Better Bounds by Worse Assumptions – Improving Network Calculus Accuracy by Adding Pessimism to the Network Model

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

***Abstract*—Safety-critical systems require certification in order to attain permission to operate. Nowadays, these systems routinely embed a communication sub-systems whose performance must be formally verified for certification. Deterministic Network Calculus (DNC) can be employed for this task. It provides a mathematical framework to derive worst-case bounds on the delay of data flows, i.e., deterministic delivery guarantees. Their accuracy is decisive as even small improvements can change the outcome of certification. In general, delay bound accuracy depends on the accuracy of the network model. While previous work assumed that a more accurate model will invariantly yield more accurate delay bounds, we show that the opposite can actually be true. In this paper, we examine a specific weakness of DNC network analysis that leads to this counter-intuitive result. We make use of this insight in a mitigation strategy called flow prolongation. By prolonging the paths of flows, we let them interfere with more other flows but may still get better results. An evaluation in differently sized networks gives information about its impact on delay bound accuracy as well as analysis effort.**

I. Introduction

Deterministic Network Calculus (DNC) provides a mathematical framework for the worst-case analysis of communication systems. These have become an integral part of larger systems where they provide networking functionality. The embedded network is used by different services in a shared fashion. Among these services, there are usually safety-critical ones that must satisfy certain deadlines. Formal verification of delay guarantees is therefore a prerequisite for certification. DNC can derive deterministic delay bounds and has already been used in certifying the Ethernet-based backbone in current Airbus aircraft, e.g., the A380. This paper investigate a problem of DNC analysis that causes inaccurate delay bounds and thus inevitably leads to overprovisioned networks.

DNC started as an analysis of single servers, deriving server-local delay bounds that can be added up in order to attain a specific flow's end-to-end delay bounds. This procedure was, however, superseded by the idea to take on the analyzed flow (flow of interest, *foi*) in its entirety. As a flow of interest crosses a sequence of servers from its source to its sink, the newly established DNC analyses are known as *tandem analyses*. They can be composed to a network analysis. From a conceptual point of view, a DNC network analysis starts with the *foi* and backtracks cross-traffic tandem-by-tandem – recursively applying a tandem analysis on each of these. There is a multitude of DNC tandem analyses for various scenarios and with varying accuracy. For example, in FIFO-multiplexing server analysis, there is the Least Upper Delay Bound (LUDB) [1]. In this paper, we focus on arbitrary multiplexing, i.e., we assume no knowledge about multiplexing and derive worst-case delay bounds accordingly. For this setting, there are multiple tandem analyses, too: Pay Multiplexing Only One Analysis (PMOOA) [2], multi-dimensional convolution [3] and the Optimization-Based Analysis (OBA) [4].

Previously, it was assumed that a more accurate network model, i.e., as little pessimistic as possible, will result in more accurate delay bounds. However, as we prove in this paper, these DNC tandem analyses have a common fundamental problem that can cause the opposite. A network model consists of worst-case descriptions of service capabilities as well as traffic arrivals. Whereas the service is shared by all flows crossing a server, we aim to derive delay bounds for a single *foi*, based on its respective arrivals. The analysis needs to consider interference with cross-traffic correctly. I.e., in order to account for demultiplexing before the *foi*'s sink correctly, cross-traffic arrivals must be bound segregately [5]. This creates worst-case assumptions in the network analysis that cannot be attained simultaneously. DNC-derived delay bounds loose accuracy and networks must be overprovisioned in order to pass certification. In this paper, we contribute the following: We prove that segregation's negative impact on the assumed cross-traffic burstiness can be alleviated by adding pessimism to the network model – representatively in an analysis that applies PMOOA on tandems. The strategy to add pessimism is called flow prolongation as it extends the path of cross-flows and thus their interference with the *foi*. Our evaluation then extends this insight by an investigation of this counter-intuitive method's impact on delay bound accuracy as well as the analysis effort.

The paper proceeds as follows: Section II presents the related work and Section III provides the required background on Deterministic Network Calculus. In Section IV, we introduce the concept of flow prolongation, prove that it alleviates the segregation problem of DNC tandem analyses and assess its costs. Then, Section V evaluates the addition of flow prolongation to the DNC analysis, with respect to delay bounds and analysis effort. Section VI concludes the paper.

*This work was supported by the Carl Zeiss Foundation.

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

II. Related Work

Recent work investigated the composition of DNC tandem analyses to a DNC analysis of feed-forward networks. This work focussed on bounding of cross-traffic arrivals. In [5], the authors propose to aggregately bound cross-flows and show its superiority to the previous approach that segregated flows before analyzing them. Aggregation was shown to be the countermeasure to segregation as it reduces the amount of worst-case left-over service computations. Applied to two flows crossing the same server, their respective left-over operations create an unattainable worst-case – in total, segregation is paid for more than once [6]. Improvements by enforcing an aggregate analysis of cross-flows was also shown in [7]. The backlog bound at a server holds for all flows crossing it; not only those that interfere with the *foi*. This work shows that, due to aggregation of all flows, this bound can be smaller than the burstiness derived for cross-traffic only. Thus, aggregation's benefits can be used to cap cross-traffic burstiness.

