Improving Cross-Traffic Bounds in Feed-Forward Networks – There is a Job for Everyone

Steffen Bondorf and Jens Schmitt

Distributed Computer Systems (DISCO) Lab,
University of Kaiserslautern, Germany

Abstract Network calculus provides a mathematical framework for deterministically bounding backlog and delay in packet-switched networks. The analysis is compositional and proceeds in several steps. In the first step, a general feed-forward network is reduced to a tandem of servers lying on the path of the flow of interest. This requires to derive bounds on the cross-traffic for that flow. Tight bounds on cross-traffic are crucial for the overall analysis to obtain tight performance bounds. In this paper, we contribute an improvement on this first bounding step in a network calculus analysis. This improvement is based on the so-called total flow analysis (TFA), which so far saw little usage as it is known to be inferior to other methods for the overall delay analysis. Yet, in this work we show that TFA actually can bring significant benefits in bounding the burstiness of cross-traffic. We investigate analytically and numerically when these benefits actually occur and show that they can be considerable with several flows' delays being improved by more than 40% compared to existing methods – thus giving TFA's existence a purpose finally.

1 Introduction

Network Calculus (NC) is a versatile methodology for queueing analysis of resource sharing systems. The high modeling power of NC has been transposed into several important applications for network engineering problems, traditionally in the Internet's Quality of Service proposals IntServ and DiffServ, and more recently in diverse environments such as wireless sensor networks [17], switched Ethernets [12], data centers [19], or System-on-Chip [15].

A network calculus analysis requires a feed-forward network in order to avoid cyclic dependencies between flows and thus be able to compute flow characteristics inside the network. In fact, the typical first step in a NC analysis, given a flow of interest (foi), is to reduce the feed-forward network to a tandem consisting of the servers on the foi's path. To that end, arrival constraints of the foi's cross-traffic burstiness and rate have to be computed. Accurate burstiness constraints are indeed crucial for the subsequent tandem analysis to achieve accurate end-to-end performance bounds. As we discuss in Section 2, much research has been invested in tightening the tandem analysis, silently assuming that the reduction step from the feed-forward network to the tandem had already been performed. However, this step becomes very important for the quality of the bounds in

larger feed-forward networks. Consequently, we deal with this reduction step in our work and present a method to tighten the bounds on the burstiness of cross-traffic. Somewhat surprisingly, we achieve this by applying the so-called total flow analysis (TFA) to compute bounds on the server backlog just before the analyzed flow's path. This is surprising because the TFA has a "bad reputation" as an overall analysis method. This is due to its inferior results when bounding a foi's end-to-end performance metrics since it cannot exploit the pay burst only once phenomenon (PBOO, see Section 3).

The beneficial effect of our burstiness bounding step is based on the following basic, intuitive insight: At the output of a server, any combination of flows can be at most as bursty as the maximum data backlog at this server. Based on this insight we formally prove how to characterize the output of a flow by its input arrival curve and the server backlog bound. The new burstiness bound can be exploited to potentially reduce cross-traffic arrival bounds that were computed conventionally with the (min,+)-deconvolution.

In fact, as we discuss below, this does not always lead to improved bounds, yet it works from certain utilizations onwards and can be considerable. The reasons why TFA can help here become clear in our detailed treatment below, but here is an intuition: TFA's aggregate (total) perspective avoids making too many assumptions on the relative priorities between flows. In contrast, the conventional method does so by separating cross-traffic flows from each other.

In short, we contribute a new method to compute arrival bounds for cross-traffic on a foi's path. It is based on backlog bounds from TFA. The rest of this paper is structured as follows: In Section 2 we discuss related work. Section 3 provides the necessary background and notation on feed-forward analysis with NC. The alternative way to calculate the output bound of a traffic flow is presented and proved in Section 4. Next, the rationale behind the new burstiness bounding procedure in feed-forward networks as well as a detailed discussion on the conditions when it can improve the existing methods is presented in Section 5. Results from numerical evaluation concerning larger feed-forward networks are reported in Section 6, before the paper is concluded in Section 7.

2 Related Work

As mentioned above, most work in network calculus focused on the second step in a feed-forward network analysis, where the problem has already been reduced to a tandem. There is a whole evolution from simple, but conservative methods to sophisticated, tight analyses which can be very involved computationally (see [6,11] for recent overviews).

However, the first step of the feed-forward network analysis, bounding the cross-traffic burstiness, has so far been largely neglected. Most work starts directly with the tandem analysis or suggests to use straightforward techniques from basic NC results (more details are given in Section 3). An exception can be found in [10], where, for a single node under arbitrary multiplexing of several flows, tight output descriptions are derived for a single flow. However, when

targeting a feed-forward network, we need to bound cross-flows that may have traversed several servers with potentially many other flows joining and leaving it. Hence, much more work is needed here.

In previous work of ours, we already addressed the cross-traffic arrival bounding. In [4], we focused on algorithmic efficiency and targeted a distributed execution of the analysis. In [5], we achieved more accurate bounds by improving the overall cross-traffic arrival bounding procedure. The results of this paper allow to further improve these bounds.

3 Network Calculus Background

Data Arrivals and Forwarding Service

Flows are characterized by functions cumulatively counting their data. They belong to the set $\mathcal{F}_0$ of non-negative, wide-sense increasing functions:

$$\mathcal{F}_0 = \left\{ f : \mathbb{R} \to \mathbb{R}_\infty^+ \mid f(0) = 0,\ \forall s \le t\ :\ f(s) \le f(t) \right\},\ \mathbb{R}_\infty^+ \coloneqq [0, +\infty) \cup \{+\infty\}\ .$$

We are particularly interested in the functions $A(t)$ and $A'(t)$ cumulatively counting a flow's data put into a server s and put out from s, both until time t. These functions allow for simple definitions of performance measures.

