Should Network Calculus Relocate? An Assessment of Current Algebraic and Optimization-based Analyses

Steffen Bondorf and Jens B. Schmitt

Distributed Computer Systems (DISCO) Lab,
University of Kaiserslautern, Germany
{bondorf,jschmitt}@cs.uni-kl.de

Abstract Network calculus (NC) offers a framework for worst-case analysis of queueing networks. It enables to derive deterministic bounds on flow delay and server backlog. The continuous evolution of NC led to a set of different analyses. In fact, it even resulted in two entirely different branches of the methodology. Both start with a common network description based on bounding functions on flow arrivals and forwarding service. Anything that follows, i.e., the actual analysis leading to a worst-case performance bound, vastly differs. For long, there was only the algebraic NC, the formalism created as a system theory for communication networks. It matured and eventually seemed to have reached its limits regarding the accuracy of bounds. The problems preventing it from attaining tight bounds in feed-forward networks were overcome with optimization-based analysis. However, this approach was proven NP-hard without an efficient analysis algorithm known for it. Therefore, it was proposed to confine to a less complex optimization-based analysis instead. Like algebraic NC analyses, it derives tight bounds for some networks and valid bounds with varying accuracy for other networks. In this paper, we investigate the consequences of this tradeoff and identify a new and crucial analysis principle that allows us to compare both NC branches more comprehensively than simply ranking delay bounds.

Keywords: Network Calculus. Algebraic Analysis. Optimization.

1 Introduction

Network calculus (NC) is a methodology for the worst-case analysis of queueing systems. It provides different analysis procedures to derive deterministic bounds on buffer requirements and flow delays. Thus, it can be employed in the verification of real-time systems. In fact, *algebraic* network calculus has seen application in avionics [8,11,12]. E.g., the Airbus A380's backbone AFDX network (Avionics Full-Duplex Ethernet) has been certified using network calculus. Additionally, tool support is available for algebraic network calculus: open-source [2], commercial [9] as well as an internal tool of a company manufacturing AFDX switches (Rockwell Collins: ConfGen [13]) and others (Hirschmann Automation

[17], SIEMENS [14]). Moreover, algebraic NC has continuously seen improvements, e.g., w.r.t. the procedure of a feed-forward analysis and the computational effort [3,4], as well as attainable features [3] and accuracy of results [4,5].

All these developments took place despite the introduction of an entirely different analysis approach based on the NC system description: optimization-based network calculus. Initially developed by [19] to prove a property of algebraic NC analysis that can inhibit deriving tight bounds, it was further developed into an alternative feed-forward analysis, the LP analysis, in [7]. The LP analysis is able to derive tight bounds, yet, it is also NP-hard with no efficient algorithm known for it. As this insight obviously prevents network calculus to relocate to optimization, an accurate (not necessarily tight) ULP analysis was proposed based on the new optimization approach. It is of course not NP-hard, however, neither was it benchmarked comprehensively against the current algebraic NC analyses. Therefore, the question of switching the fundamental analysis approach of network calculus has not been answered yet.

In this paper, we derive a new analysis principle for feed-forward networks – similar to the existing principles – that pinpoints the problem of algebraic NC, a problem theoretically solved with the LP analysis of [7]. With this principle at hand, we can provide an in-depth NC analysis evaluation. This treatment of NC also allows us to reveal the weaknesses of the ULP. We comprehensively compare it to current algebraic network calculus analyses, foremost the PMOO analysis and its extensions, and finally provide an evaluation that gives insight on the gap between both NC branches. Our work helps to better assess the severity of the shortfall of current algebraic NC and thus tool implementations. Moreover, our insights provide a guide to future work w.r.t. improving algebraic NC accuracy.

The remainder of this paper is structured as follows: Section 2 presents some background on NC: The system description common to both branches as well as the algebraic operations. In Section 3, we examine the two NC branches regarding their approaches, strengths and weaknesses when analyzing a network. This allows us to assess accuracy of the currently employed four NC analyses (SFA, PMOO, LP, and ULP) in Section 4. Section 5 concludes the paper.

2 Network Calculus Background

2.1 The System Description

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

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 up until time t. These functions allow for a straight-forward derivation of flow delays.

Definition 1. *(Flow Delay)* Assume a flow with input A crosses a server s and results in the output A'. The (virtual) delay for a data unit arriving at time t is

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

Note, that the order of data within the flow needs to be retained for the (virtual) delay calculation [18].

Network calculus 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

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

For example, periodic traffic with a maximum packet size b and a maximum arrival rate r can be bounded by token-bucket curves $\mathcal{F}_{\mathrm{TB}} = \{\gamma_{r,b} \mid \gamma_{r,b}(0) = 0, \forall d > 0 \; : \; \gamma_{r,b}(d) = b + r \cdot d\} \subseteq \mathcal{F}_0$.

Scheduling and buffering leading to the output function $A'(t)$ depend on a server's forwarding service. 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 instance, service offered by Ethernet connections can be described by rate-latency curves $\mathcal{F}_{\mathrm{RL}} = \{\beta_{R,T} \mid \beta_{R,T}(d) = \max\{0, R \cdot (d-T)\} \subseteq \mathcal{F}_0$.

A number of servers fulfill a stricter definition of service curves that guarantees a higher output during periods of queued data (backlogged periods).

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 $\beta(d)$.

2.2 Algebraic Network Calculus

Network calculus was cast in a $(\min, +)$-algebraic framework in [15,10]. We will first depict the basic operations and then present their combination for flow analysis.

