# Embedding network calculus and event stream theory in a common model

Marc Boyer, Pierre Roux

ONERA – The french aerospace lab
F 31055, Toulouse, France

***Abstract*—Network calculus (also known as real-time calculus) and event stream theory are two theories designed to compute upper bounds on response time for real-time systems. Both generalise the common periodic request arrival to any kind of workload using cumulative functions. In network calculus, a data flow is modelled by its cumulative curve, $A(t)$, representing the amount of data sent by the flow up to time $t$, whereas the event stream theory counts the number of packets sent up to $t$, denoted by $E(t)$. Of course, the size of packets is a link between both functions. This work presents a formal model embedding both $A$, $E$, the packet function, and their basic relations.**

## I. Introduction

The correct behaviour of cyber-physical systems requires some real-time control capabilities, *i.e.* the guarantee that some response to some stimulus can be computed and sent to some actuator within a given latency budget. The engineers need some method to compute the response time of the computing platform (involving calculators, busses, gateways, etc.) when designing the system and allocating resources. Several methods target this goal, and among others, event stream theory and network calculus are compositional methods, *i.e.* methods where each component is stimulated by request flows (task releases, message arrivals...), uses local resources, and where an analytic method computes the flow output and bounds on latency and memory usage.

In event stream theory [1], a flow is modelled by a function $E : \mathbb{R}^+ \to \mathbb{N}$ where $E(t)$ represents the accumulated number of task activations up to time $t$, whereas in network calculus [2], a function $A : \mathbb{R}^+ \to \mathbb{R}^+$ represents the accumulated amount of data received up to time $t$. Both CPU and network can be generalised as resources, where $E(t)$ represents the number of requests (task activation, packets...), and $A(t)$ the amount of workload (CPU cycles, number of bits...).

The response time of a component mostly depends on its workload: number of CPU cycles to execute, number of bits to send, etc. From this assessment, one may conclude that an amount-based model is the most adequate one. But in a distributed system, where a component produces some output that activates some work on a second component, two situations occur: in network elements (switches, routers), the workload is directly related to the data size, whereas in computing elements, the reception of a message is an event that will release a task, whose workload is often independent of the message size. Then, analysing a distributed system requires to handle both event arrivals and data/workload amount.

978-1-5090-1314-2/16/$31.00 

In most studies, only one function, $E$ or $A$ is modelled, and the other one is implicit and deduced from the first one using some model assumption on packets size or task execution time. For example, considering a flow of packets of size $p$, the number of packets received up to time $t$ can be deduced from $A$, setting $E(t) = \left\lfloor \frac{A(t)}{p} \right\rfloor$. Conversely, if an event sequence $E$ generates, for each event, a packet of fixed size $p$, the amount of data generated is simply $A(t) = pE(t)$. This network-based example can be translated into CPU scheduling, where $E$ counts the number of jobs and $A$ the accumulated CPU workload. In case of variable packet size, one may consider a minimal and maximal size, $\underline{p}, \overline{p}$, and consider lower and upper bounds: $\underline{p}E(t) \leq A(d) \leq \overline{p}\overline{E}(t)$.

In our experience, difficulties arise when considering flows with correlated packets sizes: either because the flow itself has different sizes (like the I,B,P frames of MPEG video encoding) or, more often, because a flow is the aggregation of sub-flows with different packet sizes.

A few works tried to generalise the minimal/maximal size approach, introducing a function capturing this packet size correlation, using either an explicit data amount and an implicit event count [3], [4] or an explicit event count and an implicit workload amount [5], [6].

This paper presents a model embedding explicitly both aspects, in order to, first, have a better understanding of each theory, and second, to be able to analyse a system with both event stream and network calculus, enhancing at each step the result of one theory with the one of the other.

This paper does not study how an input flow is transformed into an output flow: network calculus and/or event stream theory are designed for the analysis of each component. This work targets the interfaces of the components.

The expected benefits of the work are: first, a better understanding of each theory, and a way for experts of one to get knowledge on the other; second, since each theory has its own method to compute, on each component, the delay and input/output curve transformation, the model allows to analyse each component with both theories, to keep the best of each result, and to re-inject the results on the flow model from each one to the other; last, the ability to model systems that are coarsely modelled up to now.

The outline of the paper is as follows: first, Section II

gives an overview of the related work. Then, Section III introduces some basic mathematical tools required to support this framework. The model itself is presented and discussed in Section IV. Section V shows how informations on two functions can be combined to enhance our knowledge on the third one. Last, Section VI shows how to model two important elements: the packetizer and the aggregator.

## II. Related work

The event stream theory, with the pioneer work of Gresser [7], has been developed to allow a compositional analysis of real-time systems where several real-time components are hosting tasks, sharing resources, communicating through data flows. As said in [1], "Compared to the holistic approach, the compositional models are modularly structured with respect to the architecture". This work has been extended by Richter et al. [8], [9]. At this time, the goal was to rely, for each component, on the analyses of response time already existing and to use the event stream model to compose them[1]. But this first goal has been extended, and several specific analyses have been developed for specific cases [10], [11], [12]. A global overview is presented in [1] whereas the complete formal model can be found in [13].

In event stream theory, tasks are released by either some internal timing decision (period, offset, etc.) or by reception of an external packet or signal. Both notions are called events. An *event stream* $E(t)$ counts the accumulated number of events received up to time $t$, and two functions, called *event models*, denoted $\eta^+(\Delta)$ and $\eta^-(\Delta)$, are respectively an upper and lower bounds on the number of events received during any interval of length $\Delta$.

The network calculus theory has been developed independently, from the seminal paper on network calculus [14], using the term "a calculus for network delay", to compute upper bounds on network latency. In network calculus, the amount of traffic received up to time $t$, $A(t)$, is upper bounded by an *arrival curve* $\alpha$ [14], [15]. But network calculus theory does not rely on existing response time analyses, and develops its own analyses methods. A real breakthrough, from our point of view, is the introduction of the min-plus convolution, independently done in [16], [17] and [18].

The real-time calculus [19], [20] is an adaptation of network calculus to real-time time systems. Despite some changes in notations, the formal models are, in fact, equivalent [21]. The introduction of the lower bound on arrival was done in the real-time calculus community [22], as well as the main related results [19].

The event stream theory and network/real-time calculus are somehow different in the algorithms used to compute bounds, but they are very close on the way to represent requests: as see in introduction, in event stream theory, the flows are modelled by the number of events, $E(t)$, whereas in network/real-time calculus, the flows are modelled by $A(t)$ the amount of the data or workload[2].

Both notions are equivalent when considering packets of constant size, but in real systems, different flows may have different packet sizes, and even in the same flow, the size may vary from one packet to another. If only upper and lower bound are known, simple approximations are sufficient, but the sequence of sizes may have some regularity (also known as correlation in [6]), requiring a more general model.

So, to link both notions, the sizes of a sequence of packets/requests must be captured. This has been done independently in [5], [6] and in [4], [3].

