrival curves. As described in Sec. 2, it still has an effect on the performances of time-sensitive networks [28].

Two metrics are of interest when quantifying mis-ordering in time-sensitive networks: the reordering late time offset (RTO) and the reordering byte offset (RBO) [28], [33]. In this paper we focus on the mis-ordering as a consequence of the redundancy. Thus we are only interested in defining reordering metrics after the PEF, relative to a reference order defined before the PRF.

For a flow f and two vertices n [resp., o] containing the observation points v [resp., w] such that f is packetized at v and w, n is not an EP-vertex of $\mathcal{G}(f)$ and o is a diamond ancestor of n in $\mathcal{G}(f)$, we denote by $\lambda_v(f, w)$ the RTO of the data units of flow f at the observation point v, with respect to their order at w, as defined in [28], [33]. With the restrictions on v and w, $\lambda_v(f, w)$ is well defined from [28], [33] because each data unit of f is observed at most once at w and v, thus the arrival instant of each data unit at w and v is well defined. Similarly, with v and w meeting the same conditions, we denote by $\pi_v(f, w)$ the RBO, as defined in [28], [33] of the data units of flow f at v with respect to their order at the reference w.

If $\texttt{POF}_n(\{f\}, o)$ is a packet-ordering function that forces the data units of f to be in the same order as their order at the output of o, then with v being the input of $\texttt{POF}_n$, $\lambda_v(f, o^*)$ gives the minimum value for the timeout parameter T of POF algorithm and $\pi_v(f, o^*)$ gives its required buffer size [28, §IV.B]. In general, if a destination d does not support any mis-ordering, then a function $\texttt{POF}_d(\{f\}, \text{source}(f))$ that uses the reference $o = \text{source}(f)$ is placed just before delivery to the application. The end-to-end RTO and RBO $\lambda_d(f, \text{source}(f)), \pi_d(f, \text{source}(f))$ must be obtained to correctly configure this POF.

Proposition 2 (RBO $\leq \alpha$(RTO))**.** For a flow f, and two observations points v, w meeting the above conditions, if $\lambda_v(f, w) < +\infty$, then

$$\pi_v(f, w) \leq \alpha_{v,f}(\lambda_v(f, w)) \tag{4}$$

The result is directly obtained by writing the definitions of the two notions. Its formal proof is in Appendix C.4. Proposition 2 combined with our results from Sec. 5.1 show that we can focus on the effect of the PEF on the RTO to also obtain a bound on the RBO.

Theorem 2 (RTO at the output of a PEF)**.** Consider a flow f, a vertex n containing a packet-elimination function $\texttt{PEF}_n(f)$ and a diamond ancestor a of n in $\mathcal{G}(f)$. Denote by $d_f^{a\to n}$ [resp., $D_f^{a\to n}$] a lower [resp., upper] delay bound for f between the output of a and the input of $\texttt{PEF}_n(f)$, along any possible path in the graph $\mathcal{G}(f)$. Then $\lambda_{\texttt{PEF}_n(f)^*}(f, a)$, the reordering late time offset (RTO) of f at the output of the PEF, with respect to a, verifies

$$\lambda_{\texttt{PEF}_n(f)^*}(f, a^*) \leq \left| D_f^{a\to n} - d_f^{a\to n} - \alpha_{a*}^{\downarrow}(2L^{\min}) \right|^+ \tag{5}$$

where $|x|^+ \triangleq \max(0, x)$, α_{f,a^*} is an arrival curve for f at the output of the input port within a and $\alpha_{f,a^*}^{\downarrow}$ is its lower pseudo-inverse[2] defined in [34, §10].

Theorem 2 is a direct application of [28, Thm. 5] for the system located between the diamond ancestor and the output of the PEF, see Appendix C.5.

Application to the Toy Example: The lower-pseudo inverse of $\alpha_{f,B^*} = \gamma_{r_0,b_0}$ in the toy example of Fig. 2 is $\alpha_{f,B^*}^{\downarrow} : x \mapsto |x - b_0|^+/r_0$. In the toy example, all packets

2. For $f : \mathbb{R} \to \mathbb{R} \cup \{-\infty, +\infty\}$ a wide-sense increasing function, its lower pseudo inverse $f^{\downarrow}$ is defined by $f^{\downarrow}(y) = \inf\{x | f(x) \geq y\}$.

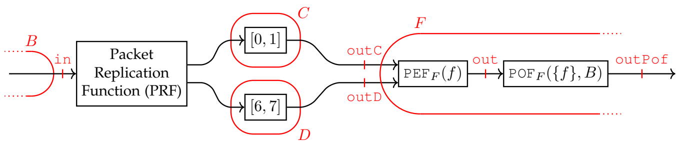

Fig. 15. Toy example of Fig. 2, with a packet-ordering function (POF) placed after the PEF to correct the mis-ordering caused by the redundancy.

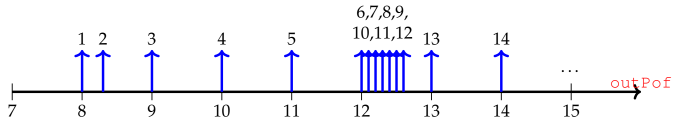

Fig. 16. Trajectory of the packets at the output of the POF of Fig. 15 when the POF processes the packets from the trajectory of Fig. 14.

have the same size of one d.u., so $\alpha^{\downarrow}_{f,B^*}(2L_{\min}) = 1$ t.u. Applying Theorem 2 proves that the RTO at the output of the PEF within F in Fig. 2 is bounded by 6 t.u. In the trajectory of Fig. 14, we observe that d.u. 6 is late by 4 t.u. with respect to d.u. 7. The worst-case RTO is hence comprised between 4 and 6 t.u.

Assume now that we place, after the PEF, the function POF$_F(\{f\}, B)$, a packet-ordering function enforcing for f the order defined at B (Fig. 15). With Theorem 2, we know that its timeout T should be of at least 6 time units and it requires a buffer of at least 14 d.u. (Proposition 2 and Fig. 12). In the trajectory of Fig. 14, the POF receives the traffic from the Line "out" and forces the data units to be in the same order as on the Line "in". The resulting output is given in Fig. 16. We observe two main characteristics of the POF; they have been widely studied in [28].