(min,+)-Operations 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

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

Quantifier	Definition
foi	Flow of interest, the flow under analysis
$\langle s_x, \ldots, s_y \rangle$	Tandem of consecutive servers s_x to s_y
$P(f)$	Path of flow f (a tandem of servers)
α^f,α_s^f	Arrival curve of flow f, arrival bound at server s
β_s	Service curve of server s
$\beta_s^{\text{l.o.}f}$, $\beta_{\langle s_x,\ldots,s_y \rangle}^{\text{l.o.}f}$	Left-over service curve for f at server s, on tandem $\langle s_x, \ldots, s_y \rangle$

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

The system description's 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 using the deconvolution $\alpha \oslash \beta$:

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

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

where the delay bound holds independent of t and α' is an arrival curve for A'.

Analyzing an entire flow with cross-traffic on its path is enabled by the following theorems. Table 1 depicts the notation we use for the network analysis.

Theorem 2. (Concatenation of Servers) *Consider a single flow f crossing a tandem of servers $s_1, \ldots, s_n$ where each s_i offers a service curve β_{s_i}. The overall service curve for f is their concatenation by convolution*

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

Theorem 3. (Left-Over Service Curve) *Consider a server s that offers a strict service curve β and that serves two input flows, f_1 and f_2 with arrival curves α^{f_1} and α^{f_2}, respectively. The minimum service f_1 is guaranteed to receive is lower bounded by the so-called left-over service curve*

$$\beta_s^{l.o.f_1} = \beta_s \ominus \alpha^{f_0},$$

with $(\beta \ominus \alpha)(d) := \sup_{0 \leq u \leq d} \{(\beta - \alpha)(u)\}$ denoting the non-decreasing upper closure of $(\beta - \alpha)(d)$.

3 Network Calculus Feed-Forward Analyses (FFA)

3.1 Algebraic Network Calculus Analysis

An algebraic network calculus FFA computes the end-to-end delay bound for a specific flow interest (foi). Conceptually, the analysis proceeds in two steps [2,3]:

1. The analysis abstracts from the feed-forward network to the analyzed flow's path (tandem of servers). This step is enabled by recursively backtracking flows, decomposing the network into tandems [2] along their paths and bounding output arrivals of cross-traffic with Theorem 1, the output bound. Then, arrival curves that bound the worst-case shape of cross-flows are known at the location of interference with the foi.
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 flow's end-to-end service curve is derived and the delay bound computed.

The second step of the algebraic feed-forward analysis procedure has seen much treatment in the literature. Effort focused on improving the ability to capture flow scheduling and cross-traffic multiplexing effects in a tandem analysis. This effort resulted in two basic principles for left-over service curve derivation of tandems that improve algebraic NC's accuracy:

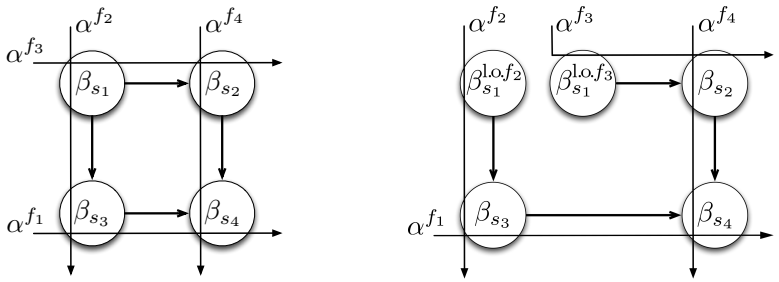
The PBOO-principle and the Separate Flow Analysis [15]: The SFA is a straight-forward, hop-by-hop application of Theorems 3 and 2: First subtract cross-traffic arrivals and then concatenate the left-over service curves. Deriving the delay bound with a single, end-to-end left-over service curve will consider the flow of interest's burst term only once. This principle is therefore called Pay Bursts Only Once (PBOO). However, for cross-flows present at multiple consecutive hops, bursts impact the derivation multiple times.

The PMOO-principle and -analysis [20]: The PMOO analysis provides an alternative derivation containing each burst term only once. Its left-over service curve derivation reverses the operations, i.e., it convolves the tandem of servers before subtracting cross-traffic. Due to this end-to-end approach for all flows on the analyzed tandem, the PMOO analysis was considered superior to SFA. Yet, [19] shows that the SFA can arbitrarily outperform a PMOO tandem analysis. Both algebraic analyses thus complement each other.

3.2 A New Principle for Feed-Forward Analysis: PSOO

Artifacts of Algebraic FFA In [4], the expression cross-flow segregation was coined for the situation where a network calculus analysis considers cross-flows to be mutually interfering in the worst case. It was identified in a procedure for the FFA step 1 that derives an individual arrival bound for each cross-flow [6]. While this situation could be avoided easily, we found further problems in the FFA, so-called *analysis artifacts*, enforcing segregation nonetheless.

Artifact 1: The FFA Procedure During the backtracking in step 1, several tandem analyses are executed such that the FFA can re-compose tandem-local results. This imposes a fundamental problem: Each tandem analysis is an independent instance of SFA or PMOO analysis, operating on its own worst-case assumptions that retain the overall FFA's worst-case modeling. The worst case is, in fact, a segregation of flows. Problems arise if flows share a server, demultiplex and rejoin



(a) *Square network.* (b) *Algebraic FFA leading to flow segregation.*

Figure 1: *The square network and the result of the algebraic FFA applied to it if* f_1 *(*α^{f_1}*) is the analyzed flow of interest.*

again – either each other directly or indirectly via another flow. For an example see Figure 1: The flows at server s_1 (service β_{s_1}) are treated independently of each other, i.e., they assume worst-case mutual interference that is not attainable in a real network. This independence is expressed by the segregation of service β_{s_1} into $\beta^{\text{l.o.}f_2}_{s_1}$ and $\beta^{\text{l.o.}f_3}_{s_1}$, both with a latency for the respective flow that is greater than β_{s_1}'s latency. Thus, they cannot sum up to the full service β_{s_1}.

