Catching Corner Cases in Network Calculus – Flow Segregation Can Improve Accuracy

Steffen Bondorf[1]*, Paul Nikolaus[2], and Jens B. Schmitt[2]

[1] National University of Singapore (NUS), Republic of Singapore
[2] Distributed Computer Systems (DISCO) Lab, TU Kaiserslautern, Germany

Abstract Worst-case bounds on flow delays are essential for safety-critical systems. Deterministic network calculus is a methodology to compute such bounds. It is actively researched regarding its modeling capabilities as well as analysis accuracy and performance. We provide a contribution to the major part of the analysis: bounding the arrivals of cross flows. In particular, it has been believed that an aggregate view on cross flows outperforms deriving a bound for each cross flow individually. In contrast, we show that the so-called cross-flow segregation, can outperform the aggregation approach under certain conditions. We give a proof of concept, combine the alternative approaches into an analysis computing best bounds, and evaluate accuracy improvements as well as computational effort increases. To that end, we show that flows known to suffer from overly pessimistic delay bounds can see this pessimism reduced by double-digit percentages.

1 Introduction

In safety-critical, distributed systems that operate in public spaces, formal verification of performance guarantees is often a prerequisite for certification. For example, bounding the end-to-end delay of data transmissions is an integral demand of x-by-wire applications such as steer-by-wire or brake-by-wire. Thus, even small gains in its accuracy can be of importance for the outcome of a certification process. Deterministic Network Calculus (DNC) provides an analytical framework to compute worst-case delay bounds on data transmissions. For instance, it has found application in the avionics industry to demonstrate fulfillment of strict aircraft network requirements. To be precise, DNC's (min,+)-algebraic branch is often applied to analyze these avionics networks. A model bounding supply and demand of the network's forwarding resources is transformed to a (min,+)-equation that bounds a specific flow's end-to-end delay.

The step deriving an equation from the model has been steadily evolved in order to improve the accuracy of the resulting delay bound. The first such improvement was to virtually separate the analyzed flow from all other flows in the network and to establish the worst-case scenario exclusively for this flow

* This work has been conducted at the Distributed Computer Systems (DISCO) Lab, TU Kaiserslautern, Germany, with support of a Carl Zeiss Foundation grant.

of interest (foi). This principle, separation of the foi, was shown to result in better bounds than the previous approach to anaylse the totality of the flows. However, subsequent work shows that only the analyzed flow of interest can be entirely removed while constructing a worst-case scenario. Efforts to implement this principle for the remainder of the network analysis were shown to result in an issue called segregation. In arbitrary multiplexing, i.e., no knowledge about the multiplexing of flows is assumed, attempting to separate multiple flows can result in situations where these simultaneously assume worst-case scenarios that are mutually exclusive in the real system. While resulting bounds remain valid, they are overly pessimistic. Therefore, the predominating objective of analyses is to aggregate all flows except the flow of interest. The literature provides a generic analysis procedure that maximizes aggregation and minimizes segregation when bounding the arrivals of the foi's cross flows. This procedure, known as Aggregate Arrival Bounding (AggrAB) [5], was extended by various analysis enhancements that can further increase aggregation of flows and improve delay bounds [6,1]. Based on the objective to maximize aggregation, an accurate and fast analysis was eventually presented [2]. The delay bounds it derives are shown to only deviate slightly from those bounds derived with the optimization branch of DNC [9]. However, while the optimization becomes computationally infeasible , the algebraic analysis scales well with increasing network size. Its execution time stays several orders of magnitude below the optimization's one [2].

In this paper, we focus on further closing the gap between algebraic and optimization-based DNC delay bounds. To be precise, we identify a peculiar corner case that is defined by a very specific combination of flow entanglements, resource demand, (left-over) resource supply and implemented DNC principle. Against the trend to prevent segregation of flows by aggregating them as much as possible, we prove that the reverse can actually result in better delay bounds. The mutually exclusive worst-case assumptions of two flows add less pessimism than the AggrAB does. We use this knowledge to contribute an arrival bounding method that catches these corner cases. It is modeled after [8] but as it relies solely on the PMOO principle, we call it SegrPMOO. Moreover, we combine it with AggrAB's latest evolutionary step, the Tandem Matching Analysis (TMA) [2], to an exhaustive search for best arrival bounds, TMA+SegrPMOO.

We evaluate our contribution by extending the numerical results providing insight on the gap between TMA and optimization. We show that the preconditions for these corner cases can be fulfilled fairly often during a network analysis, yet, in most cases it only helps to close the accuracy gap by less than 10%. A noteworthy exception to this observation can be found when investigating outliers. E.g., in the TMA evaluation's network of 20 devices, outliers are common and we show that the largest gaps to optimization can be reduced by over 30%.

The remainder of the paper is structured as follows: Section 2 provides the necessary background on DNC. In Section 3, we present the trend to improve bounds by aggregating flows. To that end, we provide the DNC analysis principles implemented in TMA. Based on these insights, Section 4 contributes and proves the potential benefit of SegrPMOO. We extend the previously applied

cross-traffic arrival bounding with it and evaluate our contribution in Section 5. We provide results on accuracy improvement as well as computational effort increase. Section 6 concludes the paper.