Definition 1. (Backlog and Delay) Assume a flow with input function A traverses a system $\mathcal{S}$ and results in the output function A'. The *backlog* of the flow at time t is defined as

$$B(t) = A(t) - A'(t).$$

The *(virtual) delay* for a data unit arriving at $\mathcal{S}$ at time t is defined as

$$D(t) = \inf \{\tau \ge 0 \mid A(t) \le A'(t + \tau)\}.$$

Note, that the order of data within the flow needs to be retained for the (virtual) delay calculation [16].

NC operates in the interval time domain, i.e., its functions of $\mathcal{F}_0$ bound the maximum data arrivals of a flow during any duration of length d.

Definition 2. *(Arrival Curve)* Given a flow with input A, a function $\alpha \in \mathcal{F}_0$ is an arrival curve for A iff

$$\forall t\, \forall d,\ 0 \le d \le t\ :\ A(t) - A(t-d) \le \alpha(d).$$

For example, sensors reporting measurement values may generate packets of size b that are periodically sent with a minimum inter-arrival time t_δ. Then, the data flow they generate has a maximum data arrival rate of $r = \frac{b}{t_\delta}$ in the fluid model of $\mathcal{F}_0$. The resulting shape of the arrival curve is commonly referred to as token bucket and belongs to the class $\mathcal{F}_{\mathrm{TB}} \subset \mathcal{F}_0$:

$$\mathcal{F}_{\mathrm{TB}} = \{\gamma_{r,b} \mid \gamma_{r,b}(0) = 0, \forall d > 0\ :\ \gamma_{r,b}(d) = b + r \cdot d\}.$$

Scheduling and buffering leading to the output function $A'(t)$ depend on a server's forwarding. It is lower bounded in interval time as well.

Definition 3. *(Service Curve)* If the service provided by a server s for a given input A results in an output A', then s offers a service curve $\beta \in \mathcal{F}_0$ iff

$$\forall t : A'(t) \geq \inf_{0 \leq d \leq t} \{A(t-d) + \beta(d)\}.$$

For example, TDMA channel access [13], duty cycling sensor nodes [2], as well as the service offered by Ethernet connections [12] can be modeled with so-called rate-latency service curves $\mathcal{F}_{\mathrm{RL}} \subset \mathcal{F}_0$:

$$\mathcal{F}_{\mathrm{RL}} = \{\beta_{R,T} \,|\, \beta_{R,T}(d) = \max\{0, R \cdot (d-T)\}.$$

A number of servers fulfill a stricter definition of service curves that guarantees a higher output during periods of queued data, the so-called backlogged periods of a server.

Definition 4. *(Strict Service Curve)* Let $\beta \in \mathcal{F}_0$. Server s offers a strict service curve β to a flow iff, during any backlogged period of duration d, the output of the flow is at least equal to $\beta(d)$.

The Network

In general, networks are modeled as graphs where a node represents a network device like a router or a switch. Devices can have multiple inputs and multiple outputs to connect to other devices. This network model does not fit well with NC' server model for queueing analysis. NC therefore analyzes so-called *server graphs*. Assuming that a network device's input buffer is served at line speed, queueing effects manifest at the output buffers. These are modeled by the graph's servers. For instance, in wireless sensor networks, nodes usually possess a single transmitter. Thus, one sensor node corresponds to one server and the transmission range defines the server graph's links [2,4].

(min,+)-Operations

Network calculus [8,9] was cast in a $(\min, +)$-algebraic framework in [14,7]. The following operations allow to manipulate arrival and service curves while retaining their worst-case semantic.

Definition 5. *((min,+)-Operations)* The (min,+) aggregation, convolution and deconvolution of two functions $f, g \in \mathcal{F}_0$ are defined as

$$\begin{aligned} \textit{aggregation:}\ (f+g)(t) &= f(t) + g(t), \\ \textit{convolution:}\ (f \otimes g)(t) &= \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\}, \\ \textit{deconvolution:}\ (f \oslash g)(t) &= \sup_{u \geq 0} \{f(t+u) - g(u)\}. \end{aligned}$$

The service curve definition then translates to $A' \geq A \otimes \beta$, the arrival curve definition to $A \otimes \alpha \geq A$, and performance characteristics can be bounded with the deconvolution $\alpha \oslash \beta$:

Quantifier	Definition
$\mathbb{F}$	Generic notation for a flow aggregate
$\{f_n, ..., f_m\}$	Flow aggregate containing flows $f_n, ..., f_m$
$\langle s_x, \ldots, s_y \rangle$	Tandem of consecutive servers s_x to s_y
α^f, $\alpha^{\mathbb{F}}$	Arrival curve of flow f, set of flows $\mathbb{F}$
α_s^f, $\alpha_s^{\mathbb{F}}$	Arrival bound at server s
β_s	Service curve of server s
$\beta^{\text{l.o.}f}$, $\beta^{\text{l.o.}\mathbb{F}}$	Left-over service curve

Table 1. Network calculus notation for flows, arrivals and service.

Theorem 1. (Performance Bounds) *Consider a server s that offers a service curve β. Assume a flow (aggregate) with arrival curve α traverses the server. Then we obtain the following performance bounds for the flow:*

$$\begin{aligned} \textit{delay: } & \forall t \in \mathbb{R}^+ : \ D(t) \leq \inf\{d \geq 0 \,|(\alpha \oslash \beta)(-d) \leq 0\} =: h(\alpha, \beta), \\ \textit{backlog: } & \forall t \in \mathbb{R}^+ : \ B(t) \leq (\alpha \oslash \beta)(0) =: v(\alpha, \beta), \\ \textit{output: } & \forall d \in \mathbb{R}^+ : \ \alpha'(d) = (\alpha \oslash \beta)(d), \end{aligned}$$

where the delay and backlog bounds are abbreviated by D and B, respectively, as they hold independent of parameter t and α' is an arrival curve for A'.