First, we note that all data units continue to have a delay upper-bounded by 7 t.u. Indeed, none of the data units has been lost for the POF thus the POF does not increase the end-to-end (ETE) latency of the data units [28, Thm. 4].

Second, we observe that the traffic at the output of the POF (Fig. 16) is much more bursty than the traffic at the output of the PEF (Line "out", Fig. 14). We observe that seven d.u. exit the POF at the same time (t.u. 12). The traffic is hence no more constrained by $\alpha_{PEF^*} = \gamma_{2r_0,4b_0} \otimes \gamma_{r_0,8b_0}$, the arrival curve of the flow at the output of the PEF, obtained by applying Theorem 1 (Sec. 5.1). We apply Corollary 1 of [28]: If none of the data units is lost for the POF (at least one replicate of each data unit reaches the PEF), then $\alpha_{POF^*} = \gamma_{2r_0,18b_0} \otimes \gamma_{r_0,15b_0} = \gamma_{r_0,15b_0}$ is an arrival curve of f at the output of the POF. The trajectory in Fig. 16 is indeed $\gamma_{r_0,15b_0}$-constrained. If both replicates of a data unit can be lost, then $\gamma_{r_0,15b_0+Tr_0}$ is an arrival curve for f at the output of the POF, with T being the timeout parameter of the POF.

Placing a POF after a PEF hence comes with benefits and drawbacks, as summarized on the first line of Table 3.

6 Analysis of the Interactions between PREFs and Traffic Regulators

Sec. 5.2 shows that a packet-ordering function (POF) can be used after a PEF to remove the mis-ordering caused by the redundancy. Similarly, regulators can be used after a PEF to remove the burstiness increase caused by the redundancy, especially if the downstream systems cannot support the worst-case traffic of the PEF output (Theorem 1).

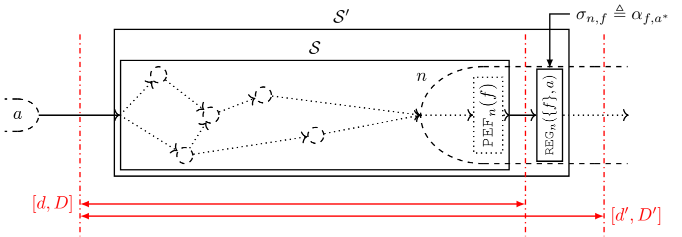

Fig. 17. Notations for the analysis of the interactions between PEF and a PFR for a flow f. Vertices of $\mathcal{G}(f)$ are shown in dashed circles/ovals and edges are shown with dotted arrows.

Traffic regulators come in two flavors: per-flow regulators (PFRs) and interleaved regulators (IRs). Both are configured with per-flow contracts, $\{\sigma_{f,n}\}_f$, and force each flow f to be $\sigma_{f,n}$-compliant, delaying the packets if required.

Hence, when the shaping curve $\sigma_{f,n}$ for each flow f equals the arrival curve that the flow had before the redundant section, then the regulators remove any burstiness increase caused by the redundancy, thus making the redundancy transparent to the downstream nodes. However, regulators are themselves queuing systems and their effect on the worst-case ETE delay should be accounted for.

In this section, we first analyze the interactions between PEF and a regulator placed directly after. We evaluate how these interactions affect the ETE delay guarantees of the flows, and we show that the conclusions highly depend on the nature of the regulator (either PFR or IR). We last analyze the effect of a POF placed after the PEF and before the REG.

6.1 Delay Bound Analysis of PREFs Combined with Per-Flow Regulators

Consider a vertex n containing a function PEF$_n(f)$ and consider a diamond ancestor a of n in $\mathcal{G}(f)$ (Fig. 17). Between a and n, the flow follows $\mathcal{G}(f)$, with potentially multiple vertices and multiple paths. Consider the system $\mathcal{S}$ between the output of a and the output of PEF$_n(f)$ (solid box in Fig. 17). Due to all the possible paths with different lengths, $\mathcal{S}$ is neither FIFO nor lossless in the general case. We denote by d [resp., D] a delay lower-bound [resp., upper-bound] for each forwarded d.u. through $\mathcal{S}$. The delays d and D are well-defined because the data units are seen at most once at the output of the PEF. Note that the PEF has no delay, hence d [resp., D] verifies $d = d_f^{a\to n}$ [resp., $D = D_f^{a\to n}$], *i.e.*, a delay bound along any possible paths $a \to n$ is a delay bound through $\mathcal{S}$.

After $\mathcal{S}$, and still within vertex n (dashed oval on the right of Fig. 17), we place a PFR: REG$_n(\{f\}, a)$ with shaping curve $\sigma_{n,f} \triangleq \alpha_{f,a^*}$. We now consider the system $\mathcal{S}'$ made of $\mathcal{S}$ followed by the PFR, and we are interested in the delay bounds $[d', D']$ for the non-lost data units through $\mathcal{S}'$. If $\mathcal{S}$ was FIFO, we could use the essential *shaping-for-free* property of regulators [9], [12]: As f is $\sigma_{f,n}$-constrained at the input of $\mathcal{S}$, the regulator would not have increased the ETE delay of the data units; we write this as $D' = D$. But, as $\mathcal{S}$ is not FIFO, the PFR does not guarantee the *shaping-for-free* property, as we show on the toy example.

Application to the Toy Example: Fig. 18 considers the toy example from Fig. 2, to which we add the PFR REG$_n(\{f\}, B)$ within vertex F (dashed oval on the right in Fig. 18), just after the function PEF$_F(f)$. With the above notations, system

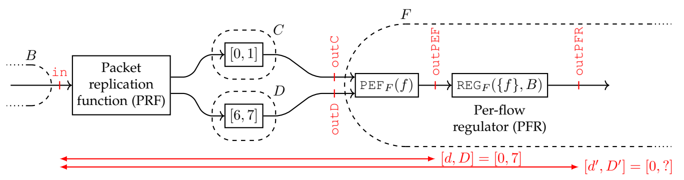

Fig. 18. Toy example of Fig. 2 with a per-flow regulator (PFR) placed after the PEF to remove the burstiness increase caused by the redundancy.