In [4], [3] a function $P$ is introduced such that $P(a)$ represents the number of packets received for the amount of data $a$, *i.e.* $P(A(t))$ is the number of packets received up to time $t$, and two functions $\underline{\pi}, \overline{\pi}$ are used to capture the minimal an maximal values of the variation of $P$ on some interval. In this work, $A$ is explicit and $E$ implicit.

Conversely, in [5], [6], two functions $\gamma^l, \gamma^u$ are used to bound the workload generated by a sequence of events: $E$ is explicit and $A$ is implicit.

This paper is a direct continuation of [4], [3]: considering that $E(t)$ is also the number of packets received up to $t$, the relation $P \circ A = E$ is explicitly set and the mathematical implications of this modelling are studied[3].

The results on aggregation of flows or packetizer are the same in [4], [3] and in this paper.

Last, this paper only deals with data flow models, and does not consider how an input flow is transformed in an output flow by a component. Such results on blind components are given in [4], [3]. And in [6], a workload-based service curve is derived from the event-based service curve, and both are used in each component.

## III. Mathematical background

After introducing some notations and definitions related to functions, section III-A presents the notion of interval bounding pair (that generalises the notion of arrival curve and event model) and section III-B presents the notion of pseudo-inverse.

The set of natural numbers is $\mathbb{N}$. The set of reals is $\mathbb{R}$ and the subset of non-negative reals is $\mathbb{R}^+$. Let us denote $\vee, \wedge$ the infix maximum and minimum operators ($a \vee b = \max(a, b)$, $a \wedge b = \min(a, b)$), and $\circ$ the composition operator $(f \circ g)(x) = f(g(x))$.

In the remainder of the paper, $\mathbb{D}$ and $\mathbb{I}$ will each denote either $\overline{\mathbb{N}} \stackrel{def}{=} \mathbb{N} \cup +\{\infty\}$ or $\overline{\mathbb{R}}^+ \stackrel{def}{=} \mathbb{R}^+ \cup \{+\infty\}$.

**Definition 1** (Non-decreasing function)**.** *A function $f : \mathbb{D} \to \mathbb{I}$ is called* non-decreasing *when for all $x, y \in \mathbb{D}$, if $x < y$, then $f(x) \leq f(y)$.*

*A non-decreasing function $f$ is* divergent *if* $\lim_{x \to \infty} f(x) = \infty$.

[1] "We don't necessarily need to develop new local analysis techniques if we can benefit from the host of work in real-time scheduling analysis." [8].

[2] The arrival curve is even defined as a generalisation of event stream model in [19].

[3] The Section IV-C will justify why we chose $P$ and not $\gamma$.

**Definition 2** (2-surjective function)**.** *A function $f : \mathbb{D} \to \mathbb{I}$ is said* 2-surjective *if*

$$\forall y \in \mathbb{I}, \exists x, x' \in \mathbb{D}, x \neq x', f(x) = f(x') = y. \tag{1}$$

**Proposition 1.** *A non-decreasing and 2-surjective function $f : \mathbb{D} \to \mathbb{I}$ satisfies $f(0) = 0$.*

### A. Min/max-plus and interval bounding pairs

**Definition 3** (Convolutions)**.** *Let $f, g : \overline{\mathbb{R}}^+ \to \mathbb{I}$ two functions. The min-plus, and max-plus convolutions are respectively defined as*

$$(f \underline{*} g)(t) \stackrel{def}{=} \inf\left\{ f(u) + g(v) \;\middle|\; u, v \in \overline{\mathbb{R}}^+, u + v = t \right\},$$
$$(f \overline{*} g)(t) \stackrel{def}{=} \sup\left\{ f(u) + g(v) \;\middle|\; u, v \in \overline{\mathbb{R}}^+, u + v = t \right\}.$$

*The respective Kleene-star closures, also known as sub-additive and sup-additive closure, are defined by*

$$f^{\underline{*}} \stackrel{def}{=} \inf\{e, f, f \underline{*} f, f \underline{*} f \underline{*} f, \ldots\},$$
$$f^{\overline{*}} \stackrel{def}{=} \sup\{-e, f, f \overline{*} f, f \overline{*} f \overline{*} f, \ldots\},$$

*with $e$ the neutral element of the min-plus convolution, $e(0) = 0$, $e(x) = \infty$ otherwise.*

In [2] is given a comprehensive list of properties of these operators (associativity, commutativity, isotonicity, etc.)

The next definition generalises the notions of arrival curves of network calculus and event model of event stream theory.

**Definition 4** (Interval bounding pair (IBP))**.** *Let $f : \overline{\mathbb{R}}^+ \to \mathbb{I}$ and $\underline{\phi}, \overline{\phi} : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$ be three functions, $\underline{\phi}$ and $\overline{\phi}$ being non-decreasing. Then $\underline{\phi}$ and $\overline{\phi}$ are respectively a lower and upper interval bounding function of $f$ if*

$$\forall t, d \in \mathbb{R}^+, \underline{\phi}(d) \leq f(t + d) - f(t) \leq \overline{\phi}(d). \tag{2}$$

*The pair $(\underline{\phi}, \overline{\phi})$ is often denoted $\phi$. This pair is called* interval bounding pair.

*Conversely, the set of functions respecting the constraint $\phi$ is denoted*

$$\mathcal{F}(\phi) \stackrel{def}{=} \left\{ f \;\middle|\; \forall t, d \in \mathbb{R}^+, \underline{\phi}(d) \leq f(t + d) - f(t) \leq \overline{\phi}(d) \right\}.$$

The condition (2) is equivalent with $f \overline{*} \underline{\phi} \leq f \leq f \underline{*} \overline{\phi}$ [2].

**Theorem 1** (Tightening interval bounding pair)**.** *Let $\underline{\phi}, \overline{\phi}, \underline{\phi}', \overline{\phi}' : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$ be two pairs of functions. Then*

$$\mathcal{F}\left(\underline{\phi}, \overline{\phi}\right) = \mathcal{F}\left(\left(\underline{\phi}\right)^{\overline{*}}, \left(\overline{\phi}\right)^{\underline{*}}\right)$$
$$\mathcal{F}(\underline{\phi}, \overline{\phi}) \cap \mathcal{F}(\underline{\phi}', \overline{\phi}') = \mathcal{F}(\underline{\phi} \vee \underline{\phi}', \overline{\phi} \wedge \overline{\phi}')$$

The proof can be found in [2]. A better result, $\mathcal{F}(\underline{\phi}, \overline{\phi}) = \mathcal{F}(\underline{\phi} \underline{\oslash} \overline{\phi}, \overline{\phi} \overline{\oslash} \underline{\phi})$ involving the min-deconvolution, $\underline{\oslash}$ and the max-deconvolution, $\overline{\oslash}$ can be found in [23], [24], but these operators are not presented in this article, by lack of space.

### B. Pseudo inverse

The core of the model presented in this paper is the function $P$ that associates to an amount of data the related count of events. But the inverse relation is also of interest. And since the function $P$ is non-decreasing, the inverse functions $P^{-1}$ may not exist. More generally, event stream theory and network calculus handle non-decreasing functions. But even if the inverse does not exist, two pseudo-inverses can be defined, and some of their properties are presented. This works partially generalises the one presented in [25] or in [2, § 3.1.4]. The intuition is quite simple, and is illustrated in Figure 1. The proofs can be found in [26].

