Some Properties Of Variable Length Packet Shapers

Jean-Yves Le Boudec

To Appear in 2002 in ACM/IEEE Transactions on Networking

Abstract— **The min-plus theory of greedy shapers has been developed after Cruz's results on the calculus of network delays. An example of greedy shaper is the buffered leaky bucket controller. The theory of greedy shapers establishes a number of properties such as the series decomposition of shapers or the conservation of arrival constraints by re-shaping. It applies in all rigor either to fluid systems, or to packets of constant size such as ATM. For variable length packets, the distortion introduced by packetization affects the theory, which is no longer valid. In this paper, we elucidate the relationship between shaping and packetization effects. We show a central result, namely, the min-plus representation of a packetized greedy shaper. We find a sufficient condition under which series decomposition of shapers and conservation of arrival constraints still hold in presence of packetization effects. This allows us to demonstrate the equivalence of implementing a buffered leaky bucket controller based on either virtual finish times or on bucket replenishment. However, we show on some examples that if the condition is not satisfied, then the property may not hold any more. This indicates that, for variable size packets, unlike for fluid systems, there is a fundamental difference between constraints based on leaky buckets, and constraints based on general arrival curves, such as spacing constraints. The latter are used in the context of ATM to obtain tight end-to-end delay bounds. In this paper, we use a min-plus theory, and obtain results on greedy shapers for variable length packets which are not readily explained with the max-plus theory of Chang.**

Keywords— **Network Calculus, Shaper, Min-Plus Algebra, Leaky Bucket**

I. Introduction

We consider some of the problems caused by traffic regulation for flows of variable length packets. While the original work by Cruz in [1] defines a leaky bucket regulator as a system handling variable length packets, the theory of regulators (which we now call "greedy shapers") that was later developed by Agrawal, Chang, Cruz, Le Boudec, Okino and Rajan [2], [3], [4], [5] either focused explicitly on flows of constant size packets (namely ATM), or applies only to fluid systems. We say that we have "bit-by-bit" greedy shapers. The theory of bit-by-bit greedy shapers is extremely powerful (in its context); it allows to establish a number of invariance or optimality properties such as the series decomposition of shapers or re-shaping keeps original arrival constraints (see Section II-A for more details). These properties have been used for example for designing schedulers [6]. The theory applies to leaky buckets, but also to other types of constraints, for example, spacing conditions used in ATM networks (see Section II-B).

For variable length packets, the distortion introduced by packetization affects the theory, which is no longer valid. We use the concept of packetizer, introduced in [7], [8], [9], which models the effect of variable length packets. In many cases, packetizers weaken the bounds obtained with a fluid model by one packet size of maximum size, however, as recalled in Section II-D, in some cases this distortion may be accumulated at every node. Shapers are important in the context of integrated as well as differentiated services; they are known under terms such as token bucket, or leaky bucket controllers. This motivates us to understand the effect of variable packet sizes on the properties of shapers, which is the objective of this paper.

Chang introduces in [10] an alternative theory for variable length packet, based on max-plus representations. As shown in Section II-E, the max-plus theory applies to slightly different shaping systems, since the concept of arrival constraints under this theory does not exactly match the usual practice for leaky bucket controllers. In addition, we find fundamental results that go beyond the results available in [10].

We take a theoretical look at the issue, and obtain both fundamental and practical results. Firstly, we introduce the concept of packetized greedy shaper and obtain their input-output representation (Theorem III.1). We find a condition under which this representation can be considerably simplified (Theorem III.3). These two findings constitute our main theoretical contribution. Secondly (Section IV-B), we find a sufficient condition under which re-shaping a flow of variable length packets does keep original arrival constraints. We use this to show that, if the condition is satisfied, the series decomposition of shapers still holds for variable size packets. However, we show on some examples that if the condition is not satisfied, then the property may not hold any more. Thirdly, we consider two commonly used alternative implementations of leaky bucket controllers, which we call the "virtual finish time" and the "bucket replenishment" implementations. We show that both implementations produce the same packet output (Corollary IV.1).

The paper is organized as follows. Section II reviews

Author Affiliations: Jean-Yves LeBoudec <jean-yves.leboudec@epfl.ch> EPFL-IC/ISC/LCA, CH-1015 Lausanne, Switzerland

the state of the art on greedy shapers, recalls the concept of packetizers, as introduced in [10], and gives our notation, together with a few minor new results on packetizers. We also describe constraints that are not based on leaky buckets but are used in practice. Section III is the main theory. First, we introduce the concept of packetized greedy shaper and obtain its input-output characterization. Then we find a condition under which the concatenation of a bit-by-bit greedy shaper and a packetizer keeps arrival constraints, which allows us to derive a number of practical results. Section IV derives some applications to leaky buckets; we examine whether a packetized greedy shaper keeps arrival constraints, and whether a series decomposition result holds. The proofs of all theorems are put in appendix, except for the proofs of Theorem IV.1 and Theorem IV.2, which are short and have some interest of their own.

II. State of the art

A. Regulator, Buffered Leaky Bucket Controller and Bit-By-Bit Greedy Shaper

We say that a flow is σ-smooth, or has σ as arrival curve, for some function $\sigma(t)$, if the number of bits observed on the flow during a time interval of duration t is $\leq \sigma(t)$. The IETF uses a generic family of arrival curves of the form $\sigma(t) = \min(M + pt, b + rt)$ where M is interpreted as a maximum packet size, p a peak rate, b a burst tolerance and r a sustainable rate [11]. Similarly, a "buffered leaky bucket controller" is a system which forces a flow to be σ-smooth, with $\sigma(t) = \min_{m=1,\ldots,M}(r_m t + b_m)$, while delaying the packets as little as possible. This is traditionally interpreted by saying that the controller observes a set of M fluid buckets, where the mth bucket is of size b_m and leaks at a constant rate r_m. Every bucket receives l_i units of fluid when packet i is released (l_i is the size of packet i). A packet is released as soon as the level of fluid in bucket m allows it, namely, has gone down below $b_m - l_i$, for all m. We will use a variant of this definition later in this paper; for clarity, we say that we have defined now a buffered leaky bucket controller based on "bucket replenishment".

A variant of the buffered leaky bucket controller was studied in a continuous time setting by Cruz in [1] under the name of "(b, r) regulator". The regulator is associated with a bucket of fluid which leaks at a constant rate r; a packet is released as soon as the level of fluid in the bucket does not exceed b. The output of the regulator has a constant rate C, which corresponds to a physical line rate. The output of such a regulator is σ-smooth, with

$$\sigma(t) = rt + b + \left(1 - \frac{r}{C}\right) l_{\max} \tag{1}$$

where $l_{\max}$ is the maximum packet size. This variant differs from the standard definition by the term $\frac{1-r}{C} l_{\max}$ and by the fact that it explicitly accounts for a line rate C at the output. As we will see in detail in Section II-E, the set of outputs of such regulators is not identical with the set of packet flows that are σ-smooth, with σ given by Equation (1). Thus this form of regulator is not exactly a buffered leaky bucket controller.