Artifact 2: Interdependence between FFA Steps We also found the need for segregation in tandem networks – although this topology does not allow flows to demultiplexing, take different paths and rejoin again. This second artifact is illustrated in the non-nested tandem with cross-traffic arrival bounding in Figure 2. Applying a PMOO FFA, i.e., using PMOO in the FFA step 2, requires knowledge about each cross-flow's arrival bounds individually. This can only be achieved by segregately executing the FFA step 1 for each cross-flow xf_1 and xf_2. This artifact thus leads to two independently computed left-over service curves $\beta^{\text{l.o.}xf_1}_{s_0}$ and $\beta^{\text{l.o.}xf_2}_{s_0}$ that do not add up to β_{s_0}, i.e., the overly pessimistic analysis does not consider usage of the entire forwarding resources.

Effect on the Analysis Result The pessimism of independent left-over service curve derivations inevitable results in less accurate delay bounds. We illustrate this situation for the tandem network of Figure 2. In the following, we compare the PMOO delay bound for f_1 delay bound with PMOO FFA (left side, Fig. 2b) against a derivation using the entire service offered by s_1 (right side, Fig. 2a):

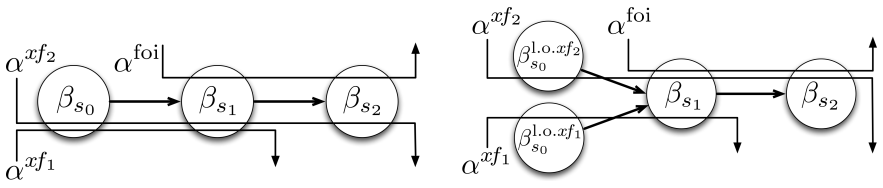
$$\beta^{\text{l.o.}xf_1}_{s_0} + \beta^{\text{l.o.}xf_2}_{s_0} < \beta_{s_0}$$

$$\Rightarrow \underbrace{\left(\alpha^{xf_1} \oslash \beta^{\text{l.o.}xf_1}_{s_0}\right)}_{=:\alpha^{xf_1}_{s_1}} + \underbrace{\left(\alpha^{xf_2} \oslash \beta^{\text{l.o.}xf_2}_{s_0}\right)}_{=:\alpha^{xf_2}_{s_1}} > \underbrace{\left(\alpha^{xf_1} + \alpha^{xf_2}\right) \oslash \beta_{s_0}}_{=:\alpha^{[xf_1,xf_2]}_{s_1}}$$

$$\Rightarrow \beta_{s_1} \ominus \left(\alpha^{xf_1}_{s_1} + \alpha^{xf_2}_{s_1}\right) < \beta_{s_1} \ominus \alpha^{[xf_1,xf_2]}_{s_1}$$

$$\Rightarrow \beta^{\text{l.o.}_{\text{segr}}\text{foi}}_{\langle s_1,s_2\rangle} < \beta^{\text{l.o.foi}}_{\langle s_1,s_2\rangle}$$

$$\Rightarrow D^{\text{foi}}_{\text{segr}} > D^{\text{foi}}$$



(a) *Tandem network.* (b) *PMOO analysis enforces segregation.*

Figure 2: *A non-nested tandem and the result of the PMOO FFA applied to it.*

As the segregated left-over service curves do not sum up to the original service curve, they cause larger output bounds for xf_1 and xf_2. This, in turn, results in a smaller left-over service curve for the foi at server s_1. Therefore, the end-to-end left-over service curve under flow segregation, $\beta^{\text{l.o.}_{\text{segr}}\text{foi}}_{\langle s_1,s_2\rangle}$, is smaller than the one without flow segregation, $\beta^{\text{l.o.foi}}_{\langle s_1,s_2\rangle}$. This results in a larger delay bound.

All of the segregated flows consider each other with isolated, local worst-case assumptions that retain the global worst case. This leads to an overall situation assumed by the analysis that cannot be attained in a realistic system. We capture this problem in a novel principle to be strived for in a feed-forward network calculus analysis, formulating it similar to PBOO and PMOO (Section 3.1).

The Pay Segregation Only Once (PSOO) principle: If the arrivals of two flows have to be bounded segregately in the feed-forward analysis and these flows both cross the same server before interfering with the flow of interest, then they should not be segregated in a way that imposes worst-case mutual interference assumptions on both. Segregation of cross-flows should only be paid for once by the ensemble of the two flows.

For instance, in the algebraic FFA equation, segregated flows should not have to consider each other fully in their respective arrival bounding. Although this leads to valid intermediate bounds on arrivals and left-over service, the according behavior is not attainable by a realistic system and thus the eventual performance bound cannot be tight.

Mitigation with by Aggregation Having derived these artifacts of algebraic FFA, we can mitigate them in different ways. On the one hand, it is possible to prevent their occurrence by routing restrictions. However, adapting the network to be analyzed may not be justifiable. Thus, we strife for a different mitigation strategy: flow aggregation. Yet, aggregation is not universally practical in algebraic NC. Therefore, we depict the state-of-the-art optimization-based NC that does not suffer from these artifacts next. Then, we benchmark it against implementations of our mitigation strategy and evaluate accuracy loss in networks where we cannot prevent algebraic NC from violating the PSOO principle.