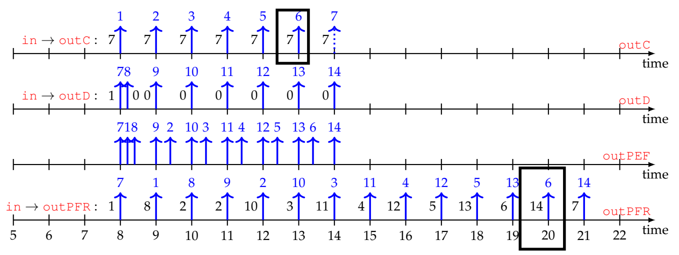

Fig. 19. An acceptable trajectory on the toy example, which shows that the delay bound D' through $\mathcal{S}'$ is at least 14 t.u. The delay of the data units from "in" to the observation points are given on the left of the packets.

$\mathcal{S}$ is between the observation points "in" and "outPEF", with the delay bounds $[d, D] = [0, 7]$ t.u. System $\mathcal{S}'$ is between the observation points "in" and "outPFR", and we seek to obtain a delay-bound D' for $\mathcal{S}'$.

Fig. 19 presents an acceptable trajectory at the different observation points using the same input "in" as in Fig. 3. The path through vertex C forwards all packets with a constant delay of 7 t.u., whereas the path through vertex D drops Packets 1 to 6, then forwards Packet 7 with a delay of 1 t.u. (its worst-case delay) and finally forwards the following packets with a delay of 0 t.u. (its best-case delay). The line "outPEF" gives the resulting trajectory at the output of the PEF that removes any duplicates.

Based on its input ("outPEF") and on its shaping curve ($\sigma_{f,F} = \alpha_{f,B^*} = \gamma_{r_0,b_0}$), the PFR outputs the packets as shown on the Line "outPFR". Recall that the PFR $\mathtt{REG}_n(\{f\}, a)$ is *itself* a FIFO system (model in Sec. 4.3).

We observe that the d.u. 6 suffers through $\mathcal{S}'$ a total delay of 14 t.u.; this is twice the delay upper-bound D through $\mathcal{S}$ alone. We note that this high delay for d.u. 6 can be explained by the time needed by the PFR to process d.u. 1 to 5 and 7 to 13 that arrived before d.u. 6 and to pace them as required by the shaping curve. This is done even though d.u.s 7 to 13 are out of order ("too early") with respect to d.u. 6. At "outPFR", the packet containing d.u. 6 is late with respect to d.u. 7 by 12 t.u. Hence, the reordering late time offset (RTO) of the flow through $\mathcal{S}'$ (*i.e.*, at the output of $\mathcal{S}'$, using the input of $\mathcal{S}'$ as reference) is at least 12 t.u., while it was bounded by only 6 t.u. through $\mathcal{S}$ alone (Sec. 5.2).

We observe that the output of the PEF is bursty and out of order, and the PFR placed afterwards paces the packets to remove the burstiness. But, by doing so, the PFR worsens the mis-ordering of the packets (12 instead of 6) and increases the delay of the late packets (Packet 6), thus increasing the worst-case ETE delay (at least 14 t.u.). As such, the regulator comes with a *delay penalty*. With PFRs configured with leaky-bucket shaping curves, we can upper-bound this delay penalty for any networks.

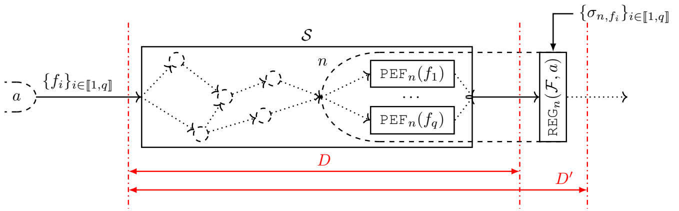
$\{\sigma_{n,f_i}\}_{i\in[\![1,q]\!]}$
$\mathcal{S}$
a
$\{f_i\}_{i\in[\![1,q]\!]}$
n
PEF_n(f_1)
...
PEF_n(f_q)
REG_n(F, a)
D
D'

Fig. 20. Notations for the analysis of the interactions between PEFs and an interleaved regulator (IR) for an aggregate of flows $\mathcal{F} = \{f_i\}_{i\in[\![1,q]\!]}$.

Theorem 3 (Bound on the delay penalty of a PFR placed after a PEF)**.** Assume that the PFR $\mathtt{REG}_n(\{f\}, a)$ is configured with a leaky-bucket shaping curve $\sigma_{n,f} = \gamma_{r,b}$, and that $\sigma_{n,f}$ is an arrival curve of f at the input of $\mathcal{S}$. If d [resp., D] is a lower [resp., an upper] bound on the delay of f through the system $\mathcal{S}$ (Fig. 17), then $d' = d$ [resp., $D' = 2D - d$] is a lower [resp., an upper] bound on the delay of f through $\mathcal{S}'$.

The proof combines Theorem 1 with the service-curve characterization of a PFR [9, §1.7.3] to obtain a delay bound within the PFR, see Appendix C.6. Combined with [28, Thm. 7], we directly obtain the following result.

Corollary 2 (Bound on the RTO at the output of a PFR placed after a PEF)**.** With the notations of Theorem 3, the RTO of f at the output of $\mathtt{PFR}_n(\{f\}, a)$, with reference a, verifies

$$\lambda_{n,\mathtt{PFR}^*}(f,a) \;\leq \lambda_{n,\mathtt{PEF}^*}(f,a) + D - d$$

with $\lambda_{n,\mathtt{PEF}^*}(f,a)$ the RTO of f at the output of the PEF, again with respect to the order of the data units at a.