2 Deterministic Network Calculus Background

DNC is based on a simple network model [12] consisting of functions from the set

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

Cumulative data arrivals are upper bounded in interval time:

Definition 1. Given a flow producing data according to function A in the time domain, a function $\alpha \in \mathcal{F}_0$ is an arrival curve for the flow iff

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

I.e., arrival curves bound the maximum data arrivals of a flow during any duration of length d. $\mathcal{F}_{\mathrm{TB}} \subseteq \mathcal{F}_0$ is a commonly used set of curve shapes to bound flow arrivals. It bounds flows shaped to comply with token bucket regulation:

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

A server's forwarding of arriving data is lower bounded in interval time:

Definition 2. If the service provided by a server S for a given input function A results in an output function A', then S offers a service curve $\beta \in \mathcal{F}_0$ iff

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

A common set of curves $\mathcal{F}_{\mathrm{RL}} \subseteq \mathcal{F}_0$ bounds service with a rate and a latency:

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

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

Definition 3. Let $\beta \in \mathcal{F}_0$. Server S offers a strict service curve β iff, during any backlogged period of duration d, its output is at least equal to $\beta(d)$.

Basic operations to manipulate curves while conserving the model's deterministic worst case were cast in a $(\min, +)$-algebraic framework [15,11]:

Definition 4. The main (min,+)-algebraic DNC operations for $f, g \in \mathcal{F}_0$ are

$$\textit{aggregation: } (f+g)(t) \coloneqq f(t) + g(t),$$
$$\textit{convolution: } (f \otimes g)(t) \coloneqq \inf_{0 \le s \le t} \{f(t-s) + g(s)\},$$
$$\textit{deconvolution: } (f \oslash g)(t) \coloneqq \sup_{u \ge 0} \{f(t+u) - g(u)\}.$$

With these operations, we can bound performance characteristics of flows:

Theorem 1. *Consider a server S offering a service curve β. Assume flow f with arrival curve α crosses S. We obtain these two performance bounds for f:*

$$\textit{Delay:} \quad \forall t \in \mathbb{R}^+ : D(t) \leq \inf\{d \geq 0 \mid (\alpha \oslash \beta)(-d) \leq 0\},$$

where it is assumed that the order of data in f is not altered.

$$\textit{Output:} \quad \forall d \in \mathbb{R}^+ : \alpha'(d) = (\alpha \oslash \beta)(d),$$

where α' is an arrival curve for A'.

Theorem 2. *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 the concatenation of servers, achieved by convolution*

$$\beta_{S_1} \otimes \cdots \otimes \beta_{S_n} = \bigotimes_{i=1}^{n} \beta_{S_i}.$$

Theorem 3. *Consider a server S offering a strict service curve β_S. Let S be crossed by two flows f_0 and f_1 with arrival curves α^{f_0} and α^{f_1}, respectively. Then f_1's worst-case residual resource share without knowledge about multiplexing (so-called* arbitrary multiplexing*) at S, i.e., its left-over service curve at S, is*

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

where $(\beta \ominus \alpha)(d) \coloneqq \sup_{0 \leq u \leq d}\{(\beta - \alpha)(u)\}$ denotes the non-decreasing upper closure of $(\beta - \alpha)(d)$.

The above left-over service curve operation is applicable to single systems only. Multiple DNC left-over service curve computations for tandems $\langle S_1, \ldots, S_n \rangle$ have been proposed in the literature. We include a common notation for these in Table 1 and discuss them as required in the next section.

Lastly, note that the optimization analyses LP and ULP [9] do not derive a left-over service curve. Instead, they each derive optimization formulations, linear programs, whose result bounds the foi's end-to-end delay.

3 Aggregation as the Objective of DNC Analyses

Basic DNC (min,+)-operations have been composed to analyses that achieve varying degrees of accuracy as they implement different sets of principles. Yet, not all principles can be fully attained at the same time. Some are even mutually exclusive under current DNC analyses. The most impactful principles and thus the best choice of analysis depends on the network scenario and the flow of interest to be bounded. Moreover, algebraic DNC analysis is compositional. It requires to combine tandem analyses to a network analysis. This paper focuses on the search for the best composition. In this section, we give detailed background on DNC weaknesses and previous attempts to mitigate or solve them by different analysis principles.

Notation	Definition
foi	Flow of interest
$\{f_n, ..., f_m\}$	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, $\alpha^{\{f_n,\ldots,f_m\}}$	Arrival curve (flow, aggregate)
α_S^f, $\alpha_S^{\{f_n,\ldots,f_m\}}$	Arrival bound at server S (flow, aggregate)
β_S	Service curve of server S
$\beta^{\text{l.o.}f}$, $\beta^{\text{l.o.}\{f_n,\ldots,f_m\}}$	Left-over service curve (flow, aggregate)
$\beta_S^{\text{l.o.}f}$, $\beta_S^{\text{l.o.}\{f_n,\ldots,f_m\}}$	Left-over service curve of server S
$\beta^{\text{l.o.}f}_{\langle S_x,\ldots,S_y\rangle}$, $\beta^{\text{l.o.}\{f_n,\ldots,f_m\}}_{\langle S_x,\ldots,S_y\rangle}$	Left-over service curve of tandem S_x to S_y