In [2], Cruz refined the concept in discrete time with constant size packets. It was found in particular that regulators can be combined to synthesize systems that force a flow to be σ-smooth where σ is any concave and piecewise linear wide-sense increasing[1] function. The optimality of regulators was established, namely, a regulator in discrete time forces its output to be σ-smooth, and does it as early as possible. This was used to show that regulators enjoy some remarkable properties, for example, if a regulated flow is passed through a second regulator, then the final output keeps the arrival curve constraint imposed by the first regulator. In the discrete time setting, the regulator is exactly a buffered leaky bucket controller.

The concept was further generalized independently in [3], [4], [5] under various names. In the context of this paper, we call them "bit-by-bit" greedy shapers. Given some wide-sense increasing function σ, a greedy shaper takes some flow as input and forces its output to be σ-smooth; it delays the input data in a buffer, whenever sending data would violate the constraint σ, but releases them as soon as possible. This generalizes the concept of buffered leaky bucket controller, this latter system corresponding to a function σ which is piecewise linear and concave. Section II-B discusses some cases that are used in practice and cannot be expressed with leaky buckets (because σ is not concave).

The theory shows that, without loss of generality, we can assume that $\sigma(0) = 0$ and that σ is sub-additive, namely, $\sigma(s + t) \leq \sigma(s) + \sigma(t)$. Wide-sense increasing functions that are "star-shaped" (namely, $\sigma(t)/t$ is wide-sense increasing) are sub-additive [10]. Concave, wide-sense increasing functions σ such that $\sigma(0) = 0$ are star-shaped, thus sub-additive, but there are useful sub-additive functions that are not concave and not even star-shaped (see Section II-B). The cornerstone result in the theory of bit-by-bit greedy shapers is the input/output characterization of greedy shapers. Call $R(t)$ the cumulative input function (namely, the number of bits observed in time interval $[0, t]$) and $R^*(t)$ the output of the greedy shaper; we have

$$R^*(t) = \inf_{0 \leq s \leq t} (\sigma(s) + R(t - s)) = (\sigma \otimes R)(t) \tag{2}$$

The right-handside in the equation is called the min-plus convolution of σ and R. The associativity and commutativity of min-plus convolution can then be used to derive a number of "physical" properties of shapers [3], [4], [5], two of which we recall now.

[1] We say that function $f(\cdot)$ is wide-sense increasing if $s \leq t$ always implies $f(s) \leq f(t)$.

1. *Reshaping keeps arrival constraints:* Consider a flow $R(t)$, which is originally α-smooth, and is input to a shaper with shaping curve σ. The output is $R^* = R \otimes \sigma$. Since the input is α-smooth, it is not modified by a shaper with shaping curve α, thus $R = R \otimes \alpha$. Thus $R^* = (R \otimes \alpha) \otimes \sigma$; by associativity and commutativity of $\otimes$, it follows that $R^* = (R \otimes \sigma) \otimes \alpha = R^* \otimes \alpha$ and thus R^* is α-smooth. We will see that this result does not generally hold in presence of packetization effects.
2. *Series decomposition of shapers:* Consider a tandem of shapers numbered $1, ..., m, ..., M$; shaper m has a shaping curve σ_m, feeds shaper $m+1$, and is fed by the output of shaper $m-1$. Call R the input; it follows immediately from Equation (2) that the final output is $R^* = R \otimes (\sigma_1 \otimes ... \otimes \sigma_M)$. Thus the tandem of shapers is equivalent to a shaper with curve $\sigma = \sigma_1 \otimes ... \otimes \sigma_M$. If σ_m is star-shaped for all m and $\sigma_m(0) = 0$ then $\sigma = \min(\sigma_1, ..., \sigma_M)$. This establishes that a buffered leaky bucket controller with shaping curve $\sigma(t) = \min_{m=1,...,M}(r_m t + b_m)$ can be implemented as a series of single leaky bucket controllers. Note that the series decomposition can be made in any order. We will show that this results still holds (for leaky buckets, not in general) in presence of packetization effects.

The theory in [2], [3] is for discrete time systems with constant packet sizes, and thus applies without restriction to ATM systems. In contrast, and unlike the original results in [1], [12], the theory of greedy shapers in [4], [5] applies to continuous time and flows with variable packet size, but does not account for packetization effects. In this context, the greedy shaper characterized by Equation (2) outputs a continuous stream of bits, not entire packets; see for example Figure 1. This is why we call it a "bit-by-bit greedy shaper". In some cases, it is a good model: for example, a constant bit rate trunk with rate C is modeled with a bit-by-bit greedy shaper with $\sigma(t) = Ct$ (for $t \geq 0$). In general, though, regulators used in various packet scheduling methods output entire packets and cannot strictly be modeled with a bit-by-bit greedy shaper, but can be captured by the concept of "packetizer", recalled in Section II-D.

B. Non concave arrival curves

Much attention has been given to concave arrival curves, because they naturally appear with leaky buckets. However, non concave arrival curves are sometimes used in the context of ATM or of switch dimensioning. As an example, we use in this paper the "stair function" v_T, defined by

$$v_T(t) = \begin{cases} \lceil \frac{t}{T} \rceil \text{ if } t > 0 \\ 0 \text{ if } t \leq 0 \end{cases}$$

It can easily be seen that, for any T and k there is equivalence between saying that a flow $R(t)$ is kv_T-smooth and saying that, for any time $t \geq 0$:

$$R(t+T) - R(t) \leq k \tag{3}$$

For packets of fixed size equal to k data units, this is equivalent to a spacing condition [13]. Figure 2 shows an example of shaper with shaping curve of the form $\sigma = kv_T$.

Consider an ATM switch receiving n connections, each of them perfectly shaped with peak rate T.[2]The aggregate flow is nv_T-smooth; characterizing this aggregate flow with a concave arrival curve would lead to looser dimensioning bounds. This is used for example in [14], [13] to obtain tight end-to-end delay jitter bounds for an ATM network which are not obtainable if leaky bucket characterizations would be used instead.

C. Distortions dues to packetization

In this section we illustrate the fact, mentioned earlier, that the theory of bit-by bit greedy shapers in [3], [4], [5] does not apply to systems that have the constraint of releasing entire packets.