Application to the Toy Example: Applying Theorem 3 shows that $2D - d = 14$ t.u. is an upper delay bound through $\mathcal{S}'$. As it is achieved by d.u. 6 in Fig. 19, it is also the worst-case delay. Applying Corollary 2 to the toy example gives that 13 t.u. is an upper-bound on the RTO of the flow at the output of the PFR, with respect to the order of the packets at B. Data Unit 6 in the trajectory achieves a reordering offset of 12 t.u. (with respect to d.u. 7), thus the worst-case RTO at the output of $\mathcal{S}'$ in the toy example is between 12 and 13 t.u.

When a PFR is used after a PEF, the current subsection shows that the *shaping-for-free* property does not hold, but Theorem 3 captures the delay penalty by using the service-curve characterization of PFRs, combined with the arrival curve obtained from Theorem 1. As we do not know any service-curve characterization for an IR, we cannot apply the Theorem 3 to interleaved regulators (IRs).

6.2 Instability of the Interleaved Regulator Placed after a Set of PEFs

With an interleaved regulator (IR), several flows $\mathcal{F} = \{f_i\}_{1\leq i\leq q}$, sharing the same redundant section $a \to n$ are processed by the same IR $\mathtt{REG}_n(\mathcal{F}, a)$, after their respective elimination function $\mathtt{PEF}_n(f_i)$ for $i \in [\![1, m]\!]$ (see Fig. 20).

When the aggregate contains a unique flow, then the IR is a PFR. Therefore, we do not expect the *shaping-for-free* property to be valid with the IR either. However, as opposed to the PFR, we exhibit an adversarial model in which any IR placed after the PEFs and processing several flows yields unbounded latencies.

Theorem 4 (Instability of the IR placed after the PEFs)**.** Consider a network with graph $\mathcal{G}$ and consider $q \in \mathbb{N}$ flows $f_1, \dots, f_q$ (see Fig. 20). Take two vertices a and n such that, for each $i \in [\![1, q]\!]$, a is a diamond ancestor of n in $\mathcal{G}(f_i)$. Assume that

(a) for each $i \in [\![1, q]\!]$, vertex n contains $\texttt{PEF}_n(f_i)$, a PEF for f_i,
(b) vertex n contains $\texttt{REG}_n(\{f_i\}_{i\in[\![1,q]\!]}, a)$, an interleaved regulator (IR) for the aggregate, placed after the PEFs, with the same leaky-bucket shaping curve for each flow: $\forall i \in [\![1, q]\!], \sigma_{f_i,n} = \gamma_{r,b}$,
(c) all graphs $\{\mathcal{G}(f_i)\}_{i\in[\![1,q]\!]}$ share at least two different paths P_1, P_2 to reach n from a.

For $q \in \mathbb{N}$ and $r, b, d_1, d_2, D_1, D_2 \in \mathbb{R}^+$ with $d_1 \leq D_1$, $d_2 \leq D_2$ and $D_1 \leq D_2$ (flipping the indexes if required), if

(d) b is greater than the minimum packet length,
(e) d_1, D_1, d_2, D_2 are not all equal, and
(f) $q \geq q_{\min}$ with

$$q_{\min} \triangleq \left\lfloor \frac{2r\,|d_2 - D_1|^+}{b} + 2 \right\rfloor + 1$$

then there exists an adversarial traffic arrival at a for each of the q flows and an adversarial implementation of the paths $\{P_j\}_j$ such that

1/ each flow f_i is $\gamma_{r,b}$-constrained at a,
2/ for each data unit m belonging to one of the flows $\{f_i\}_{i\in[\![1,q]\!]}$, if m is not lost on P_1 [resp., on P_2], then its delay along P_1 [resp., along P_2] is within $[d_1, D_1]$ [resp., within $[d_2, D_2]$],
3/ flows $\{f_i\}_i$ have an unbounded latency within the IR,
4/ P_1 and P_2 are both FIFO,
5/ the system $\mathcal{S}$ made of the sub-graph of $\mathcal{G}$ between a and the output of the PEFs (Fig. 20) remains lossless and FIFO-per-flow for each f_i.

The proof is in Appendix C.7. It relies on the trajectory developed for the proof of [14, Prop. 7.3]. The main idea is to use the mis-ordering caused by PREFs and the property that the IR looks only at the head-of-line packet to generate blocking situations with always-increasing packet delays.

Note that only Properties 1/ to 3/ of Theorem 4 are required to prove the validity of the adversarial model. However, our adversarial model provides additional Properties 4/ and 5/; they are of interest when considering the solutions for preventing the instability, as we illustrate in Sec. 6.3. Theorem 4 also provides a mean to obtain the following wider result, whose proof is in Appendix C.8.

Corollary 3 (Instability of the interleaved regulator after a non-FIFO system, even if the system is FIFO-per-flow and lossless)**.** For any $D_{\max} > 0$, $r > 0$, b greater than the minimum packet length, and for any IR that processes 3 or more flows $\{f_i\}_i$ using the same leaky-bucket shaping curve $\gamma_{r,b}$, there exists a lossless FIFO-per-flow system $\mathcal{S}$ and a $\gamma_{r,b}$-constrained adversarial generation of each flow at the input of $\mathcal{S}$ such that, when the IR is placed after $\mathcal{S}$, the delay of the flows through $\mathcal{S}$ is upper-bounded by $D_{\max}$ but the delay of the flows through the IR is not bounded.

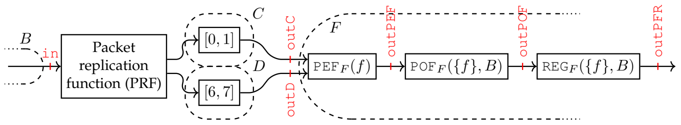

Fig. 21. Toy example of Fig. 2, to which we added a POF followed by a PFR.

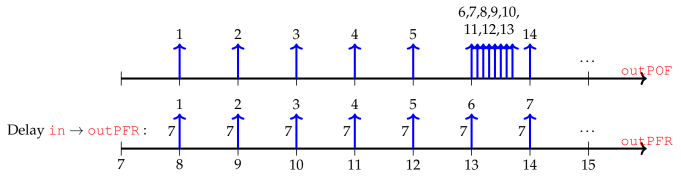

Fig. 22. Output of the POF and of the PFR of Fig. 21 when they process the trajectory of Fig. 19.