3.3 Optimization-based Network Calculus Analysis and PSOO

An optimization-based FFA was proposed in [7]. It transforms the NC system description of Section 2.1 into a set of linear programs as follows:

1. Starting from the foi's sink server, flows and their cross-flows are recursively backtracked. For every link traversed backwards, the start of backlogged periods at the connected servers is related. This results in a partial order where there is no given order relation for servers on parallel paths.
2. Next, the partial order is extended to the set of all compatible total orders. In contrast to algebraic FFA, the backtracking result is not directly used to derive performance bounds. The extension enumerates all potential relations of backlogged periods on parallel paths to attain all potential entanglement in the network. Total orders of particular interest are those assuming an equal (start of) backlogged period for flows that later are demultiplexed. I.e., in Figure 1's network, the flows f_2 and f_3 are related with a common start of the backlogged period at server s_1 in order to implement PSOO.
3. Based on the network calculus model, each total order is converted into one linear program. Strict service curves, arrival curves, non-decreasing functions, non-negativity and the flow constraint derive LP constraints.
4. The set of LPs represents all potential entanglements in the network, not only worst-case ones. Therefore, all linear programs must be solved in the final step. The maximum among their solutions is a valid worst-case delay for the foi.

The LP FFA was, however, shown to be NP-hard due to step 2. Therefore, the authors propose to confine to a less complex, accurate analysis by skipping the extension of the partial order. This analysis is known as the *unique* LP (ULP). The ULP is, of course, less constrained than any single linear program of the LP; the constraints are chosen such that it guarantees to derive an upper bound on delay and backlog.

4 Accuracy Evaluation of Network Calculus Analyses

The LP analysis and the ULP analysis both implement the PBOO and the PMOO principle. Yet, the ULP skips the LP's step 2 that is crucial for the PSOO principle. Therefore, it must operate on worst-case assumptions when bounding the arrivals of cross-traffic. As arrival bounding is not distinguishable from the foi analysis during an optimization-based analysis, we aim to gain knowledge about the attained worst case by evaluations. In contrast to the literature, we add the PMOO analysis that implements PBOO and PMOO principles but not PSOO in order to observe the impact of LP's PSOO implementation on the accuracy of NC delay bounds.

As there is no comprehensive tool support for the LP or the ULP available, we need to rely on the tooling provided by the authors of these analyses[1]. It

[1] http://perso.bretagne.ens-cachan.fr/~bouillar/NCbounds/

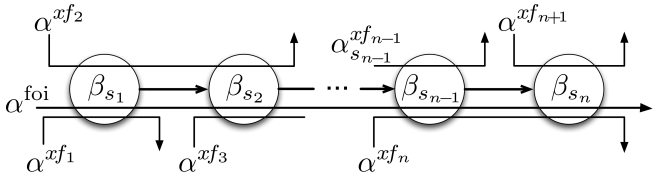


Figure 3: *Tandem network with non-nested cross-traffic arrivals.*

allows to analyze arbitrary tandem networks. Additionally, LpSolve lp files for the square network are available.

In this paper, we benchmark instances of these networks against the advanced algebraic NC that has been implemented in the DiscoDNC [2] in the meantime. Additionally, we provide the equations created by different analyses where it is helpful for illustration of analysis principle violations. We are the first to contribute PMOO delay bounds to a comparison of the two NC branches.

4.1 The Non-nested Tandem with Overlapping Interference

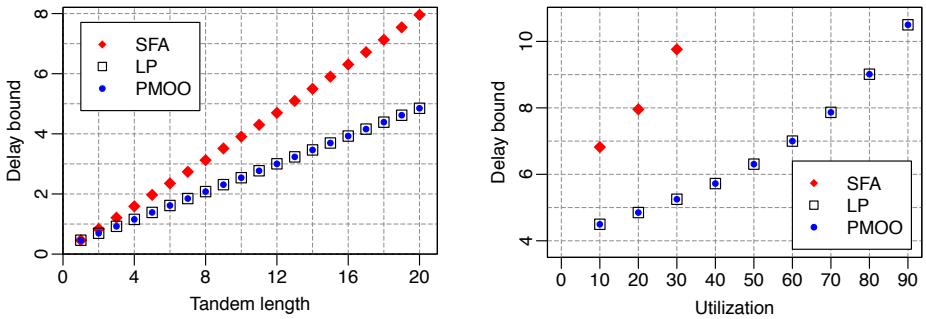
In [7], the so-called non-nested tandem was analyzed first. It consists of a sequence of servers crossed by the flow of interest and overlapping interference of cross-flows such that there are three flows at every server (see Figure 3). This is a classic example in network calculus; it was already used when introducing the PMOO principle and analysis [20]. Two evaluations are carried out: one that investigates the impact of the tandem's length and another one varying the utilization for the tandem of length 20. Arrival curves and service curves are taken from [7]: Both evaluations assign rate-latency service $\beta_{R,T} = \beta_{10,\frac{1}{10}}$. The evaluation with varying utilization $u \in \{0.1, \ldots, 0.9\}$ assigns token-bucket arrival curves of $\alpha = \gamma_{r,b} = \gamma_{\frac{10u}{3},1}$ where $\frac{10u}{3}$ is rounded to two decimal digits. The evaluation of tandem length impact is carried out at a utilization of 0.2, i.e., all arrivals are shaped to $\alpha = \gamma_{0.67,1}$.

