Computation of a (min,+) multi-dimensional convolution for end-to-end performance analysis.

Anne Bouillard
ENS Cachan / IRISA
Campus de Beaulieu
35000 Rennes, France
Anne.Bouillard@irisa.fr

Laurent Jouhet
ENS Lyon / IXXI
46 Allée d' Italie
69007 Lyon, France
Laurent.Jouhet@ens-lyon.fr

Eric Thierry
LIAFA & ENS Lyon / IXXI
46 Allée d' Italie
69007 Lyon, France
Eric.Thierry@ens-lyon.fr

ABSTRACT

Network Calculus is an attractive theory to derive deterministic bounds on end-to-end performance measures. Nevertheless bounding tightly and quickly the worst-case delay or backlog of a flow over a path with cross-traffic remains a challenging problem.

This paper carries on with the study of configurations where a main flow encounters some cross-traffic flows which interfere over *connected sub-paths*. We also assume that no information is available about scheduling policies at the nodes (*blind multiplexing*). Such configurations were first analyzed in [25, 27] where a "Pay Multiplexing Only Once" (PMOO) phenomenon was identified, and then in [6, 7] where a $(\min, +)$ multi-dimensional operator was introduced to compute a minimum service curve for the whole path. Under usual assumptions (concave arrival curves and convex service curves), we prove some properties of this new operator and we show how to use it to derive bounds on delays and backlogs in polynomial time.

We also discuss the simpler case when there is no cross-traffic. Then the analysis is known to boil down to the $(\min, +)$ convolution of all the service curves over the path. For convex and piecewise affine service curves, a specific theorem enables to compute efficiently the convolution. This theorem has been used by several authors [6, 8, 17, 21, 22, 25, 27], but they all refer to a proof which is unfortunately incomplete [5]. To set definitely this theorem, we provide three different proofs. We also investigate the complexity of computing performances bounds in this case.

1. INTRODUCTION

Network Calculus can be presented as a deterministic queuing theory based on the $(\min, +)$ semi-ring, and aimed at worst-case performance analyzes in communication networks. From a mathematical point of view, it consists in combining curves which locally describe the shape of the traffic and the services, with $(\min, +)$ or $(\max, +)$ operations in order to predict the global behavior of the networks, and in particular end-to-end measures. This theory has managed to gather several results of the performance evaluation literature within a common framework and it has yielded interesting new results about QoS in networks [12, 13, 9, 5]. Originally intended for Internet quality of service [14], its use seems promising for other types of networks: embedded systems like networks processors [30], switched Ethernet networks [15], sensor networks [24, 28]. Besides those achievements and potential applications, some mathematical and algorithmic issues remain open. One main issue is the difficulty to analyze networks where several flows are multiplexed at some nodes. Such configurations also involve the type of scheduling policy used to serve data from those different flows when they cross. As an example, the complexity of deciding the *stability* of a network (i.e. checking that the amount of data backlogged in the network will never grow to infinite) is still open for simple Network Calculus constraints and FIFO policy everywhere [1, 5]. Even for feed-forward networks where a simple criteria of stability is known [5], providing quickly tight bounds on end-to-end measures like delays is very challenging in Network Calculus.

The analysis of one flow which encounters some cross-traffic on its path is one basic issue that has been addressed with various assumptions. For FIFO multiplexing at the nodes of the path (which are the servers), configurations with arbitrary cross-traffic have been studied in [3, 10, 16] where bounds on end-to-end delays can be derived only for small utilization factors. The reader is also referred to [18, 19, 20] for the latest results about FIFO multiplexing: tight bounds are provided for configurations where each cross-traffic flow interferes over a sub-path which is a suffix of the main path. Our work rather lies in the framework of *blind multiplexing*, that is nothing is known about multiplexing policies at the nodes. Despite this lack of information, one can provide bounds on end-to-end performance measures like delays or backlogs. The analysis of a path reduced to a single node with cross-traffic and under blind multiplexing can be found in [5]. This analysis was extended to configurations where cross-traffic flows interfere over *connected sub-paths* [25, 27]. Those authors identified a phenomenon called "Pay Multiplexing Only Once" (PMOO). To give a short description of this phenomenon, let us say that when the routed flow merges with some cross-traffic flows, its service may be strongly reduced at the first node. However at the next nodes, the interference due to the cross-traffic cannot be as severe since the competition for the resource has already been partially resolved at the first node (see [25, 27] for a discussion about the PMOO appellation). The PMOO phenomenon can be quantified in the Network Calculus framework: a formula provided in [27] for a small example was generalized in [6, 7] into an explicit formula of an end-to-end service curve offered by the whole path to the main flow. This formula is written under the form of a multi-dimensional $(\min, +)$ convolution. It was claimed in [6] that this new operator, and consequently the end-to-end service curve, could be computed in polynomial time. Unfortunately the algorithm had a flaw explained in [7]. The first part of our paper (Section 2) comes back to the properties and the computation of this multi-dimensional convolution under usual assumptions (concave arrival curves and convex service curves): we show that convexity is preserved (Theorem 3) and although the complexity of computing the whole output curve remains an open problem, one can compute end-to-end delay or backlog bounds in polynomial time thanks to linear programming (Theorem 4). It is a new evidence of the importance of algorithmic geometry for Network Calculus [8].

The second part of the paper (Section 4) focuses on paths without any cross-traffic. Such configurations represent the brand image of Network Calculus: it is well-known that one can analyze the performances by replacing the whole path by a single node offering as minimum service curve the convolution of all the service curves of the initial path. It captures for instance the famous “Pay Burst Only Once” phenomenon [5]. When the initial service curves are convex and piecewise affine with a finite number of pieces, the convolution comes to concatenating all the pieces of the different curves sorted by non-decreasing slopes (up to removing a few pieces at the end). This theorem stated in [5] has been used by various authors [6, 8, 17, 21, 22, 25, 27]. Unfortunately the proof presented in [5] is incomplete (Remark 2). Consequently we present three different proofs of this theorem (Theorem 5) which illustrate different types of tools that can be used. Then we investigate the complexity of computing performance bounds in this case with no cross-traffic (Theorem 6).