The delay bound equals the horizontal deviation between α and β, $h(\alpha, \beta)$. In case the arrival curve belongs to a single flow, the order of data within this flow must be retained (FIFO per µFlow property [16]). In case α belongs to a flow aggregate, FIFO multiplexing between the aggregated flows is additionally required (cf. Definition 1). In contrast, for the backlog bound, i.e., the vertical deviation $v(\alpha, \beta)$, no FIFO assumptions are required.

Analyzing a flow in an end-to-end fashion while considering cross-traffic on its path is enabled by the following theorems. Table 1 provides the notation required to analyze such a path tandem of servers.

Theorem 2. (Concatenation of Servers) *Consider a flow (aggregate) $\mathbb{F}$ crossing a tandem of servers $\langle s_1, \ldots, s_n \rangle$ and assume that each s_i, $i \in \{1, \ldots, n\}$, offers a service curve β_{s_i}. The overall service curve offered to $\mathbb{F}$ is their concatenation*

$$\beta_{s_1} \otimes \ldots \otimes \beta_{s_n} = \bigotimes_{i=1}^{n} \beta_{s_i}$$

Theorem 3. *(Left-Over Service Curve) Consider a server s that offers a strict service curve β_s. Let s be crossed by two flow aggregates $\mathbb{F}_0$ and $\mathbb{F}_1$ with aggregate arrival curves $\alpha^{\mathbb{F}_0}$ and $\alpha^{\mathbb{F}_1}$, respectively. Then $\mathbb{F}_1$'s worst-case residual resource share under arbitrary multiplexing at s, i.e., its left-over service curve at s, is*

$$\beta_s^{\text{l.o.}\mathbb{F}_1} = \beta_s \ominus \alpha^{\mathbb{F}_0}$$

with $(\beta \ominus \alpha)(d) \coloneqq \sup\{0 \leq u \leq d \,|\, (\beta - \alpha)(u)\}$ denoting the non-decreasing upper closure of $(\beta - \alpha)(d)$.

Network Analysis

A network calculus analysis computes the end-to-end delay bound for a specific flow (flow of interest, foi). Conceptually, algebraic NC is compositional and its feed-forward analyses proceed in two steps [3,4]:

1. First, the analysis abstracts from the feed-forward network to the flow of interest's path (a tandem of servers). This step is enabled by recursively decomposing the server graph into tandems [5] and bounding the output arrivals of cross-traffic with Theorem 1, the output bound. After this step, a bound on the worst-case shape of cross-flows is known at the location of interference with the foi. Then, the following step need not consider the part of the network traversed by cross-flows nor the potentially complex interference patterns they are subject to.
2. The foi's end-to-end delay bound in the feed-forward network can now be calculated with a less complex *tandem analysis*. The foi's end-to-end left-over service curve is derived and the delay bound computed.

The second step of the feed-forward analysis (FFA) procedure has seen much treatment in the literature. Effort constantly focused on improving the ability to capture flow scheduling and cross-traffic multiplexing effects and thus provide more accurate delay bounds. One of the earliest improvements was made with the step from the total flow analysis to the separate flow analysis.

Total Flow Analysis (TFA) [9]: The Total Flow Analysis directly applies the basic results from Theorem 1. Given the arrival curve for the totality of flows (a flow aggregate) present at a server and the server's service curve, TFA allows to derive deterministic worst-case bounds on the delay a flow (aggregate) experiences when crossing the analyzed server as well as the server's buffer requirement for handling all traffic without suffering from overflows. The backlog bound coincides with the total buffer demand of a server. The TFA is a server-local analysis, i.e., all bounds it derives hold for a specific server and the totality of traffic crossing it, not for a single flow of interest because flows are not analyzed individually. When TFA is used as a tandem analysis in FFA-step 2 of the above scheme, the flow of interest's end-to-end delay bound is computed by summing up the server-local delay bounds on its path.

The Separate Flow Analysis (SFA) and the PBOO-effect [14]: The TFA delay bound can be improved by separating the analysis' flow of interest from its cross-traffic. In this preparatory step, the so-called left-over service curve calculation, cross-traffic arrivals are subtracted from the service curves in the foi's path. The SFA is a straight-forward, hop-by-hop application of Theorems 3 and 2: First subtract cross-traffic arrivals such that βs become $\beta^{\text{l.o.foi}}$s and then concatenate the left-over service curves. Deriving the delay bound with a single, end-to-end left-over service curve considers the flow of interest's burst term only once. This effect is therefore called Pay Bursts Only Once (PBOO).

Note, that TFA and SFA both define the procedure for FFA-step 2 only. In the first step of the feed-forward analysis procedure, only flows that eventually interfere with the flow of interest are considered – cross-traffic arrival bounding is

therefore limited to these flows. They are separated from their own cross-traffic and bounded in an aggregate fashion. The former defines the difference to the TFA backlog bounding where all flows at a server are considered, regardless their subsequent hop [14]. The latter defines the *aggregate PBOO Arrival Bounding* (PBOO-AB) [5]. Thus, both approaches incorporate different degrees of flow aggregation. We exploit a combination of both, yet without explicitly tracing them throughout the entire arrival bounding [**?**] but with the TFA's additional benefits for bounding a server's output burstiness.

4 An Alternative Output Bound

In this section, we derive an alternative output bound. As presented in Section 5 and numerically evaluated in Section 6, this alternative output bound enables an improved arrival bounding step (FFA-step 1).

Let A, A' be input and output to/from a system. We assume to have an arrival curve α for the arrivals A and a service curve β offered by the system. Let us further assume that the arrival curve α is such that for $d > 0$ it can be written as

$$\alpha(d) = \tilde{\alpha}(d) + \alpha(0^+),$$

with $\tilde{\alpha}$ being a concave function (defined for $d > 0$ by the above equation and with $\tilde{\alpha}(0) = 0$), and $\alpha(0^+) = \lim_{d \to 0^+} \alpha(d)$. Clearly, this means that α is also a concave function. Further note that, for instance, any concave piecewise-linear arrival curve meets this condition, hence it is not restrictive in practice (e.g., the Disco Deterministic Network Calculator, DiscoDNC, uses such functions as arrival curves [3]). As $\tilde{\alpha} \in \mathcal{F}_0$ and is concave, it is also sub-additive, which is crucial as we see below.