**Definition 5** (Pseudo inverse)**.** *Let $f : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$ be a non-decreasing function. Then, $f^{\underline{-1}}$ and $f^{\overline{-1}} : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$ are defined for all $y \in \overline{\mathbb{R}}^+$ as*

$$f^{\underline{-1}}(y) = \inf\left\{ x \in \overline{\mathbb{R}}^+ \;\middle|\; f(x) \geq y \right\}$$
$$f^{\overline{-1}}(y) = \sup\left\{ x \in \overline{\mathbb{R}}^+ \;\middle|\; f(x) \leq y \right\}$$

*with the convention that $\sup \emptyset = 0$ and $\inf \emptyset = \infty$.*

**Proposition 2** (Pseudo inverses are non-decreasing)**.** *For any non-decreasing function $f : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$, the functions $f^{\underline{-1}}$ and $f^{\overline{-1}}$ are non-decreasing.*

**Proposition 3** (Continuity of pseudo inverses)**.** *For any non-decreasing function $f : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$, $f^{\underline{-1}}$ is left-continuous and $f^{\overline{-1}}$ is right-continuous.*

**Proposition 4** (Pseudo inverses and composition)**.** *For any non-decreasing functions $f, g : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$*

$$f^{\underline{-1}} \circ f(x) \leq x \leq f^{\overline{-1}} \circ f(x) \tag{3}$$

*Moreover, if $f$ is right-continuous*

$$f \circ f^{\underline{-1}} \circ f = f \tag{4}$$

*and if it is left-continuous*

$$f \circ f^{\overline{-1}} \circ f = f. \tag{5}$$

**Proposition 5** (Pseudo inverses and IBP)**.** *For any non-decreasing function $f : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+$, if $\left(\underline{\phi}, \overline{\phi}\right)$ is an IBP for $f$, then $\left(\overline{\phi}^{\overline{-1}}, \underline{\phi}^{\underline{-1}}\right)$ is an IBP for both $f^{\underline{-1}}$ and $f^{\overline{-1}}$. That is, for all $y, \delta \in \overline{\mathbb{R}}^+$*

$$\overline{\phi}^{\overline{-1}}(\delta) \leq f^{\underline{-1}}(y + \delta) - f^{\underline{-1}}(y) \leq \underline{\phi}^{\underline{-1}}(\delta) \tag{6}$$
$$\overline{\phi}^{\overline{-1}}(\delta) \leq f^{\overline{-1}}(y + \delta) - f^{\overline{-1}}(y) \leq \underline{\phi}^{\underline{-1}}(\delta). \tag{7}$$

*Moreover, for all $y, \delta \in \overline{\mathbb{R}}^+$*

$$\overline{\phi}^{\underline{-1}}(\delta) \leq f^{\overline{-1}}(y + \delta) - f^{\underline{-1}}(y) \leq \underline{\phi}^{\overline{-1}}(\delta). \tag{8}$$

## IV. The linking model

This section presents the data flow model, made of three curves, $A, E, P$. The definition, presented in Section IV-A, will be discussed just after.

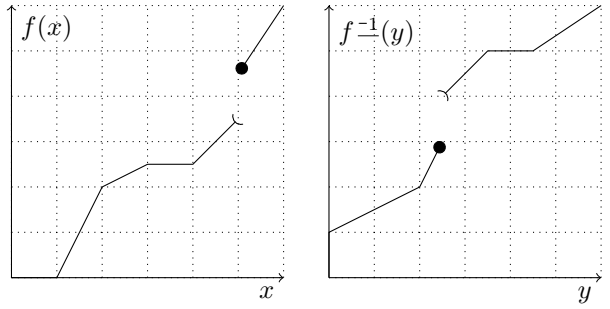


Fig. 1. Illustration of pseudo-inverse.

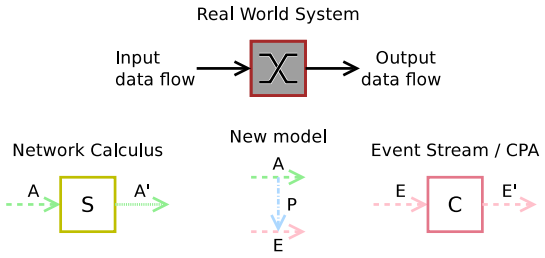


Fig. 2. Model relations

### A. A mathematical definition

Before giving a formal definition of the model, a data-flow based interpretation is proposed.

In this model, a data flow is modelled by three curves, $A$ such that $A(t)$ represents the cumulative amount of data/bits sent by the flow up to time $t$, $E$ such that $E(t)$ represents the cumulative number of (full) packets sent by the flow up to time $t$, and $P$ such that $P(a)$ is the number of (full) packets in the $a$ first bits of the flow. These three functions are linked by the relation $P \circ A = E$.

While considering the workload generated by a task, $E(t)$ represents the number of task instances (often called "jobs") released up to time $t$, $A(t)$ represents the total workload associated to these jobs, and $P(a)$ the number of task instances associated with the global workload $a$.

On may wonder why not to derive $A$ from $E$ and individual packets sizes or task instance workload, that uses a more intuitive semantics than the one of $P$. This will be discussed in Sections IV-C and IV-D.

*Naming:* The $A$ function represents the *amount* of data. In previous network calculus publications, the $A$ was used for *arrival*. The $P$ is of course for *packet* count. The $E$ comes from the *event stream* theory.

**Definition 6** (Flow tuple)**.** *A flow is modelled by a tuple* $\langle A, E, P\rangle$ *such that*

$$A : \overline{\mathbb{R}}^+ \to \overline{\mathbb{R}}^+, \quad E : \overline{\mathbb{R}}^+ \to \mathbb{N}, \quad P : \overline{\mathbb{R}}^+ \to \mathbb{N}, \quad P \circ A = E.$$

*The three functions,* $A, E, P$ *are non-decreasing, piecewise continuous and* $A(0) = 0$*. Moreover, the* $P$ *function is 2-surjective.*

**Corollary 1.** *For any flow* $\langle A, E, P\rangle$*,* $P(0) = E(0) = 0$*.*

*Proof:* By Proposition 1, $P(0) = 0$, and by definition $E(0) = P(A(0)) = 0$. ∎

### B. On function domain

We chose a dense time domain, whereas some other real-time models assume a discrete time domain, $\mathbb{N}$. Since a computer-based system is driven by a discrete clock, a discrete time domain is a sound assumption. Nevertheless, while considering a distributed system, made of different sub-systems, each sub-system may have its own clock, and hence its own clock drift. There is then no guarantee that a common discrete clock exists. This justifies our choice of a dense time domain.