2. “PAY MULTIPLEXING ONLY ONCE”

2.1 The Network Calculus framework

In Network Calculus, flows and services in the network are modelled by non-decreasing functions $t \mapsto f(t)$ where t is *time* and $f(t)$ an amount of *data*. Time as well as data can be discrete (values in $\mathbb{N}$) or continuous (values in $\mathbb{R}$). In this paper, we will focus on the continuous time model and to cover both discrete and continuous data we will work in $\mathcal{F}$, the set of functions from $\mathbb{R}_+$ to $\mathbb{R}_{\min} = \mathbb{R} \cup \{+\infty\}$. The core of Network Calculus consists in providing bounds on worst-case performance measures by combining constraint functions on flows (*arrival curves*) and services (*service curves*) with $(\min, +)$ operations. Beyond usual operations over $\mathcal{F}$ like the minimum or the addition of functions, Network Calculus makes use of several classical operations [2] which are the translations of $(+, \times)$ filtering operations into the $(\min, +)$ setting. The *convolution*, denoted $*$, and the *deconvolution*, denoted $\oslash$, are defined as: for all f, g in $\mathcal{F}$, $\forall t \in \mathbb{R}_+$,

- Convolution: $(f * g)(t) = \inf_{0 \le s \le t}(f(s) + g(t-s))$.
- Deconvolution: $(f \oslash g)(t) = \sup_{u \ge 0}(f(t+u) - g(u))$.

Arrival curves. Given a data flow traversing a system, let $A \in \mathcal{F}$ be its *cumulative arrival function*, *i.e.* $A(t)$ is the total amount of data that has arrived in the system until time t, with $A(0) = 0$. A function $\alpha \in \mathcal{F}$ is an *arrival curve* for A if $\forall\, s, t \in \mathbb{R}_+$, $0 \le s \le t$, we have $A(t) - A(s) \le \alpha(t-s)$. It means that the amount of data arriving between time s and t is at most $\alpha(t-s)$. An example of arrival curve is the affine function $\alpha(t) = \sigma + \rho t$, $\sigma, \rho \in \mathbb{R}_+$ (sometimes called *leaky-bucket* arrival curve). In this case, σ can be interpreted as the maximal number of bits that can arrive simultaneously (maximal burst) and ρ the maximal long-term rate of arrivals.

Service curves. Consider B the *cumulative departure function* of the flow, *i.e.* the amount of data $B(t)$ that has left the system until time t, with $B(0) = 0$. The system is said to provide a (minimum) *service curve* $\beta \in \mathcal{F}$ if $B \ge A * \beta$. A *strict service curve* β for the system is a non-negative function in $\mathcal{F}$ such that during any backlogged period of duration u, at least $\beta(u)$ data is served. More formally, for any $s, t \in \mathbb{R}_+$, $s \le t$, if $\forall s < v < t$, $A(v) - B(v) > 0$, then $B(t) \ge B(s) + \beta(t-s)$. A strict service curve is clearly always a service curve for the system (by considering starts of backlogged periods), but the converse is not true [5].

Note that although it may seem unusual or hard to interpret, nothing prevents from using service curves and strict service curves which fail to be non-decreasing or non-negative.

Convex/concave/piecewise affine assumptions. In this paper, we will often assume that the input curves are piecewise affine with a finite number of pieces, and convex or concave. These are recurrent assumptions in Network Calculus [6, 7, 9, 5, 25, 30] due to the fact that such functions can be easily generated from combinations of linear functions and that convex/concave properties are often preserved by the Network Calculus operations.

Note that since we consider functions from $\mathbb{R}_+$ into $\mathbb{R}_{\min}$, our definition of a *convex* (resp. *concave*) function f is that its *support* (the set where it is finite) is $\mathbb{R}_+$ or an interval $[0, T]$, $T \in \mathbb{R}_+$, and f is convex on its support. In the same way, a function f from $\mathbb{R}_+$ into $\mathbb{R}_{\min}$ is *piecewise affine* if its support (the set where it is finite) is $\mathbb{R}_+$ or an interval $[0, T]$, $T \in \mathbb{R}_+$, and f is piecewise affine on its support. The number of affine pieces of f is denoted $|f|$.

Bounds on delays and backlogs. Bounds on worst-case backlogs and delays can be easily derived from the Network Calculus constraints.

Definition 1. *Let A be the cumulative arrival function of a flow entering a system and let B be its corresponding cumulative departure function. Then the* backlog *of the flow at time t is*

$$b(t) \overset{\text{def}}{=} A(t) - B(t)$$

and the delay (assuming FIFO order when serving data of *the flow) at time t is*

$$d(t) \overset{\text{def}}{=} \inf\{s \geq 0 \mid A(t) \leq B(t+s)\}.$$

Given an arrival curve and a service curve, it is possible to compute with the Network Calculus operations an upper bound on the maximal backlog (resp. delay). Moreover, one can also compute the arrival curve of the departure flow.

THEOREM 1 ([5, 9]). *Let $A \in \mathcal{F}$ be the cumulative arrival function with an arrival curve α for a flow entering a system with service curve β. Let $B \in \mathcal{F}$ be the cumulative departure function. Then,*

1. *B has an arrival curve $\alpha \oslash \beta$.*
2. *$b(t) \leq B_{\max}(\alpha, \beta) \overset{\text{def}}{=} \sup\{\alpha(t) - \beta(t) \mid t \geq 0\} = (\alpha \oslash \beta)(0)$.* [1]
3. *$d(t) \leq D_{\max}(\alpha,\beta)$* $\overset{\text{def}}{=} \inf\{d \geq 0 \mid \forall t \geq 0,\ \alpha(t) \leq \beta(t+d)\}$ [2] $= \inf\{d \geq 0 \mid (-\beta) \oslash (-\alpha)(d) \leq 0\}$.

As illustrated on Fig. 1, the backlog bound $B_{\max}(\alpha,\beta)$ is the maximal vertical distance between α and β while the delay bound $D_{\max}(\alpha,\beta)$ is given by the maximal horizontal distance between those two functions.

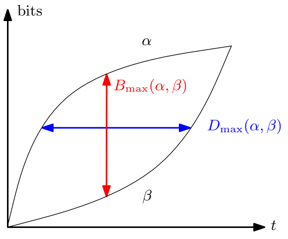

Figure 1: Bounds on worst-case backlog and delay.