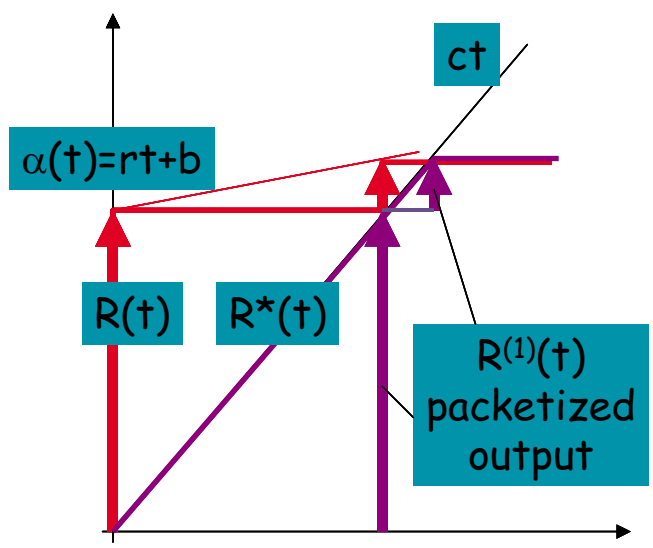

Fig. 1. A constant rate server, followed by a packetizer, does not keep arrival constraints. The input $R(t)$ is constrained to be σ-smooth. The output of the bit-by-bit constant rate server, $R^*(t)$, is α-smooth, as predicted by the theory of bit-by-bit shapers. The final, packetized output $R^{(1)}(t)$ is *not* α-smooth.

A first example is pointed out by Pla and Guérin in [15]. Consider a flow, which is known to be α-smooth, for some function α, fed into a server of constant rate c. For a fluid or ATM system, we know from "re-shaping keep arrival constraints" that the output must also be α-smooth. Now if we assume that the output is in packet form, and that packet sizes are not all identical, then this is no longer true. The details of an example, taken from [15], are as follows (see Figure 1) . The input flow $R(t)$ sends a first packet of size $l_1 = l_{\max}$ at time $T_1 = 0$, and a second packet of size l_2 at

[2]A more involved example would use $v_{T,\tau}(t) = \lceil \frac{t+\tau}{T} \rceil$ for $t > 0$ in order to account for cell delay variation tolerance

time $T_2 = \frac{l_2}{r}$. Assume that $\alpha(t) = l_{\max} + rt$ with $r < C$, thus the flow R is indeed α-smooth. The departure time for the first packet is $T_1' = \frac{l_{\max}}{C}$. Assume that the second packet l_2 is small, namely $l_2 < \frac{r}{C} l_{\max}$; then the two packets are sent back-to-back and thus the departure time for the second packet is $T_2' = T_1' + \frac{l_2}{C}$. Now the spacing $T_2' - T_1'$ is less than $\frac{l_2}{r}$, thus the second packet is not conformant, in other words, $R^{(1)}$ is not α-smooth. Note that this example is not possible if all packets have the same size.

A second (original) example shows how distortion may occur even if all packets have the same size. In Figure 2, we see that the arrival curve constraint imposed by a bit-by-bit greedy shaper can be lost after packetization.

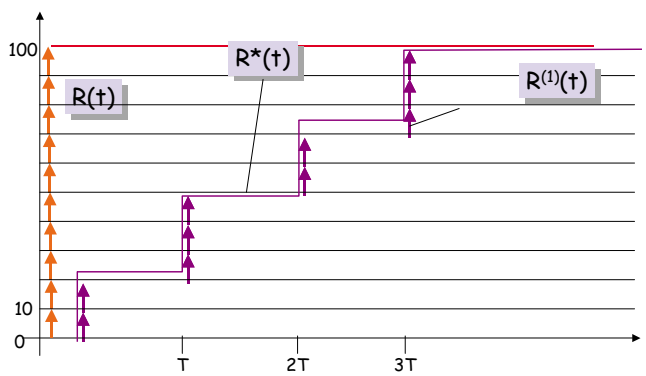

Fig. 2. Packetization may undo shaping. The input $R(t)$ is made of a burst of 10 packets of size 10 data units each. It is fed to a bit-by bit shaper with shaping curve $\sigma = 25v_T$. The output of the bit-by-bit shaper is $R^*(t)$. It is made of 25 data units every T time units. The final output $R^{(1)}(t)$ results from packetization. It is made of a burst of 2 packets at time 0, followed by a burst of 3 packets at time T, etc. It is not σ-smooth because of the bursts of 3 packets, which corresponds to 30 data units, whereas σ-smoothness requires that no more than 25 data units are sent every T time units.

These two examples illustrate a relationship between packetization and shaping. We will establish general results in Section III-B.

D. The packetizer

Our analysis uses the concept of packetizer, [7], [8], [9], [10] as a means to model packetization effect. We say that a sequence $L = (L(0) = 0, L(1), L(2), ...)$ is a "sequence of cumulative packet lengths" if it is wide sense increasing and

$$l_{\max} := \sup_n \{L(n+1) - L(n)\}$$

is finite. We interpret $L(n) - L(n-1)$ as the length of the nth packet.

Now for any sequence of cumulative packet lengths L we define function P^L by

$$P^L(x) = \sup_{n \in \mathbb{N}} \{L(n) 1_{\{L(n) \leq x\}}\} \tag{4}$$

Intuitively, $P^L(x)$ is the largest cumulative packet length which is entirely contained in x. Function P^L is right-continuous; if R is right-continuous, then so is $P^L(R(t))$. For example, if all packets have unit length, then $L(n) = n$ and for $x > 0$: $P^L(x) = \lfloor x \rfloor$. An equivalent characterization of P^L is

$$P^L(x) = L(n) \Longleftrightarrow L(n) \leq x < L(n+1) \tag{5}$$

Note also that

$$x - l_{\max} < P^L(x) \leq x \tag{6}$$

We also say that a flow $R(t)$ is L-packetized if $P^L(R(t)) = R(t)$ for all t.

An "L-packetizer" [7], [8], [9], [10] is defined as the system which transforms an input $R(t)$ into $P^L(R(t))$. Figure 1 and Figure 2 both include a packetizer in tandem with a bit-by-bit greedy shaper.

Some bounds for the effect of a packetizer can easily be derived, based on the maximum packet size can easily be derived. The following items come mostly from [10]. Consider a system (*bit-by-bit system*) with L-packetized input R and bit-by-bit output R^*, which is then L-packetized to produce a final packetized output $R^{(1)}$. We call *combined system* the system which maps R into $R^{(1)}$. Assume both systems are first-in-first-out and lossless.