Separate Flow Analysis (SFA) First, we derive the flow of interest's end-to-end left-over service curve (FFA step 2):

$$\begin{aligned}\beta^{\text{l.o.foi}}_{P(\text{foi})} &= \beta^{\text{l.o.foi}}_{s_1} \otimes \beta^{\text{l.o.foi}}_{s_2} \otimes \ldots \otimes \beta^{\text{l.o.foi}}_{s_n} \\ &= \left(\beta_{s_1} \ominus \left(\alpha^{xf_1}_{s_1} + \alpha^{xf_2}_{s_1}\right)\right) \otimes \left(\beta_{s_2} \ominus \left(\alpha^{xf_2}_{s_2} + \alpha^{xf_3}_{s_2}\right)\right) \otimes \ldots \otimes \left(\beta_{s_n} \ominus \left(\alpha^{xf_n}_{s_n} + \alpha^{xf_{n+1}}_{s_n}\right)\right) \\ &= \left(\beta_{s_1} \ominus \left(\alpha^{xf_1} + \alpha^{xf_2}\right)\right) \otimes \left(\beta_{s_2} \ominus \left(\alpha^{xf_2}_{s_2} + \alpha^{xf_3}\right)\right) \otimes \ldots \otimes \left(\beta_{s_n} \ominus \left(\alpha^{xf_n}_{s_n} + \alpha^{xf_{n+1}}\right)\right) \quad (1)\end{aligned}$$

In the equation, we can see that the PMOO property is not fulfilled. We need to pay for xf_m arrivals, $m \in \{2, \ldots n\}$, at both servers the cross-flow shares with the foi. Thus, we need to compute the xf_m arrivals at the respective second server of interference (FFA step 1; Equation 1: αs with server indices). For each server s_i, $i \in \{2, \ldots, n\}$, we get:

$$\begin{aligned}&\beta_{s_i} \ominus \left(\alpha^{xf_i}_{s_i} + \alpha^{xf_{i+i}}_{s_i}\right) \\ &\quad = \beta_{s_i} \ominus \left(\left(\alpha^{xf_i} \oslash \left(\beta_{s_{i-1}} \ominus \left(\alpha^{xf_{i-2}} \oslash \left(\ldots \left(\beta_{s_1} \ominus \alpha^{xf_1}\right) \ldots\right)\right)\right)\right) + \alpha^{xf_{i+1}}\right) \quad (2)\end{aligned}$$



(a) *20% utilization, increasing tandem size.* (b) *20 servers, increasing utilization.*

Figure 4: *Delay bounds in the non-nested tandem with overlapping interference.*

Note, that cross-traffic arrival bounding demands to recursively backtrack cross-flows of cross-flows at every server, yet, this backtracking never leaves the foi's path. The recursion terminates when the sources of flows are reached. See Equation 2 where αs do not have server indices as we know $\alpha^{xf_1}_{s_1} = \alpha^{xf_1}$, $\alpha^{xf_{n+1}}_{s_n} = \alpha^{xf_{n+1}}$ and $\alpha^{xf_m}_{s_{m-1}} = \alpha^{xf_m}$ for $m \in \{2, \dots n\}$ from the network description.

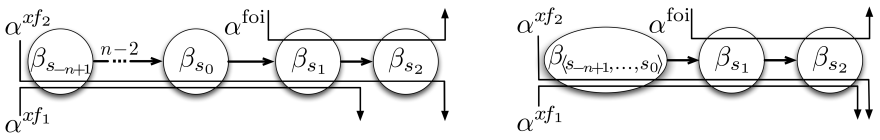
Pay Multiplexing Only Once (PMOO) analysis PMOO was specifically designed to counteract the burstiness increase being considered in the analysis multiple times. I.e, each xf_m's burstiness only appears once in its $\beta^{\text{l.o.foi}}_{P(\text{foi})}$-derivation [20]. The PMOO analysis does not demand the cross-flow arrival bounding (Equation 2) and thus performs better on this non-nested tandem.

Linear Programming (LP) analysis and comparison The results for the LP analysis are depicted alongside PMOO and SFA in Figure 4. They all scale linearly with the length of the tandem (Figures 4a) – the PBOO principle is responsible for this behavior. The second evaluation shows that all delay bounds grow super-linearly when increasing the utilization. Yet, the PMOO principle implemented by the eponymous analysis and the LP analysis leads to much smaller, better scaling delay bounds compared to the SFA (Figure 4b).

For both parts of this evaluation, the PMOO analysis that was omitted in [7] performs equal to the LP analysis.

4.2 The Non-nested Tandem with Cross-traffic Arrival Bounding

In this section, we will evaluate a tandem network with arbitrarily many PSOO violations due to algebraic FFA Artifact 2. To do so, we generalize the network of Figure 2a by adding a variable number of servers to be traversed by the two cross-flows (see Figure 5a). Each server constitutes one PSOO violation. Parameters are chosen according to the previous tandem evaluation: We assign rate-latency service $\beta_{R,T} = \beta_{10,\frac{1}{10}}$ and token-bucket arrival curves $\alpha = \gamma_{r,b}$.



(a) *Tandem network, n PSOO violations.* (b) *Flow extension counteracts segregation.*

Figure 5: *Tandem with algebraic FFA Artifact 2 and mitigation by flow extension.*

These are either fixed to $\gamma_{2,1}$ for a network utilization of 60% or kept variable at $\gamma_{\frac{10u}{3},1}$, $u \in \{0.1, \ldots, 0.9\}$, where $\frac{10u}{3}$ is rounded to two decimal digits.

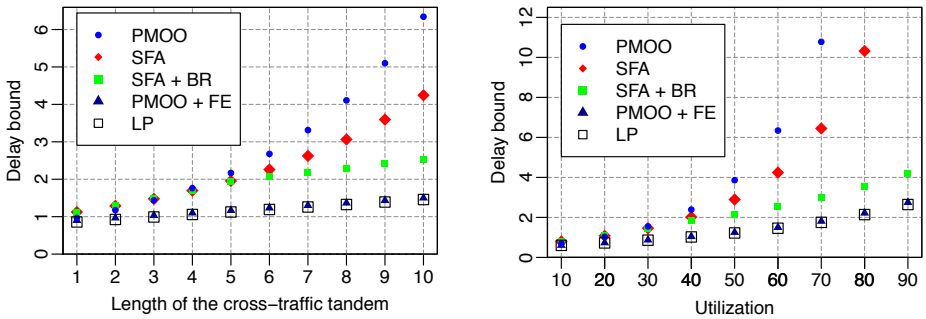
Separate Flow Analysis (SFA) While SFA can aggregately bound both cross-flows xf_1 and xf_2 at server s_1, its left-over service curve derivation at s_2 requires segregation in order to only consider xf_2 there. [5] presents a new countermeasure to this artifact called Burst Reduction (BR). It caps xf_2's burstiness at s_2 with the backlog bound aggregately caused by both flows at server s_1.