2.2 Quantifying the "Pay Multiplexing Only Once" phenomenon

There exists an interesting formula which gives a end-to-end service curve for a path in case the cross-traffic flows interfere over *sub-paths*, *i.e.* sets of consecutive nodes. It generalizes a formula obtained in [27] for a small example with three nodes and two cross-traffic flows. Here are the configurations we study (and called *PMOO*) and the notation we use, illustrated by Fig. 2:

PMOO Configurations:

- The main flow F_0 follows a path $\mathfrak{p}$ of n nodes indexed 1,2,...,n with respective service curves β_j, $1 \leq j \leq n$, which can be *strict* or not.
- The cross-traffic interfering with the path is composed of k flows F_i, $1 \leq i \leq k$, with respective arrival curve α_i.
- Each cross-traffic flow F_i intersects the path $\mathfrak{p}$ over a set of consecutive nodes of $\mathfrak{p}$. It joins the path at node s_i and leaves it just after node e_i (in particular $s_0 = 1$ and $e_0 = n$).
- Consider a flow i, $1 \leq i \leq k$, the amount of data that have been served by node j until time t is denoted $A_i^{(j)}(t)$. The amount of data that have entered node s_i until time t is denoted $A_i^{(s_i-1)}(t)$.
- As in the rest of the paper, we assume *blind multiplexing* at the nodes: the scheduling policy applied to the input flows entering a node is unknown (we only assume that, for each flow, it is FIFO with regard to data from this flow).

[1]with the convention $(+\infty) - (+\infty) = -\infty$
[2]with the convention $\inf \emptyset = +\infty$

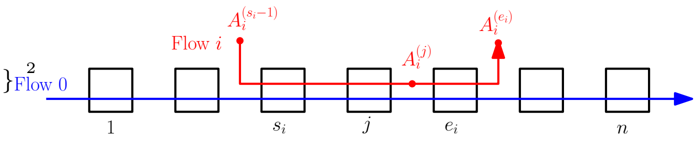

Figure 2: Path where each cross-traffic flow i interferes over a sub-path.

The next theorem is proved in [6] with the assumption that all the service curves β_j are strict. A careful look at the proof enables to rewrite the theorem as follows (it emphasizes the role of the *strict service curve* assumption):

THEOREM 2 (PMOO MULTI-DIMENSIONAL OPERATOR [6]). *For a PMOO configuration, a service curve offered by the whole path $\mathfrak{p}$ to flow F_0 is:*

$$\psi(t) = \inf_{\substack{u_1,\ldots,u_n \geq 0 \\ u_1+\cdots+u_n = t}} \sum_{j=1}^{n} \beta_j(u_j) - \sum_{i=1}^{k} \alpha_i\Big(\sum_{j=s_i}^{e_i} u_j\Big).$$

If all the service curves β_j are strict, *a service curve offered by the whole path is $\phi(t) = \psi(t)_+$, where $x_+ \overset{\text{def}}{=} \max(x, 0)$.*

Two classical results of Network Calculus can be seen as corollaries of this theorem. The first one corresponds to $n = 2$ and no cross traffic, and the second one to $n = k = 1$.

COROLLARY 1 (TANDEM [5, 9]). *Consider a flow crossing two nodes in tandem with respective service curves β_1 and β_2. Then the concatenation of the two nodes offers a minimum service curve $\beta_1 * \beta_2$ to the flow.*

COROLLARY 2 (RESIDUAL SERVICE [5, 9]). *Consider a node offering a* strict *service curve β and two flows entering that server, with respective arrival curves α_1 and α_2. Then a service curve for flow 1 is $\beta_1 = (\beta - \alpha_2)_+$.*

3. PERFORMANCE ANALYSIS WITH A MULTI-DIMENSIONAL CONVOLUTION

3.1 Multi-dimensional convolution

Definition 2. *Let $J = \{1,\dots,n\}$ and $I = \{1,\dots,k\}$. Let $\{f_i\}_{i\in I}$ be a family of functions in $\mathcal{F}$, and $\{J_i\}_{i\in I}$ be a family of subsets of J. The* multi-dimensional convolution *associated with those families is the function $\psi \in \mathcal{F}$ defined as:*

$$\psi(t) = \min_{\substack{u_1,\dots,u_n \geq 0\\ u_1+\cdots+u_n = t}} \sum_{i\in I} f_i\Big(\sum_{j\in J_i} u_j\Big).$$

Theorem 3. *With the notation of Definition 2, if all the functions f_i are convex, then ψ is also convex. Moreover if they are convex and piecewise affine with a finite number of pieces, then ψ is a convex piecewise affine function which can be computed in exponential time.*

Proof. For all $j \in J$, we will denote $I_j = \{i \in I \mid j \in J_i\}$. We can suppose w.l.o.g. that for all $1 \leq i \leq k$, $f_i(0) = 0$. Otherwise it is sufficient to notice that ψ is equal to the same formula with $\tilde{f}_i(x) = f_i(x) - f_i(0)$ plus the constant $\sum_{i=1}^k f_i(0)$. Then we may also suppose w.l.o.g. that all functions are non-decreasing. If some functions f_i are not non-decreasing, let $\lambda < 0$ be the smallest slope among them. For all $t \in \mathbb{R}_+$, it can be checked that

$$\psi(t) = k\lambda t + \min_{\substack{u_1,\dots,u_n \geq 0\\ u_1+\cdots+u_n = t}} \sum_{i=1}^k \tilde{f}_i\Big(\sum_{j\in J_i} u_j\Big) + \sum_{j=1}^n \lambda(|I_j| - k)u_j.$$

In this way, up to adding the negative linear function $t \mapsto k\lambda t$, ψ is the multi-dimensional convolution of $n+k$ non-decreasing convex piecewise functions. Note that those transformations yield new instances which can be stored in linear space with regard to the initial instance.