1. The *per-packet delay* for the combined system is defined as $\sup_i (T_i' - T_i)$, where T_i, T_i' are the arrival and departure time for the ith packet. It is identical to the maximum virtual delay[3] for the bit-by-bit system, without packetizer. In other words, for computing packet delay, we may ignore the packetizer.
2. Call B^* the maximum backlog for the bit-by-bit system and B' the maximum backlog for the combined system. We have $B^* \leq B' \leq B^* + l_{\max}$.
3. Assume that the bit-by-bit system offers to the flow a maximum service curve γ and a minimum service curve β. The combined system offers to the flow a maximum service curve γ and a minimum service curve β' given by $\beta'(t) = [\beta(t) - l_{\max}]^+$. Thus packetizing weakens the service curve guarantee by one maximum packet length. For example, if a system offers a rate-latency service curve with rate R, then appending a packetizer to the system has the effect of increasing the latency by $\frac{l_{\max}}{R}$. The rate-latency service curve with rate R and latency T is defined by $S(t) = R(t - T)^+$. It is commonly used to model a generic scheduler.

[3]The virtual delay at time t is the time it takes to output all bits present at time t. If the system is FIFO, it is the delay incurred by a hypothetical bit that would arrive at time t

4. If some flow $S(t)$ has $\sigma(t)$ as arrival curve, then $P^L(S(t))$ has $\sigma(t) + l_{\max} 1_{\{t>0\}}$ as arrival curve. For Figure 1 this tells us that the final output $R^{(1)}$ has $\sigma'(t) = \sigma(t) + l_{\max} 1_{\{t>0\}}$ as arrival curve, which is consistent with our observation that $R^{(1)}$ is not σ-smooth, even though R^* is. We will see in Section III that there is a stronger result, in relation with the concept of "packetized greedy shaper".

At this point, the alert reader may ask herself why it is worth studying the distortion introduced by packetization in the theory of shapers, since its effect may seem to be limited to one packet. However, firstly, shaping variable size packets does occur with differentiated and integrated services at network ingress, and it is important to understand it from a theoretical viewpoint. Secondly, we show now that distortions due to packetization may accumulate over network paths and lead to important discrepancies.

Item 1 says that appending a packetizer to a node does not increase the packet delay at this node. However, packetization does increase the end-to-end delay. Consider the concatenation of the theoretical GPS node, with guaranteed rate R [16] and an L-packetizer (Figure 3). Assume this system receives a flow of variable length packets. This models a theoretical node which would work as a GPS node but is constrained to deliver entire packets. This is not very realistic, and we will see later in this section how to deal with a realistic example, but it is sufficient to show one important effect of packetizers.

In the fluid model, the GPS node offers a rate latency service curve with some rate R and 0 latency (in the simplest case in [16]). By application of item 3, we find that the combined node offers a rate-latency service curve with rate R and latency $T = \frac{l_{\max}}{R}$. Now concatenate m such identical nodes, as illustrated on Figure 3. By application of standard results, the end-to-end service curve is the rate latency-function with rate R and latency $T = m\frac{l_{\max}}{R}$. However, for the computation of the end-to-end delay bound, we need to take into account item 1, which tells us that we can forget the last packetizer. Thus, a bound on end-to-end delay is obtained by considering that the end-to-end path offers a service curve equal to the latency-function with rate R and latency $T_0 = (m-1)\frac{l_{\max}}{R}$. For example, if the original input flow is constrained by one leaky bucket of rate r and bucket pool of size b, then an end-to-end delay bound is

$$\frac{b + (m-1)l_{\max}}{R} \tag{7}$$

There is a straightforward generalization of the above reasoning to the family of "guaranteed rate" (GR) nodes [17], which contains PGPS or SCFQ schedulers, etc. and their combination. A GR node is characterized by a rate R and a latency v which captures the scheduler lateness, compared to GPS; for example, for PGPS, $v = \frac{M}{C}$ where M is the maximum packet size across all flows and C is the total server rate. It is shown in [18], Theorem 2.1.1, that a GR node is the concatenation of a node offering the rate-latency service curve with rate R and latency v, followed by a packetizer. Using items 1 and 3 above, we find that the concatenation of m GR nodes guarantees a delay bound of $\frac{b+(m-1)l_{\max}}{R} + mv$ (with the same notation as for Equation (7))[4].

We see on this example that the additional latency introduced by one packetizer is indeed of the order of one packet length; however, this effect is multiplied by the number of hops, roughly speaking.

E. The max-plus theory of shapers

A dual approach to account for variable length packets is introduced in [10]. It consists in replacing the definition of σ-smoothness, mentioned above, by the concept of g-regularity. Consider a flow of variable length packets, with cumulative packet length L and call T_i the arrival epoch for the ith packet. The flow is said to be g-regular if $T_j - T_i \geq g(L(j) - L(i))$ for all packet numbers $i \leq j$. A theory is then developed with concepts similar to the greedy shaper. The theory uses max-plus convolution instead of min-plus convolution. The (b, r) regulator of Cruz is a shaper in this theory, whose output is g-regular, with $g(x) = \frac{(x-b)}{r}^+$.

Note that this theory does not exactly correspond to the usual concept of leaky bucket controllers. More specifically, there is not an exact correspondence between the set of flows that are g-regular on one hand, and that are σ-smooth on the other. We explain why on an example. Consider the set of flows that are g-regular, with $g(x) = \frac{x}{r}$. The minimum arrival curve we can put on this set of flows is $\sigma(t) = rt + l_{\max}$ [10]. But conversely, if a flow is σ-smooth, we cannot guarantee that it is g-regular. Indeed, the following sequence of packets is a flow which is σ-smooth but not g-regular: the flow has a short packet (length $l_1 < l_{\max}$) at time $T_1 = 0$, followed by a packet of maximum size $l_{\max}$ at time $T_2 = \frac{l_1}{r}$. In fact, if a flow is σ-smooth, then it is g'-regular, with $g'(x) = \frac{(x-l_{\max})}{r}^+$. Nonetheless, this theory is a very elegant and powerful complement to the min-plus theory in [3], [4], [5].

In this paper we focus on the properties of regulators, and on the min-plus theory rather than the max-plus. As a consequence, our results apply to the usual definition of leaky bucket controllers and arrival constraints as given in Section II-A.

III. The Packetized Greedy Shaper

We now give a more fundamental look at the issue of packetized shaping. We introduce the concept of Packetized Greedy Shaper as a natural abstraction of the buffered

[4] In [10] there is a slightly weaker formula, with $ml_{\max}$ instead of $(m-1)l_{\max}$.

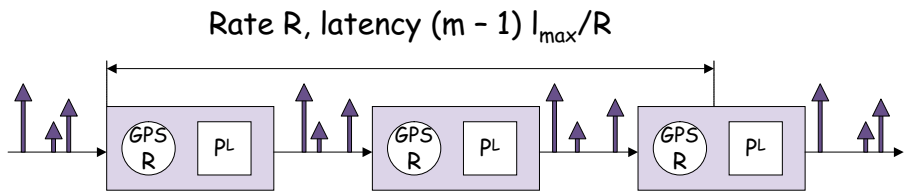

Fig. 3. The concatenation of several GPS fluid nodes with packetized outputs

leaky bucket controller.