The literature also reveals a potentially negative interdependency between individual DNC tandem analyses that are composed to a feed-forward network analysis [6]: The prerequisite to segregately bound the arrivals of cross-flows in order to account for demultiplexing correctly. Yet, it did not formally prove that this leads to overly-pessimistic bounds on cross-traffic arrivals. We provide this prove and show that the proposed flow prolongation counteracts this segregation by allowing for aggregation during the analysis.

Flow prolongation transforms the network model into a more pessimistic one by assuming that cross-flows take more hops than they actually do. This assumption can be found in the DNC literature, yet for different reasons than reducing the burstiness of cross-traffic. In [8], prolongation is presented as a computational reduction technique for arbitrary multiplexing networks. After prolonging, bounding the arrivals of a cross-traffic aggregate is less costly in terms of computational effort than bounding individual cross-flow arrivals. The potentially positive effect on DNC accuracy is, however, not considered. In previous work on FIFO multiplexing analysis, flow prolongation has been proven to potentially improve the delay bounds [9]. Key is the aggregation of the flow of interest with its cross-traffic. However, this is not possible in arbitrary multiplexing as we show in Section III. The DNC literature separately provides some insights on the benefits of aggregate analysis of flows as well as on flow prolongation. We complete this with a formal depiction of these aspects in conjunction with the DNC network analysis for arbitrary multiplexing.

III. Deterministic Network Calculus Background

DNC is based on a simple network model [10] with its operations cast in a $(\min,+)$-algebraic framework [11], [12].

A. The Network Model

Data Arrivals and Forwarding Service: Flows are characterized by functions cumulatively counting their data. They are belong to the set $\mathcal{F}_0$:

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

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*

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

Note, that the order of data within the flow needs to be retained for the (virtual) delay calculation [13].

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*

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

For example, flows periodically sending a maximum packet size b with a minimum inter-arrival time t_δ are upper bounded by the data arrival rate $r = \frac{b}{t_\delta}$. Their arrival curve is commonly referred to as token bucket and belongs to the set $\mathcal{F}_{\text{TB}} \subseteq \mathcal{F}_0$:

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

A server's forwarding service results in the output function $A'(t)$. This service 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*

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

For instance, service offered by Ethernet connections can be described by rate-latency curves from $\mathcal{F}_{\text{RL}} \subseteq \mathcal{F}_0$:

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

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 4 (Strict Service Curve)**.** *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 $\beta(d)$.*

B. (min,+)-Algebraic Deterministic Network Calculus

Definition 5 ((min,+)-Operations)**.** *The main (min,+)-algebraic DNC operations are for $f, g \in \mathcal{F}_0$ are*

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

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

Theorem 6 (*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:*

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

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

Graphically, delay bounding translates to the horizontal deviation between α and β. It assumes that there is no reordering of data within the flow, i.e., FIFO. Bounding the delay of a flow aggregate with the horizontal deviation thus demands that flows in the aggregate a multiplexed FIFO, too.

Analyzing a single flow under arbitrary multiplexing is enabled by the following theorem. With it, the *foi* can be segregated from its cross-traffic by deriving a lower bound on its share of service – a so-called left-over service curve.

Theorem 7 (*Left-Over Service Curve*)**.** *Consider a server s that offers 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 under arbitrary multiplexing at s, i.e., its left-over service curve at s, is*

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

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

Simultaneously assuming f_0's respective left-over service curve $\beta_s^{\text{l.o.}f_0}$ establishes mutual worst-case interference that cannot be attained by a realistic system [6].

The above left-over service curve operation is applicable to single servers only. In its evolution towards the analysis of tandems, multiple DNC tandem left-over service curves have been established: PMOOA [2], OBA [4], multi-dimensional convolution [3] or LUDB [1]. A recent overview over the principles they implement can be found in [14]. For a tandem of n servers $\langle s_1, \ldots, s_n \rangle$ with m cross-flows to be subtracted, they compute a left-over service curve

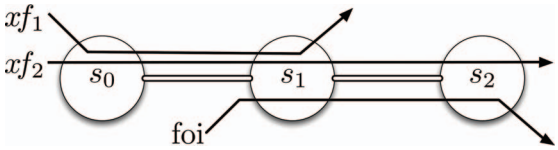
$$\beta_{\langle s_1,\ldots,s_n\rangle}^{\text{l.o.}} \quad = \quad \beta \ominus \left(\alpha^{f_1}, \ldots, \alpha^{f_m}\right)$$

where α^{f_i}, $i \in \{1, \ldots, m\}$ are the cross-flow arrival curves. In this paper, we focus on the derivation of these curves.