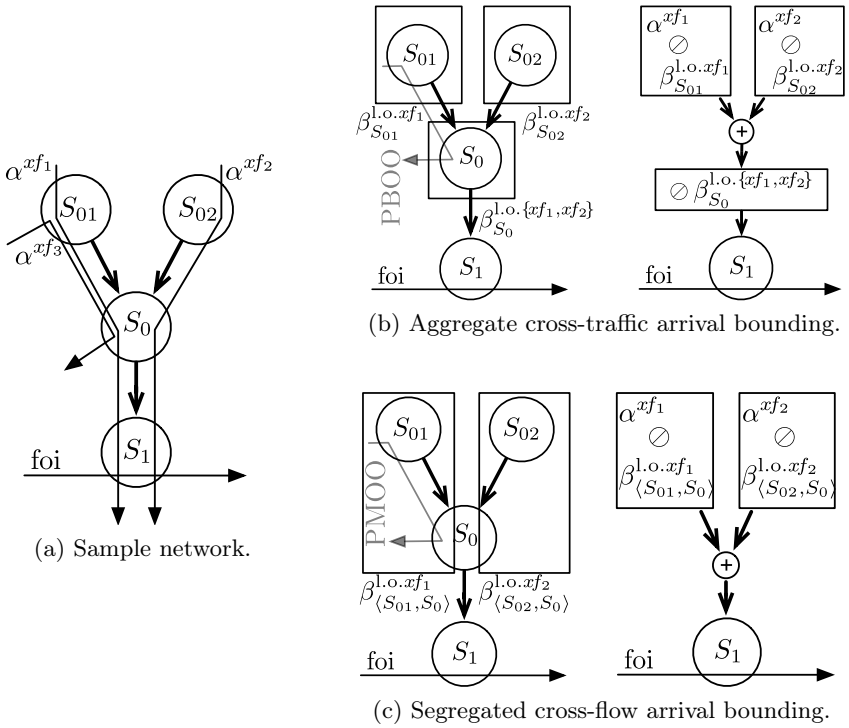
Table 1: Deterministic Network Calculus Notation.

3.1 DNC Analyses and Principles

The ultimate goal of a DNC analysis is to give an upper bound on the delay for the foi. This flow is thus the starting point of an analysis, its path defines the first tandem of servers whose left-over service curve has to be derived. This computation requires bounds on data arrivals of interfering flows. Therefore, these cross flows are backtracked, their respective left-over service curve is derived, and the output from their path – another tandem of servers – is bounded. This procedure repeats recursively in order to consider all flows that impact the foi directly or indirectly [4]. Thus, a compositional network analysis is composed of many tandem analyses, each computing a left-over service curve $\beta^{\text{l.o.}}_{\langle S_1,\ldots,S_n\rangle}$. To allow for selecting the best analysis per tandem, we present common principles.

Aggregation of Flows (Agg) [12,13] Aggregation of flows crossing a server (Definition 4) is the earliest principle of DNC. Total Flow Analysis (TFA), the first DNC analysis, proposes to aggregate the totality of flows at each server. This is generally beneficial for output bounding, yet, an aggregate's delay bound depends on the multiplexing discipline. In FIFO multiplexing, the horizontal deviation between aggregate arrival curve and service curve gives a valid bound (Theorem 1). For arbitrary multiplexing as considered in this paper, the intersection does. Separating the foi with a left-over service operation (Theorem 3) allows for horizontal deviation and results in better delay bounds.

Pay Bursts Only Once (PBOO) [15] Computing the foi's left-over service curve enables a key principle of algebraic DNC analysis: convolving all left-over service curves of servers crossed by the foi (Theorem 2). Then, the tandem analysis does not compute the foi's arrivals at every server, but its burst term is only considered once and the principle is known as pay bursts only once. Separation of the foi and convolution of left-over service curves are key to the



(a) Sample network.

(b) Aggregate cross-traffic arrival bounding.

(c) Segregated cross-flow arrival bounding.

Figure 1: Decomposition of a network (a) for cross-traffic arrival bounding: (b) depicts the alternative to aggregate cross-flows [8] that restricts to PBOO, (c) shows the segregation approach that can benefit from the PMOO principle.

Separate Flow Analysis (SFA). Thus, the SFA implements the PBOO principle. Its delay bounds invariably outperform TFA delay bounds.

Pay Multiplexing Only Once (PMOO) [17] In case cross flows share multiple consecutive servers with the foi, their burst terms appear in each server's left-over computation. I.e., multiplexing with cross-traffic bursts is paid for multiple times and the principle to counteract this issue is known as pay multiplexing only once. The eponymic analysis, PMOO Analysis (PMOOA), suggests to reverse SFA's order of operations. Servers are convolved before cross traffic is subtracted. [17] provides a tandem left-over service curve computation achieving PMOO. Note, that PMOO implies PBOO due to foi separation and convolution.

Order of Servers (Order) [16] (min,+)-convolution is a commutative operation. Thus, the order of crossed servers is lost when applying it. This impacts the PMOOA as the slowest server defines the tandem service. For that reason, SFA's per-hop cross-traffic considerations can arbitrarily outperform

PMOOA [16]. Therefore, in [16] the first optimization-based analysis (OBA) is proposed that derives a tandem left-over service curve implementing PBOO, PMOO and accounting for the order of servers.

Output Burstiness Cap (OBC) [6] While separation of flows is beneficial for delay bounding, it was shown in [6] that the output bound computation suffers. A subset of flows' output burstiness after a server can supersede the maximum amount of backlogged data by the totality of flows – a cap for a server's output burstiness that also holds for any subset of flows crossing the respective server.

Pay Segregation Only Once (PSOO) [7] Applying the left-over service computation implicitly assumes higher priority for the subtracted flow. If, at a single server, two flows applying the left-over thus assume incompatible priorities, segregation is paid for more than once. Counteracting this issue results in better performance bounds.

Flow Prolongation (FP) [1] A strategy to benefit from aggregation, similar to OBC, is to prolong flows over servers they do not cross in reality. The analysis has to work with increased load assumptions at these servers, however, if prolonged flows then share a path with other flows, aggregation is possible and its benefits can supersede the pessimism added to the model. FP was shown to be inherently infeasible [1] and is thus not included as a viable principle in Table 2.

3.2 Compositional Approaches for Arrival Bounding

As stated above, DNC composes tandem left-over service computations to a feed-forward network analysis. The literature proposes two alternatives. Both start with the foi but differ beyond its path, in the so-called cross-traffic arrival bounding. It analyzes the network between locations of interference with the foi and sources of relevant cross flows. It is usually the largest part of the analysis, as exemplified in Figure 1.