A. Definition and Representation of Packetized Greedy Shaper

Definition III.1: [Packetized Greedy Shaper] Consider an input sequence of packets. Call L the cumulative packet length. We call *packetized shaper*, with shaping curve σ, a system which forces its output to have σ as arrival curve and be L-packetized. We call *packetized greedy shaper* a packetized shaper which delays the input packets in a buffer, whenever sending a packet would violate the constraint σ, but releases them as soon as possible.

The buffered leaky bucket controller defined in Section II-A is clearly a packetized greedy shaper with

$$\sigma(t) = \min_{m=1,\dots,M} (r_m t + b_m)$$

If some bucket size b_m is less than the maximum packet size, then it is never possible to output a packet: all packets remain stuck in the packet buffer, and the output is 0. In general, if $\sigma(0+) < l_{\max}$ then the the packetized greedy shaper blocks all packets for ever.[5] Thus, for practical cases, we have to assume that the arrival curve σ has a discontinuity at the origin at least as large as one maximum packet size. Our first main result is the following.

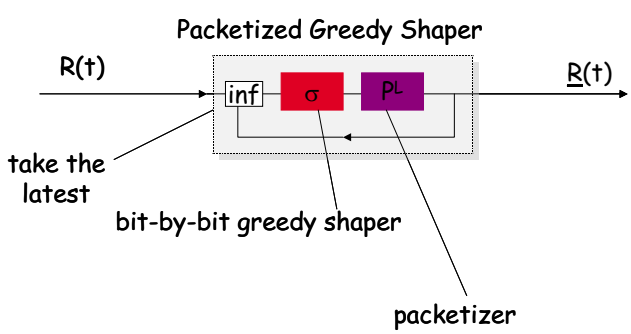

Fig. 4. Representation of a packetized greedy shaper, as given by Theorem III.1.

Theorem III.1: (I/O Characterisation of Packetized Greedy Shapers) Consider a packetized greedy shaper with shaping curve σ and cumulative packet length L. Assume that σ is sub-additive and $\sigma(0) = 0$. The output $\overline{R}(t)$ of the packetized greedy shaper is given by

$$\overline{R} = \inf \left\{ R^{(1)}, R^{(2)}, R^{(3)}, \dots \right\} \tag{8}$$

with $R^{(1)}(t) = P^L((\sigma \otimes R)(t))$ and $R^{(i)}(t) = P^L((\sigma \otimes R^{(i-1)})(t))$ for $i \geq 2$.

Figure 4 illustrates the theorem; Figure 5 shows the iterative construction of the output on one example. As another example, the reader can check that if $\sigma(t) = Ct$ for $t \geq 0$ (thus the condition $\sigma(0+) < l_{\max}$ is satisfied) then the result of Equation (8) is 0.

Theorem III.1 is of theoretical nature and will be used in the rest of this paper to derive some applications; as such, it does not explain how to implement a packetized greedy shaper in practice. However, we show in the next section that there is an important special case where the construction in Theorem III.1 can be considerably simplified.

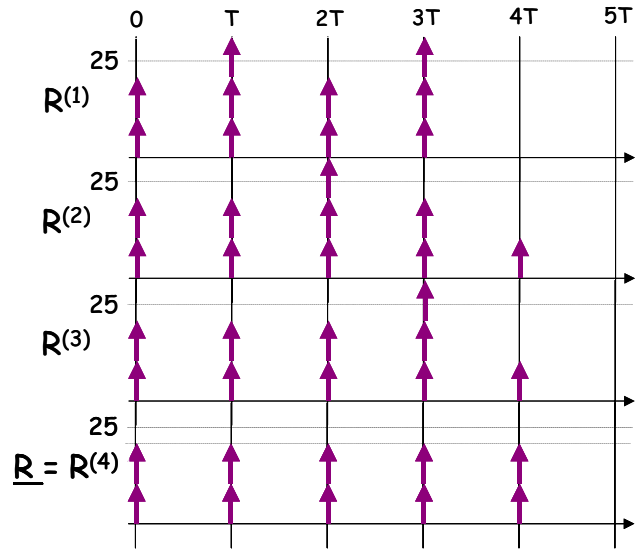

Fig. 5. Example of output of a packetized greedy shaper. The input data $R(t)$ (a burst of 10 packets of size 10 data units, at time 0), the shaping curve ($\sigma = 25v_T$) and thus $R^{(1)}$ are the same as on Figure 2. $R^{(i)}$ is obtained by passing $R^{(i-1)}$ first through a bit by bit greedy shaper, then trough a packetizer, as illustrated by Figure 4. $R^{(4)}$ is σ-smooth, therefore $R^{(5)} = R^{(4)}$ and the iteration stops.

[5] $\sigma(0+) = \inf_{t>0} \sigma(t)$ is the limit to the right of σ at 0.

B. A special case of interest

We have seen that appending a packetizer to a greedy shaper weakens the arrival curve property of the output. There is however a case where this is not true. This case is important by its application to Theorem III.3, but also has practical applications of its own. The following constitutes our second main theoretical result.

Theorem III.2: Consider a sequence L of cumulative packet lengths and a sub-additive function σ with $\sigma(0) = 0$. Assume that

$$\begin{cases} \text{There exists a sub-additive function } \sigma_0 \\ \text{and a number } l \geq l_{\max} \text{ such that} \\ \sigma(t) = \sigma_0(t) + l1_{t>0} \end{cases} \tag{9}$$

For any L-packetized input, the output of the concatenation of the bit-by-bit greedy shaper with shaping curve σ, followed by an L-packetizer, is σ-smooth.

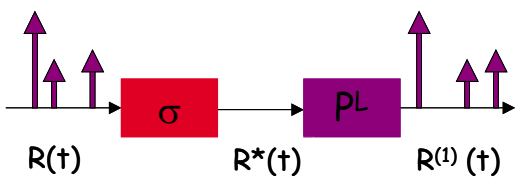

Fig. 6. Theorem III.2 says that if $R(t)$ is L-packetized and σ satisfies Equation (9), then $R^{(1)}$ is σ-smooth.

Figure 6 illustrates the theorem. Note that in general the output of the bit-by-bit greedy shaper is *not* L-packetized, even if σ satisfies the condition in the theorem. Similarly, if we relax the assumption that the input be L-packetized, then the output of the concatenation of the bit-by-bit greedy shaper and an L-packetizer is not σ-smooth, in general (finding counter-examples is easy and is left to the reader's enjoyment).

Discussion of the Condition in Equation (9): Equation (9) is satisfied in practice if σ is concave and $\sigma(0+) \geq l_{\max}$. This occurs for example if the shaping curve is defined by the conjunction of leaky buckets, with all bucket sizes at least as large as the maximum packet size.