We also set $\overline{\mathbb{R}}^+$ as the domain of $P$. When a flow sends an infinite amount of data, *i.e.* $\lim_{t\to\infty} A(t) = \infty$, it is an obvious requirement, but event with a flow sending a finite amount of data, *i.e.* $\lim_{t\to\infty} A(t) = M \in \overline{\mathbb{R}}^+$, the model requires that $P$ is defined even for values on $[M, \infty)$. The theoretical model may have been defined to weaken this requirement, but it would imply a more complex definition. Nevertheless, the Definition 9 will later have to restrict some condition on $P$ on its relevant prefix. This is a trade-off between including the specific case of finite flow in the flow definition or further.

### C. On information redundancy

It is clear that the use of an explicit function $E$ creates information redundancy, but one of the purposes of this model is to build an explicit link between network calculus and event stream theories. Then, existing results in one theory can be transferred to the other. Moreover, when dealing with contracts, three interval bounding pairs capture more information than only two, as will be shown in Section V.

Note also that even if, obviously, $E$ can be deduced from $A$ and $P$, the converse is not true. First, $P$ can not be deduced from $A$ and $E$, in general: if several packets are received at the same time (when it exists $t$ such that $E(t+) - E(t-) \geq 2$), there is no way to know the individual size of each packet, it exists an infinite number of $P$ satisfying the relation. Second, neither can $A$ be deduced from $E$ and $P$: $E$ represents the instant of the end of packet arrival, and $P$ the size of individual packets, but the information of bit arrival rate "inside" each packet is lost.

### D. On the packet function and individual packet sizes

We have justified, while giving the definition of a flow, why it uses the $P$ function, whose semantics is not obvious at first glance. Here is presented the equivalence between this notion and the individual packet sizes.

Assume that the data flow is a sequence of packets, the $i$-th packet having length[4] $l_i$. Let $L : \mathbb{N} \to \overline{\mathbb{R}}^+$ defined as $L(n) = \sum_{i=1}^{n} l_i$. This cumulative packet size function is equivalent to the list of individual packets sizes, since $l_i = L(i) - L(i-1)$, but easier to handle in our context.

[4]One often uses the term "size" of packet, but the letter "s" is commonly used to denote servers.

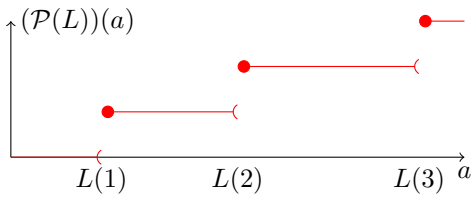


Fig. 3. The packet function built from cumulative length.

**Definition 7** (Cumulative packet length)**.** *A function* $L : \mathbb{N} \to \overline{\mathbb{R}}^+$ *is a* cumulative packet length function *if*

1) *it is increasing* ($n < n' \implies L(n) < L(n')$)*,*
2) *it is null at 0* ($L(0) = 0$)*,*
3) *it is divergent* ($\lim_{n\to\infty} L(n) = \infty$)*.*

**Definition 8** (From packet function to cumulative packet length and back)**.** *Let* $\mathcal{P}$ *the operator which maps each cumulative packet length function* $L$ *to the non-decreasing piecewise continuous 2-surjective function*

$$(\mathcal{P}(L))(a) \stackrel{def}{=} \max\{n \in \mathbb{N} \mid L(n) \leq a\}. \tag{9}$$

*Let* $\mathcal{L}$ *the operator which maps each non-decreasing piecewise continuous 2-surjective function to the cumulative packet length function*

$$(\mathcal{L}(P))(n) \stackrel{def}{=} \inf\left\{d \in \overline{\mathbb{R}}^+ \;\middle|\; P(d) \geq n\right\}. \tag{10}$$

Note that eq. 10 is equivalent to $\mathcal{L}(P) = P^{\overline{-1}}$.

Note that we could have defined $(\mathcal{P}(L))(a)$ as $(\min\{n \mid L(n) \geq a\} - 1) \vee 0$, which is left-continuous, whereas the definition used is right-continuous.

We have to prove that the definition is well formed, that is to say that $\mathcal{P}(L)$ is a non-decreasing piecewise continuous 2-surjective function, and that $\mathcal{L}(P)$ is a cumulative packet length function. To do so, the following Lemma gives the main arguments. The proof itself can be found in [26].

**Lemma 1** (Equivalent definitions of $\mathcal{P}$)**.** *Let* $L$ *be a* cumulative packet length function*, and* $n \in \mathbb{N}$*, then:*

$$\forall a \in [L(n), L(n+1)) : (\mathcal{P}(L))(a) = n \tag{11}$$

$$(\mathcal{P}(L))(a) = \sup\left\{x \in \overline{\mathbb{R}}^+ \;\middle|\; L(\lceil x \rceil) \leq a\right\} \tag{12}$$

The Figure 3 illustrates the eq. (11). The eq. 12 is useful since it allows to define the operator $\mathcal{P}$ as a pseudo-inverse: $\mathcal{P}(L) = (L \circ \lceil\cdot\rceil)^{\overline{-1}}$.

*Proof:* Let $L$ be a cumulative packet length function, and $P = \mathcal{P}(L)$.

For the eq. (11), if $a = L(n)$, $n \in \{n' \mid L(n') \leq L(n)\}$, and since $L(n)$ is increasing, given $m > n$, $L(m) > L(n)$ and $m \notin \{n' \mid L(n') \leq L(n)\}$.

Otherwise, if $a \in (L(n), L(n+1))$ and to ease readability, set $\mathcal{N} = \{n \in \mathbb{N} \mid L(n) \leq a\}$.

Then, $L(n) \leq a$, so $n \in \mathcal{N}$, and $n \leq \max\mathcal{N} = P(a)$.

Consider $m = n+1$: $a < L(m)$, so $m \notin \mathcal{N}$, so $m > \max\mathcal{N}$ *i.e.* $n + 1 > P(a)$.

Combining $n + 1 > P(a) \geq n$ leads to $P(a) = n$.

For eq (12), pick some $a$, keep the same definition of $\mathcal{N}$ and set $\mathcal{X} = \left\{x \in \overline{\mathbb{R}}^+ \;\middle|\; L(\lceil x \rceil) \leq a\right\}$. Given $n \in \mathcal{N}$, $\lceil n \rceil = n$ so $n \in \mathcal{X}$, *i.e.* $\mathcal{N} \subseteq \mathcal{X}$ and $\max\mathcal{N} \leq \sup\mathcal{X}$. Conversely, given $x \in \mathcal{X}$, it exists $n \in \mathcal{N}$ (consider $n = \lceil x \rceil$) such that $x \leq n$. Then, $\sup\mathcal{X} \leq \max\mathcal{N}$. ∎

**Theorem 2** (Equivalence between cumulative packet length and packet functions)**.** *For any cumulative packet length function* $L$*,* $(\mathcal{L} \circ \mathcal{P})(L) = L$*.*

This theorem states that cumulative packet length functions and the non-decreasing piecewise continuous 2-surjective functions are equivalent, that is to say, in our context, that modelling the packet sizes either by a $P$ or an $L$ function is equivalent. The proof can be found in [26].