Now for each function f_i, $1 \leq i \leq k$, with $\ell_i \overset{\text{def}}{=} |f_i|$ pieces on its support, let $r_{i,\ell}$ (resp. $\tau_{i,\ell}$) be the slope (resp. the horizontal length, possibly $+\infty$) of the ℓ-th piece starting from the left. Since all f_i's are convex and non-decreasing, for a fixed $t \in \mathbb{R}_+$, the value $\psi(t)$ is exactly the solution of a linear programming instance (the *primal* one). It uses the non-negative variables $(u_j)_{1\leq j\leq n}$, $(v_{i,\ell})_{1\leq \ell\leq \ell_i}$ for each $1 \leq i \leq k$. It can be easily checked that the value $\psi(t)$ is equal to $\inf \sum_{i=1}^k \sum_{\ell=1}^{\ell_i} r_{i,\ell} v_{i,\ell}$ under the linear constraints: $u_1 + \cdots + u_n \leq t$ and for all $1 \leq i \leq k$,

$$\begin{cases} \sum_{j\in J_i} u_j \leq \sum_{\ell=1}^{\ell_i} v_{i,\ell} \\ v_{i,\ell} \leq \tau_{i,\ell} \text{ for all } 1 \leq \ell \leq \ell_i \end{cases}$$

We first wish to find the support of ψ: we only have to replace the objective function by $\sup t$ with the same constraints, the solution T (possibly $+\infty$) can be found in polynomial time by solving this linear programming instance. Whenever $t > T$ we know that $\psi(t) = +\infty$.

Now the Strong Duality theorem for linear programming [29] ensures that, when it is finite, $\psi(t)$ is also equal to the optimum of the *dual* instance defined for the non-negative variables y, $(y_i)_{1\leq i\leq k}$ and $(y_{i,\ell})_{1\leq \ell\leq \ell_i}$. The value $\psi(t)$ is equal to $\sup yt - \sum_{i=1}^k \sum_{\ell=1}^{\ell_i} y_{i,\ell}\tau_{i,\ell}$ under the constraints:

$$\begin{cases} y_i \leq y_{i,\ell} + r_{i,\ell} & \text{for all } 1 \leq i \leq k, 1 \leq \ell \leq \ell_i, \\ y \leq \sum_{i\in I_j} y_i & \text{for all } 1 \leq j \leq n \end{cases}$$

Those $c = n + \sum_{i=1}^k \ell_i$ linear constraints over $d = c + k + 1$ variables define a convex polyhedron in $\mathbb{R}_+^d$ which is independent of t (note that this polyhedron is not bounded). We already know the support of ψ, thus we are only interested in finite values of ψ. It is known that the supremum (if finite) is necessarily reached at an extremal point of the polyhedron in $\mathbb{R}_+^d$. Moreover the set of extremal points is finite, its cardinal can be exponential in d, and this set can be computed in exponential time in d [4, 29]. Each extremal point $(y, (y_i)_{1\leq i\leq k}, (y_{i,\ell})_{1\leq \ell\leq \ell_i}) \in \mathbb{R}_+^d$ defines an affine function $t \mapsto yt - \sum_{i=1}^k \sum_{\ell=1}^{\ell_i} y_{i,\ell}\tau_{i,\ell}$ and over its support, $\psi(t)$ is the supremum of this finite set of affine functions. Consequently ψ is a convex piecewise affine function which can be computed in exponential time in d. $\square$

For now, when the functions f_i are convex and piecewise affine, if one wishes to compute and store the whole function ψ, we only have an exponential algorithm except from very specific cases where a polynomial algorithm is known:

- $J_i = J$ for all i (it comes to the addition of convex piecewise affine functions).
- $I = J$ and $J_i = \{i\}$ for all i (it comes to the classical convolution - see Section 4).

Nevertheless we conjecture that it is possible to compute the whole function ψ in polynomial time (which implies that the number of pieces of ψ is polynomial with the size of the input functions). As a matter of fact, we can already compute, in polynomial time, worst-case bounds on delays and backlogs using ψ but without computing explicitly the whole curve, as shown in the next section.

3.2 Bounds on delays and backlogs

Theorem 4. *In a PMOO configuration, if all the service (resp. arrival) curves $|\beta_j|$ (resp. $|\alpha_i|$, including α the arrival curve of the main flow F_0) are convex (resp. concave) piecewise affine functions with a finite number of pieces, then the worst-case bounds $D_{\max}(\alpha,\psi)$ and $B_{\max}(\alpha,\psi)$ for the main flow F_0 can be computed in polynomial time by solving one linear programming instance with $\mathcal{O}(\sum_{i=1}^k |\alpha_i| + \sum_{j=1}^n |\beta_j|)$ linear constraints over $\mathcal{O}(\sum_{i=1}^k |\alpha_i| + \sum_{j=1}^n |\beta_j|)$ variables (replace ψ by $\phi = \psi_+$ if service curves are strict).*

Proof. Consider $C(\alpha,\psi) \overset{\text{def}}{=} \{(t,z) \in \mathbb{R}_+ \times \mathbb{R} \mid \psi(t) \leq z \leq \alpha(t)\}$ the area between the arrival curve α and the end-to-end service curve ψ defined by Theorem 2. This function ψ is a particular case of the multi-dimensional convolution and due to the hypotheses on α_i and β_j, Theorem 3 ensures that ψ is convex (as well as ψ_+). Since α is concave, $C(\alpha,\psi)$ is a convex set. As a matter of fact, let $\alpha(t) \overset{\text{def}}{=} \min_{1\leq p\leq|\alpha|}(a_p t + b_p)$, then $(t,z) \in C(\alpha,\psi)$ if and only if $z \leq a_p t + b_p$ for all $1 \leq p \leq |\alpha|$ and (t,z) satisfies the constraints of the primal linear programming instance presented in the proof of Theorem 3 applied to our PMOO formula. Just replace the objective function by the constraint $z \geq \inf \sum_{i=1}^{k} \sum_{\ell=1}^{\ell_i} r_{i,\ell} v_{i,\ell}$ up to the small transformations presented at the beginning of that proof. Then $D_{\max}(\alpha, \psi) = \sup\{t' - t \mid (t,z) \in C(\alpha,\psi),\ (t',z) \in C(\alpha,\psi),\ z \in \mathbb{R}_+\}$ and $B_{\max}(\alpha,\psi) = \sup\{z' - z \mid (t,z) \in C(\alpha,\psi),\ (t,z') \in C(\alpha,\psi),\ t \in \mathbb{R}_+\}$. In both case, the value is the optimum of a linear programming instance with $\leq 2(2n + k + 2 + |\alpha| + \sum |\alpha_i| + \sum |\beta_j|)$ constraints over $\leq 2(n + \sum |\alpha_i| + \sum |\beta_j|) + 3$ variables.