This also sheds some light on the example in Figure 1: the problem occurs because the shaping curve $\sigma(t) = ct$ does not satisfy the condition.

The alert reader will ask herself whether a sufficient condition for Equation (9) to hold is that σ is sub-additive and $\sigma(0+) \geq l_{\max}$. Unfortunately, the answer is no. Consider for example the stair function $\sigma = l_{\max} v_T$. We have $\sigma(0+) = l_{\max}$ but if we try to decompose σ into $\sigma(t) = \sigma_0(t) + l1_{\{t>0\}}$ we must have $l = l_{\max}$ and $\sigma_0(t) = 0$ for $t \in (0, T]$; if we impose that σ_0 is sub-additive, the latter implies $\sigma_0 = 0$ which is not compatible with Equation (9).[6]

Counter-example: We can find shaping curves σ that do satisfy $\sigma(t) \geq l_{\max}$ for $t > 0$ but which still do not satisfy Equation (9). From the previous paragraph, such functions have to be non-concave. In such cases, the conclusion of Theorem III.2 may not hold in general. Figure 2 provides an example where this occurs.

Realization of Packetized Greedy Shaper : If Equation (9) is satisfied, then the realization of a packetized greedy shaper is simplified.

Theorem III.3: Consider a sequence L of cumulative packet lengths and a sub-additive function σ with $\sigma(0) = 0$. Assume that σ satisfies Equation (9). Consider only inputs that are L packetized. Then the packetized greedy shaper for σ and L can be realized as the concatenation of the bit-by-bit greedy shaper with shaping curve σ and the L-packetizer.

IV. Applications

In this section we apply the previous theory to some properties of leaky buckets and to two properties of shapers. We give the proofs inline, as they illustrate in a compact way the use of our method based on min-plus operators.

A. Buffered Leaky Buckets

Theorem III.2 gives us an alternative implementation of the buffered leaky bucket controller. We build it as the concatenation of a buffered leaky bucket controller operating bit-by-bit and a packetizer. We compute the output time for the last bit of a packet (= finish time) under the bit-by-bit leaky bucket controller, and release the entire packet instantly at this finish time. If each bucket pool is at least as large as the maximum packet size then Theorem III.2 tells us that the final output satisfies the leaky bucket constraints. Note that this implementation differs from the buffered leaky bucket controller based on bucket replenishment introduced in Section II-A. In the former, during a period where, say, bucket m only is full, fragments of a packet are virtually released at rate r_m, bucket m remains full, and the (virtual) fragments are then re-assembled in the packetizer; in the latter, if a bucket becomes full, the controller waits until it empties by at least the size of the current packet. Thus we expect that the level of fluid in both systems is not the same, the former being an upper bound. However, we have the following equivalence, which directly follows from Theorem III.3 and the discussion after Theorem III.2.

Corollary IV.1: For packetized inputs, the implementations of buffered leaky bucket controllers based on bucket

[6]The same conclusion unfortunately also holds if we replace sub-additive by "star-shaped" [10].

replenishment and virtual finish times are equivalent, provided that all bucket sizes are at least as large as the maximum packet size.

B. Does a packetized greedy shaper keep arrival constraints ?

In general, we cannot expect a positive answer. Indeed, Figure 7 shows a counter-example, namely a variable length packet flow which has lost its initial arrival curve constraint after traversing a packetized greedy shaper.

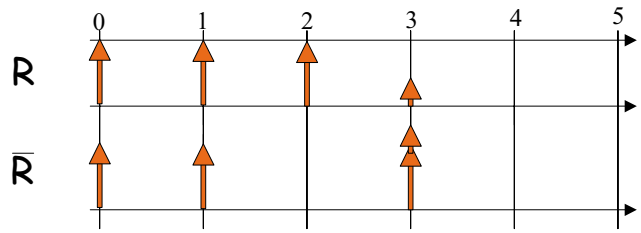

Fig. 7. Non-conservation of arrival constraints by a packetized greedy shaper. The input flow is shown above; it consists of 3 packets of size 10 data units and one of size 5 data units, spaced by one time unit. It is α-smooth with $\sigma = 10v_1$. The bottom flow is the output of the packetized greedy shaper with $\sigma = 25v_3$. The output has a burst of 15 data units packets at time 3. It is σ-smooth but *not* α-smooth.

However, if arrival curves are defined by leaky buckets that are large enough, we do have a positive result.

Theorem IV.1: (Conservation of Concave Arrival Constraints) Assume an L-packetized flow with arrival curve α is input to a packetized greedy shaper with cumulative packet length L and shaping curve σ. Assume that α and σ are concave with $\alpha(0+) \geq l_{\max}$ and $\sigma(0+) \geq l_{\max}$. Then the output flow is still constrained by the original arrival curve α.

Proof: We use the notation in Theorem III.1. Since σ satisfies Equation (9), it follows from Theorem III.3 that

$$\overline{R} = P^L(\sigma \otimes R)$$

Now R is α-smooth thus it is not modified by a bit-by-bit greedy shaper with shaping curve α:

$$R = \alpha \otimes R$$

Combining the two and using the associativity of $\otimes$ gives

$$\overline{R} = P^L[\sigma \otimes (\alpha \otimes R)] = P^L[(\sigma \otimes \alpha) \otimes R]$$

From our hypothesis, $\sigma \otimes \alpha = \min(\sigma, \alpha)$[10] and thus $\sigma \otimes \alpha$ satisfies Equation (9). Thus, by Theorem III.2, $\overline{R}$ is $\sigma \otimes \alpha$-smooth, and thus α-smooth. $\diamond$

If the condition in the previous theorem is not satisfied, then the conclusion may not hold. A first example was given at the beginning of this section, with non-concave shaping curves. A second example, with a concave shaping curve σ, but where $\sigma(0+)$ is too small, was given in Figure 1 (with $\sigma(t) = ct$).

C. Series decomposition of shapers

Theorem IV.2: Consider a tandem of M packetized greedy shapers in series; assume that the shaping curve σ_m of the mth shaper is concave with $\sigma_m(0+) \geq l_{\max}$. For L-packetized inputs, the tandem is equivalent to the packetized greedy shaper with shaping curve $\sigma = \min_m \sigma_m$.

Proof: We do the proof for $M = 2$ as it extends without difficulty to larger values of M. Call $R(t)$ the packetized input, $R'(t)$ the output of the tandem of shapers, and $\overline{R}(t)$ the output of the packetized greedy shaper with input $R(t)$.