## V. Flow contracts

The real behaviour of a data flow, or request sequence, is in general unknown, or too complex to be handled. Then, some contracts, or patterns, on the behaviour are used, involving often some period, jitter, minimal or maximal size, etc. Such constraints are called "arrival curves" in network calculus, or "event model" in event stream theory. Here is defined a notion of *contract* that embraces the previous notions.

**Definition 9** (Interval bounding tuple of flow)**.** *Let* $\langle A, E, P\rangle$ *be a flow, and* $\alpha, \eta, \pi$ *three IBP. Then, the flow* $\langle A, E, P\rangle$ respects the contract $\langle \underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi}\rangle$ *if the pairs* $(\underline{\alpha}, \overline{\alpha})$*,* $(\underline{\eta}, \overline{\eta})$ *are respectively interval bounding pairs for functions* $A$ *and* $E$*, and*

$$\begin{aligned}\forall a, d \in \overline{\mathbb{R}}^+, a + d \leq \lim_{x\to\infty} A(x) :\\ \underline{\pi}(d) \leq P(a+d) - P(d) \leq \overline{\pi}(d)\end{aligned} \tag{13}$$

*Conversely, the (possibly empty) set of flow satisfying theses constraints is denoted* $\mathcal{F}(\alpha, \eta, \pi)$*.*

The semantics of $\alpha$ and $\eta$ are quite intuitive: they are lower and upper bounds on the amount of data or event received on some interval. The semantics of $\pi$ deserves a discussion: $\pi$ could be counter-intuitive since a flow with large packets will have a smaller $\pi$ function than a flow with small packets. The mathematical reason is that $P$ is the pseudo-inverse of the packet length function $L$, as presented in Section IV-D. The semantics of $\pi$ could be the following: imagine that there exists an "end of packet" delimiter in the data flow just after each packet. Then, given any slice of a trace of length $a$, $\underline{\pi}(a)$ and $\overline{\pi}(a)$ are respectively lower and upper bounds on the number of "end of packet" delimiters than can be found in the slice.

Once defined this contract notion, this section will show how to automatically tighten these contracts using algebraic properties, and using information on some functions to get a better characterisation of the other ones.

First, the equations (14) and (15) allow to tighten a contract, pair per pair. Second, the Theorems 4, 5 and 6 allow to transfer information from two contracts on the third one. Combining all these results with eq. (16) enables to build a sequence of contracts, converging to a fixpoint.

**Theorem 3** (Tightening contracts)**.** *Let $\alpha, \alpha', \eta, \eta', \phi, \phi'$ three interval bounding pairs functions. Then*

$$\mathcal{F}(\underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi}) = \mathcal{F}(\underline{\alpha}^{\overline{*}}, \overline{\alpha}^{*}, \underline{\eta}^{\overline{*}}, \overline{\eta}^{*}, \underline{\pi}^{\overline{*}}, \overline{\pi}^{*}) \tag{14}$$

$$\mathcal{F}(\underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi}) = \mathcal{F}(\underline{\alpha}, \overline{\alpha}, \lceil \underline{\eta} \rceil, \lfloor \overline{\eta} \rfloor, \lceil \underline{\pi} \rceil, \lfloor \overline{\pi} \rfloor) \tag{15}$$

$$\begin{aligned}\mathcal{F}(\underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi}) \cap \mathcal{F}(\underline{\alpha}', \overline{\alpha}', \underline{\eta}', \overline{\eta}', \underline{\pi}', \overline{\pi}') = \\ \mathcal{F}(\underline{\alpha} \vee \underline{\alpha}', \overline{\alpha} \wedge \overline{\alpha}', \underline{\eta} \vee \underline{\eta}', \overline{\eta} \wedge \eta', \underline{\pi} \vee \underline{\pi}', \overline{\pi} \wedge \overline{\pi}')\end{aligned} \tag{16}$$

The eq. (15) allows to restrict values of bounds on $P$ and $E$ to integer values. We may have enforced it by setting $\mathbb{N}$ as the image of these functions in the definitions, but when implementing tools, it may be easier to handle functions with real values[5]. Then, the formal model allows to use $\overline{\mathbb{R}}^{+}$ as image and also to restrict it to $\mathbb{N}$.

*Proof:* The eq. (14) and (16) are a direct application of Theorem 1.

Consider first the restriction to $\lceil \underline{\eta} \rceil, \lfloor \overline{\eta} \rfloor$. Let $\langle A, E, P \rangle \in \mathcal{F}(\underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi})$ and $t, d \in \overline{\mathbb{R}}^{+}$. Since $E(t), E(t+d) \in \mathbb{N}$, and $E$ is non-decreasing, then $E(t+d) - E(t) \in \mathbb{N}$. So, $\underline{\eta}(d) \leq E(t+d) - E(t) \leq \overline{\eta}(d)$ implies $\lceil \underline{\eta} \rceil (d) \leq E(t+d) - E(t) \leq \lfloor \overline{\eta} \rfloor (d)$. Conversely, since $\underline{\eta}(d) \leq \lceil \underline{\eta} \rceil (d)$ and $\overline{\eta}(d) \geq \lfloor \overline{\eta} \rfloor (d)$, $\lceil \underline{\eta} \rceil (d) \leq E(t+d) - E(t) \leq \lfloor \overline{\eta} \rfloor (d) \implies \underline{\eta}(d) \leq E(t+d) - E(t) \leq \overline{\eta}(d)$. Then $\mathcal{F}(\underline{\alpha}, \overline{\alpha}, \underline{\eta}, \overline{\eta}, \underline{\pi}, \overline{\pi}) \subseteq \mathcal{F}(\underline{\alpha}, \overline{\alpha}, \lceil \underline{\eta} \rceil, \lfloor \overline{\eta} \rfloor, \underline{\pi}, \overline{\pi})$. The restriction on $\lceil \underline{\pi} \rceil, \lfloor \overline{\pi} \rfloor$ is done the same way. ∎

**Theorem 4** (Bounding event count from data and packet bounds)**.** *Let $\alpha, \pi$ two IBP. Then*

$$\mathcal{F}(\alpha, \cdot, \pi) \subseteq \mathcal{F}\left(\alpha, (\underline{\pi} \circ \underline{\alpha}, \overline{\pi} \circ \overline{\alpha}), \pi\right). \tag{17}$$

This theorem allows, for each flow $\langle A, E, P \rangle$, to compute an IBP for $E$ from the ones for $A$ and $P$.

*Example of event count with linear arrival:* Consider a flow made of packets of maximal size 3 and minimal size $\frac{1}{2}$. Then, $\underline{\pi}(a) = \left\lceil \frac{a}{3} \right\rceil$ and $\overline{\pi}(a) = \lfloor 2a \rfloor$. Consider that the flow throughput is at most 2 and at least $\frac{1}{5}$, leading to $\underline{\alpha}(t) = \frac{t}{5}$ and $\overline{\alpha}(t) = 2t$. Then, the Theorem 4 states that $\lfloor 4t \rfloor$ is an upper bound on the number of event: since the maximal rate is 2 per time unit, the flow can send one packet of size $\frac{1}{2}$ every $\frac{1}{4}$ time unit. Conversely, $\left\lceil \frac{t}{15} \right\rceil$ is a lower bound: it may take 15 time units to send a packet of size 3 with a throughput of $\frac{1}{5}$.