If all the service curves are strict, considering $\phi = \psi_+$ instead of ψ provides tighter bounds and only adds one linear constraint. □

Note that, in potential applications of Network Calculus, it is often assumed that $|\alpha_i|$ and $|\beta_j|$ are equal to 1 or 2 leading to small linear programming instances

Remark 1. *The first issue of Network Calculus is to provide bounds on worst-case performance measures. Then a very difficult problem is to find tight bounds. The first analyzes taking into account PMOO raised this question for the configurations we consider here [25, 27]. The works [6, 7] proving the PMOO formula of Theorem 2 showed on a small example that in some specific cases the end-to-end service curve $\phi = \psi_+$ could give worse bounds on delays and backlogs than some other service curves also offered by the path but better adapted to the arrival curve of the main flow. In fact, very recent and accurate work [26] suggests that the situation is much more complicated than expected and requires the development of new methods to get tight bounds, losing part of the algebraic flavor of Network Calculus. Nevertheless the use of our PMOO formula still provides better bounds than former approaches as indicated by experiments in [25, 27, 7] and, as shown in Theorem 4, those bounds can be computed in polynomial, whereas the recent solution presented in [26] requires exponential time ($\prod_{i=1}^{k} |\alpha_i| \prod_{j=1}^{n} |\beta_j|$ linear programming instances to solve).*

4. A PARTICULAR CASE: THE CLASSICAL (MIN,+) CONVOLUTION

4.1 The convolution of piecewise affine convex functions

Theorem 5 ([5]). *Let f and g be two convex piecewise affine functions over $\mathbb{R}_+$ with a finite number pieces. Let ρ_f (resp. ρ_g) be the slope of the last semi-infinite segment of f (resp. g) if it exists, or $\rho_f = +\infty$ (resp. $\rho_g = +\infty$) if f (resp. g) is equal to $+\infty$ from a point. Then the convolution $f * g$ first consists in concatenating, in the increasing order of the slopes and starting from $f(0) + g(0)$, all the segments of slope $< \min(\rho_f, \rho_g)$ of f and g. It ends by concatenating a semi-infinite segments of slope $\min(\rho_f, \rho_g)$ if this value is finite.*

First proof (Local differentiation). Let $f, g \in cF$ be two convex functions. As convex functions, f and g are continuous and admit a left and a right derivative. For a function f admitting a derivative on the left (resp. right),

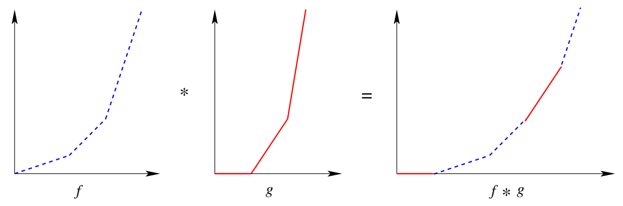

Figure 3: Convolution of convex piecewise affine functions.

let us denote by f'_ℓ (resp. f'_r) the left (resp. right) derivative function of f. If f is convex, f'_ℓ and f'_r are non-decreasing and $\forall u \in \mathbb{R}_+$, $f'_\ell(u) \leq f'_r(u)$ and are respectively left and right-continuous. By convention, one set $f'_\ell(0) = g'_\ell(0) = -\infty$.

The next two lemmas apply to arbitrary convex functions in $\mathcal{F}$.

Lemma 1 (Folklore). *Let f be a convex function, $\forall u, v \in \mathbb{R}_+$, with $u < v$, one has*

$$f'_r(u) \leq \frac{f(v) - f(u)}{v - u} \leq f'_\ell(v).$$

Lemma 2. *Let $f, g \in \mathcal{F}$ be two convex functions. Then, for all $t \in \mathbb{R}_+$ and $u, v \in \mathbb{R}_+$ s.t. $u + v = t$, the next two points are equivalent:*

- $(f * g)(t) = f(u) + g(v)$.
- $g'_l(v) \leq f'_r(u)$ *and* $f'_l(u) \leq g'_r(v)$.

For all $t \in \mathbb{R}_+$, there always exist such $u, v \in \mathbb{R}_+$.

Proof. Since f and g are continuous and $f * g(t)$ is an infimum over the compact set $[0, t]$, the existence of u and v stated at the end is proved.

Suppose that one can find u and v such that $u + v = t$, $g'_l(v) \leq f'_r(u)$ and $f'_l(u) \leq g'_r(v)$. Consider u' and v' such that $f * g(t) = f(u') + g(v')$ with $u' + v' = t$ (then $u - u' = v' - v$). Without loss of generality, one can suppose that $u' \leq u$. Then,

$$\begin{aligned}
&f(u) + g(v)\\
&= f(u') + [f(u) - f(u')] + g(v') + [(g(v) - g(v')]\\
&= f(u') + g(v') + (u - u')\left[\tfrac{f(u) - f(u')}{u - u'} - \tfrac{g(v') - g(v)}{v' - v}\right]\\
&\leq f(u') + g(v') + (u - u')(f'_l(u) - g'_r(v))\\
&\leq f(u') + g(v').
\end{aligned}$$

As a consequence, $f(u) + g(v) = f * g(t)$.

Conversely, let $f * g(t) = f(u) + g(v)$, $u + v = t$. Then $f(u) + g(v) \leq \inf_{0 < \varepsilon < u} f(u - \varepsilon) + g(v + \varepsilon)$. It implies $0 \leq \inf_{\varepsilon > 0} [-f(u) + f(u - \varepsilon) + g(v + \varepsilon) - g(v)]$. Due to convexity, $g(v + \varepsilon) - g(v) \leq \varepsilon g'_r(v + \varepsilon)$ and $f(u) - f(u - \varepsilon) \geq \varepsilon f'_l(u - \varepsilon)$. Thus $f'_l(u - \varepsilon) \leq g'_r(v + \varepsilon)$. The function f'_l (resp. g'_r) is continuous on the left (resp. on the right). By letting ε go to 0, we get $f'_l(u) \leq g'_r(v)$. In a symmetric way, we can prove $g'_l(v) \leq f'_r(u)$.