Segregated Arrival Bounding (SegrAB) [8] A straightforward extension of the SFA was proposed for cross-traffic arrival bounding [8]. Each flow interfering with the foi is analyzed using SFA's approach: compute per-server left-over service curves from source to the server before meeting the foi, convolve these service curves, compute an output bound. Then, all flows' arrival bounds are summed up. However, it was shown that, for $\alpha \in \mathcal{F}_{\mathrm{TB}}$ and $\beta \in \mathcal{F}_{\mathrm{RL}}$, the PSOO violations explicitly enforced by this approach cannot be set off by the implemented PBOO principle [5]. In Figure 1c, the PSOO violation takes place at S_0. Due to the SFA-based extension, only PBOO is implemented in the two left-over service curves required to bound arrivals of cross flows, $\beta^{\mathrm{l.o.}xf_1}_{\langle S_{01},S_0\rangle}$ and $\beta^{\mathrm{l.o.}xf_2}_{\langle S_{02},S_0\rangle}$.

Aggregate Arrival Bounding (AggrAB) [5] The approach immediately resulting from the insights on segregated PBOO arrival bounding is to strongly prefer aggregation. To maximize aggregation benefits, the length of tandems is reduced such that all analyzed flows take the same path over it and can thus be considered forming a single flow aggregate. Figure 1b depicts this approach: Instead of a PSOO violation, a single left-over service curve for xf_1 and xf_2 suffices at S_0. To achieve this, the flows' arrivals to S_0 need to be computed, PBOO is enforced. In total, the arrival bounding will thus consist of three left-over operations on shorter tandems instead of two on longer tandems.

3.3 Network Analyses

The above compositions of tandem analyses mostly take static, tandem-local information such as flow entanglement into account. In contrast, network analyses take a more network-wide view and break with the strict composition rules of SegrAB and AggrAB.

Tandem Matching Analysis (TMA) [2] We abbreviate the algebraic search for best bounds presented in [2] as Tandem Matching Analysis (TMA). From a conceptual point of view, it matches differently sized tandems onto the entire feed-forward network to define an order of tandems to be analyzed. This is done in an exhaustive fashion, yet, based on PBOO-applying segregation's inferiority, paths of flow *aggregates* are a restricting factor. Thus, AggrAB becomes one of the alternatives TMA considers – in Figure 1, no (sub)tandem has length >1 and thus TMA behaves exactly like AggrAB. TMA also leverages information about the order of subtandems and employs the output burstiness cap. This added flexibility in the tandem decomposition proved key for the most accurate algebraically derived DNC delay bounds to date. TMA also mitigates the combinatorial explosion of effort and shows good scaling of the analysis.

Linear Program / Unique Linear Program (LP / ULP) [9] Instead of searching for the best combination of algebraic operations and analyses to apply, LP and ULP analysis directly search for the best delay bound. I.e., these optimization-based analyses do not derive a left-over service curve (unlike OBA). They convert the entire network into an optimization formulation, a (set of) linear program(s), that relates backlogged periods of servers. Yet, the ultimately tight LP analysis suffers from unmitigated combinatorial explosion. In evaluations, the proposed heuristic ULP has been shown to only contribute little accuracy over TMA but at significantly longer analysis execution times [2].

4 Accuracy Improvements by SegrPMOO Cross-traffic Arrival Bounding

We investigate the only not yet investigated principle in the TMA column in Table 2. In Figure 1c, it enforces a segregation at S_1 but it allows for the PMOO

Principle	Tandem Analysis				Network Analysis		
	TFA	SFA	PMOOA	OBA	TMA	ULP	LP
Agg	✓	(✓)	(✓)	(✓)	(✓)	✓	✓
PBOO	✗	✓	✓	✓	✓	✓	✓
PMOO	✗	✗	✓	✓	(✓)	✓	✓
Order	✗	✓	✗	✓	(✓)	✓	✓
OBC	✓	✗	*NA*		✓	*NA*	
PSOO	*NA*	✗[1]	*NA*		(✓)	(✓)[7]	✓
SegrAB	*NA*				✗	*NA*	
AggrAB	*NA*				✓	*NA*	
good scaling	✓	✓	✓	✗[14]	✓	✗	✗

Table 2: Feature matrix of all current, mutually exclusive DNC analyses. Principle implementation: ✓ full, (✓) partial/selective, ✗ none, *NA* not applicable. [1]SFA requires arrival bounding for servers on the analyzed tandem.

principle for xf_1 and xf_2. In case these suffer from cross-traffic themselves, xf_3 interfering with xf_1 in our example, we can benefit from PMOO where aggregate arrival bounding (Figure 1b) is only capable of implementing the PBOO principle. In this section, we show that the segregation/PMOO-tradeoff in SegrPMOO can outperform the aggregation/PBOO-tradeoff provided by current AggrAB.

4.1 Introducing SegrPMOO

We call the SegrAB strategy that exclusively applies the PMOO analysis for each left-over service curve derivation SegrPMOO. Next we give a proof that SegrPMOO can indeed outperform AggrAB.