*Proof:* Let $\langle A, E, P \rangle \in \mathcal{F}(\alpha, \cdot, \pi)$ a flow. Given $t, \Delta \in \overline{\mathbb{R}}^{+}$, by definition of a flow,

$$E(t+\Delta) - E(t) = P(A(t+\Delta)) - P(A(t)).$$

By definition of $\underline{\alpha}$ and $\overline{\alpha}$

$$\begin{aligned}&\underline{\alpha}(\Delta) \leq A(t+\Delta) - A(t) \leq \overline{\alpha}(\Delta) \\ \iff &\underline{\alpha}(\Delta) + A(t) \leq A(t+\Delta) \leq \overline{\alpha}(\Delta) + A(t)\end{aligned}$$

[5]For example, consider two period flow, one sending one packet every 5 time unit, and the other every 7 time unit. Then, $\overline{\eta}_1 = \left\lceil \frac{\cdot}{5} \right\rceil$ and $\overline{\eta}_1 = \left\lceil \frac{\cdot}{7} \right\rceil$ are two upper bounds on the event number with values in $\mathbb{N}$ and $\overline{\eta}'_1(t) = \frac{t}{5} + 1, \overline{\eta}'_2(t) = \frac{t}{7} + 1$ are two others, wider, with values in $\overline{\mathbb{R}}^{+}$. But the sum of the two last ones is easier to represent in computer than the sum of the two first ones.

then, since $P$ is a non-decreasing function

$$P(\underline{\alpha}(\Delta) + A(t)) \leq P(A(t+\Delta)) \leq P(\overline{\alpha}(\Delta) + A(t)).$$

Now, add $-P(A(t))$ to each term

$$\begin{cases} P(\underline{\alpha}(\Delta) + A(t)) - P(A(t)) \leq P(A(t+\Delta)) - P(A(t)), \\ P(A(t+\Delta)) - P(A(t)) \leq P(\overline{\alpha}(\Delta) + A(t))) - P(A(t)). \end{cases}$$

Consider now the upper bound, by definition of $\overline{\pi}$

$$P(\overline{\alpha}(\Delta) + A(t))) - P(A(t)) \leq \overline{\pi}(\overline{\alpha}(\Delta)).$$

The same way, for the lower bound:

$$P(\underline{\alpha}(\Delta) + A(t))) - P(A(t)) \geq \underline{\pi}(\underline{\alpha}(\Delta)).$$

Then, $E \in \mathcal{F}(\overline{\pi} \circ \overline{\alpha}, \underline{\pi} \circ \underline{\alpha})$, and $\langle A, E, P \rangle \in \mathcal{F}\left(\alpha, (\underline{\pi} \circ \underline{\alpha}, \overline{\pi} \circ \overline{\alpha}), \pi\right)$. ∎

**Theorem 5** (Bounding data amount from event and packet bounds)**.** *Let $\eta, \pi$ two interval bounding pairs, then*

$$\mathcal{F}(\cdot, \eta, \pi) \subseteq \mathcal{F}\left(\left(\overline{\pi}^{\underline{-1}} \circ \underline{\eta}, \underline{\pi}^{\overline{-1}} \circ \overline{\eta}\right), \eta, \pi\right). \tag{18}$$

*Proof:* Let $\langle A, E, P \rangle \in \mathcal{F}(\alpha, \cdot, \pi)$ a flow. From relation $P \circ A = E$, it comes $P^{\underline{-1}}(E) \leq A \leq P^{\overline{-1}}(E)$. Then, given $t, \Delta \in \overline{\mathbb{R}}^{+}$:

$$A(t+\Delta) - A(t) \leq P^{\overline{-1}}(E(t+\Delta)) - P^{\underline{-1}}(E(t)) \tag{19}$$

by definition of $\overline{\eta}$, and monotony of $P^{\overline{-1}}$

$$\leq P^{\overline{-1}}(E(t) + \overline{\eta}(\Delta)) - P^{\underline{-1}}(E(t)) \tag{20}$$

and from Prop. 5, eq. (8),

$$\leq \underline{\pi}^{\overline{-1}}(\overline{\eta}(\Delta)) \tag{21}$$

The proof for the lower bound is done the same way. ∎

*Example of arrival curve with unit model:* Consider a flow sending one packet of size one every time instant. That is to say $E(t) = \lfloor t \rfloor$ and $P(d) = \lfloor d \rfloor$. It admits IBP $\overline{\eta} = \overline{\pi} = \lceil \cdot \rceil$, and $\underline{\eta} = \underline{\pi} = \lfloor \cdot \rfloor$. What can be the arrival curve of such a flow? One may expect to get no more than one unit of data per unit of time ($A = \lceil \cdot \rceil$), but it is not necessarily the case.

Applying Thm. 5 leads to the arrival curve $\overline{\alpha} = \underline{\pi}^{\overline{-1}} \circ \overline{\eta} = \lceil \cdot \rceil + 1$, enabling a burst of one packets in null time and two packets on any interval of arbitrary small length $\epsilon > 0$.

Using sub-additive closure, one may enhance slightly the result, and consider $(\overline{\alpha})^{*} = (\lceil \cdot \rceil + 1) \wedge \delta_0$, but it still allows the arrival of two packets in any interval of arbitrary small length. This is somehow counter intuitive.

But it may happen, up to some $\epsilon$. Keep in mind that $E$ is increased when a packet is fully received. One can build a flow $A$ that send one "full" packet of size 1 each time unit but that can send $2-\epsilon$ data on some interval of width $\epsilon$: see the $A$ drawn in Figure 4 with some linear slopes to connect the points $A(0) = 0$, $A(1-\epsilon) = \epsilon$, $A(1+\epsilon) = 2-\epsilon$, $A(3-\epsilon) = 2+\epsilon$, etc.

**Theorem 6** (Bounding packet sizes from event and data bounds)**.** *Let $\alpha, \eta$ two interval bounding pairs, then*

$$\mathcal{F}(\alpha, \eta, \cdot) \subseteq \mathcal{F}\left(\alpha, \eta, \left(\underline{\eta_l} \circ \overline{\alpha}^{\underline{-1}}, \overline{\eta_r} \circ \underline{\alpha}^{\overline{-1}}\right)\right). \tag{22}$$

The proof can be found in [26].

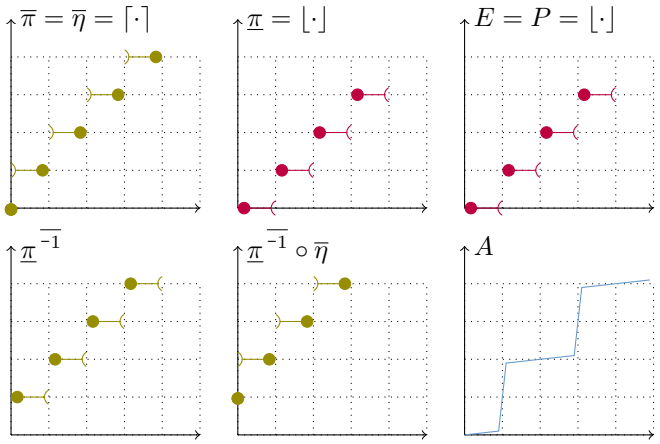


Fig. 4. Example of periodic constant size packet.