□

We now apply this theorem to piecewise affine functions. Let $t \in \mathbb{R}_+$ and u and v be such that $f * g(t) = f(u) + g(v)$. We look at continuity of the derivative of f and g at those points.

- if f' exists at u and is continuous on $[u, u+\varepsilon]$, then $f'(u) = f'_l(u+\varepsilon)$ and $f_l(u+\varepsilon) \leq g'_r(v)$ and $g'_l(v) \leq f'_r(u) \leq f'_r(u+\varepsilon)$. So, $f*g(t+\varepsilon) = f(u+\varepsilon)+g(v)$ and $f*g$ is affine of slope $f'(u)$ on $[t, t+\varepsilon]$. One can chose ε such that there is a change of slope in f at $u+\varepsilon$.
- the same holds for g and v if g' is continuous on an interval $[v, v+\varepsilon]$ by symmetry.
- If f' and g' don't exist at u and v, and if $f'_r(u) \leq g'_r(v)$ (the role of f and g are symmetric), then $f'_l(u+\varepsilon) = f'_r(u) \leq g'_r(v)$ and $g'_l(v) \leq f'_r(u) = f'_r(u+\varepsilon)$. So $f*g(t) = f(u+\varepsilon)+g(v)$ and the slope of $f*g$ after t is $f'_r(u)$. This holds for any ε smaller that the length of the segment of slope $f'_r(u)$ in f. If this segment is of infinite length, it holds for every $\varepsilon \in \mathbb{R}_+$.

The three items show that one can construct $f*g$ by letting both u and v grow (sometimes alternatively). The first two items show that the complete pieces of f and g appear in the function $f*g$. The third item indicates the rule of concatenation: the piece of smallest slope appears first. ■

Second proof (Algorithmic proof). Consider the first segment of f, of slope s_f and the first segment of g of slope s_g and suppose that $s_f \leq s_g$ and that the length of the first segment of f is $\ell_f \in]0, \infty]$. Then, for every $t \in [0, \ell_f]$,

$$\begin{aligned} f*g(t) &= \inf_{u+v=t, u,v\geq 0} f(u)+g(v) \\ &= \inf_{u+v=t,u,v\geq 0} f(0)+s_f u + g(0) + v\frac{g(v)-g(0)}{v-0}. \end{aligned}$$

As g is convex, $\frac{g(v)-g(0)}{v-0} \geq s_g \geq s_f$ so $f*g(t) = g(0)+f(t)$.

If $\ell_f = \infty$, then $f*g$ is an affine function consisting in the segment of f and g if smallest slope.

Otherwise, $f*g$ consist of the segment of smallest slope on $[0, \ell_f]$.

Let $t > \ell_f$ and suppose that $f*g(t) = f(u)+g(v)$, $u+v=t$ and $u < \ell_f$.

$$\begin{aligned} & f(\ell_f) + g(t-\ell_f) \\ & = [f(\ell_f) - f(u)] + f(u) + [g(t-\ell_f) - g(v)] + g(v) \\ & = f(u) + g(v) + s_f(\ell_f - u) - (\ell_f - u)\tfrac{g(t-\ell_f)-g(v)}{t-\ell_f-v}. \end{aligned}$$

But, $\frac{g(t-\ell_f)-g(v)}{t-\ell_f-v} \geq s_g \geq s_f$, so $f(\ell_f)+g(t-\ell_f) \leq f(u) + g(v) = f*g(t)$.

As a consequence, $f*g(t) = f(\ell_f)+g(t-\ell_f)$ and one always can write

$$f*g(t) = f(\ell) + \tilde{f}*g(t-\ell_f),$$

with $\tilde{f}(t-\ell_f) = f(t) - f(\ell_f)$. In other words, $\tilde{f}$ is constructed from f by removing the first segment in f (and $\tilde{f}(0)=0$). $\tilde{f}$ is also convex, so one can compute iteratively the remaining of $f*g$. The segments of f and g are clearly concatenated in increasing order of the slopes. If there is a segment of infinite length, then the segments of greater slopes are ignored. ■

Third proof (Legendre-Fenchel transform). Without loss of generality, we assume that f and g are non-decreasing and $f(0) = g(0) = 0$. Suppose actually that f or g have pieces with negative slopes and let $R < 0$ be the smallest slopes over all their pieces. Then one can easily check that for all $t \in \mathbb{R}_+$, $(f*g)(t) - Rt = \inf_{u+v=t,u\geq 0,v\geq 0}((f(u) - Ru) + (g(v) - Rv))$. The functions $u \mapsto f(u) - Ru$ and $v \mapsto g(v) - Rv$ remain convex piecewise affine and become non-decreasing. One can reason on those functions and then come back to $f*g$ by adding $t \mapsto Rt$ which preserves the concatenation construction. In the same way, suppose that $f(0) \neq 0$ or $g(0) \neq 0$, then one can reason with the functions $u \mapsto f(u) - f(0)$ and $v \mapsto g(v) - g(0)$ and then deduce the theorem statement since $(f*g)(t) - f(0) - g(0) = \inf_{u+v=t,u\geq 0,v\geq 0}((f(u) - f(0)) + (g(v) - g(0)))$.

Let $\mathcal{F}^+$ be the set of non-decreasing functions h from $\mathbb{R}_+$ into $\mathbb{R}_+ \cup \{+\infty\}$ such that $h(0) = 0$. We use the Legendre-Fenchel transform restricted to $\mathcal{F}^+$. It associates with any function $f \in \mathcal{F}^+$ the function $\widehat{f} \in \mathcal{F}^+$ defined by $\widehat{f}(\lambda) \overset{\text{def}}{=} \sup_{t\geq 0}(\lambda t - f(t))$ for all $\lambda \in \mathbb{R}_+$. This powerful tool of convex analysis [23, 31] has very interesting properties:

(LF1) For all $f, g \in \mathcal{F}^+$, $\widehat{f*g} = \widehat{f} + \widehat{g}$.

(LF2) For all $f \in \mathcal{F}^+$, $\widehat{\widehat{f}} = f$ if and only if f convex.

A consequence of the second property is that $f \mapsto \widehat{f}$ is an involution (and thus a bijection) on $\mathcal{C}^+$ the set of convex functions of $\mathcal{F}^+$. The next lemma provides a formula for the Legendre-Fenchel transform of convex piecewise affine functions.