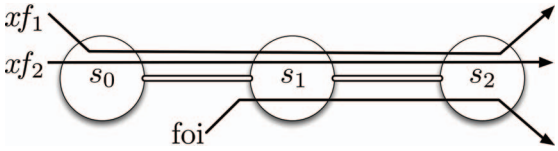
C. Feed-Forward Network Analysis

DNC is able to derive end-to-end delay bounds for individual flows traversing a feed-forward network. From a high-level point of view, the analysis consists of three parts:

1) Backtracking of flows to their respective sources where their arrival curve is known. This step results information about the interference between flows. DNC tandem analyses recursively backtrack in a tandem-by-tandem fashion. Higher recursion levels may enforce segregation on lower ones. E.g., xf_1 and xf_2 in Figure 1a need to be backtracked segregately instead of aggregately when using the previously mentioned DNC tandem analyses.



(a) Original, accurate network model.



(b) Overly-pessimistic network model after xf_1 was prolonged.

Figure 1: Sample network showing the basic setting where flow prolongation can result in more accurate delay bounds. Prolonging xf_1 influences the backtracking step such that it now operates on a flow aggregate instead of segregated flows.

2) Conversion of the previous step's interference information to a (min,+)-algebraic equation that bounds the flow of interest's delay. Part of this equation naturally bounds the arrivals of cross-traffic. In the example of Figure 1a, PMOOA computes cross-traffic arrivals as $\left(\alpha^{xf_1} \oslash \left(\beta_{s_0} \ominus \alpha_{s_0}^{xf_2}\right)\right) + \left(\alpha_{s_0}^{xf_2} \oslash \left(\beta_{s_0} \ominus \alpha_{s_0}^{xf_1}\right)\right)$ [5], i.e., it assumes unattainable mutual interference.
3) Solving the equation, e.g., by employing one of the available DNC implementations [15], [16], [17].

IV. Flow Prolongation

Flow prolongation transforms the network model into a more pessimistic one by assuming that cross-flows take more hops than they actually do. This is illustrated in Figure 1.

We depict the basic setting that enables flow prolongation (Figure 1a). In a network analysis, the first DNC tandem analyses is naturally applied to the *foi*. In order to correctly account for demultiplexing of cross-traffic on the *foi*'s path, cross-flows are segregated according to the (sub-)path they share with the *foi*. I.e., arrivals of xf_1 and xf_2 are derived segregately although they both share the same server just before interfering with the *foi*. After flow prolongation, xf_1 and xf_2 share the same (sub-)path of the *foi* (Figure 1b). This transformation crucially influences the backtracking step of DNC analysis. The cross-flows be bounded aggregately as $[xf_1, xf_2]$ – segregation's unattainable worst-case is circumvented. In this section, we briefly depict the added pessimism. Then, we prove reduced cross-traffic arrival bounds and show that they can result in more accurate delay bounds.

A. Additional Pessimism in the Network Model

Adding pessimism is decisive for the transformation by flow prolongation in order not to invalidate the worst-case of the original, accurate network model. Prolonging flows adds more interference to the analyzed *foi* at servers where its left-over service should actually be larger. This decreases the stability region of the network as overloaded servers can lead infinite delay bounds.

B. Improved Network Calculus Accuracy

Next, we show that the flow prolongation we propose is capable of improving the accuracy despite the pessimism added to the network model. We use the PMOOA tandem analysis of [2] to represent the class of DNC tandem analyses enforcing the flow segregation depicted above. Unlike [9], we do not prove an improved *foi* delay bound directly and specific to one multiplexing discipline. We rather prove the more general notion of reduced cross-traffic burstiness.

Theorem 8 (Flow Prolongation)**.** *Flow prolongation changes a DNC analysis' backtracking, allowing for aggregate cross-traffic arrival bounding that can improve the analyzed flow of interest's left-over service curve as well as delay bound.*

Proof: We will prove this statement by constructing a network with a *foi* whose delay bound become more accurate when prolonging a flow. We strive for a basic flow interference pattern on a tandem that can be found oftentimes in more involved network models; i.e., the one depicted in Figure 1a. Moreover, we instantiate this network model such that we can argue exclusively over the proceedings of the PMOOA [2]. For simplicity, we assume that the network is homogeneous with token-bucket arrival curves, $\alpha^{\text{foi}}, \alpha^{xf_1}, \alpha^{xf_2} \in \mathcal{F}_{\text{TB}}$, and rate-latency service curves, $\beta_{s_0}, \beta_{s_1}, \beta_{s_2} \in \mathcal{F}_{\text{RL}}$. Then, the PMOOA's left-over service curves for the *foi* are rate-latencies as well: $\beta^{\text{l.o.orig}}_{\langle s_1,s_2\rangle} = \beta_{R^{\text{l.o.}}_{\text{orig}},T^{\text{l.o.}}_{\text{orig}}} \in \mathcal{F}_{\text{RL}}$ is the original one of Figure 1a and $\beta^{\text{l.o.FP}}_{\langle s_1,s_2\rangle} = \beta_{R^{\text{l.o.}}_{\text{FP}},T^{\text{l.o.}}_{\text{FP}}} \in \mathcal{F}_{\text{RL}}$ is the flow prolongation one of Figure 1b. In this setting, these PMOOA left-over service curves are derived as follows.