Noting that we can bound the backlog for any given arrival process A by

$$B(t) = A(t) - A'(t) \leq A(t) - (A \otimes \beta)(t) = \sup_{0 \leq u \leq t} \{A(t) - A(u) - \beta(t-u)\},$$

we provide the alternative output bound in the following theorem.

Theorem 4. *Under the above assumptions and notations, an output bound on the departure flow (aggregate) A' can be calculated as*

$$\alpha'(d) = \alpha(d) + (v(\alpha, \beta) - \alpha(0^+)) \cdot 1_{\{d>0\}}.$$

Proof. Let $s < t$:

$$\begin{aligned}
A'(t) - A'(s) &= A(t) - A(s) + B(s) - B(t) \\
&\leq A(t) - A(s) + B(s) \\
&\leq A(t) - A(s) + \sup_{0 \leq u \leq s} \{A(s) - A(u) - \beta(s-u)\} \\
&= \sup_{0 \leq u \leq s} \{A(t) - A(u) - \beta(s-u)\} \\
&\leq \sup_{0 \leq u \leq s} \{\alpha(t-u) - \beta(s-u)\}
\end{aligned}$$

$$
\begin{aligned}
&= \sup_{0\le u\le s}\{\tilde{\alpha}(t-u)+\alpha(0^+)-\beta(s-u)\}\\
&\le \sup_{0\le u\le s}\{\tilde{\alpha}(t-s)+\tilde{\alpha}(s-u)+\alpha(0^+)-\beta(s-u)\}\\
&= \tilde{\alpha}(t-s)+\sup_{0\le u\le s}\{\alpha(s-u)-\beta(s-u)\}\\
&\le \tilde{\alpha}(t-s)+v(\alpha,\beta)\\
&= \alpha(t-s)+v(\alpha,\beta)-\alpha(0^+)=\alpha'(t-s).
\end{aligned}
$$

For $s=t: A'(t)-A'(s)=0=\alpha'(t-s)$. q.e.d.

Note that this result resembles a known basic result that can be found in Chang's textbook in Lemma 1.4.2 [7]. This lemma states that for a server with a bound on the queue $\bar{q}$ and a $\gamma_{r,b}$-constrained input, an output bound can be given as $\gamma_{r,b+\bar{q}}$. Besides generalizing this lemma, we point out that we actually improve it, as we basically get rid of the burst term and would obtain $\gamma_{r,\bar{q}}$ as an output bound under Chang's assumptions.

5 TFA-assisted PBOO Arrival Bounding

In this section, we demonstrate how to exploit the basic insight about the alternative output characterization from the previous section. It gives us the choice between the existing PBOO arrival bounding (PBOO-AB), which applies the conventional output bound, and an approach where we use a backlog bound for the cross-traffic and apply Theorem 4. This backlog bound is obtained from TFA, i.e., it actually considers flows that demultiplex from cross-traffic and do not interfere with the foi. In the following we discuss why and when this can actually lead to an improvement.

Consider the network configuration of Figure 1 where f is the flow of interest, xf is its cross-flow and xxf is the cross-traffic of xf. Although the network is depicted as a tandem, we cannot apply a simple tandem analysis because the flow of interest f does not cross all servers, i.e., cross-traffic arrival bounding is necessary in this network: Deriving f's performance bounds with the SFA requires bounding xf's arrival at s_2, $\alpha^{xf}_{s_2}$, (FFA-step 1) with PBOO-AB first. It's result is used to separate f by computing f's left-over service curve at s_2 that is then used to derive f's delay bound (FFA-step 2).

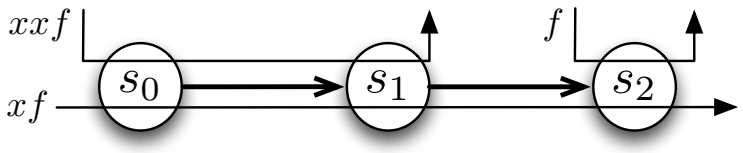

Figure 1. Sample network.

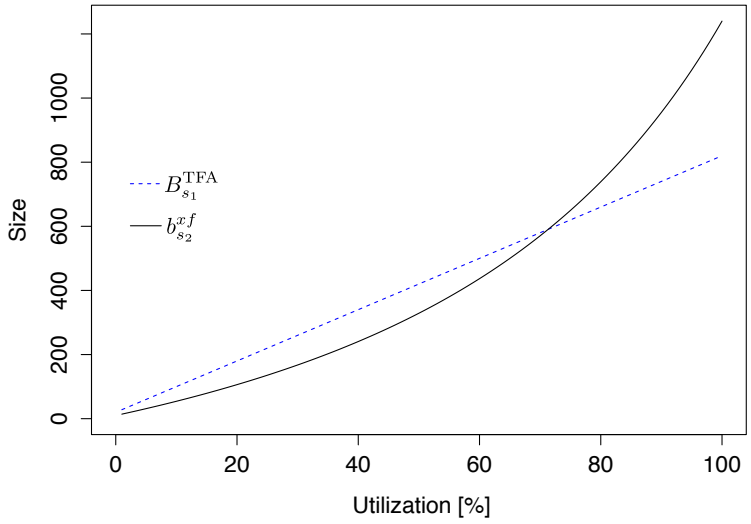

Figure 2. Different scaling behaviors of $B_{s_1}^{\mathrm{TFA}}$ and $b_{s_2}^{xf}$ with respect to the network utilization.