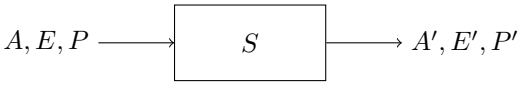


Fig. 5. Packetizer.

## VI. Application: Modeling Packetizers and Aggregation

Once defined the model and its main intrinsic properties, this section presents two basic applications, the modelling of a packetizer, and its interval bounding tuple, and the same for aggregation of two flows.

### A. Packetizer

A packetizer, as in Figure 5, takes as input a flow and outputs another flow with the same packets but in which data are released only when a full input packet is received. An example of input and output is displayed in Figure 6.

**Definition 10.** *A packetizer maps any input flow $\langle A, E, P\rangle$ such that $P$ is right-continuous to an output flow $\langle A', E', P'\rangle$*

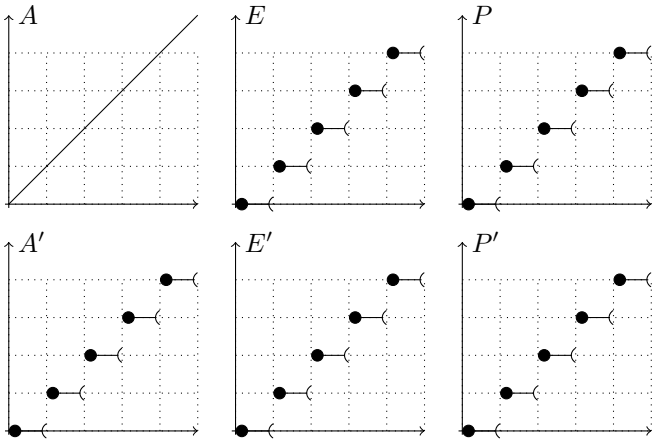


Fig. 6. Example of input $\langle A, E, P\rangle$ and output $\langle A', E', P'\rangle$ for a packetizer.

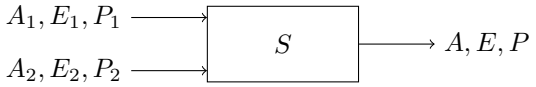


Fig. 7. Aggregation.

*such that*

$$A' = P^{\underline{-1}} \circ P \circ A, \qquad E' = E, \qquad P' = P.$$

*Proof:* We have to prove that $\langle A', E', P'\rangle$ defined as above is a flow. In particular, we need to prove that $P'(A') = E'$, that is $P \circ P^{\underline{-1}} \circ P \circ A = P \circ A$. This holds according to Proposition 4 since $P$ is right-continuous. ∎

**Remark 1.** *If $P$ is not right-continuous, the above property may not hold. For instance, for $P = \lceil \cdot \rceil$, we have $P(1.5) = 2$ whereas $\left(P \circ P^{\underline{-1}} \circ P\right)(1.5) = \left(P \circ P^{\underline{-1}}\right)(2) = P(1) = 1$.*

**Proposition 6** (Contract at packetizer output)**.** *Given a flow $\langle A, E, P\rangle \in \mathcal{F}(\alpha, \eta, \pi)$ entering a packetizer, and let $\langle A', E', P'\rangle$ be the output, then*

$$\langle A', E', P'\rangle \in \mathcal{F}\left(\left(\overline{\pi}^{\overline{-1}} \circ \underline{\eta}, \underline{\pi}^{\underline{-1}} \circ \overline{\eta}\right), \eta, \pi\right).$$

*Proof:* For any $t, \delta \in \overline{\mathbb{R}}^+$, $A'(t+\delta) - A'(t) = P^{\underline{-1}} \circ E(t+\delta) - P^{\underline{-1}} \circ E(t)$ and since $E(t+\delta) - E(t) \le \overline{\eta}(\delta)$ and $P^{\underline{-1}}$ is non-decreasing, we then have

$$\begin{aligned} A'(t+\delta) - A'(t) &\le P^{\underline{-1}}(E(t) + \overline{\eta}(\delta)) - P^{\underline{-1}}(E(t)) \\ &\le \underline{\pi}^{\underline{-1}}(\overline{\eta}(\delta)) \end{aligned}$$

by Prop. 5, eq (6). The proof for the lower bound is similar. ∎

### B. Aggregation

Aggregation takes two flows $A_1, E_1, P_1$ and $A_2, E_2, P_2$ as input and immediately puts their packets in a common output flow $A, E, P$. This is pictured in Figure 7. In network calculus, as in the event stream theory, the aggregation of two flows is defined as the sum of the modelling functions, that is $A = A_1 + A_2$ and $E = E_1 + E_2$.

The definition of $P$ is less obvious. Of course, for $A, E, P$ to be a flow according to Definition 6, $P$ must satisfy $P \circ A = E$, that is $P(A_1 + A_2) = E_1 + E_2 = P_1(A_1) + P_2(A_2)$. However, this does not precisely define $P$, particularly when $A_1$ and $A_2$ present discontinuities, which happens for instance when they are taken as output of packetizers. As we would like to keep the fact that the packets in the ouput flow are just an interleaving of the one in the two input flows, we additionally require that $L := \mathcal{L}(P)$ is an interleaving of $L_1 := \mathcal{L}(P_1)$ and $L_2 := \mathcal{L}(P_2)$. We thus end up with the following definition.

**Definition 11** (Aggregation)**.** *Aggregation maps any two input flows $(A_1, E_1, P_1)$ and $(A_2, E_2, P_2)$ to an output flow $(A, E, P)$ such that*

$$A = A_1 + A_2,\ E = E_1 + E_2,\ P(A_1 + A_2) = P_1(A_1) + P_2(A_2)$$

*and there exists* $\varphi_1, \varphi_2 : \mathbb{N} \to \mathbb{N}$ *such that for all* $n \in \mathbb{N}$, $0 \leq \varphi_1(n+1) - \varphi_1(n) \leq 1$, $\varphi_2(n) = n - \varphi_1(n)$ *and*

$$P = \mathcal{P}\left(\mathcal{L}\left(P_1\right)\left(\varphi_1(n)\right) + \mathcal{L}\left(P_2\right)\left(\varphi_2(n)\right)\right).$$

Intuitively, $\varphi_1(n)$ (resp. $\varphi_2(n)$) denotes the number of packets taken from the first (resp. second) flow among the first $n$ packets of the output flow.