$$\begin{aligned} R^{\text{l.o.}}_{\text{orig}} &= \left(R_{s_1} - r^{xf_1}_{s_1} - r^{xf_2}_{s_1}\right) \wedge \left(R_{s_2} - r^{xf_2}_{s_2}\right) \\ T^{\text{l.o.}}_{\text{orig}} &= T_{s_1} + T_{s_2} + \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}_{\text{orig}}} \\ &\quad + \frac{\left(r^{xf_1}_{s_1} + r^{xf_2}_{s_1}\right) \cdot T_{s_1} + r^{xf_2}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}_{\text{orig}}} \end{aligned}$$

and

$$\begin{aligned} R^{\text{l.o.}}_{\text{FP}} &= \left(R_{s_1} - r^{[xf_1,xf_2]}_{s_1}\right) \wedge \left(R_{s_2} - r^{[xf_1,xf_2]}_{s_1}\right) \\ T^{\text{l.o.}}_{\text{FP}} &= T_{s_1} + T_{s_2} + \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}_{\text{FP}}} \\ &\quad + \frac{r^{[xf_1,xf_2]}_{s_1} \cdot T_{s_1} + r^{[xf_1,xf_2]}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}_{\text{FP}}} \end{aligned}$$

Due to homogeneity, we get $R^{\text{l.o.}}_{\text{orig}} = R^{\text{l.o.}}_{\text{FP}} =: R^{\text{l.o.}}$. Thus, the difference between the respective left-over latencies, $T^{\text{l.o.}}_{\text{orig}}$ and $T^{\text{l.o.}}_{\text{FP}}$, is decisive for DNC performance bounds. The backlog bound mentioned in Section II is the maximum vertical deviation and for rate-latency service, it is found at the latency term. I.e., a smaller latency will result in more accurate backlog and output bounds. The delay bound, i.e, the maximum horizontal deviation, decreases with decreasing left-over latency which that can be found in the fixed as well as cross-traffic-dependent part of the derivations.

The left-over latency increases if larger cross-traffic burstiness needs to be subtracted from the original service (see Theorem 7). Therefore, we show that $T^{\text{l.o.}}_{\text{orig}} > T^{\text{l.o.}}_{\text{FP}}$ can occur; depending on the way to bound the two cross-flows in our basic example. Cross-traffic aggregation is superior to cross-traffic segregation:

$$\begin{aligned} & T^{\text{l.o.}}_{\text{orig}} > T^{\text{l.o.}}_{\text{FP}} \\ \Leftrightarrow\ & T_{s_1} + T_{s_2} + \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} \\ & \quad + \frac{\left(r^{xf_1}_{s_1} + r^{xf_2}_{s_1}\right) \cdot T_{s_1} + r^{xf_2}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ & > T_{s_1} + T_{s_2} + \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} \\ & \quad + \frac{r^{[xf_1,xf_2]}_{s_1} \cdot T_{s_1} + r^{[xf_1,xf_2]}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ \Leftrightarrow\ & \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} + \frac{\left(r^{xf_1}_{s_1} + r^{xf_2}_{s_1}\right) \cdot T_{s_1} + r^{xf_2}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ & > \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} + \frac{r^{[xf_1,xf_2]}_{s_1} \cdot T_{s_1} + r^{[xf_1,xf_2]}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ \Leftrightarrow\ & (\mathcal{F}_{\text{TB}} \text{ rate aggregation == addition}) \\ & \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} + \frac{\left(r^{xf_1}_{s_1} + r^{xf_2}_{s_1}\right) \cdot T_{s_1} + r^{xf_2}_{s_2} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ & > \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} + \frac{\left(r^{xf_1}_{s_1} + r^{xf_2}_{s_1}\right) \cdot T_{s_1} + \left(r^{xf_1}_{s_2} + r^{xf_2}_{s_2}\right) \cdot T_{s_2}}{R^{\text{l.o.}}} \\ \Leftrightarrow\ & \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} > \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} + \frac{r^{xf_1}_{s_1} \cdot T_{s_2}}{R^{\text{l.o.}}} \\ \Leftrightarrow\ & (\text{stability condition: } \frac{r^{xf_1}_{s_1}}{R^{\text{l.o.}}} \leq 1) \\ & \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} > \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} + T_{s_2} \\ \Leftrightarrow\ & (\text{assume } T_{s_2} = 0) \\ & \frac{b^{xf_1}_{s_1} + b^{xf_2}_{s_1}}{R^{\text{l.o.}}} > \frac{b^{[xf_1,xf_2]}_{s_1}}{R^{\text{l.o.}}} \\ \Leftrightarrow\ & b^{xf_1}_{s_1} + b^{xf_2}_{s_1} > b^{[xf_1,xf_2]}_{s_1} \end{aligned}$$