6.3 Effect of the Packet-Ordering Function on the Combination of a PEF with Traffic Regulators

Table 3 summarizes the benefits and drawbacks of using regulators after a PEF, as analyzed in Sections 6.1 and 6.2. We observe that the drawbacks of the regulators appear symmetrical with respect to those of the POF. For example, a main issue of the POF is the burstiness of the traffic at its output; this can be corrected by using a regulator. A main issue of the REGs is the delay penalty caused by the out-of-order input; this can be solved by placing a POF just before.

The combination PEF + POF + REG appears as a potential solution for keeping the benefits of both the POF and the REG without their main drawbacks. We first analyze this new configuration on the toy example.

Application to the Toy Example: Let us first add a POF before the PFR of the single-flow situation in Fig. 18. It gives the situation presented in Fig. 21. The POF enforces the order of the data units as seen at B. Assume for example that it receives the traffic defined by the line "`outPEF`" of Fig. 19. Then the POF outputs the data units as on Line "`outPOF`" of Fig. 22. The PFR further processes this trajectory to spread the data units as per the flow's contract and outputs them as on the Line "`outPFR`" of Fig. 22. The resulting traffic is compliant with the initial arrival curve α_{r_0,b_0}. We observe that all the data units have kept an ETE delay below 7 t.u.

When using an interleaved regulator, Property 5/ of Theorem 4 shows that the re-sequencing must be performed globally on the aggregate processed by the IR, and not for each flow individually. The above observations are summarized in the following result, valid for both PFRs and IRs.

Theorem 5 (Elimination-resequencing-reshaping is for free)**.** Consider a network with graph $\mathcal{G}$ and consider a set of one or more flows $\mathcal{F}$. Take a and n two vertices of $\mathcal{G}$ such that for each flow $f \in \mathcal{F}$, a is a diamond ancestor of n in $\mathcal{G}(f)$ (see Fig. 23). Assume that the CBQS within n is preceded by the following functions, in this order: a set of parallel packet-elimination functions $\{\texttt{PEF}_n(f)\}_{f\in\mathcal{F}}$, followed by a unique packet-ordering function with configuration $\texttt{POF}_n(\mathcal{F}, a)$, and finally a regulator with configuration $\texttt{REG}_n(\mathcal{F}, a)$. Denote by d [resp., D] a lower bound [resp., an upper bound] for the delay of the non-lost data units of $\mathcal{F}$ through the system $\mathcal{S}$ between a and the output of the PEFs.

• If $\mathcal{S}$ is lossless for $\mathcal{F}$ (*i.e.* for every data unit m of the aggregate, at least one packet containing m reaches the PEFs), then d [resp., D] is also a lower bound [resp.,

TABLE 3
Benefits and Drawbacks of Several Configurations, Compared to the Situation with the PEF(s) only.

Configuration	Benefits with respect to the PEF alone	Drawbacks with respect to the PEF alone
PEF + POF	• Destination receives the data units in order • Reordering-for-free: the POF does not increase the end-to-end delay bounds (when at least one replicate per d.u. is received).	• The POF worsens the arrival curve; this can lead to higher delay bounds in downstream nodes. • Increased hardware complexity (Figure 8).
PEF + REG	• Output traffic keeps the arrival constraints it had before the redundant section, resulting in smaller delay bounds in downstream nodes.	• Delay penalty due to mis-ordering: with PFR: delay penalty with a guaranteed maximum delay; with IR: unbounded delay. • Increased hardware complexity (Figure 9).
PEF + POF + REG	• Destination receives the data units in order • Reordering-for-free and shaping-for-free: [POF + REG] does not increase the delay bounds (when at least one replicate per data unit is received). • Output traffic keeps the same arrival constraints as it had before the redundant section.	• Increased hardware complexity (Figures 8 and 9).

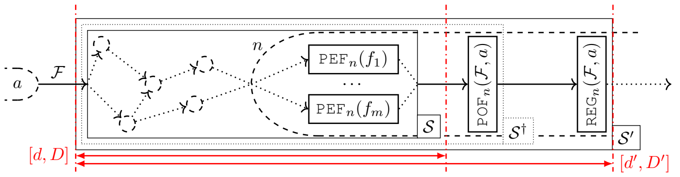

Fig. 23. Notations of Theorem 5. An aggregate re-sequencing followed by a REG is placed after the PEFs. We are interested in the delay bounds through system $\mathcal{S}'$.

an upper bound] for the delay of the non-lost data units through $\mathcal{S}'$, which we note $[d', D'] = [d, D]$.

• Otherwise, denote by T the timeout value of the POF [28, §III.D]. Then d [resp., $D + T$] is a lower bound [resp., an upper bound] for the delay of the data units through $\mathcal{S}'$, *i.e.*, $[d', D'] = [d, D + T]$.

The proof in Appendix C.9 first applies [28, Theorem 4] to obtain the delay bounds through the system $\mathcal{S}^\dagger$ on Fig. 23. This system is FIFO thus [9, Thm. 5] can be applied.

Therefore, the "PEF + POF + REG" configuration provides all the benefits on the network performance bounds associated with the "PEF + POF" and the "PEF + REG" configurations, removing most of their drawbacks. This is summarized on the last line of Table 3. Only the hardware cost remains a drawback, as the models of Figures 8 and 9 must be implemented.

7 Evaluation of the Framework on an Industrial Use-Case

In this section, we use a modified version of FP-TFA [13, §VI] that implements the results from Sections 5 and 6 to compute end-to-end delay bounds in a representative industrial use-case that contains PREFs. FP-TFA has been chosen because it can compute delay bounds for general topologies, *i.e.* even for those with cyclic dependencies [13].

Network Description: We consider the Volvo core TSN network [35]. Its physical topology is given in Fig. 24. The network contains two redundant control units `P1` and `P2` [35, Page 4]. Each of the four micro-controller units (MCUs) acts as a gateway between the core TSN network and the local networks running on legacy protocols. We hence assume that the MCUs are legacy devices that support only 100Mbps full-duplex links and cannot implement the recent technologies of TSN or DetNet, such as PREOFs. We assume that their applications cannot handle any duplicate.