Proposition 1. *Cross-flow segregation paired with a PMOO analysis is able to obtain lower bounds on flow arrivals than its aggregating counterpart. That is, none of these arrival bounding alternatives is a dominating approach.*

Proof. The superiority of AggrAB employing PBOO over the segregated version has been discussed in [5]. For the case that AggrAB implements either PBOO or PMOO and SegrAB implements PMOO, we give an example where cross-flow segregation yields a better result. Let us therefore consider the setting as in Figure 1 with token-bucket arrivals ($\mathcal{F}_{\text{TB}}$) and rate-latency service ($\mathcal{F}_{\text{RL}}$). First, we derive the arrival bound when aggregating cross flows:

$$
\begin{aligned}
&\alpha_{S_1}^{\text{Aggr}\{xf_1,xf_2\}} \\
&\quad = \alpha_{S_0}^{\{xf_1,xf_2\}} \oslash \beta_{S_0}^{\text{l.o.}\{xf_1,xf_2\}} \\
&\quad = \left(\left(\alpha^{xf_1} \oslash \beta_{S_{01}}^{\text{l.o.}xf_1}\right) + \left(\alpha^{xf_2} \oslash \beta_{S_{02}}^{\text{l.o.}xf_2}\right)\right) \oslash \left(\beta_{S_0} \ominus \alpha_{S_0}^{xxf_1}\right) \\
&\quad = \left(\left(\alpha^{xf_1} \oslash \left(\beta_{S_{01}} \ominus \alpha^{xf_3}\right)\right) + \left(\alpha^{xf_2} \oslash \beta_{S_{02}}\right)\right) \oslash \left(\beta_{S_0} \ominus \left(\alpha^{xf_3} \oslash \beta_{S_{01}}^{\text{l.o.}xf_3}\right)\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \left(\left(\alpha^{xf_1} \oslash \left(\beta_{S_{01}} \ominus \alpha^{xf_3}\right)\right) + \left(\alpha^{xf_2} \oslash \beta_{S_{02}}\right)\right) \oslash \left(\beta_{S_0} \ominus \left(\alpha^{xf_3} \oslash \beta_{S_{01}}\right)\right) \\
&= \left(\gamma_{r_1,b_1} \oslash \left(\beta_{R_{01},T_{01}} \ominus \gamma_{r_3,b_3}\right)\right) \\
&\quad + \left(\left(\gamma_{r_2,b_2} \oslash \beta_{R_{02},T_{02}}\right) \oslash \left(\beta_{R_0,T_0} \ominus \left(\gamma_{r_3,b_3} \oslash \beta_{R_{01},T_{01}}\right)\right)\right).
\end{aligned}
$$

We continue with

$$
\begin{aligned}
&\alpha_{S_1}^{\mathrm{Aggr}\{xf_1,xf_2\}} \\
&= \left(\left(\gamma_{r_1,b_1} \oslash \beta_{R_{01}-r_3,\frac{R_{01}\cdot T_{01}+b_3}{R_{01}-r_3}}\right) + \gamma_{r_2,b_2+r_2\cdot T_{02}}\right) \oslash \left(\beta_{R_0,T_0} \ominus \gamma_{r_3,b_3+r_3\cdot T_{01}}\right) \\
&= \left(\gamma_{r_1,b_1+r_1\cdot\frac{R_{01}\cdot T_{01}+b_3}{R_{01}-r_3}} + \gamma_{r_2,b_2+r_2\cdot T_{02}}\right) \oslash \beta_{R_0-r_3,\frac{R_0\cdot T_0+b_3+r_3\cdot T_{01}}{R_0-r_3}} \\
&= \gamma_{r_1+r_2,b_1+b_2+r_1\cdot\frac{R_{01}\cdot T_{01}+b_3}{R_{01}-r_3}+r_2\cdot T_{02}} \oslash \beta_{R_0-r_3,\frac{R_0\cdot T_0+b_3+r_3\cdot T_{01}}{R_0-r_3}} \\
&= \gamma_{r_1+r_2,b_1+b_2+r_1\cdot\frac{R_{01}\cdot T_{01}+b_3}{R_{01}-r_3}+r_2\cdot T_{02}} + (r_1+r_2)\cdot\frac{R_0\cdot T_0+b_3+r_3\cdot T_{01}}{R_0-r_3}.
\end{aligned}
$$

At this point, please note that the PBOO property is preserved as b_1 and b_2 occur only once. The PMOO property, on the other hand, does not hold anymore, as b_3 is included twice. The segregated version yields

$$
\begin{aligned}
\alpha_{S_1}^{\mathrm{Segr}\{xf_1,xf_2\}} &= \alpha_{S_1}^{xf_1} + \alpha_{S_1}^{xf_2} \\
&= \left(\alpha^{xf_1} \oslash \beta_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1}\right) + \left(\alpha^{xf_2} \oslash \beta_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2}\right) \\
&= \left(\gamma_{r_1,b_1} \oslash \beta_{R_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1},T_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1}}\right) + \left(\gamma_{r_2,b_2} \oslash \beta_{R_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2},T_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2}}\right) \\
&= \gamma_{r_1,b_1+r_1\cdot T_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1}} + \gamma_{r_2,b_2+r_2\cdot T_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2}} \\
&= \gamma_{r_1+r_2,b_1+b_2+r_1\cdot T_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1}+r_2\cdot T_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2}}.
\end{aligned}
$$