Firstly, by Theorem III.3

$$R' = P^L[\sigma_2 \otimes P^L(\sigma_1 \otimes R)]$$

Now $\sigma_m \geq \sigma$ for all m thus

$$R' \geq P^L[\sigma \otimes P^L(\sigma \otimes R)]$$

Again by Theorem III.3, we have $\overline{R} = P^L(\sigma \otimes R)$. Moreover $\overline{R}$ is L-packetized and σ-smooth, thus $\overline{R} = P^L(\overline{R})$ and $\overline{R} = \sigma \otimes \overline{R}$. Thus finally

$$R' \geq \overline{R} \tag{10}$$

Secondly, R' is L-packetized and by Theorem IV.1, it is σ-smooth. Thus the tandem is a packetized (possibly non greedy) shaper. Since $\overline{R}(t)$ is the output of the packetized greedy shaper, we must have $R' \leq \overline{R}$. Combining with Equation (10) ends the proof. $\diamond$

It follows that a shaper with shaping curve $\sigma(t) = \min_{m=1,\ldots,M}(r_m t + b_m)$, where $b_m \geq l_{\max}$ for all m, can be implemented by a tandem of M individual leaky buckets, in any order. Furthermore, by Corollary IV.1, every individual leaky bucket may independently be based either on virtual finish times or on bucket replenishment.

If the condition in the theorem is not satisfied, then the conclusion may not hold. Indeed, for the example in Figure 7, the tandem of packetized greedy shapers with curves α and σ does not have an α-smooth output, therefore it cannot be equivalent to the packetized greedy shaper with curve $\min(\alpha, \sigma)$.

Thus, we have proven that the conservation and decomposition properties established in [12] for (b, r)-regulators holds for the more usually accepted definition of buffered leaky bucket controller. However, we have also found that, unlike the results for constant size packets in [3], [4], [5], we cannot, in general, extend this property to any arbitrary arrival curve.

V. Conclusion

We have extended the min-plus theory of variable length packet shaping and shown some fundamental results which account for the distortion introduced by packetization effects. Our main results are the min-plus representation of a packetized greedy shaper (Theorem III.1) and a condition under which shaping and packetization combine well (Theorem III.2). This allows us to prove that, under some assumptions, re-shaping a flow of variable length packets does keep original arrival constraints, and that the series decomposition of shaper holds. However, we show on some examples that if the assumptions are not satisfied, then the property may not hold any more. We also demonstrate the equivalence of implementing a buffered leaky bucket controller based on either virtual finish times or on bucket replenishment.

Our results suggest that, for systems with variable size packets, results on leaky buckets constraints do not extend to general arrival curve constraints such as ones based on spacing, contrary to what happens with fluid systems or ATM. For example, in presence of variable size packets, the series decomposition of shapers holds for leaky buckets, but, in general, does not for other constraints such as the spacing constraints that were used in the context of ATM to obtain tight end-to-end delay bounds. In some sense, leaky buckets may be more specific than the theory of bit-by-bit greedy shapers may lead one to think.

VI. Appendix: Proof of Theorems

A. Proof of Theorem III.1

A straightforward proof is a direct application of the general method in [19] or [18], after showing that the packetizer is upper-semi-continuous and isotone. We give here an alternative proof, which is essentially the same, but is self contained. The theorem follows immediately from Lemma VI.1, which in turn uses Lemma VI.2.

Lemma VI.1: Consider a sequence L of cumulative packet lengths and a sub-additive function σ with $\sigma(0) = 0$. Among all flows $x(t)$ such that

$$\begin{cases} x \leq R \\ x \text{ is } L\text{-packetized} \\ x \text{ has } \sigma \text{ as arrival curve} \end{cases} \tag{11}$$

there is one flow $\overline{R}(t)$ which upper-bounds all. It is given by Equation (8).

Proof: If x is a solution, then it is straightforward to show by induction on i that $x(t) \leq R^{(i)}(t)$ and thus $x \leq \overline{R}$. The difficult part is now to show that $\overline{R}$ is indeed a solution. We need to show that the three conditions in Equation (11) hold. Firstly, $R^{(1)} \leq R(t)$ and by induction on i, $R^{(i)} \leq R$ for all i; thus $\overline{R} \leq R$.

Secondly, consider some fixed t; $R^{(i)}(t)$ is L-packetized for all $i \geq 1$. Let $L(n_0) := R^{(1)}(t)$. Since $R^{(i)}(t) \leq R^{(1)}(t)$, $R^{(i)}(t)$ is in the set $\{L(0), L(1), L(2), ..., L(n_0)\}$. This set is finite, thus, $\overline{R}(t)$, which is the infimum of elements in this set, has to be one of the $L(k)$ for $k \leq n_0$. This shows that $\overline{R}(t)$ is L-packetized, and this is true for any time t.

Thirdly, we have, for all i

$$\overline{R}(t) \leq R^{(i+1)}(t) = P^L((\sigma \otimes R^{(i)})(t)) \leq (\sigma \otimes R^{(i)})(t)$$

thus

$$\overline{R} \leq \inf_i (\sigma \otimes R^{(i)})$$

From Lemma VI.2, $\inf_i(\sigma \otimes R^{(i)}) = \sigma \otimes \overline{R}$ thus

$$\overline{R} \leq \sigma \otimes \overline{R}$$

which shows the third condition. ◇

Lemma VI.2: (Convolution By A Fixed Function Is Upper-Semi-Continuous) Consider some function $\sigma(t)$ with $\sigma(0) = 0$. Consider also a sequence of functions $x_n(t)$ such that $x_{n+1}(t) \leq x_n(t)$ for all t and n and call $x = \inf_n x_n$. Then

$$\inf_n (\sigma \otimes x_n) = \sigma \otimes x$$

Proof: By definition of min-plus convolution, we have, for all $t \geq 0$:

$$(\sigma \otimes x_n)(t) = \inf_{s \in [0,t]} (\sigma(s) + x_n(t-s))$$

thus, by "Fubini" formula for infimum [18], Theorem 3.1.1:

$$\begin{aligned} \inf_n (\sigma \otimes x_n)(t) = & \inf_{s \in [0,t], n \in \mathbb{N}} [\sigma(s) + x_n(t-s)] \\ = & \inf_{s \in [0,t]} \{\inf_{n \in \mathbb{N}} [(\sigma(s) + x_n(t-s)]\} \\ = & \inf_{s \in [0,t]} \{\sigma(s) + \inf_{n \in \mathbb{N}} [x_n(t-s)]\} \\ = & \inf_{s \in [0,t]} [\sigma(s) + x(t-s)] \\ = & (\sigma \otimes x)(t) \end{aligned}$$

◇

B. Proof of Theorem III.2

We use the notation in Figure 6. We want to show that $R^{(1)}$ is σ-smooth. We have $R^* = R \otimes \sigma$. Consider now some arbitrary s and t with $s < t$. From the definition of min-plus convolution, for all $\epsilon > 0$, there exists some $u \leq s$ such that

$$(R \otimes \sigma)(s) \geq R(u) + \sigma(s-u) - \epsilon \tag{12}$$

Now consider the set E of $\epsilon > 0$ such that we can find one $u < s$ satisfying the above equation. Two cases are

possible: either 0 is an accumulation point for E[7] (case 1) , or not (case 2).