Next, we need to compare the derivation of these cross-traffic burstiness terms. The left-hand side derives both bursts segregately:

$$\begin{aligned} b^{xf_1}_{s_1} + b^{xf_2}_{s_1} &= \left(\alpha^{xf_1} \oslash \left(\beta_{s_0} \ominus \alpha^{xf_2}_{s_0}\right)\right)(0) \\ &\quad + \left(\alpha^{xf_2}_{s_0} \oslash \left(\beta_{s_0} \ominus \alpha^{xf_1}_{s_0}\right)\right)(0) \\ &= \left(\alpha^{xf_1}_{s_0} \oslash \beta^{\text{l.o.}xf_1}_{s_0}\right)(0) + \left(\alpha^{xf_2}_{s_0} \oslash \beta^{\text{l.o.}xf_2}_{s_0}\right)(0) \\ &\quad \text{(service curves} \in \mathcal{F}_{\text{RL}} \text{ and arrival curves} \in \mathcal{F}_{\text{TB}}\text{)} \\ &= \left(\alpha^{xf_1}_{s_0} \oslash \beta^{\text{l.o.}xf_1}_{s_0} + \alpha^{xf_2}_{s_0} \oslash \beta^{\text{l.o.}xf_2}_{s_0}\right)(0) \\ &\quad \text{(arrival homogeneity: } \beta^{\text{l.o.}xf_1}_{s_0} = \beta^{\text{l.o.}xf_2}_{s_0} =: \beta^{\text{l.o.}xf}_{s_0}\text{)} \\ &= \left(\alpha^{xf_1}_{s_0} \oslash \beta^{\text{l.o.}xf}_{s_0} + \alpha^{xf_2}_{s_0} \oslash \beta^{\text{l.o.}xf}_{s_0}\right)(0) \\ &\quad \text{(distributivity of } \oslash \text{ w.r.t. + [18])} \\ &= \left(\left(\alpha^{xf_1}_{s_0} + \alpha^{xf_2}_{s_0}\right) \oslash \beta^{\text{l.o.}xf}_{s_0}\right)(0) \\ &\quad \text{(arrival curves} \in \mathcal{F}_{\text{TB}}\text{)} \\ &= \left(\alpha^{[xf_1,xf_2]}_{s_0} \oslash \beta^{\text{l.o.}xf}_{s_0}\right)(0) \end{aligned}$$

In contrast, the right-hand burstiness is derived aggregately thanks to flow prolongation:

$$b_{s_1}^{[xf_1,xf_2]} = \left(\alpha_{s_0}^{[xf_1,xf_2]} \oslash \beta_{s_0}\right)(0)$$

Last, as we do not consider the trivial case of null arrivals, we know that $\beta_{s_0}^{\text{l.o.}xf} < \beta_{s_0}$. Thus, under the given assumptions, we get $b_{s_1}^{xf_1} + b_{s_1}^{xf_2} > b_{s_1}^{[xf_1,xf_2]}$ which leads to $\beta_{\langle s_1,s_2\rangle}^{\text{l.o.orig}} < \beta_{\langle s_1,s_2\rangle}^{\text{l.o.FP}}$ and finally to better performance bounds in the more pessimistic, flow prolonged network model. ■

In the above proof of concept, we constructed one specific instantiation of the network model where flow prolongation outperforms the original network analysis. The assumptions of this instantiation might not always be fulfilled. However, flow prolongation can still have a positive impact in many more instantiations of the model given in Figure 1. We illustrate this by an example where we break with the above assumptions..

Example 9. Take the following instantiations of the rate-latency service curves and token-bucket arrival curves in the networks depicted in Figure 1:

- $\beta_{s_0} = \beta_{8,4}$, $\beta_{s_1} = \beta_{13,5}$, $\beta_{s_2} = \beta_{12,2}$, and
- $\alpha^{\text{foi}} = \gamma_{2,2}$, $\alpha^{xf_1} = \gamma_{3,8}$, $\alpha^{xf_2} = \gamma_{4,10}$

I.e., we have a heterogeneous network where flow prolongation creates a bottleneck at s_2 – unlike the flow prolongation proposed in [8]. Nonetheless, using the derivations given in the above proof, we get the following left-over service curves:

- $\beta_{\langle s_1,s_2\rangle}^{\text{l.o.orig}} = \beta_{6,27.75}$ and
- $\beta_{\langle s_1,s_2\rangle}^{\text{l.o.FP}} = \beta_{5,26}$.

In this example, $\beta_{\langle s_1,s_2\rangle}^{\text{l.o.FP}}$ is not strictly smaller than $\beta_{\langle s_1,s_2\rangle}^{\text{l.o.orig}}$, both curves intersect. However, given the low arrival rate and small burstiness of the *foi*, the smaller latency derived in the flow prolongated network is still decisive as the bounds show:

- Delay bounds are $D^{\text{orig}} = 28\frac{1}{12} > 26\frac{2}{5} = D^{\text{FP}}$ and
- backlog bounds are $B^{\text{orig}} = 57\frac{1}{2} > 54 = B^{\text{FP}}$.

Departing from the clear setting we used in the proof might reduce the segregation's impact but the potential for improvement persists – even if the network is heterogeneous or if flow prolongation creates a bottleneck that reduces the entire path's left-over service rate in the PMOOA derivation.