Using

$$
\begin{aligned}
T_{\langle S_{01},S_0\rangle}^{\mathrm{l.o.}xf_1} &= T_{01} + T_0 + \frac{b_2+b_3+r_3\cdot T_{01}+(r_2+r_3)\cdot T_0}{(R_{01}-r_3)\wedge(R_0-r_2-r_3)}, \\
T_{\langle S_{02},S_0\rangle}^{\mathrm{l.o.}xf_2} &= T_{02} + T_0 + \frac{b_1+b_3+(r_1+r_3)\cdot T_0}{R_{02}\wedge(R_0-r_1-r_3)}
\end{aligned}
$$

computed with [17] gives us

$$
\begin{aligned}
\alpha_{S_1}^{\mathrm{Segr}\{xf_1,xf_2\}} &= \gamma_{r_1+r_2,b_1+b_2+r_1\cdot\left(T_{01}+T_0+\frac{b_2+b_3+r_3\cdot T_{01}+(r_2+r_3)\cdot T_0}{(R_{01}-r_3)\wedge(R_0-r_2-r_3)}\right)} \\
&\qquad +r_2\cdot\left(T_{02}+T_0+\frac{b_1+b_3+(r_1+r_3)\cdot T_0}{R_{02}\wedge(R_0-r_1-r_3)}\right) \\
&= \gamma_{r_1+r_2,b_1+b_2+r_1\cdot T_{01}+r_1\cdot T_0+r_1\cdot\frac{b_2+b_3+r_3\cdot T_{01}+(r_2+r_3)\cdot T_0}{(R_{01}-r_3)\wedge(R_0-r_2-r_3)}} \\
&\qquad +r_2\cdot T_{02}+r_2\cdot T_0+r_2\cdot\frac{b_1+b_3+(r_1+r_3)\cdot T_0}{R_{02}\wedge(R_0-r_1-r_3)}
\end{aligned}
$$

where the PMOO principle is implemented per flow xf_1 and xf_2. Yet, overall, b_3 appears twice. We bound all arrivals with equal token buckets and continue by comparing burst terms. As we are free to choose parameters, we set

$T_0 = T_{01} = T_{02} = b_1 = b_2 = 0$ and the arrival rates to be homogeneous ($r_1 = r_2 = r_3 =: r > 0$). We further assume the burst term b_3 to be >0. Assume now that the claim does not hold true yielding for the burst term

$$\begin{aligned} & b_{S_1}^{\mathrm{Aggr}\{xf_1, xf_2\}} < b_{S_1}^{\mathrm{Segr}\{xf_1, xf_2\}} \qquad (1) \\ \Leftrightarrow \quad & r \cdot \frac{b_3}{R_{01} - r} + r \cdot \frac{b_3}{R_0 - r} + r \cdot \frac{b_3}{R_0 - r} \\ & \qquad < r \cdot \frac{b_3}{(R_{01} - r) \wedge (R_0 - 2r)} + r \cdot \frac{b_3}{R_{02} \wedge (R_0 - 2r)} \\ \Leftrightarrow \quad & \frac{1}{R_{01} - r} + \frac{2}{R_0 - r} < \frac{1}{(R_{01} - r) \wedge (R_0 - 2r)} + \frac{1}{R_{02} \wedge (R_0 - 2r)}. \end{aligned}$$

In order to contradict the claim and prove the proposition, it is sufficient to give an example where Equation (1) cannot hold. For this, see Example 1 below.

Example 1. Choosing $r = 1$, $R_0 = 4$, and $R_{01} = R_{02} = 2$ in Equation (1) results in $\frac{5}{3} \overset{!}{<} \frac{3}{2}$.

5 Numerical Evaluation

Previous evaluations of the TMA [2] showed that its delay bounds are close to those computed with the ULP optimization. Yet, there is still a gap that can be significant for some outliers. In this section, we extend the previous evaluation by SegrPMOO to check if it can mitigate the cause for this gap and the outliers; either by being independently executed as a stand-alone arrival bounding or by deeply integrating it into to the search executed by TMA.

Analyzed Networks [2] provides Internet-like topologies, generated according to the general linear preference GLP model [10] ($m_0 = 20$, $m = 1$, $p = 0.4695$, $\beta_{\mathrm{GLP}} = 0.6447$). We extend the analysis of these homogeneous networks of sizes 20 and 40 devices (Table 3) with arrival curves set to $\gamma_{5\mathrm{Mbps},5\mathrm{Mb}} \in \mathcal{F}_{\mathrm{TB}}$ and service curves set to $\beta_{10\mathrm{Gbps},0} \in \mathcal{F}_{\mathrm{RL}}$.

Accuracy Metric We are interested in the gap that algebraic DNC analyses have to close to achieve the ULP optimization's accuracy. Our metric of choice is therefore

$$\text{Gap Closing [\%]} = \frac{|\mathrm{Delay}_{\mathrm{TMA}} - \mathrm{Delay}_{\mathrm{NewAnalysis}}|}{|\mathrm{Delay}_{\mathrm{TMA}} - \mathrm{Delay}_{\mathrm{ULP}}|}. \qquad (2)$$

Execution Time Metric To allow for meaningful extension and comparison of execution time measurements, computations were executed with the same tools (DiscoDNC [3] v2.2.3 and IBM CPLEX version 12.6.2) on the same hardware platform (2x Intel Xeon E5420 CPU, 12GB main memory) as in [2]. We measure the time it takes to analyze all flows in a given network.

5.1 Accuracy

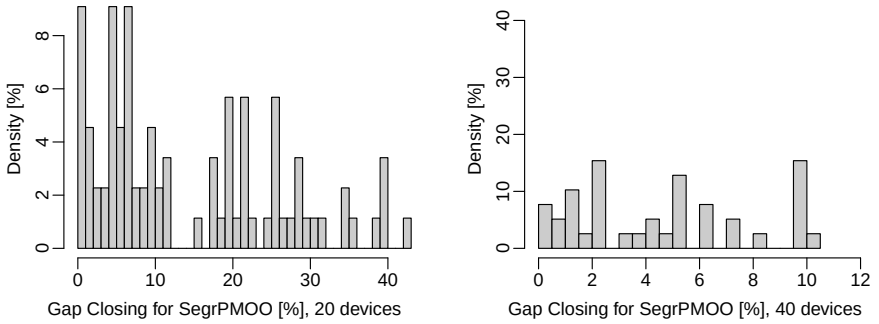
SegrPMOO Arrival Bounding vs. Optimization Analysis We first evaluate SegrPMOO in isolation. That is, we analyze the foi with TMA and compute the required cross-traffic arrival bounds with SegrPMOO only. This strategy defines the first *NewAnalysis* in Equation (2). Table 3a provides the results for networks of size 20 and 40 devices, translating to 152 and 472 flows to be analyzed respectively. While the share of improved delay bounds in the smaller network exceeds 50%, it already decreases to 10% in the larger network. Table 3 also gives the max and mean gap closing whereas Figure 2a shows the gap closing distributions for the flows that showed improved delay bounds.