LEMMA 3. *Let $f \in \mathcal{F}^+$ be a convex piecewise affine function with ℓ pieces over its support. For each $1 \leq i \leq \ell$, r_i (resp. τ_i) denotes the slope (resp. the horizontal length) of the i-th piece starting from the left (with the convention $\tau_\ell = +\infty$ if the last piece is semi-infinite). Then for all $\lambda \in \mathbb{R}_+$:*

$$\widehat{f}(\lambda) = \sum_{i=1}^{\ell} (\lambda - r_i)_+ \tau_i$$

with the convention $0 \times +\infty = 0$.

The proof of this lemma is easy from the definition of the Legendre-Fenchel transform. It uses the fact that the sequence $(r_i)_{1\leq i\leq \ell}$ is increasing to locate the value t for which the supremum is reached.

From Property (LF1) and Lemma 3, for all $\lambda \in \mathbb{R}_+$, $\widehat{f*g}(\lambda) = \widehat{f}(\lambda)+\widehat{g}(\lambda) = \sum_{(r,\tau)\in C_f}(\lambda-r)_+\tau+\sum_{(r,\tau)\in C_g}(\lambda-r)_+\tau$ where C_f (resp. C_g) contains the couples (slope,horizontal length) of all the pieces of f (resp. g).

We also know that $f*g$ is convex. This is true for any convex functions $f, g \in \mathcal{F}$ (even if not piecewise affine) since $f*g$ is the infimum of a convex function over a convex set [23].

We now introduce h the concatenation from point $(0,0)$ of all the pieces of f and g, sorted by non-decreasing slopes. If f or g has a semi-infinite piece, that is $(\rho_f, +\infty) \in C_f$ or $(\rho_g, +\infty) \in C_g$, then we ignore the pieces with slopes larger than ρ_f or ρ_g and h ends by the semi-infinite piece of smallest slope (denoted ρ_h). Since h sorts the pieces by non-decreasing slopes h is convex by construction and Lemma 3 applies: for all $\lambda \in \mathbb{R}_+$, $\widehat{h}(\lambda) = \sum_{(r,\tau)\in C_h}(\lambda-r)_+\tau$ where C_h denotes the set of the couples (slope,horizontal length) of the pieces of h .

One can easily check that, for all $\lambda \in \mathbb{R}_+$, $\widehat{f*g}(\lambda) = \widehat{h}(\lambda)$. If f and g have no semi-infinite, $C_h = C_f \cup C_g$. Otherwise $C_h \subseteq C_f \cup C_g$, but the terms which only appear in the formula of $\widehat{f*g}(\lambda)$ do not play any role: due to their form $(\lambda-r)_+\tau$ with $r > \rho_h$, they may come up only for $\lambda > \rho_h$, but both $\widehat{f*g}$ and $\widehat{h}$ contain the term $(\lambda-\rho_h)_+ \times +\infty$ which is equal $+\infty$ for such λ.

To sum up, $f*g$ and h are both convex functions of $\mathcal{F}^+$ and $\widehat{f*g} = \widehat{h}$. Since the Legendre-Fenchel transform is injective over $\mathcal{C}^+$ the set of convex functions of $\mathcal{F}^+$, we have $f*g = h$.

■

Remark 2. *Theorem 5 is presented in [5] (Theorem 3.1.6 rule 9, pages 113-115). However the proof for convex piecewise affine functions is incomplete. Two functions are defined from the convex functions f and g: $h \overset{\text{def}}{=} f*g$ and h' which is the concatenation from $f(0)+g(0)$ of all the pieces of f and g sorted by non-decreasing slopes. The proof defines the respective epigraphs $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}$, $\mathcal{S}'$ of f, g, h, h' (and their boundaries $\partial\mathcal{S}_1$, $\partial\mathcal{S}_2$, $\partial\mathcal{S}$, $\partial\mathcal{S}'$). It is shown that $\partial\mathcal{S}' \subseteq \partial\mathcal{S}_1 + \partial\mathcal{S}_2$. It implies that $h' \geq f*g$, but unfortunately it does not prove the converse inequality. The proof that $\partial\mathcal{S}' \subseteq \partial\mathcal{S}_1 + \partial\mathcal{S}_2$ only uses the fact that h' is a prefix of a shuffle of the two sequences of pieces of f and g (seen as two words). It does not use the sorting of slopes. Although at some point it is noted that h' is convex by construction, this property is not used later.*

4.2 Bounds on delays and backlogs

Theorem 6. *In the simple PMOO configurations where there is no cross-traffic, a service curve for the whole path is $\beta = \beta_1 * \beta_2 * \cdots \beta_n$. If all the service curves are convex piecewise affine functions with a finite number of pieces, the size of ψ satisfies $|\psi| \leq \sum_{j=1}^n |\beta_j|$ and ψ can be computed in $\mathcal{O}(\log_2 n \sum_{j=1}^n |\beta_j|)$ time.*

Given an concave piecewise affine arrival curve α with a finite number of pieces, if the pieces of α (resp. ψ) are stored in an array sorted w.r.t. abscisses, one can compute $D_{\max}(\alpha,\beta)$ and $B_{\max}(\alpha,\beta)$ in $\mathcal{O}((\log^2(|\alpha|+|\beta|)))$ time.

Proof. Thanks to Theorem 5, the computations of $\beta = \beta_1 * \beta_2 * \cdots \beta_n$ comes to merging n sorted lists of respective sizes $|\beta_j|$. The complexity stated in the theorem can be achieved by progressing from left to right on the lists while maintaining a binary heap which contains the first available piece of each list and enable to find efficiently the one of smallest slope [11].