C. The Generic Flow Prolongation Scheme

In Figure 1, the tandem consisting of servers s_1 and s_2 represents the basic building block of flow prolongation. The tandem of servers where flows might be prolonged is defined by the analyzed flow in the current step of the feed-forward network analysis (cf. Section III-C). In the depicted case, the *foi* takes two hops and there is one flow that does not fully take them with the *foi* to the end. This flow is then prolonged by one hop. In general feed-forward networks, this scheme is extended to cover all potential prolongations for all cross-flows. I.e., every flow not sharing the tandem to the end will be prolonged hop-by-hop. For more involved flow interference patterns, this scheme generates a large set of flow prolongation alternatives, each of which results in a valid left-over service curve. These need to be computed in order to later judge their impact on the entire feed-forward analysis (Figure 1: aggregation of flows at server s_0) and the eventually derived performance bounds.

D. Effort Considerations

Previous work did not consider flow prolongation in the compositional DNC analysis of feed-forward networks. This consideration reveals that the search for the best flow prolongation alternative is prone to combinatorial explosion. During the recursive tandem-by-tandem backtracking in a feed-forward analysis, we additionally execute a hop-by-hop flow prolongation for every cross-flow on every tandem. Especially the naive, exhaustive approach depicted in the previous Subsection becomes infeasible easily. Therefore, we present countermeasures to the combinatorial explosion problem that do not impact the accuracy of our results. As we intend to evaluate the potential impact of flow prolongation, we leave the work on flow prolongation heuristics to future work.

From the network model itself, we cannot derive the best flow prolongation alternative for every tandem in the entire network analysis a priori. Neither can we simply derive the involved parameters for the alternatives and decide on-the-fly because every flow prolongation is followed by its specific backtracking recursion deriving the required cross-traffic arrival bounds (see Section III-C). However, we can counteract the combinatorial explosion by excluding flow prolongation alternatives that cannot increase aggregation of cross-flows:

1) We do not prolong flows entering the network at a server on the flow of interest's path. These flows' arrival curves are already known and cannot be derived differently – neither aggregately nor segregately.
2) If a cross-traffic aggregate interferes with the *foi*, we do not segregate flows from it and prolong them individually. I.e., already existing cross-traffic aggregates are retained by flow aggregate prolongation.
3) Prolongation is not strictly done hop-by-hop but such that aggregation is enabled during arrival bounding. This rule considers the inlink cross-flows join the *foi*'s path from. Different inlinks define different servers prior to the interference and thus inhibit aggregation.

We also combine these theoretical countermeasures with practical improvements of the DiscoDNC tool we use for our numerical evaluations [15]. We parallelized flow prolongation as well as convolve and cache alternative intermediate arrival bounds. With these efficiency improvements, we aim for an exhaustive search for the best flow prolongation analysis.

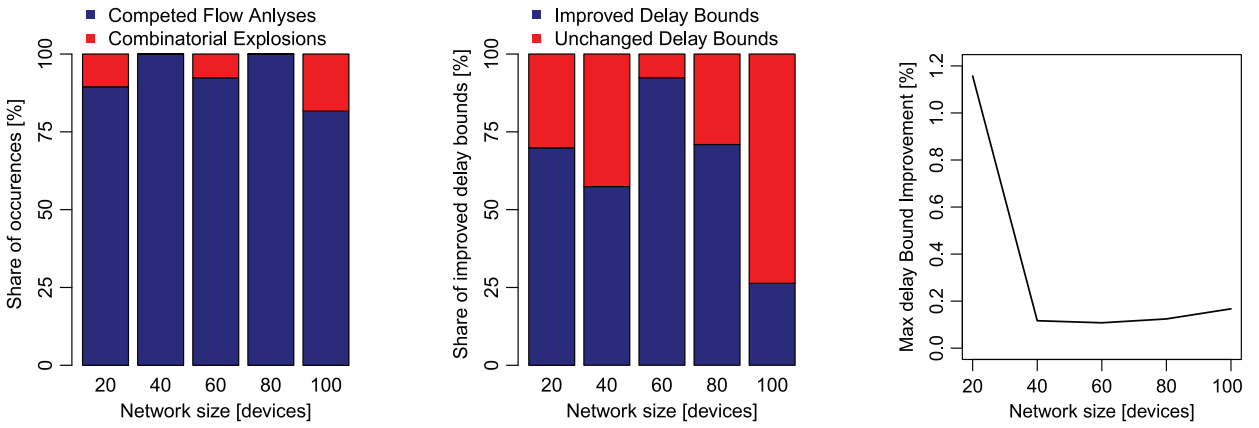
V. Numerical Evaluation

In this section, we investigate the potential improvement as well as effort of flow prolongation by numerical evaluation.

Evaluation Methodology: For our numerical investigation, we created Internet-like topologies according to the general linear preference (GLP) model [19]. We applied the default

Table I: Evaluated Network Sizes.

Devices	Servers	Flows
20	38	152
40	118	472
60	164	656
80	282	1128
100	364	1456



(a) Share of combinatorial explosions. (b) Share of improved delay bounds. (c) Maximum delay bound improvements.

Figure 2: Numerical evaluation of GLP networks of different sizes.

