Generalizing Network Calculus Analysis to Derive Performance Guarantees for Multicast Flows

Steffen Bondorf
Distributed Computer Systems (DISCO) Lab
University of Kaiserslautern
D-67653 Kaiserslautern, Germany
bondorf@cs.uni-kl.de

Fabien Geyer
Airbus Group Innovations
Dept. TX2P
D-81663 Munich, Germany
fabien.geyer@airbus.com

ABSTRACT

Guaranteeing performance bounds of data flows is an essential part of network engineering and certification of networks with real-time constraints. A prevalent analytical method to derive guarantees for end-to-end delay and buffer size is Deterministic Network Calculus (DNC). Due to the DNC system model, one decisive restriction is that only unicast flows can be analyzed. Previous attempts to analyze networks with multicast flows circumvented this restriction instead of overcoming it. E.g., they replaced the system model with an overly-pessimistic one that consists of unicast flows only. Such approaches impair modeling accuracy and thus inevitably result in inaccurate performance bounds.

In this paper, we approach the problem of multicast flows differently. We start from existing DNC analyses and generalize them to handle multicast flows. We contribute a novel analysis procedure that leaves the network model unaltered, preserves its accuracy, allows for DNC principles such as pay multiplexing only once, and therefore derives more accurate performance bounds than existing approaches.

CCS Concepts

•**Networks** → **Network performance evaluation;** *Network performance analysis;* Network performance modeling; •**Computing methodologies** → **Symbolic and algebraic algorithms;** *Symbolic calculus algorithms;*

Keywords

Delay bounds, deterministic network calculus, feed-forward networks, multicast flows

1. INTRODUCTION

Distributed embedded electronic applications communicating via packet networks have become the norm in various industries such as automotive, avionic or automation. In such industrial applications, real-time constraints on packet delay and jitter are usually required in order to ensure the specified processes behavior. Due to certification of systems as well as reliability demands, formal methods are applied to validate these timing constraints. They allow for hard guarantees via upper bounds. While different analytical methods have been proposed in the literature, Deterministic Network Calculus (DNC) established itself as common tool to analyze asynchronous communications in packet networks. A concrete example of this is Avionic Full-Duplex Ethernet (AFDX), a communication technology based on Ethernet and already deployed in avionic systems. Network calculus has proven a key tool for the certification of the deterministic property of the networks used for fly-by-wire [10].

An important property of those industrial networks is that communications are usually based on the multicast paradigm, where packets being sent by one sender are duplicated by switching elements in the network and received by multiple receivers. Using DNC on such multicast protocols requires some adaptations, since this method is restricted to the analysis of unicast communications. As detailed later, in Section 3, previous attempts for using DNC to analyze multicast communications only circumvented its current restriction. They do not provide a solution to overcome this limitation. Those approaches cannot benefit from all DNC capabilities to provide accurate end-to-end guarantees and networks designed based on them will be over-dimensioned.

We address the open issue of multicast flow analysis with DNC. We contribute two approaches that turn out to be steps generalizing existing analyses. The first one, Explicit Intermediate Bounds (EIB), is an approach where multicast flows are cut into sequences of unicast sub-flows. End-to-end performance bounds are then derived from sub-flow results. It does not require a transformation of the network, however, it amends a step to the analysis. Our second generalization finally leads to a DNC multicast feed-forward analysis. Neither transforming the network nor cutting any flows is required. Therefore, more accurate bounds are obtained since existing DNC principles can be applied in order to reduce effects such as flow multiplexing or burstiness. We numerically evaluate our proposed methods on two AFDX networks given in the literature and show that our DNC results are on par with other analytical methods or outperform them.

This paper is organized as follows: Section 2 gives a brief background on deterministic network calculus and we derive the foundation for our generalizations. In Section 3, we present related work on multicast flow analysis. Sections 4 and 5 contribute generalizations of DNC analyses for the study of multicast flow guarantees. We evaluate our approaches in Section 6 and Section 7 concludes the paper.

ACM ISBN .
DOI:

VALUETOOLS 2016, October 25-28, Taormina, Italy

DOI 10.4108/eai.25-10-2016.2266598

2. NETWORK CALCULUS BACKGROUND

We present in this section a brief overview of deterministic network calculus. For a more in-depth description, we refer the reader to [9] and [14].

2.1 Flow and Server Modeling

In network calculus, flows correspond to unidirectional and unicast communications between two servers. They are modeled as functions of their cumulative arrival of data. More formally, those functions belong to the following set $\mathcal{F}_0$ of non-negative, wide-sens increasing functions:

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

In order to compute bounds on the flows, we are interested in the functions $A(t)$ corresponding to the data arriving in a given server s at time t, and $A'(t)$ the amount of data processed by the server at time t. Using this formalism, the following delay definition can then be derived:

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

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

Instead of directly working with A, network calculus makes use of the concept of arrival curves, which is a function bounding the maximal arrivals of a flow:

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

$$A(t) - A(s) \leq \alpha(t-s), \forall t, s, 0 \leq s \leq t$$

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*

$$A'(t) \geq \inf_{0 \leq s \leq t} \{A(t-s) + \beta(s)\}, \forall t$$

2.2 (min, +) Algebra

Network calculus was formalized as a $(\min, +)$-algebraic framework in [9, 14], enabling an easier description of operations on flow and server descriptions.