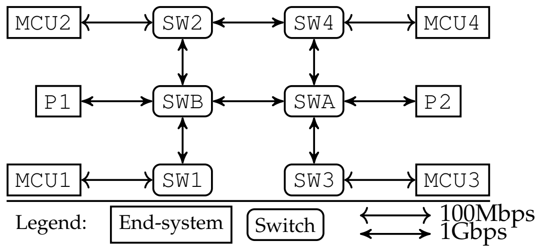

Fig. 24. Simplified physical topology of the Volvo core TSN Network. From [35].

TABLE 4
Traffic Profiles. Realistic Use-Case Based on the Values for *Command and Control* Flows in [35, Page 13].

Name	Payload size	Period at source	Deadline
`S`	64B	0.5ms	0.2ms
`M1`	92B	2ms	0.8ms
`M2`	120B	3.5ms	1.4ms
`B`	150B	5ms	2ms

Flow Description: We focus on the *Command and Control* class and consider four different periodic traffic profiles within the class. Their characteristics are based on [35, Page 13] and listed in Table 4. For each traffic profile and for each `MCU`, there exist a multicast flow that carries the sensor data from the MCU to both `P1` and `P2` and a unicast flow per control unit (2 in total) that carries the commands from the control unit to the MCU (see Table 5).

To meet stringent loss-ratio requirements, flows are redounded by using PREFs, whenever two alternative paths can be found for a (source, destination) tuple. In total, the network contains 48 flows, including 40 redounded flows, 16 of which are also multicast.

Service Description: As the class of interest is of highest priority, each CBQS offers to the aggregate a service rate equal to the capacity of the transmission link (either 100Mbps or 1Gbps). We also assume that the technological latency within each output port is below $2\mu s$, and we neglect input-port and switching-fabric latencies.

Comparison of the Analytical Models: We first set the load of the network at 5.2%. We compare the *intuitive approach* from Sec. 2.1 with the *tight model* that relies on Theorem 1. In Fig. 25, we provide the deterministic lower and upper

TABLE 5
Flow Path for $i \in \{1, 2, 3, 4\}$, $p \in \{\mathtt{S}, \mathtt{M1}, \mathtt{M2}, \mathtt{B}\}$.

Name	Source	Dest.	Redundancy
`C_MCU`i`_P12_`p	`MCU`i	`P1, P2`	For `C_MCU3_P12_`p [resp., `C_MCU4_P12_`p], dest. `P2` [resp., `P1`] is not protected
`C_P1_MCU`i`_`p	`P1`	`MCU`i	Except for `C_P1_MCU1_`p
`C_P2_MCU`i`_`p	`P2`	`MCU`i	Except for `C_P2_MCU3_`p

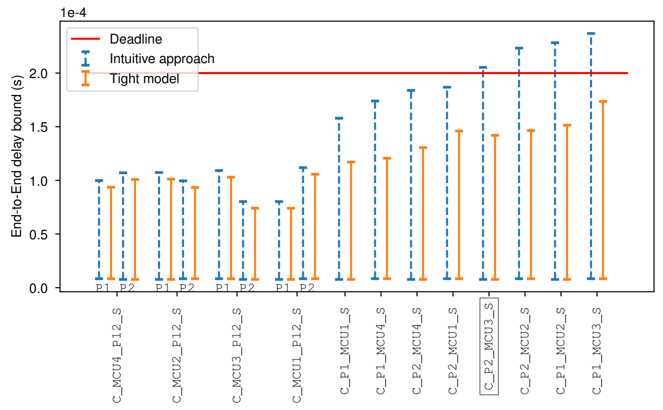

Fig. 25. Comparison of the guaranteed end-to-end (ETE) latency intervals (upper and lower bounds) for each flow and each destination, obtained by using either the intuitive approach or the tight model.

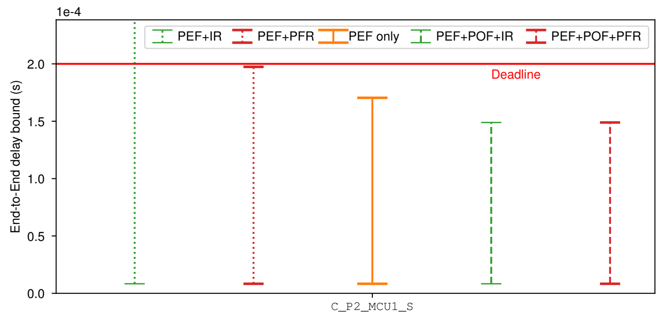

Fig. 26. Comparison of the guaranteed ETE latency intervals with several technological choices. Delay bounds without any REG are shown in the middle. The bars on the left are the guaranteed intervals when the flows are regulated after the PEF, but without any POF. When a POF is additionally placed between the PEF and the REG, we obtain the results on the right of the baseline.

bounds of the latency of each flow for each of its destinations. The delay upper-bounds are obtained by using the fix-point version of FP-TFA [13, § VI.C], modified for taking into account the effect of PREFs with either the intuitive approach or the tight model. The exact best-case and worst-case latencies for the flow are guaranteed to be within the provided interval, thus the smaller the guaranteed interval the better the model.

We observe that an analysis of the network by using the tight model concludes that all flows meet their deadline, whereas the same analysis that uses the intuitive approach shows that four flows may violate their deadlines. The delay bounds for all flows, including those that are not redounded by PREFs, are improved with the tight model. For example, the flow in a box in Fig. 25, from P2 to MCU3, is not redounded, but the tight model still computes a guaranteed delay interval tighter than with the intuitive approach. Indeed, the flow shares the link SWA → SW3 and SW3 → MCU3 with redounded flows, for which the burst bounds obtained with the tight model are smaller. Hence, the delay that this flow suffers in SWA and SW3 has a better bound with the tight model than with the intuitive one.