Combined AggrAB and SegrPMOO vs. Optimization Analysis Second, we integrate SegrPMOO into AggrAB-based TMA arrival bounding. At every recursion level of TMA, i.e., when flows split up, a new arrival bounding is started (see Figure 1b, above server S_0). Here, we additionally execute SegrPMOO. For these SegrPMOO arrival boundings, the same holds vice versa. When they need to recursively bound arrivals of cross traffic, bounds are additionally derived with TMA. In both cases only the smaller of the derived bounds is considered.

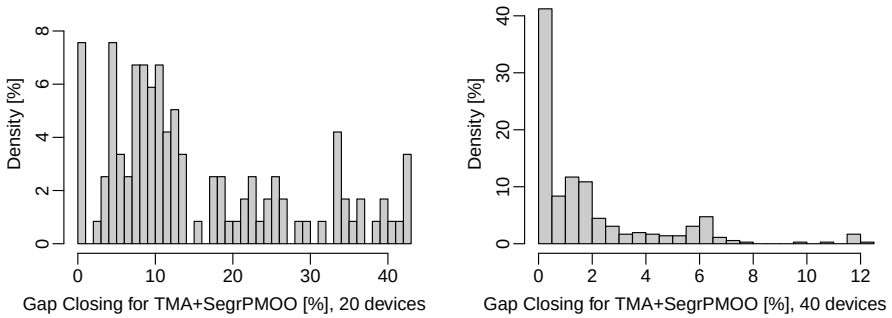
Results are depicted in Table 3b: a rather steady share of all flows, 78.3% (20 devices) and 72.5% (40 devices), respectively, sees improvements. I.e., in these cases, applying at least one cross-flow segregation during the entire arrival bounding process was beneficial over TMA only. Also, the shares are considerably larger than with SegrPMOO only. This result reveals a rather large amount of situations leading to segrPMOO superiority, although it depends on specific flow entanglements and parameter combinations. Note, that the latter solely occurs due to curve transformations as we evaluate homogeneous networks. The improvements themselves, however, are in proximity of the SegrPMOO results and considerably less pronounced in the larger network. The maximum reduction of the gap to the ULP shrinks from 42.9% to 12% and the mean from 15.5% to 1.88%. Figure 2b shows the gap closing distribution. Compared to SegrPMOO, mostly flows with small improvements are added. Yet, the 20 devices network sees a noticeable growth of the flows having their gap closed by >40%.

Devices	Servers	Flows		Gap Closing [%]	
		Total	Improved	Max	Mean [improved]
(a) TMA foi Analysis, only SegrPMOO Arrival Bounding					
20	38	152	88	42.36299	14.514300
40	118	472	39	10.03384	4.594554
(b) TMA foi Analysis, TMA+SegrPMOO Arrival Bounding					
20	38	152	119	42.92102	15.50147
40	118	472	342	12.00559	1.877968

Table 3: Closing the gap between TMA and ULP.



(a) Gap closing density of SegrPMOO for 20 devices (left) and 40 devices (right).



(b) Gap closing density of TMA+SegrPMOO for 20 devices (left) and 40 devices (right).

Figure 2: Closing the gap to ULP delay bounds.

Catching Outliers We also evaluated where TMA+SegrPMOO's impact is concentrated. Figure 3 depicts old and new deviation as well as gap closing for the 10 outlier flows that previously suffered from the largest gap to ULP. 8 out of 10 see their gap closed by more than 20% and the largest outlier even benefits from an improvement narrowing its gap from 4.26% to 3.21%. Our results show that the peculiar corner cases where segregation helps concentrate at the outliers.

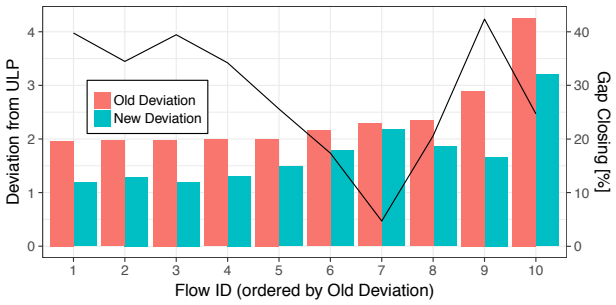
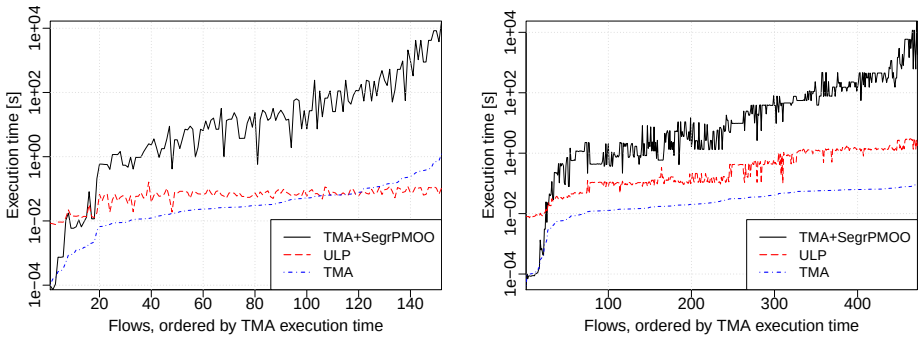


Figure 3: Gap closing with TMA+SegrPMOO for outliers, 20 devices network.