Pay Multiplexing Only Once (PMOO) analysis The PMOO analysis enforces PSOO violations in this network, too (see Section 3.2). This can be mitigated by a preceding step to the analysis called Flow Extension (FE) [1]. This step transforms the analyzed network to a different one that is worse form the foi's point of view: Interference of cross-flow xf_1 is extended to server s_2 as depicted in Figure 5b. While this reduces the service available to the foi at s_2 in the network, the PMOO FFA does not suffer from Artifact 2 anymore. It can now handle the cross-flows aggregately and convolve the servers they traverse. Therefore, it computes a better delay bound in the worse network setting. We are the first to show FE's benefits in arbitrary multiplexing networks.

Linear Programming (LP) analysis and comparison Equal to the previous tandem network, we carry out two evaluations: one that increases the length of the cross-traffic tandem $s_{-n+1}, \ldots, s_0$ and thus the amount of PSOO violations and another one that increases the maximum utilization in the network.

The scaling with respect to the tandem length is depicted in Figure 6a. We fixed the network utilization, defined by server s_1 crossed by three flows, at 60%. SFA and PMOO analysis both scale super-linearly with the length of the cross-traffic tandem, i.e., with the amount of PSOO violations. Burst reduction (BR) significantly decreases the delay bound of the SFA for long tandems, however, it is outperformed by PMOO with flow extension (FE). The difference between the LP analysis and PMOO + FE remains small and steady for all tandem lengths – it is the penalty of assuming xf_1 crossing server s_2, too.

For the second evaluation, we fixed the tandem length at 12 servers, i.e., 10 servers with PSOO violations. Again, plain analyses (SFA and PMOO) scale super-linearly and the deviation from LP delays becomes large with growing



(a) *60% utilization, increasing tandem size.* (b) *10 cross-traffic servers, incr. util.*

Figure 6: *Delays in the non-nested tandem with cross-traffic arrival bounding.*

utilization. The two analyses amended with aggregation-based countermeasures perform considerably better; PMOO + FE derives nearly identical bounds to the LP analysis. The difference does not remain steady, yet, it does not grow much. In Figure 6b, PMOO + FE's triangles stay within LP's squares, i.e., current algebraic NC analyses counteract the PSOO violations with small amendments and their results are highly competitive with the ones derived by optimization.

4.3 The Square Network

Next, we evaluate the square network from [7] (see Figure 1). This network is evaluated for varying utilizations: service curves remain $\beta_{s_i} = \beta_{R,T} = \beta_{10,\frac{1}{10}}$, $i \in \{1,2,3,4\}$ and arrival curves are adapted to the utilization $u \in \{0.1,\ldots,0.9\}$. As there are only two flows per server, this setting translates to $\alpha^{f_i} = \gamma_{r,b} = \gamma_{\frac{10u}{2},1}$.

Separate Flow Analysis (SFA) We start with the SFA left-over service curve derivation steps shared by every utilization's analysis:

$$\beta^{\text{l.o.}f_1}_{\langle s_3,s_4\rangle} = \beta^{\text{l.o.}f_1}_{s_3} \otimes \beta^{\text{l.o.}f_1}_{s_4} = \left(\beta_{s_3} \ominus \alpha^{f_2}_{s_3}\right) \otimes \left(\beta_{s_4} \ominus \alpha^{f_4}_{s_4}\right) \tag{3}$$

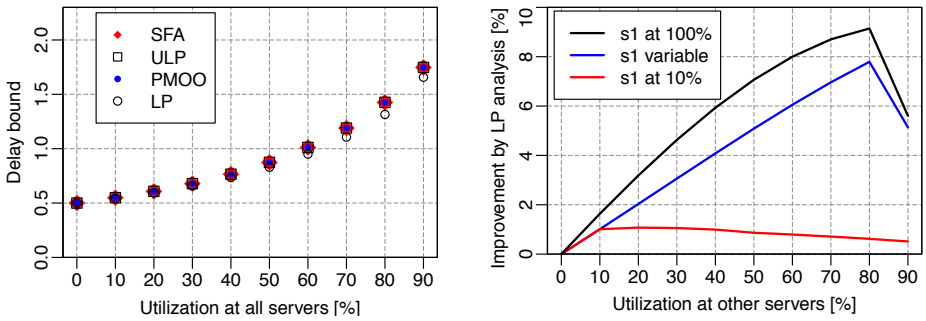
with the following cross-traffic arrival boundings:

$$\alpha^{f_2}_{s_3} = \alpha^{f_2} \oslash \beta^{\text{l.o.}f_2}_{s_1} = \alpha^{f_2} \oslash \left(\beta_{s_1} \ominus \alpha^{f_3}\right) \tag{4}$$

$$\alpha^{f_4}_{s_4} = \alpha^{f_4} \oslash \beta^{\text{l.o.}f_4}_{s_2} = \alpha^{f_4} \oslash \left(\beta_{s_2} \ominus \left(\alpha^{f_3} \oslash \left(\beta_{s_1} \ominus \alpha^{f_2}\right)\right)\right) \tag{5}$$

The cross-traffic arrival boundings show mutual interference assumptions of Figure 1: It computes $\beta_{s_1} \ominus \alpha^{f_3}$ and $\beta_{s_1} \ominus \alpha^{f_2}$, both will be in the SFA left-over service curve derivation of Equation 3 – the PSOO principle is not implemented.