Comparison of the Technological Solutions: Fig. 25 shows that, at low network load, the network edges withstand the peak rate an increased burstiness at the output of the PEFs, even if they rely only on 100Mbps links.

We now consider the same network but we increase the load up to 88% by reducing the period of each flow. We focus on the four redounded flows from P2 to MCU1. Each of them is processed by a PEF within SWB to eliminate the duplicates coming from SW2 and SWA and each of them present a peak rate and and increased burstiness after its PEF.

We evaluate the opportunity to shape the four flows with their source profile before they compete with the four other flows coming from P1 in the output port of SWB. We can either use four per-flow regulators (PFRs) (each processing a unique flow), or we can use a unique interleaved regulator (IR), because they all share the same reference point P2.

Fig. 26 focuses on flow C_P2_MCU1_S. The baseline guaranteed delay interval (in the middle) is obtained from the application of the tight model without any regulator. We note that the flow is schedulable, but as the network load is higher, its safety margin is reduced with respect to Fig. 25.

The dotted bars on the left of the baseline represent the guaranteed delay intervals obtained when the flows are processed, either with an IR (far-left), or with four independent PFRs, but without using any POF. For the IR, no guarantee can be obtained per Theorem 4. For the PFR, the flow remain schedulable but its safety margin is drastically reduced by the delay penalty of the PFR (Theorem 3).

The dashed bars on the right of the baseline represent the guaranteed delay bounds when using the combination POF+REG after the PEF, assuming that for each data unit, at least one replicate is not lost. On the far-right, the delay bounds with four per-flow POFs placed before the PFRs (as in Fig. 11), and the other bar represents the delay bounds with a unique POF for the aggregate before the IR (as in Fig. 10). The shaping-for-free property holds in both cases, thus their delay bounds are equal. They represent a 13% improvement with respect to the baseline. Indeed, the regulators reduce the downstream burst, thus reducing the worst-case delay in the low-capacity link SW1→MCU1.

8 Conclusion

We provide a toolbox of network-calculus results that give theoretical foundations for the worst-case analysis of DetNet PREOF (*packet replication, elimination and ordering functions*) and TSN FRER (*frame replication and elimination for redundancy*). The toolbox contains an output-arrival-curve characterization of the packet-elimination function that is tighter than any other variable-bit-rate or leaky-bucket arrival curves. It also contains a quantification of the amount of mis-ordering caused by the redundancy.

We further analyze the interactions between the packet-elimination function, the packet-ordering function and traffic regulators. We show that the latter can cancel the burstiness increase caused by the redundancy. But when traffic regulators are placed immediately after the packet-elimination function, they do not enjoy the shaping-for-free property: Per-flow regulators induce a delay penalty that we upper-bound, whereas interleaved regulators (such as TSN *Asynchronous Traffic Shapers*) induce unbounded latencies. Shaping-for-free can be retrieved if the data units are re-ordered after the elimination function and prior to shaping.

The users of TSN FRER and TSN asynchronous traffic shaping (ATS) are invited to bear in mind the conflicting interactions outlined in this paper, as no packet-ordering function is available within TSN at the time of writing.

We finally apply our theoretical and practical results on a representative industrial use-case. The latency bounds obtained with the toolbox are significantly tighter than those obtained with an intuitive approach. We also highlight the end-to-end latency gain obtained on the use-case when traffic regulators are placed after the redundant section with a reordering function in between.

REFERENCES

[1] IEEE and SAE, "P802.1DP – TSN for Aerospace Onboard Ethernet Communications —." https://1.ieee802.org/tsn/802-1dp/.

[2] IEC and IEEE, "IEC/IEEE 60802 - Time-Sensitive Networking Profile for Industrial Automation," vol. IEC/IEEE 60802 (D1.1), 2019. http://www.ieee802.org/1/files/private/60802-drafts/d1/60802-d1-1.pdf.

[3] IEEE, "Draft Standard for Local and metropolitan area networks — Time-Sensitive Networking Profile for Automotive In-Vehicle Ethernet Communications," *IEEE P802.1DG™/D1.1*, vol. In IEEE802.1 private repository. To obtain the access credentials, visit https://www.ietf.org/proceedings/52/slides/bridge-0/tsld003.htm or contact the IEEE802.1 chair., Oct. 2019. http://www.ieee802.org/1/files/private/dg-drafts/d1/802-1DG-d1-1.pdf.

[4] N. Finn, P. Thubert, B. Varga, and J. Farkas, "Deterministic Networking Architecture," no. RFC 8655, 2019. https://www.rfc-editor.org/info/rfc8655.

[5] J. Farkas, "TSN Basic Concepts." https://www.ieee802.org/1/files/public/docs2018/detnet-tsn-farkas-tsn-basic-concepts-1118-v01.pdf, Nov. 2018.

[6] A. Bouillard, M. Boyer, and E. Corronc, *Deterministic Network Calculus: From Theory to Practical Implementation*. Networks and Telecommunications, Wiley, 2018. http://doi.org/10.1002/9781119440284.

[7] "IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks," *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)*, pp. 1–1993, July 2018.

[8] N. Finn, J.-Y. L. Boudec, E. Mohammadpour, J. Zhang, B. Varga, and J. Farkas, "DetNet bounded latency," Internet-Draft draft-ietf-detnet-bounded-latency-08, Internet Engineering Task Force / Internet Engineering Task Force, Jan. 2022. https://datatracker.ietf.org/doc/html/draft-ietf-detnet-bounded-latency-08.

[9] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Lecture Notes in Computer Science, Lect.Notes Computer. Tutorial, Berlin Heidelberg: Springer-Verlag, 2001. https://www.springer.com/us/book/9783540421849.

[10] AEE Committee and others, "Aircraft Data Network Part 7, Avionics Full Duplex Switched Ethernet (AFDX) Network, ARINC Specification 664," *Annapolis, Maryland: Aeronautical Radio*, 2002.

[11] E. Mohammadpour, E. Stai, M. Mohiuddin, and J. 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. 02, pp. 1–6, Sept. 2018. http://doi.org/10.1109/ITC30.2018.10053.