**Proposition 7** (Contract on aggregation output)**.** *Given two flows* $\langle A_1, E_1, P_1\rangle \in \mathcal{F}\left(\alpha_1, \eta_1, \pi_1\right)$ *and* $\langle A_2, E_2, P_2\rangle \in \mathcal{F}\left(\alpha_2, \eta_2, \pi_2\right)$*, and let* $\langle A, E, P\rangle$ *be one aggregation, then*

$$\langle A, E, P\rangle \in \mathcal{F}\left(\alpha_1 + \alpha_2, \eta_1 + \eta_2, \left(\left\lfloor \underline{\pi_1} \mathbin{\underline{*}} \underline{\pi_2} \right\rfloor, \left\lceil \overline{\pi_1} \mathbin{\overline{*}} \overline{\pi_2} \right\rceil\right)\right).$$

A proof can be found in [26]

## VII. Conclusion

As presented in the introduction, we were looking for a model that embeds the flow models of network calculus, based on data amount, and event stream theory, based on event count.

The first step was to define a suitable model: such a model has been presented with the justifications of the choices (Section IV). Moreover, the "robustness" of the model has been tested, on the intrinsic properties of the model (Section V), and on its ability to formally model the packetization and the aggregation flows (Section VI). The results involve compositions, convolutions, and have required some work on pseudo-inverses of functions. Such operations have been implemented for the ultimately pseudo-periodic class of functions [27], except the composition, which is a on-going work.

We now plan to test the accuracy of this model on some realistic examples, and especially some where the resource consumption is a function of both the workload and the number of requests: for example, some switch where the memory allocation is done by blocks of size $b$, *i.e.* the memory used to store a packet of size $s$ is $b\left\lceil \frac{s}{b} \right\rceil$. An accurate bound requires both packet sizes and the number of packets. Another example is a gateway that performs some data format conversion: in this case, the task workload is a constant, plus a part proportional to the packet size. Moreover, such a gateway often extracts some payload, removing some header, and forwards the payload with a new header. This model is also designed to model such header change.

## Thanks

We would like to thank Anne Bouillard, for the pioneer work done in [4], [3], and some informal discussions on the subject, and Sophie Quinton, for her great interest on this thematic that gave us the energy to formalise it.

This work as been partially funded by French agency DGA, under project IREHDO 2.

## References

[1] J. Rox and R. Ernst, "Compositional performance analysis with improved analysis techniques for obtaining viable end-to-end latencies in distributed embedded systems," *International Journal on Software Tools for Technology Transfer*, vol. 15, no. 3, pp. 171–187, 2013.

[2] J.-Y. Le Boudec and P. Thiran, *Network Calculus*, ser. LNCS. Springer Verlag, 2001, vol. 2050, http://lrcwww.epfl.ch/PS_files/NetCal.htm.

[3] A. Bouillard, N. Farhi, and B. Gaujal, "Packetization and aggregate scheduling," INRIA, Tech. Rep. 7685, 2011.

[4] ——, "Packetization and packet curves in network calculus," in *Proc. of the 6th Int. Conf. on Performance Evaluation Methodologies and Tools (ValueTools 2012)*, France, Oct. 2012, invited Paper.

[5] E. Wandeler, A. Maxiaguine, and L. Thiele, "Quantitative characterization of event streams in analysis of hard real-time applications," *Real-Time Systems*, vol. 29, no. 2-3, pp. 205–225, 2005.

[6] E. Wandeler and L. Thiele, "Characterizing workload correlations in multi processor hard real-time systems." in *Proc. of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2005)*, 2005, pp. 46–55.

[7] K. Gresser, "An event model for deadline verification of hard real-time systems," in *Proc. of the Fifth Euromicro Workshop on Real-Time Systems.*, 1993, pp. 118–123.

[8] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis - the SymTA/S approach," *IEEE Proc. on Computers and Digital Techniques*, vol. 152, march 2005.

[9] K. Richter, "Compositional scheduling analysis using standard event model – the SymTA/S approach," Ph.D. dissertation, Technical university Carolo-Wilhelmina of Braunschweig, 2005.

[10] J. Rox and R. Ernst, "Exploiting inter-event stream correlations between output event streams of non-preemptively scheduled tasks," in *Proc. of Int. Conf. on the Design, Automation and Test in Europe (DATE 2010)*, march 2010, pp. 226 –231.

[11] ——, "Formal timing analysis of full duplex switched based ethernet network architectures," in *Proc. of the SAE 2010 AeroTech Congress and Exhibition*. SAE International, 2010.

[12] P. Axer, D. Thiele, R. Ernst, and J. Diemer, "Exploiting shaper context to improve performance bounds of Ethernet AVB networks," in *Proc. of the 51st Annual Design Automation Conf. (DAC'14)*. ACM, 2014.

[13] S. Schliecke, "Performance analysis of multiprocessor real-time systems with shared resources," Ph.D. dissertation, TU Braunschweig, 2011.

[14] R. L. Cruz, "A calculus for network delay, part I: Network elements in isolation," *IEEE Transactions on information theory*, vol. 37, no. 1, pp. 114–131, January 1991.

[15] C.-S. Chang, "Stability, queue length, and delay of deterministic and stochastic queueing networks," *IEEE Transactions on Automatic Control*, vol. 39, no. 5, pp. 913–931, May 1994.

[16] R. Cruz and C. Okino, "Service guarantees for window flow control," in *Proc. of the annual Allerton Conf. on communication control and computing*, vol. 34, 1996, pp. 10–21.

[17] J.-y. Le Boudec, "Network calculus made easy," Ecole Polytechnique Fdrale de Lausanne (EPFL), Tech. Rep. EPFL-DI 96/218, Dec. 1996.

[18] C.-S. Chang, "A filtering theory for deterministic traffic regulation," in *INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution., Proceedings IEEE*, vol. 2, Apr 1997, pp. 436–443 vol.2.

[19] E. Wandeler, "Modular performance analysis and interface-based design for embedded real-time systems," Ph.D. dissertation, ETH Zurich, September 2006.

[20] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems." in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2000, pp. 101–104.

[21] A. Bouillard, L. Jouhet, and E. Thierry, "Service curves in Network Calculus: dos and don'ts," INRIA, Research Report RR-7094, 2009.

[22] S. Chakraborty, S. Knzli, L. Thiele, A. Herkersdorf, and P. Sagmeister, "Performance evaluation of network processor architectures: Combining simulation with analytical estimation," *Computer Networks*, vol. 41, no. 5, pp. 641—-665, 2003.

[23] M. Moy and K. Altisen, "Arrival curves for real-time calculus: The causality problem and its solutions," pp. 358–372, 2010, 10.1007/978-3-642-12002-2_31.

[24] ——, "Arrival curves for real-time calculus: the causality problem and its solutions," Verimag, Tech. Rep. TR-2009-15, 2009.

[25] P. Embrechts and M. Hofert, "A note on generalized inverses," *Mathematical Methods of Operations Research*, vol. 77, no. 3, 2013.

[26] M. Boyer and P. Roux, "A common framework embedding network calculus and event stream theory," May 2016, working paper. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01311502

[27] A. Bouillard and E. Thierry, "An algorithmic toolbox for network calculus," INRIA, Rapport de recherche 6094, January 2007.