PBOO-AB retains the worst-case when arbitrarily multiplexing of flows, i.e., in contrast to FIFO multiplexing, data of xf may always be served after xxf's data – independent of their relative arrival times. Thus, burstiness of $\alpha_{s_2}^{xf}$, denoted by $b_{s_2}^{xf} := \alpha_{s_2}^{xf}(0^+)$, increases when more data of xxf arrives in shorter intervals, i.e., its arrival curve α^{xxf} increases. In our illustrative numerical evaluation of this section, service curves are chosen to be rate latency functions $\beta_{R,T} = \beta_{20,20}$ and arrival curves to be token buckets $\alpha = \gamma_{r,10}$ where the rate r is variable. In this parameterized homogeneous setting, α^{xxf} increases with parameter r that we use to show xf's worst-case burstiness increase with a growing network utilization.

Figure 2 shows the utilization's impact on the PBOO-AB burstiness of f's cross-traffic, $b_{s_2}^{xf}$, and on the TFA backlog bound at server s_1, $B_{s_1}^{\mathrm{TFA}}$. TFA considers all flows at s_1 and derives the backlog bound based on their aggregate arrival curve. Being the backlog of all incoming traffic at the server, i.e., a superset of f's cross-traffic xf, $B_{s_1}^{\mathrm{TFA}}$ is also a backlog bound for xf. In Figure 2, $B_{s_1}^{\mathrm{TFA}}$ scales linearly whereas $b_{s_2}^{xf}$ scales super-linearly with the utilization. Consequently, both curves intersect and $b_{s_2}^{xf}$ exceeds $B_{s_1}^{\mathrm{TFA}}$, such that using the TFA backlog bound and Theorem 4 indeed achieves an improvement over PBOO-AB.

This can be explained by the derivation of the two values, $B_{s_1}^{\mathrm{TFA}}$ and $b_{s_2}^{xf}$. For detailed information on how to compute the result of $(\min, +)$-operations for token-bucket arrival curves and rate-latency service curves, please refer to the DiscoDNC documentation [1].

$$
\begin{aligned}
b_{s_2}^{xf} &= \left(\alpha^{xf} \oslash \beta_{\langle s_0, s_1 \rangle}^{\mathrm{l.o.}xf}\right)(0) \\
&= \left(\alpha^{xf} \oslash \left(\beta_{s_0}^{\mathrm{l.o.}xf} \otimes \beta_{s_1}^{\mathrm{l.o.}xf}\right)\right)(0)
\end{aligned}
$$

$$
\begin{aligned}
&= \left(\alpha^{xf} \oslash \left(\left(\beta_{s_0} \ominus \alpha_{s_0}^{xxf}\right) \otimes \left(\beta_{s_1} \ominus \alpha_{s_1}^{xxf}\right)\right)\right)(0) \\
&= \left(\alpha^{xf} \oslash \left(\left(\beta_{s_0} \ominus \alpha^{xxf}\right) \otimes \left(\beta_{s_1} \ominus \left(\alpha_{s_0}^{xxf} \oslash \beta_{s_0}^{\text{l.o.}xxf}\right)\right)\right)\right)(0) \\
&= \left(\alpha^{xf} \oslash \left(\left(\beta_{s_0} \ominus \alpha^{xxf}\right) \otimes \left(\beta_{s_1} \ominus \left(\alpha^{xxf} \oslash \left(\beta_{s_0} \ominus \alpha_{s_0}^{xf}\right)\right)\right)\right)\right)(0) \\
&= \left(\gamma_{r,10} \oslash \left(\left(\beta_{20,20} \ominus \gamma_{r,10}\right) \otimes \left(\beta_{20,20} \ominus \left(\gamma_{r,10} \oslash \left(\beta_{20,20} \ominus \gamma_{r,10}\right)\right)\right)\right)\right)(0) \\
&\overset{(1)}{=} \left(\gamma_{r,10} \oslash \left(\beta_{20-r,\frac{410}{20-r}} \otimes \left(\beta_{20,20} \ominus \left(\gamma_{r,10} \oslash \beta_{20-r,\frac{410}{20-r}}\right)\right)\right)\right)(0) \\
&\overset{(2)}{=} \left(\gamma_{r,10} \oslash \left(\beta_{20-r,\frac{410}{20-r}} \otimes \left(\beta_{20,20} \ominus \gamma_{r,\frac{410r}{20-r}+10}\right)\right)\right)(0) \\
&\overset{(3)}{=} \left(\gamma_{r,10} \oslash \left(\beta_{20-r,\frac{410}{20-r}} \otimes \left(\beta_{20,20} \ominus \gamma_{r,\frac{400r+200}{20-r}}\right)\right)\right)(0) \\
&\overset{(4)}{=} \left(\gamma_{r,10} \oslash \left(\beta_{20-r,\frac{410}{20-r}} \otimes \beta_{20-r,\frac{8200}{(20-r)^2}}\right)\right)(0) \\
&= \left(\gamma_{r,10} \oslash \beta_{20-r,\frac{410}{20-r}+\frac{8200}{(20-r)^2}}\right)(0) \\
&\overset{(5)}{=} \left(\gamma_{r,10} \oslash \beta_{20-r,\frac{16400-410r}{(20-r)^2}}\right)(0) \\
&\overset{(6)}{=} \frac{4000 + 16000r - 400r^2}{400 - 40r + r^2}
\end{aligned}
$$

We can see that $b_{s_2}^{xf}$ monotonically increases because the numerator is larger as well as faster growing than the denominator and the stability condition $r \leq 10$ leads to an always positive denominator.

Next, let us see how the polynomial expression's degree builds up during the above derivation. Multiplication by the arrival rate is required to compute the burstiness of an output arrival curve, i.e., every time we deconvolve – see steps from (1) to (2) and from (5) to (6). Subsequent left-over service curve operations, e.g., from (3) to (4), retain the rate in the latency term's denominator, as does the convolution of service curves in the step from (4) to (5). Deconvolution is required for output bounding and thus occurs at every level of the recursive arrival bounding procedure. In this example, xf is bounded in the first recursion level and it requires bounding xxf in a second level; hence, we obtain a rational function of degree 2 (with a pole at $r = 20$).