(a) 20 devices. (b) 40 devices.

Figure 4: Execution time comparison.

5.2 Computational Effort

The enhanced delay bound computation of TMA+SegrPMOO comes at the price of additional computational cost, as more arrival bounds are computed at each recursion level of the TMA analysis. We measured the execution time of each full analysis, i.e., for all flows in each network. The results are shown in Figure 4, ordered by the previously known TMA execution times. We observe that for cases with a very fast TMA computation, the added SegrPMOO overhead is negligible. In these cases, the recursion is not deep as little flows are involved and paths of relevant cross flows are rather short. Yet, the ULP analysis does not seem to be accelerated for these flows and their smaller optimization problem.

However, the amount of flows not requiring much arrival bounding is very small. Thus, we observe a sharp increase of computation times for the majority of flow analyses. TMA+SegrPMOO even takes considerably more time than the ULP analysis. Analyzing the 20 and 40 devices network only seems manageable due to their small sizes. In fact, we also observed that the 60 devices network presented in [2] becomes infeasible to fully analyze in acceptable time. This means design space exploration with TMA+SegrPMOO is out of scope although this network only consists of 164 servers and 656 flows. Nonetheless, if only a small number of flow delay bounds exceed their predefined deadlines, a selective, additional analysis of these flows comes at an acceptable execution overhead.

A TMA+SegrPMOO Heuristic Last, let us remark the potential for a heuristic that trades accuracy for faster computation. The later SegrPMOO is applied in the recursive arrival bounding, the shorter the tandem it analyzes. Thus, it becomes less likely that AggrAB enforces PBOO where the analysis could benefit from PMOO. This is reflected in Table 3 where the improvement from SegrPMOO to TMA+SegrPMOO seems small but might still be decisive. We leave heuristics selectively removing SegrPMOO from TMA for future work.

6 Conclusion

In this paper, we demonstrate that cross-flow segregation combined with the PMOO principle can outperform the predominating objective to aggregate flows. We contribute an analysis that incorporates this SegrPMOO approach into the existing TMA. Our numerical evaluations show that this new analysis outperforms others for the majority of analyzed flows. However, the improvement's amplitude can be small and comes at a considerably increased analysis cost. Our new TMA+SegrPMOO is thus most suitable for small networks or a follow-up analysis for selected flows, for instance to ensure strict certification requirements.

References

1. S. Bondorf. Better bounds by worse assumptions - improving network calculus accuracy by adding pessimism to the network model. In *Proc. of IEEE ICC*, 2017.
2. S. Bondorf, P. Nikolaus, and J. B. Schmitt. Quality and cost of deterministic network calculus – design and evaluation of an accurate and fast analysis. *ACM POMACS*, 1(1):16:1–16:34, 2017.
3. S. Bondorf and J. B. Schmitt. The DiscoDNC v2 – A Comprehensive Tool for Deterministic Network Calculus. In *Proc. of EAI ValueTools*, 2014.
4. S. Bondorf and J. B. Schmitt. Boosting Sensor Network Calculus by Thoroughly Bounding Cross-Traffic. In *Proc. of IEEE INFOCOM*, 2015.
5. S. Bondorf and J. B. Schmitt. Calculating Accurate End-to-End Delay Bounds – You Better Know Your Cross-Traffic. In *Proc. of EAI ValueTools*, 2015.
6. S. Bondorf and J. B. Schmitt. Improving Cross-Traffic Bounds in Feed-Forward Networks – There is a Job for Everyone. In *Proc. of GI/ITG MMB & DFT*, 2016.
7. S. Bondorf and J. B. Schmitt. Should network calculus relocate? an assessment of current algebraic and optimization-based analyses. In *Proc. of QEST*, 2016.
8. A. Bouillard. Algorithms and Efficiency of Network Calculus. Habilitation thesis, École Normale Supérieure, 2014.
9. A. Bouillard, L. Jouhet, and E. Thierry. Tight Performance Bounds in the Worst-Case Analysis of Feed-Forward Networks. In *Proc. of IEEE INFOCOM*, 2010.
10. T. Bu and D. Towsley. On Distinguishing between Internet Power Law Topology Generators. In *Proc. of IEEE INFOCOM*, 2002.
11. C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer, 2000.
12. R. L. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Trans. Inf. Theory*, 37(1):114–131, 1991.
13. R. L. Cruz. A Calculus for Network Delay, Part II: Network Analysis. *IEEE Trans. Inf. Theory*, 37(1):132–141, 1991.
14. A. Kiefer, N. Gollan, and J. B. Schmitt. Searching for Tight Performance Bounds in Feed-Forward Networks. In *Proc. of GI/ITG MMB & DFT*, 2010.
15. J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer, 2001.
16. J. B. Schmitt, F. A. Zdarsky, and M. Fidler. Delay Bounds under Arbitrary Multiplexing: When Network Calculus Leaves You in the Lurch ... In *Proc. of IEEE INFOCOM*, 2008.
17. J. B. Schmitt, F. A. Zdarsky, and I. Martinovic. Improving Performance Bounds in Feed-Forward Networks by Paying Multiplexing Only Once. In *Proc. of GI/ITG MMB*, 2008.