Consider case 1; there exists a sequence (ϵ_n, s_n), with $s_n < s$,

$$\lim_{n\to+\infty} \epsilon_n = 0$$

and

$$(R \otimes \sigma)(s) \geq R(s_n) + \sigma(s - s_n) - \epsilon_n$$

Now since $s_n \leq t$:

$$(R \otimes \sigma)(t) \leq R(s_n) + \sigma(t - s_n)$$

Combining the two:

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma(t - s_n) - \sigma(s - s_n) + \epsilon_n$$

Now $t - s_n > 0$ and $s - s_n > 0$ thus

$$\sigma(t - s_n) - \sigma(s - s_n) = \sigma_0(t - s_n) - \sigma_0(s - s_n)$$

We have assumed that σ_0 is sub-additive. Now $t \geq s$ thus

$$\sigma_0(t - s_n) - \sigma_0(s - s_n) \leq \sigma_0(t - s)$$

we have thus shown that, for all n

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma_0(t - s) + \epsilon_n$$

and thus

$$(R \otimes \sigma)(t) - (R \otimes \sigma)(s) \leq \sigma_0(t - s)$$

Now from Equation (6), it follows that

$$R^{(1)}(t) - R^{(1)}(s) \leq \sigma_0(t - s) + l_{\max} \leq \sigma(t - s)$$

which ends the proof for case 1.

Now consider case 2. There exists some ϵ_0 such that for $0 < \epsilon < \epsilon_0$, we have to take $u = s$ in Equation (12). This implies that

$$(R \otimes \sigma)(s) = R(s)$$

Now R is L-packetized by hypothesis. Thus

$$R^{(1)}(s) = P^L((R \otimes \sigma)(s)) = P^L(R(s)) = R(s) = (R \otimes \sigma)(s)$$

thus

$$\begin{aligned} R^{(1)}(t) - R^{(1)}(s) = & \quad P^L((R \otimes \sigma)(t) - (R \otimes \sigma)(s) \\ \leq & \quad (R \otimes \sigma)(t) - (R \otimes \sigma)(s) \end{aligned}$$

now $R \otimes \sigma$ has σ as arrival curve thus finally

$$R^{(1)}(t) - R^{(1)}(s) \leq \sigma(t - s)$$

which ends the proof for case 2. ◇

[7] namely, there exists a sequence of elements in E which converges to 0

C. Proof of Theorem III.3

Call $R(t)$ the packetized input; the output of the bit-by-bit greedy shaper followed by a packetizer is $R^{(1)}(t) = P^L(R \otimes \sigma)(t))$. Call $\overline{R}(t)$ the output of the packetized greedy shaper. We have $\overline{R} \leq R$ thus $\overline{R} \otimes \sigma \leq R \otimes \sigma$ and thus

$$P^L(\overline{R} \otimes \sigma) \leq P^L(R \otimes \sigma)$$

But $\overline{R}$ is σ-smooth, thus $\overline{R} \otimes \sigma = \overline{R}$, and is L-packetized, thus $P^L(\overline{R} \otimes \sigma) = \overline{R}$. Thus the former inequality can be rewritten as $\overline{R} \leq R^{(1)}$. Conversely, from Theorem III.2, $R^{(1)}$ is also σ-smooth and L-packetized. The definition of the packetized greedy shaper implies that $\overline{R} \geq R^{(1)}$ (for a formal proof, see Lemma VI.1) thus finally $\overline{R} = R^{(1)}$. ◇

References

[1] R.L. Cruz, "A calculus for network delay, part i: Network elements in isolation," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 114–131, January 1991.

[2] R.L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE JSAC*, pp. 1048–1056, August 1995.

[3] C.S. Chang, "On deterministic traffic regulation and service guarantee: A systematic approach by filtering," *IEEE Transactions on Information Theory*, vol. 44, pp. 1096–1107, August 1998.

[4] J.-Y. Le Boudec, "Application of network calculus to guaranteed service networks," *IEEE Transactions on Information Theory*, vol. 44, pp. 1087–1096, May 1998.

[5] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan, "Performance bounds for flow control protocols," *IEEE/ACM Transactions on Networking (7) 3*, pp. 310–323, June 1999.

[6] R. L. Cruz, "Sced+ : Efficient management of quality of service guarantees," in *IEEE Infocom'98, San Francisco*, March 1998.

[7] L. Georgiadis, Gurin R., and Peris V., "Efficient network provisioning based on per node traffic shaping," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 482–501, 1996.

[8] Agrawal R. and Rajan R., "A general framework for analyzing schedulers and regulators in integrated services network," in *34th Allerton Conf of Comm., Cont., and Comp. Monticello, IL*, Oct 1996, pp. 239–248.

[9] R. L. Cruz, "Lecture notes on quality of service guarantees," 1998.

[10] C.S. Chang, *Performance Guarantees in Communication Networks*, Springer-Verlag, New York, 2000.

[11] R. Guérin S. Shenker, C. Partridge, "Specification of guaranteed quality of service," Septembre 1997, RFC 2702, IETF.

[12] R.L. Cruz, "A calculus for network delay, part ii: Network analysis," *IEEE Trans. Inform. Theory, vol 37-1*, pp. 132–141, January 1991.

[13] I. Chlamtac, A. Faragó, H. Zhang, and A. Fumagalli, "A deterministic approach to the end-to-end analysis of packet flows in connection oriented networks," *IEEE/ACM transactions on networking*, vol. (6)4, pp. 422–431, 08 1998.

[14] J.-Y. Le Boudec and G. Hebuterne, "Comment on a deterministic approach to the end-to-end analysis of packet flows in connection oriented network," *IEEE/ACM Transactions on Networking*, February 2000.

[15] R. Guérin and V. Pla, "Aggregation and conformance in differentiated service networks – a case study," Tech. Rep. Research Report, U Penn, http://www.seas.upenn.edu:8080/ guerin/publications/aggreg.pdf, August 2000.

[16] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Trans. Networking, vol 1-3*, pp. 344–357, June 1993.

[17] P. Goyal, S. S. Lam, and H. Vin, "Determining end-to-end delay bounds in heterogeneous networks," in *5th Int Workshop on Network and Op. Sys support for Digital Audio and Video, Durham NH*, April 1995.

[18] J.-Y. Le Boudec and P. Thiran, *Network Calculus*, Springer Verlag Lecture Notes in Computer Science volume 2050 (available online at http://icawww.epfl.ch), July 2001.

[19] Baccelli F., Cohen G., Olsder G. J., , and Quadrat J.-P., *Synchronization and Linearity, An Algebra for Discrete Event Systems*, John Wiley and Sons, 1992.