This derivation illustrates the different approaches applied by algebraic NC to model the flow of interest's worst-case scenario in a feed-forward network. On the foi's path (i.e., the foi tandem analysis part of step 2), the left-over service curve



(a) *Uniform increase of utilization.* (b) *Relative improvement of bounds.*

Figure 7: *Delay bounds in the square network.*

derivation assumes lowest priority for the flow of interest (cf. Theorem 3). In the arrival bounding, each flow's worst-case interference is modeled individually. Therefore, at server s_1, flows f_2 and f_3 are considered as mutual interference: $\beta_{s_1}^{\text{l.o.}f_2} = \left(\beta_{s_1} \ominus \alpha^{f_3}\right)$ and $\beta_{s_1}^{\text{l.o.}f_3} = \left(\beta_{s_1} \ominus \alpha^{f_2}\right)$. Note, that we cannot exploit burst reduction because flows only interfere with each other on single servers.

Pay Multiplexing Only Once (PMOO) analysis The PMOO left-over service curve [20] is derived as follows:

$$\beta_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1} = \beta_{R_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1},T_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1}} \tag{6}$$

$$R_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1} = \left(R_{s_3} - r^{f_2}\right) \wedge \left(R_{s_4} - r^{f_4}\right) \tag{7}$$

$$T_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1} = T_{s_3} + T_{s_4} + \frac{b_{s_3}^{f_2} + b_{s_4}^{f_4} + r_{s_3}^{f_2} \cdot T_{s_3} + r_{s_4}^{f_4} \cdot T_{s_4}}{R_{\langle s_3,s_4\rangle}^{\text{l.o.}f_1}} \tag{8}$$

The mutual interference modeling persists. It can be seen in the cross-flow burst terms $b_{s_3}^{f_2}$ and $b_{s_4}^{f_4}$ in Equation 8 that equal the one of the according arrival bounds $\alpha_{s_3}^{f_2}$ and $\alpha_{s_4}^{f_4}$ used in the SFA. Cross-traffic is bounded with Equations 4 and 5, again, as it belongs to FFA step 1. I.e., the PMOO left-over service curve derivation of Equation 6 will also suffer from the mutual interference problem depicted in Figure 1 due to Equation 8. In this network, flow extension is not beneficial as both cross-flows of f_1 arrive from different links.

Linear Programming (LP, ULP) analysis and comparison Based on the derivation of algebraic analysis of the non-nested tandems and the square network (Equations 1 to 8) as well as the observed properties and delay bounds, we can predict the relative performance of NC analyses in the square network:

Recall that the LP approach enumerates all potential entanglements of flows by extending the partial order (defined by consecutive hops of flows) to the set

of all compatible total orders – the benefit is pointed out in Section 3.3, step 2. The ULP, in contrast, is solely based on the partial order, the extension step is omitted due to its combinatorial explosion. This results in a smaller set of ULP constraints, especially the potential entanglements of f_2 and f_3 at s_1 are not considered anymore. Instead, they are assumed to constitute the respective flow's worst case. Based on this insight, it is not surprising that the ULP actually does not beat the PMOO analysis in the square network (see Figure 7a). In fact, even the SFA yields the same delays. The reason for SFA's accuracy is the lack of multi-hop interference, the effect captured with the PMOO principle does not manifest in this network.

Our evaluation shows that the ULP models the worst-case interference between f_2 and f_3 in the same way as the algebraic analyses do.

The square network also illustrates the case where we cannot mitigate the PSOO violation in algebraic NC. Thus, we conclude our evaluation by investigating the potential superiority of LP delay bounds over PMOO delay bounds. Figure 7b depicts three different settings:

- First, we fixed the utilization of server s_1 at 100%. For the mutual interference assumptions violating the PSOO principle, this leads to the maximum burstiness increase attainable for flows f_2 and f_3. As depicted in Section 3, the burstiness will propagate through the analysis and eventually result in a loose delay bound. However, the final impact also depends on the utilization of servers that forward the flows with overly pessimistic burstiness. We evaluated utilizations of the remaining servers s_2, s_3 and s_4 ranging from 10% to 90% (see [7]). As expected, we can first observe an amplification of the PSOO violation's effect that lets the LP analysis outperform the algebraic ones by an increasing margin. Yet, it reaches its maximum of 9.15% at 80% utilization. After this peak, its impact declines to a level below the 40% network utilization.
- A similar decline can be observed in the second evaluation, taken from [7] and depicted in Figure 7a, where the rate at s_1 varies with the other servers.
- We added another utilization level to the evaluation to confirm this observation. With a fixed utilization of 10% at s_1, the peak is already reached at a network utilization of 10%.

Thus, our results show that the benefit of implementing the PSOO principle is bounded as the network utilization becomes more impactful. Compared to the impact of previous analysis principles, the PSOO did not lead to a vast decrease of delay bounds. Moreover, implementing it requires huge effort. There are already eleven linear programs derives from this small square network alone.

4.4 Outlook

Evaluation of Larger, more Involved Networks In this paper, we restricted our evaluation to tandem networks and the square network. This restriction is caused by a practical problem of the LP analysis. Besides being NP-hard, the

effort to derive its linear programs them scales super-exponentially with the network size. The authors of this analysis point out two particular problems [6]:

1. The LPs relate the start of backlogged periods at servers. The amount of these dates to be related to each other grows exponentially with the network size. This problem directly impact the second one.
2. Dates are not totally ordered. This problem corresponds to step 2 given in Section 3.3: extension of a partial order to the set of all compatible total orders. Even for rather small networks, this step suffers from combinatorial explosion [16] and known algorithms for linear extensions like the Varol-Rotem algorithm [21] do not allow for the analysis of larger networks. Computational effort becomes prohibitive.