The TFA backlog bound derivation for server s_1 proceeds as follows:

$$
\begin{aligned}
B_{s_1}^{\text{TFA}} &\overset{(1)}{=} v\left(\left\{\alpha_{s_1}^{xf}, \alpha_{s_1}^{xxf}\right\}, \beta_{s_1}\right) \\
&= v\left(\left\{\alpha_{s_0}^{xf}, \alpha_{s_0}^{xxf}\right\} \oslash \beta_{s_0}^{\text{l.o.}\left\{\alpha_{s_1}^{xf}, \alpha_{s_1}^{xxf}\right\}}, \beta_{s_1}\right) \\
&\overset{(2)}{=} v\left(\left\{\alpha_{s_0}^{xf}, \alpha_{s_0}^{xxf}\right\} \oslash \beta_{s_0}, \beta_{s_1}\right) \\
&\overset{(3)}{=} v\left(\left(\gamma_{r,10} + \gamma_{r,10}\right) \oslash \beta_{20,20}, \beta_{20,20}\right) \\
&\overset{(4)}{=} v\left(\gamma_{2r,20} \oslash \beta_{20,20}, \beta_{20,20}\right) \\
&\overset{(5)}{=} v\left(\gamma_{2r,20+2r\cdot 20}, \beta_{20,20}\right) \\
&= 80r + 20
\end{aligned}
$$

The derivation takes advantage of aggregation in (1) and (3), which prevents recursive cross-traffic arrival bounding in our example. xxf is not considered cross-traffic of xf as both belong to the same flow aggregate and therefore no action has to be taken to derive the left-over service curve at s_0 in (2). The only relevant deconvolution in $B_{s_1}^{\text{TFA}}$'s derivation is found in the computation of the aggregate's output bound after crossing s_0. The deconvolution in the backlog bounding operation $v\left(\left\{\alpha_{s_1}^{xf}, \alpha_{s_1}^{xxf}\right\}, \beta_{s_1}\right)$ executed in the step from (4) to (5) is, in contrast to the $b_{s_2}^{xf}$-derivation not affecting the polynomial expression's degree because its latency is not depending on r. Thus, the entire term grows linearly with the flow arrival rate.

Remark 1. It is not possible to improve xf's output bound by using the backlog bound for flow xf at server s_1, i.e., $B_{s_1}^{xf}$, because $B_{s_1}^{xf}$ and $b_{s_2}^{xf}$ are equal due to [14], Theorem 3.1.12, Rule 12:

$$\begin{aligned} B_{s_1}^{xf} &= \left(\left(\alpha^{xf} \oslash \beta_{s_0}^{\text{l.o.}xf}\right) \oslash \beta_{s_1}^{\text{l.o.}xf}\right)(0) \\ &= \left(\alpha^{xf} \oslash \left(\beta_{s_0}^{\text{l.o.}xf} \otimes \beta_{s_1}^{\text{l.o.}xf}\right)\right)(0) \;=\; b_{s_2}^{xf} \end{aligned}$$

From this reformulated derivation of $b_{s_2}^{xf}$ we obtain another explanation for its function being of degree 1 in the above example: There is only one deconvolution.

Remark 2. Theorem 1.4.5 in Le Boudec and Thiran's text book [14] presents conditions for tight output arrival bounds. These are satisfied in both our derivations above, yet, we improve xf's output bound by incorporating $B_{s_1}^{\text{TFA}}$. At first glance, this may seem like a contradiction, however, we gain tightness from additional considerations of a feed-forward analysis that are not addressed in [14], Theorem 1.4.5. It remains valid, yet only with respect to the given service curves that, in turn, might be tightness-compromising left-overs like in Remark 1.

In a more complex feed-forward network, we often have multi-level recursions for cross-traffic of cross-traffic in the arrival bounding phase of the derivations [5] – also for the backlog bound at a server – and therefore polynomial expressions of higher degrees occur in both alternative bounds on the output burstiness. For the ease of presentation, we continue to illustrate the impact of the differing scaling behaviors as well as the service curve latency and the initial burstiness of flows in the simple network from Figure 1. In Section 6, we extend our evaluation to more involved feed-forward networks.

Above, we discussed that left-over service curve computations retain the arrival rate in their results' latency term. For instance, the left-over latency at server s_0 is $\frac{T_{s_0} \cdot R_{s_0} + b^{xxf}}{R_{s_0} - r^{xxf}} = T_{s_0} + \frac{r^{xxf} \cdot T_{s_0} + b^{xxf}}{R_{s_0} - r^{xxf}}$, i.e, it consists of a fixed and a variable part. The fixed part is defined by the service curves' initial latency $T_{s_0} = T_{s_1} = T_{s_2} =: T$ (equal for all servers in out homogeneous sample network) whose influence on the total burstiness we evaluate – increasing T naturally decreases the impact of the variable part containing the crucial factor r. We check $T = 0$, i.e., the natural lower limit of the latency, and $T = 10^6$, a value several orders of magnitude larger than the service rate $R = 20$ and thus safe to be

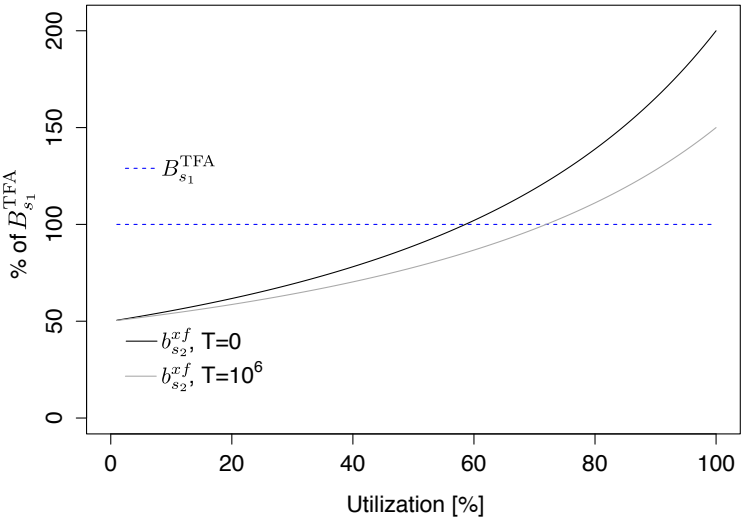

Figure 3. Relative difference: Influence of β's latency T.