GLP parameter setting ($m_0 = 20$, $m = 1$, $p = 0.4695$, $\beta_{\text{GLP}} = 0.6447$) and used the aSHIIP tool [20] to generate these topologies. From DNC's point of view, nodes of the topology are network devices, each consisting of multiple servers that forward data via a specific outlink of the device. Therefore, this topology is converted to a server graph that directly connects the servers. The server graph is not necessarily feed-forward, this is achieved by applying turn prohibition to it [21]. Traffic was created with a fixed server-to-flow ratio of 1:4 to generate load – a server graph with flows is called network. Flows are routed on the shortest server graph path between two randomly chosen network devices. Table I shows the devices, servers and flows for all of the GLP networks we evaluate. Service curves correspond to 10Gbps Ethernet connections ($\beta_{10Gbps,0}$) and arrival are uniformly shaped to token buckets with rate 5Mbps and bucket size 5Mb ($\gamma_{5Mbps,5Mb}$).

Analysis Setting: The DiscoDNC [15] (version 2.2.6) automates the three steps of a compositional DNC feed-forward analysis. It is written in Java, we ran experiments with OpenJDK 8 on CentOS 7.2. Moreover, we extended the DiscoDNC by parallelization of two analysis aspects: Delay bounding of all flows in a network and the prolongation alternatives on every tandem in the analysis. These adaptations naturally increase the resource demand by creating numerous threads. Therefore, we ran our evaluation on a Colfax Ninja Workstation with an Intel Xeon Phi 7210 CPU and 110GB RAM. The CPU offers 64 multithreading-enabled physical cores, resulting in 256 logical cores. We allocated 96GB to the Java heap, an exhaustion of this large heap that, in turn, terminates the DNC analysis signals a combinatorial explosion.

A. Delay Bound Improvements

We executed entire network analyses, i.e., for every network size we attempted to compute the delay bound for every flow – with and without flow prolongation. None of our networks suffered from a bottleneck that impeded stability on a tandem with prolonged flows. However, some flows suffered from combinatorial explosion and thus we are not able to compare delay bounds for all flows. The share of flows we were able to analyze is between 81.6% and 100% (see Figure 2a and Table II). Among these flows, we can see a considerable share of improved delay bounds, especially in the networks smaller than 100 devices. Figure 2b and Table II show that in these networks, between 57.4% and 92.4% flows gained delay bound accuracy improvements larger than 10^{-12} (safety margin to rule out double rounding errors). In the 100 devices network, the share drops to 26.3%.

While the amount of improvements is considerable, our numerical evaluation reveals that their magnitude is relatively small. Figure 2c provides an overview over the maximum delay bound improvements and Figure 3 extents this depiction by two means: one taken over all flows and one taken over the improvements only. Most of these three improvement values are close together and well below 0.2% (see Table II). The only exception is the maximally observed delay bound improvement in the GLP network with 20 devices at just below 1.16%. Although the actual impact of these improvements can vary vastly, depending on the analyzed network and the usage of the analysis results, these observations suggest a low probability of decisive impact. Therefore, our numerical evaluation shows that the problem counteracted by flow prolongation does not have a severe impact in Internet-like topologies. If a considerable improvement of certain delay bounds is required,

Table II: Overview of delay bounding results.

Devices	Flows		
	Total	Analyzed	Improved
20	152	136	95
40	472	472	271
60	656	606	560
80	1128	1128	800
100	1456	1189	313

Devices	Improvements of flow delay bounds [%]		
	Max	Mean [Analyzed]	Mean [Improved]
20	1.157184	0.1855357	0.265609
40	0.1167132	0.009803669	0.01707502
60	0.1079412	0.008994453	0.009733283
80	0.1243899	0.005160176	0.007275849
100	0.1673514	0.00360915	0.01374475

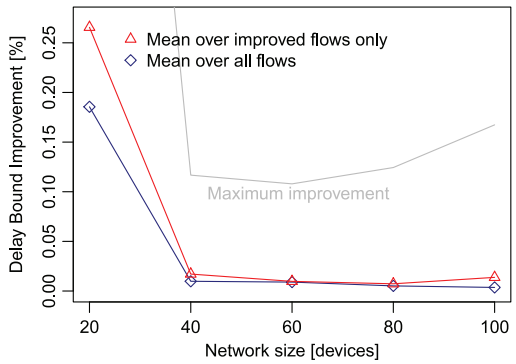


Figure 3: Comparison of delay bound improvements.

the network design needs to be improved instead of the DNC analysis. Next, we investigate if the involved effort justifies flow prolongation in the search for these small improvements we could observe.

B. Computational Effort Observations

Concerning the computational effort to analyze every flow of every network, we can report observations suggesting unpredictability. The common DNC measure for computational effort is analysis run-time [22] and most flow-prolongation delay bounds in the smallest network were computed within a day. However, we already experienced combinatorial explosions preventing the analysis of 16 flows (see Figure 2a and Table II). On the other hand, the GLP network with 40 devices was completely analyzed after 7 minutes and 16 seconds. The following network of size 60 was the first to require multiple days of computations for one of the flows' delay bounding, yet without exhausting the provided 96GB of heap space. This behavior carried forward to the final network with 100 devices. We ran its computations for two weeks straight before experiencing exhaustion of the heap space. Again, most of the individual flow analyses finished within a much shorter time of a couple of days while some others were responsible for this long computation time. Summing up, the computational cost of flow prolongation cannot be predicted. However, these observations suggest that our effort reductions presented in Section IV-D make an analysis feasible for the majority of flows.