For these reasons, network calculus lacks comprehensive tool support for the only analysis implementing the PSOO principle as of today. In tandem networks, flows cannot take parallel paths and thus there is only one total order. This fact is exploited by the tool we used for the evaluations in Sections 4.1 and 4.2. For the square network evaluation of Section 4.3, the authors of the LP analysis provide the required linear programs. Analyzing larger networks remains an open issue.

Improving the Accuracy of the ULP Analysis The ULP constitutes the return to accurate, yet, untight bounds in general feed-forward networks. However, for some special networks it derives tight bounds nonetheless. The special case holds in tandem networks (see Figure 1) where there is only one order. For more involved networks, we only know that the ULP has less constraints than any of the LPs. Improving the ULP's result can only be achieved by adding more constraints to it. The addition of constraints found in a total order can, however, result in a linear program that produces an invalid bound. Identifying constraints that improve the derived bound while guaranteeing to retain its validity is an open research topic of optimization-based NC.

5 Conclusion

In this paper, we assessed both branches of network calculus, the algebraic and the optimization-based analysis branch. We aimed at a more comprehensive comparison of those generally incomparable alternatives to derive delay bounds from a NC system description. A new principle for feed-forward analysis, the Pay Segregation Only Once (PSOO), enabled us to derive new insights on both branches of NC such that we were able to predict the relative results between analyses. Moreover, we provide evidence that the PSOO principle that is only implemented by the NP-hard LP analysis does not lead to vastly improved delay bounds when it is compared to the PMOO analysis of algebraic network calculus. Thus, there is currently no clear proof for the necessity to relocate network calculus by abandoning the idea of an algebraic system theory. In current NC, the LP analysis is rather a tool to benchmark algebraic NC analysis in small networks and as such it helps to find algebraic NC's weak spots, e.g., the violation of the PSOO principle we present and evaluate in this work.

References

1. L. Bisti, L. Lenzini, E. Mingozzi, and G. Stea. Estimating the Worst-case Delay in FIFO Tandems Using Network Calculus. In *Proc. ValueTools*, 2008.
2. S. Bondorf and J. B. Schmitt. The DiscoDNC v2 – A Comprehensive Tool for Deterministic Network Calculus. In *Proc. ValueTools*, 2014.
3. S. Bondorf and J. B. Schmitt. Boosting Sensor Network Calculus by Thoroughly Bounding Cross-Traffic. In *Proc. IEEE INFOCOM*, 2015.
4. S. Bondorf and J. B. Schmitt. Calculating Accurate End-to-End Delay Bounds – You Better Know Your Cross-Traffic. In *Proc. ValueTools*, 2015.
5. S. Bondorf and J. B. Schmitt. Improving Cross-Traffic Bounds in Feed-Forward Networks – There is a Job for Everyone. In *Proc. GI/ITG MMB & DFT*, 2016.
6. A. Bouillard. *Algorithms and Efficiency of Network Calculus*. Habilitation thesis, ENS, 2014.
7. A. Bouillard, L. Jouhet, and E. Thierry. Tight Performance Bounds in the Worst-Case Analysis of Feed-Forward Networks. In *Proc. IEEE INFOCOM*, 2010.
8. M. Boyer and C. Fraboul. Tightening end-to-end delay upper bound for AFDX network calculus with rate latency FIFO servers using network calculus. In *Proc. IEEE WFCS*, 2008.
9. M. Boyer, N. Navet, X. Olive, and E. Thierry. The PEGASE project: precise and scalable temporal analysis for aerospace communication systems with network calculus. In *Proc. ISoLA*, 2010.
10. C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer, 2000.
11. F. Frances, C. Fraboul, and J. Grieu. Using Network Calculus to Optimize AFDX Network. In *ERTS*, 2006.
12. F. Geyer and G. Carle. Network engineering for real-time networks: comparison of automotive and aeronautic industries approaches. *IEEE Communications Magazine*, 54(2):106–112, February 2016.
13. J. Grieu. *Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques*. PhD thesis, INPT, 2004.
14. S. Kerschbaum, K.-S. J. Hielscher, U. Klehmet, and R. German. A Framework for Establishing Performance Guarantees in Industrial Automation Networks. In *GI/ITG MMB & DFT*, 2014.
15. J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer, 2001.
16. F. Ruskey. *Combinatorial Generation*. 2003.
17. M. Schmidt, S. Veith, M. Menth, and S. Kehrer. DelayLyzer: A Tool for Analyzing Delay Bounds in Industrial Ethernet Networks. In *GI/ITG MMB & DFT*, 2014.
18. J. B. Schmitt, N. Gollan, S. Bondorf, and I. Martinovic. Pay Bursts Only Once Holds for (Some) non-FIFO Systems. In *Proc. IEEE INFOCOM*, 2011.
19. J. B. Schmitt, F. A. Zdarsky, and M. Fidler. Delay Bounds under Arbitrary Multiplexing: When Network Calculus Leaves You in the Lurch ... In *Proc. IEEE INFOCOM*, 2008.
20. J. B. Schmitt, F. A. Zdarsky, and I. Martinovic. Improving Performance Bounds in Feed-Forward Networks by Paying Multiplexing Only Once. In *Proc. GI/ITG MMB*, 2008.
21. Y. L. Varol and D. Rotem. An Algorithm to Generate all Topological Sorting Arrangements. *The Computer Journal*, 24(1), 1981.