[12] J.-Y. Le Boudec, "A Theory of Traffic Regulators for Deterministic Networks With Application to Interleaved Regulators," *IEEE/ACM Transactions on Networking*, vol. 26, pp. 2721–2733, Dec. 2018. http://doi.org/10.1109/TNET.2018.2875191.

[13] L. Thomas, J.-Y. Le Boudec, and A. Mifdaoui, "On Cyclic Dependencies and Regulators in Time-Sensitive Networks," in *2019 IEEE Real-Time Systems Symposium (RTSS)*, pp. 299–311, Dec. 2019.

[14] L. Thomas and J.-Y. Le Boudec, "On Time Synchronization Issues in Time-Sensitive Networks with Regulators and Nonideal Clocks," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, pp. 27:1–27:41, June 2020. https://doi.org/10.1145/3392145.

[15] L. Maile, K.-S. Hielscher, and R. German, "Network calculus results for tsn: An introduction," in *2020 Information Communication Technologies Conference (ICTC)*, pp. 131–140, IEEE, 2020.

[16] L. Zhao, P. Pop, and S. Steinhorst, "Quantitative Performance Comparison of Various Traffic Shapers in Time-Sensitive Networking," *arXiv:2103.13424 [cs]*, Mar. 2021. http://arxiv.org/abs/2103.13424.

[17] "Time-Sensitive Networking (TSN) Task Group —." https://1.ieee802.org/tsn/.

[18] IEEE, "IEEE Standard for Local and metropolitan area networks–Frame Replication and Elimination for Reliability," *IEEE Std 802.1CB-2017*, pp. 1–102, Oct. 2017. https://doi.org/10.1109/IEEESTD.2017.8091139.

[19] D. L. Black, Z. Wang, M. A. Carlson, W. Weiss, E. B. Davies, and S. L. Blake, "An Architecture for Differentiated Services," Request for Comments RFC 2475, Internet Engineering Task Force, Dec. 1998. https://datatracker.ietf.org/doc/rfc2475.

[20] IEEE, "Draft Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment: Asynchronous Traffic Shaping," *IEEE P802.1Qcr/D2.0*, vol. In IEEE802.1 private repository. Access credentials: User: 'p8021' Password: 'go_wildcats', Dec. 2019. http://www.ieee802.org/1/files/private/cr-drafts/d2/802-1Qcr-d2-0.pdf.

[21] P. Heise, *Real-Time Guarantees, Dependability and Self-Configuration in Future Avionic Networks*. PhD thesis, Universitätsbibliothek der Universität Siegen, Siegen, 2018.

[22] P. Heise, N. Tobeck, O. Hanka, and S. Schneele, "SAFDX: Deterministic high-availability ring for industrial low-cost networks," in *2014 7th International Workshop on Communication Technologies for Vehicles (Nets4Cars-Fall)*, pp. 40–44, Oct. 2014.

[23] IEC, "IEC 62439-3:2016 — IEC Webstore — smart manufacturing, industrie 4.0, industry 4.0." https://webstore.iec.ch/publication/24447.

[24] M. Pahlevan and R. Obermaisser, "Redundancy Management for Safety-Critical Applications with Time Sensitive Networking," in *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, pp. 1–7, Nov. 2018.

[25] J. Täubrich and R. von Hanxleden, "Formal Specification and Analysis of AFDX Redundancy Management Algorithms," in *Computer Safety, Reliability, and Security* (F. Saglietti and N. Oster, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 436–450, Springer, 2007.

[26] R. Hofmann, B. Nikolić, and R. Ernst, "Challenges and Limitations of IEEE 802.1CB-2017," *IEEE Embedded Systems Letters*, vol. 12, pp. 105–108, Dec. 2020. https://doi.org/10.1109/LES.2019.2960744.

[27] P. Heise, F. Geyer, and R. Obermaisser, "TSimNet: An Industrial Time Sensitive Networking Simulation Framework Based on OMNeT++," in *2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–5, Nov. 2016.

[28] E. Mohammadpour and J.-Y. Le Boudec, "On Packet Reordering in Time-Sensitive Networks," *IEEE/ACM Transactions on Networking*, pp. 1–13, 2021.

[29] B. Varga, J. Farkas, S. Kehrer, and T. Heer, "Deterministic Networking (DetNet): Packet Ordering Function," Internet Draft draft-varga-detnet-pof-02, Internet Engineering Task Force, Oct. 2021. https://datatracker.ietf.org/doc/draft-varga-detnet-pof-02.

[30] E. Mohammadpour, E. Stai, and J. L. Boudec, "Improved Delay Bound for a Service Curve Element with Known Transmission Rate," *IEEE Networking Letters*, pp. 1–1, 2019. http://doi.org/10.1109/LNET.2019.2927143.

[31] 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.

[32] A. Mifdaoui and T. Leydier, "Beyond the Accuracy-Complexity Tradeoffs of Compositional Analyses using Network Calculus for Complex Networks," in *10th International Workshop on Compositional Theory and Technology for Real-Time Embedded Systems (Co-Located with RTSS 2017)*, (Paris, France), pp. pp. 1–8, Dec. 2017. https://hal.archives-ouvertes.fr/hal-01690096.

[33] A. Morton, G. Ramachandran, S. Shalunov, L. Ciavattone, and J. Perser, "Packet reordering metrics." RFC 4737, Nov. 2006.

[34] J. Liebeherr, *Duality of the Max-Plus and Min-Plus Network Calculus*. now, 2017. https://ieeexplore.ieee.org/document/8187214.

[35] N. Navet, H. H. Bengtsson, and J. Migge, "Early-stage Bottleneck Identification and Removal in TSN Networks," Feb. 2020. https://orbilu.uni.lu/handle/10993/46282.

[36] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks - Amendment 34:Asynchronous Traffic Shaping," *IEEE Std 802.1Qcr-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, IEEE Std 802.1Qcy-2019, and IEEE Std 802.1Qcx-2020)*, pp. 1–151, Nov. 2020. https://doi.org/10.1109/IEEESTD.2020.9253013.