VI. Conclusion

In this paper, we proved that a specific form of flow segregation enforced by DNC tandem analyses can be alleviated by flow prolongation – a technique transforming the network model into a more pessimistic one that positively impacts the DNC network analysis. Its impact is able supersede the added pessimism such that DNC performance bounds, in particular delay bounds, can be improved. Our numerical evaluation shows that this situation appears very often in a network analysis, yet, the observed delay bound reductions are small. While effort of flow prolongation is mostly manageable, it might suffer from combinatorial explosion that inhibits the derivation of bounds. Having added these two insights to the state-of-the art in DNC, we recommend to apply flow prolongation in a subsequent analysis of selected flows whose bounds require small improvements, e.g., to fulfill pre-defined delay guarantees, in order to maximize its potentially decisive impact while simultaneously minimizing the involved effort.

References

[1] L. Lenzini, E. Mingozzi, and G. Stea, "A Methodology for Computing End-to-End Delay Bounds in FIFO-Multiplexing Tandems," *Elsevier Performance Evaluation*, vol. 65, no. 11–12, pp. 922–943, 2008.

[2] 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*, March 2008, pp. 1–15.

[3] A. Bouillard, B. Gaujal, S. Lagrange, and E. Thierry, "Optimal Routing for End-to-End Guarantees Using Network Calculus," *Elsevier Performance Evaluation*, vol. 65, no. 11–12, pp. 883–906, 2008.

[4] 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*, April 2008.

[5] S. Bondorf and J. B. Schmitt, "Calculating Accurate End-to-End Delay Bounds – You Better Know Your Cross-Traffic," in *Proc. of the 9th International Conference on Performance Evaluation Methodologies and Tools (ValueTools)*, December 2015.

[6] ——, "Should Network Calculus Relocate? An Assessment of Current Algebraic and Optimization-based Analyses," in *International Conference on Quantitative Evaluation of Systems (QEST)*, 2016.

[7] ——, "Improving Cross-Traffic Bounds in Feed-Forward Networks – There is a Job for Everyone," in *Proc. GI/ITG MMB & DFT*, 2016.

[8] J. B. Schmitt, F. A. Zdarsky, and I. Martinovic, "Performance Bounds in Feed-Forward Networks under Blind Multiplexing," University of Kaiserslautern, Germany, Technical Report 349/06, 2006.

[9] L. Bisti, L. Lenzini, E. Mingozzi, and G. Stea, "Estimating the Worst-case Delay in FIFO Tandems Using Network Calculus," in *Proc. of the 3rd International Conference on Performance Evaluation Methodologies and Tools (ValueTools)*, October 2008.

[10] R. L. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–131, 1991.

[11] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer, 2001.

[12] C.-S. Chang, *Performance Guarantees in Communication Networks*. Springer, 2000.

[13] J. B. Schmitt, N. Gollan, S. Bondorf, and I. Martinovic, "Pay Bursts Only Once Holds for (Some) non-FIFO Systems," in *Proc. of IEEE INFOCOM*, April 2011.

[14] S. Bondorf, "Worst-Case Performance Analysis of Feed-Forward Networks – An Efficient and Accurate Network Calculus," Ph.D. dissertation, 2016.

[15] S. Bondorf and J. B. Schmitt, "The DiscoDNC v2 – A Comprehensive Tool for Deterministic Network Calculus," in *Proc. of the 8th International Conference on Performance Evaluation Methodologies and Tools (ValueTools)*, December 2014.

[16] L. Thiele, S. Chakraborty, and M. Naedele, "Real-Time Calculus for Scheduling Hard Real-Time Systems," in *Proc. ISCAS*, March 2000.

[17] 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. of the 4th international conference on Leveraging applications of formal methods, verification, and validation (ISoLA)*, October 2010.

[18] S. Bondorf and J. B. Schmitt, "Boosting sensor network calculus by thoroughly bounding cross-traffic," in *Proc. of IEEE INFOCOM*, April 2015.

[19] T. Bu and D. Towsley, "On Distinguishing between Internet Power Law Topology Generators," in *Proc. of IEEE INFOCOM*, June 2002.

[20] J. Tomasik and M.-A. Weisser, "Internet Topology on AS-level: Model, Generation Methods and Tool," in *Proc. of the 29th IEEE International Performance Computing and Communications Conference (IPCCC)*, December 2010.

[21] D. Starobinski, M. Karpovsky, and L. A. Zakrevski, "Application of Network Calculus to General Topologies Using Turn-Prohibition," *IEEE/ACM Transactions on Networking*, vol. 11, no. 3, pp. 411–421, 2003.

[22] S. Bondorf, P. Nikolaus, and J. B. Schmitt, "Delay bounds in feed-forward networks – a fast and accurate network calculus solution," *arXiv:1603.02094 [cs.NI]*, Mar. 2016.