assumed as a realistic upper bound on T. The resulting range of T's impact is depicted by the relative difference between $B^{\mathrm{TFA}}_{s_1}$ and $b^{xf}_{s_2}$ in Figure 3. Most notably, the network utilization required for the TFA backlog bound to outperform the separated flow's output burstiness is between 59% to 72% – that is, it always exists and resides at utilizations considerably lower than the network's capacity limit. Moreover, $b^{xf}_{s_2}$'s relative benefit of 50% over $B^{\mathrm{TFA}}_{s_1}$ for low utilizations is in fact small in absolute values (cf. Figure 2) whereas its disadvantage (right of the intersection) grows fast to become large in absolute numbers.

Last, we evaluate the impact of the remaining variable parameter besides utilization and the service curve latency: The initial burstiness of flows b in the homogeneous network. We reduced the service curve latency's influence by assigning $\beta = \beta_{20,0.1}$. Arrival curves are $\alpha = \gamma_{r,b}$ where r is defined by the network utilization (i.e., relative to the service rate R) and b is slowly increased from 0 to the previously used value of 10. Figure 4 depicts the relative difference between $B^{\mathrm{TFA}}_{s_1}$ and $b^{xf}_{s_2}$ for three levels of network utilization: 59% and 72% (the intersections of both values in the latency evaluation of Figure 3) as well as 100%. We can see that the TFA backlog bound at server s_1 is in fact always within the output burstiness of the same utilizations found for the latency – for 59%, $B^{\mathrm{TFA}}_{s_1}$ is an asymptote when increasing b, and for 72% the $b^{xf}_{s_2}$-value starts at the server backlog bound. The impact of initial burstiness of flows is similar to the latency's impact. For the maximum network utilization, $b^{xf}_{s_2}$ always exceeds $B^{\mathrm{TFA}}_{s_1}$ by at least 50% in our sample network, i.e., utilization remains most impactful.

Based on these observations, we propose to improve the arrival bound of a flow (aggregate) with the TFA backlog bound and Theorem 4 applied at the last hop of this flow (aggregate) – of course, only if it actually improves the bound. We call this new method: *TFA-assisted PBOO Arrival Bound*.

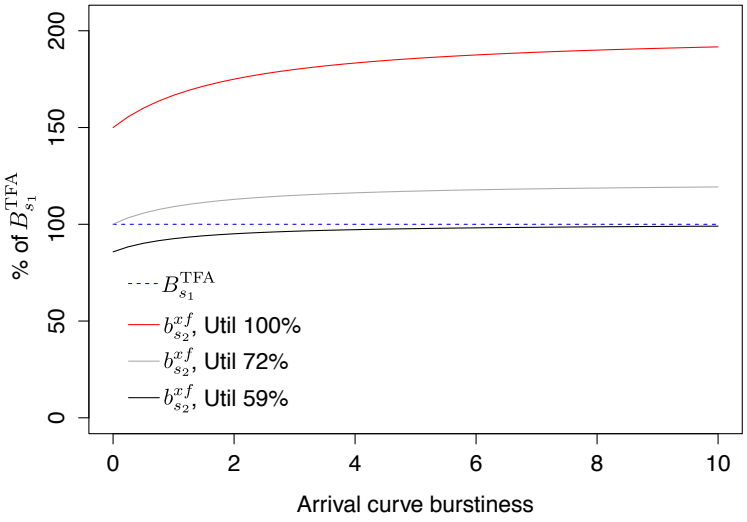

Figure 4. Relative difference: Influence of the arrival burstiness on the backlog bound at s_1 and xf's worst-case burstiness assumed at s_2.

6 Feed-Forward Network Evaluation

The potential improvement of cross-traffic bounds can be quite considerable in the small scenario of Section 5. Now we turn to the investigation of the impact on the end-to-end delay bound of flows traversing larger feed-forward networks. That is, we evaluate the improvements gained by reduced cross-traffic interference that ultimately tightens delay bounds. We have extended the Disco Deterministic Network Calculator (DiscoDNC) [3] with the TFA-assisted PBOO Arrival Bounding in order to benchmark the resulting new variant against the existing one without this improvement (plain PBOO-AB).

The exemplary network we generated for evaluation consists of 150 homogeneous servers with service curves $\beta_{R,T} = \beta_{200,0.1}$. 600 flows with random paths and arrival curve $\alpha = \gamma_{2,0.1}$ were added to the network. They are supposed to randomly generate hotspots of considerable, yet, uncontrolled utilization for the evaluation. These hotspots see the highest numbers of flows such that the impact of separation vs. aggregation can be observed – similar to heterogeneous networks where some flows outweigh others. We chose a small initial burstiness to additionally check the above claim that unavoidable burstiness increases are sufficient to cause impact of the TFA's assistance to the delay analysis.

The TFA-assisted PBOO-AB improved 369 out of 600 flow delay bounds over those derived with plain PBOO-AB (see Figure 5). In total, 61.5% of flows cross a hotspot that: 1) enables the TFA to aggregate flows such that its backlog bounding requires less recursion levels, making it grow slower with the utilization, and 2) has a utilization large enough to allow for its backlog bound to fall below the output bound burstiness. For the 33% of flows with largest delay bound