In order to compute the bound $B_{\max}(\alpha,\beta)$, one can use a kind of dichotomic algorithm. Two pointers $l_\alpha \leq r_\alpha$ (resp. $l_\beta \leq r_\beta$) are associated with each sorted array, they points at two pieces of the corresponding function and indicates the extremities on the interval where the maximum vertical distance may be reached. Those pointers will move during the search of this maximum, they are initialized at both extremities of each array, i.e. $l_\alpha = 1$ and $r_\alpha = |\alpha|$, $l_\beta = 1$ and $r_\beta = |\beta|$. A elementary step of the algorithm can be decomposed as follows:

- Choose the function which maximizes $\max(r_\alpha - l_\alpha, r_\beta - l_\beta)$. Say for instance that it is α.
- Consider the median piece between l_α and r_α, i.e. the piece of index $\lceil l_\alpha + r_\alpha \rceil$, and find the piece(s) of β below each extremities of this piece thanks to a dichotomic search in the sorted array of β.
- By comparing the slopes of the median piece of α and of the piece(s) of below its extremities (in some cases one may have to check the slopes of the adjacent pieces), one can decide whether the maximum vertical distance is located on the left, on the right or below the median piece. The pointers of α can be appropriately updated.

Each step removes at least 1/4 of the number of pieces of α and β, and each dichotomic search to locate pieces w.r.t. the median piece is in $\mathcal{O}(\log_2(|\alpha|+|\beta|))$ time. Thus the overall time complexity is bounded by $\mathcal{O}(\log_2(|\alpha|+|\beta|)\log_{3/4}(|\alpha|+|\beta|))$.

The same kind of algorithm applies to the computation of $D_{\max}(\alpha,\beta)$. □

5. REFERENCES

[1] M. Andrews. Instability of fifo in the permanent sessions model at arbitrarily small network loads. In *Proceedings of SODA'07*, 2007.

[2] F. Baccelli, G. Cohen, G.Y. Olsder, and J.P. Quadrat. *Synchronization and linearity.* Wiley, 1992.

[3] J.C.R. Bennett, K. Benson, A. Charny, W.F. Courtney, and J.-Y. Le Boudec. Delay jitter bounds and packet scale rate guarantee for expedited forwarding. *IEEE/ACM Transactions on Networking*, 10(4):529–540, 2002.

[4] J.-D. Boissonnat and M. Yvinec. *Algorithmic Geometry.* Cambridge University Press, 2001.

[5] J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, volume LNCS 2050. Springer-Verlag, revised version 4, may 10, 2004 edition, 2001.

[6] A. Bouillard, B. Gaujal, S. Lagrange, and E. Thierry. Optimal routing for end-to-end guarantees: the price of multiplexing. In *Proceedings of Valuetools'07*, 2007.

[7] A. Bouillard, B. Gaujal, S. Lagrange, and E. Thierry. Optimal routing for end-to-end guarantees using network calculus. Technical report, INRIA, 2008.

[8] A. Bouillard and E. Thierry. An algorithmic toolbox for network calculus. *Discrete Event Dynamic Systems*, 18(1):3–49, 2008.

[9] C. S. Chang. *Performance Guarantees in Communication Networks*. TNCS, 2000.

[10] A. Charny and J.-Y. Le Boudec. Delay bounds in a network with aggregate scheduling. In *Proceedings of QoFIS'00*, volume LNCS 1922, pages 1–13, 200.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

[12] R. L. Cruz. A calculus for network delay, part i: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, 1991.

[13] R. L. Cruz. A calculus for network delay, part ii: Network analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, 1991.

[14] V. Firoiu, J.-Y. Le Boudec, D. Towsley, and Zhi-Li Zhang. Theories and models for internet quality of service. *Proceedings of the IEEE*, 90(9):1565–1591, 2002.

[15] J.-P. Georges, E. Rondeau, and T. Divoux. Evaluation of switched ethernet in an industrial context by using the network calculus. In *Proceedings of 4th IEEE Workshop on Factory Communication Systems*, pages 19–26, 2002.

[16] Y. Jiang. Delay bounds for a network of guaranteed rate servers with fifo aggregation. *Computer Networks*, 40(6):683–694, 2002.

[17] H. Kim and J. C. Hou. Network calculus based simulation for tcp congestion control: Theorems, implementation and evaluation. In *Proceedings of INFOCOM'2004*, 2004.

[18] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea. A novel approach to scalable cac for real-time traffic in sink-tree networks with aggregate scheduling. In *Proceedings of Valuetools'06*, 2006.

[19] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea. Tight end-to-end per-flow delay bounds in fifo multiplexing sink-tree networks. *Performance Evaluation*, 63(9-10):956–987, 2006.

[20] L. Lenzini, E. Mingozzi, and G. Stea. End-to-end delay bounds in fifo-multiplexing tandems. In *Proceedings of Valuetools'07*, 2007.

[21] F. Nemeth, P. Barta, R. Szabo, and J. Biro. Network internal traffic characterization and end-to-end delay bound calculus for generalized processor sharing scheduling discipline. *Computer Networks*, 48(6):910–940, 2005.

[22] K. Pandit, C. Kirchner, J. Schmitt, and R. Steinmetz. A transform for network calculus and its application to multimedia networking. In *Proceddings of SPIE's Multimedia Computing and Networking Conference 2006 (MMCN'06)*, 2006.

[23] R. T. Rockfellar. *Convex Analysis*. Princeton University Press, 1996.

[24] J. B. Schmitt and U. Roedig. Sensor network calculus: A framework for worst case analysis. In *Proceedings of 1st International Conference on Distributed Computing in Sensor Systems*, 2005.

[25] J. B. Schmitt and F. A. Zdarsky. The disco network calculator: a toolbox for worst case analysis. In *Proceedings of Valuetools'06*, 2006.

[26] J. B. Schmitt, F. A. Zdarsky, and M. Fidler. Delay bounds under arbitrary multiplexing. Technical report, University of Kaiserslautern, 2007.

[27] J. B. Schmitt, F. A. Zdarsky, and I. Martinovic. Performance bounds in feed-forward networks under blind multiplexing. Technical Report 349/06, University of Kaiserslautern, Germany, 2006.

[28] J. B. Schmitt, F. A. Zdarsky, and U. Roedig. Sensor network calculus with multiple sinks. In *Proceedings of IFIP Networking'2006*, 2006.

[29] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

[30] L. Thiele, S. Chakraborty, M. Gries, A. Maxiaguine, and J. Greutert. Embedded software in network processors models and algorithms. In *Proceedings of Embedded Software Workshop EMSOFT'2001*, 2001.

[31] H. Touchette. Legendre-fenchel transforms in a nutshell. Technical report, School of Mathematical Sciences, University of London, 2005.