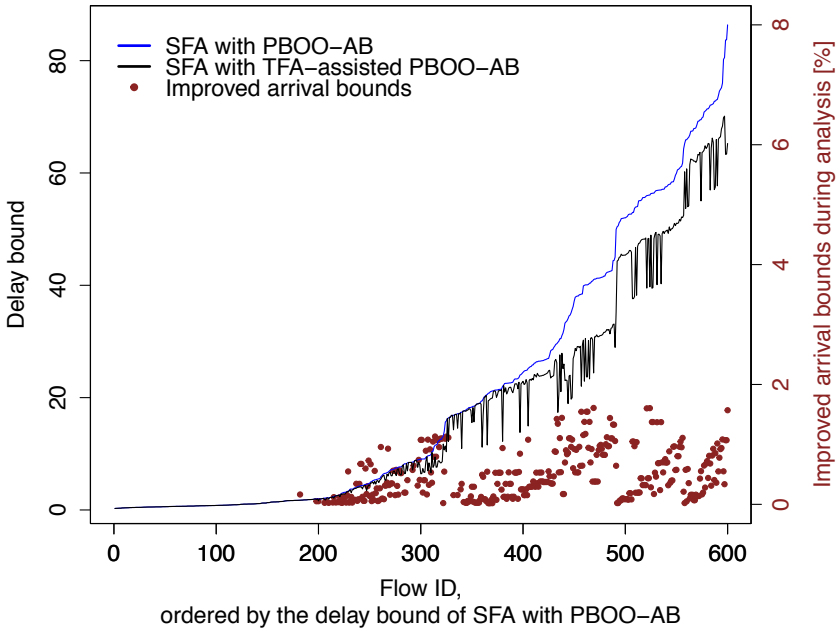

Figure 5. Delay analysis of a feed-forward network.

(using plain PBOO-AB), we achieved an average improvement of 17.93%, with a maximum improvement of 44.41%.

The distribution of brown dots for these rightmost 200 flows in Figure 5 shows that this improvement was achieved without ever capping more than 2% of the arrival bounds derived during the entire feed-forward analysis (right y-axis). Moreover, it is clearly visible that an increased share of burstiness improvements causes a larger delay bound reduction. For the rightmost 200 flows in Figure 5, the dots form a pattern of three "peaks" whose beginning and end both demarcate a step in the improved delay bounds depicted above them.

Another interesting observation is that these distinguishable peaks in improved worst-case burstiness cause a non-uniform decrease of delay bounds. The global network delay bound – the maximum delay bound of all flows in the network – is not defined by the same flow anymore. Applying our new analysis, 11 flows that had a smaller delay bound than this flow now have a larger one. This reordering indicates that even when delay bounds are just used as a relative figure of merit, such as in design space explorations [18], an accurate network delay analysis is important and the first step of the FFA procedure is crucial.

7 Conclusion

In network calculus, the Total Flow Analysis (TFA) had been abandoned since it is inferior to other methods for overall network delay analysis. In this paper, we

demonstrate that the TFA can actually be very useful to improve the bounding of cross-traffic arrivals in a feed-forward network. The trick is to use TFA's backlog bound as an upper bound on the burstiness at the servers where cross-traffic joins the analyzed flow of interest. We showed that the improvement can be quite significant, with some delay bounds reduced by more than 40%. So, we see: There is a job for everyone!

References

1. The Disco Deterministic Network Calculator. Available online: http://disco.cs.uni-kl.de/index.php/projects/disco-dnc.
2. S. Bondorf and J. Schmitt. Statistical Response Time Bounds in Randomly Deployed Wireless Sensor Networks. In *Proc. IEEE LCN*, 2010.
3. S. Bondorf and J. Schmitt. The DiscoDNC v2 – A Comprehensive Tool for Deterministic Network Calculus. In *Proc. ValueTools*, 2014.
4. S. Bondorf and J. Schmitt. Boosting Sensor Network Calculus by Thoroughly Bounding Cross-Traffic. In *Proc. IEEE INFOCOM*, 2015.
5. S. Bondorf and J. Schmitt. Calculating Accurate End-to-End Delay Bounds – You Better Know Your Cross-Traffic. In *Proc. ValueTools*, 2015.
6. A. Bouillard. Algorithms and efficiency of Network calculus. Habilitation thesis, École Normale Supérieure, 2014.
7. C.-S. Chang. *Performance Guarantees in Communication Networks.* Springer, 2000.
8. R. L. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions on Information Theory*, 1991.
9. R. L. Cruz. A Calculus for Network Delay, Part II: Network Analysis. *IEEE Transactions on Information Theory*, 1991.
10. J. Echagüe and V. Cholvi. Tight Arrival Curve at the Output of a Work-Conserving Blind Multiplexing Server. *Informatica*, 2010.
11. M. Fidler. Survey of Deterministic and Stochastic Service Curve Models in the Network Calculus. *Communications Surveys Tutorials*, 2010.
12. F. Frances, C. Fraboul, and J. Grieu. Using Network Calculus to Optimize AFDX Network. In *Proc. ERTS*, 2006.
13. N. Gollan and J. Schmitt. Energy-Efficient TDMA Design Under Real-Time Constraints in Wireless Sensor Networks. In *Proc. IEEE/ACM MASCOTS*, 2007.
14. J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet.* Springer, 2001.
15. A. Maxiaguine, S. Kunzli, S. Chakraborty, and L. Thiele. Rate Analysis for Streaming Applications with On-chip Buffer Constraints. In *Proc. ASP-DAC*, 2004.
16. J. Schmitt, N. Gollan, S. Bondorf, and I. Martinovic. Pay Bursts Only Once Holds for (Some) non-FIFO Systems. In *Proc. IEEE INFOCOM*, April 2011.
17. J. Schmitt, F. Zdarsky, and L. Thiele. A Comprehensive Worst-Case Calculus for Wireless Sensor Networks with In-Network Processing. In *Proc. IEEE RTSS*, 2007.
18. L. Thiele, S. Chakraborty, M. Gries, and S. Künzli. Design Space Exploration of Network Processor Architectures. In *In Network Processor Design: Issues and Practices, Volume 1*, pages 30–41. Morgan Kaufmann Publishers, 2002.
19. T. Zhu, A. Tumanov, M. Kozuch, M. Harchol-Balter, and G. Ganger. PriorityMeister: Tail Latency QoS for Shared Networked Storage. In *Proc. ACM SOCC*, 2014.