DEFINITION 4 ($(\min, +)$ OPERATIONS). *The $(\min, +)$ convolution and deconvolution of two functions $f, g \in \mathcal{F}_0$ are defined as:*

$$\textit{Convolution: } (f \otimes g)(t) = \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\}$$

$$\textit{Deconvolution: } (f \oslash g)(t) = \sup_{s \geq 0} \{f(t+s) - g(s)\}$$

Using those $(\min, +)$ operations, one can rewrite the previous definitions as $A' \geq A \otimes \beta$ and $A \otimes \alpha \geq A$.

Moreover, $(\min, +)$ convolution allows DNC to concatenate the service of consecutive servers $\langle 1, \ldots, n \rangle$, so-called tandems, into a single service curve:

THEOREM 1 (CONCATENATION OF SERVERS). *Consider a single flow f crossing a tandem of servers $s_1, \ldots, s_n$ where each server s_i offers a service curve β_i. The overall service curve for f is their concatenation by convolution:*

$$\beta_i \otimes \cdots \otimes \beta_n = \bigotimes_{i=1}^{n} \beta_i$$

Given a strict service curve that guarantees a minimum output of β if data is present at a server, we lower bound the service left-over for a specific flow:

THEOREM 2 (LEFT-OVER SERVICE CURVE). *Consider a server s that offers a strict service curve β. Let s be crossed by flows f_0 and f_1, with arrival curves α_0, respectively α_1. Then the worst-case residual resource share under arbitrary multiplexing of f_1 at s is:*

$$\beta^{\text{l.o.}f_1} = \beta \ominus \alpha_0$$

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

Last, we use these curves to derive performance bounds.

THEOREM 3 (PERFORMANCE BOUNDS [14]). *Consider a flow f with arrival curve α traversing a server s with a service curve β. The following bounds can be derived:*

$$\textit{Backlog: } Q(t) \leq \sup_{s \geq 0}\{\alpha(s) - \beta(s)\} = (\alpha \oslash \beta)(0)$$

$$\textit{Delay: } D(t) \leq \inf\{d \geq 0 \mid (\alpha \oslash \beta)(-d) \leq 0\}$$

$$\textit{Output: } \alpha'(d) = (\alpha \oslash \beta)(d)$$

with α' being an output arrival curve for A'.

2.3 Network Analysis

Using the definitions and theorems presented above, the end-to-end performances of flows interacting on a network of servers can be computed. We call the analyzed flow *flow of interest*, abbreviated foi.

2.3.1 Tandems of Servers

The foi's path defines the sequence (tandem) of servers that defines its end-to-end delay. For bounding this delay, different methods have been proposed in the literature.

Total Flow Analysis (TFA) [14].

The TFA first computes per-server delay bounds. Each one holds for the sum of all the traffic arriving to a server, i.e., these bounds are independent of the foi. The flow's end-to-end delay bound is derived by summing up the individual server delay bounds on its path. The TFA's server-isolating approach constitutes a direct application of Theorem 3; it is known to be inferior to the following analyses [14, 18].

Separated Flow Analysis (SFA) [14].

The SFA is a direct application of other theorems: first compute the left-over service of each server on the foi's path using Theorem 2, then concatenate them using Theorem 1 and finally derive the end-to-end delay bound using Theorem 3. Deriving the end-to-end delay bound using only one service curve will consider the burst term of the foi only once, a property called Pay Burst Only Once (PBOO).

Pay Multiplexing Only Once (PMOO) [18].

The PMOO analysis first convolves the tandem of servers before subtracting the cross-traffic. Using this order, the bursts of the cross-traffic appear only a single time compared to the SFA analysis where the bursts are included at each server. Therefore, multiplexing with cross-traffic is only paid for once. However, [17] showed that the PMOO method does not necessarily outperform the SFA.

2.3.2 Feed-forward Networks

For more involved feed-forward networks, a procedure to combine tandem analyses to a network analysis exists. In order to integrate the analysis of multicast flows into DNC, we derive this procedure in great detail. Here, we contribute the following result: a precise structure of the steps taken by any DNC network analysis. The well-elaborated structure we establish in this section, also serves us to judge and compare different approaches to aiming for performance bounds in feed-forward networks with multicast flows.

In previous work, two basic steps of the analysis have already been identified [5]: *1)* cross-traffic arrival bounding and *2)* flow of interest performance bounding. They are tailored to a compositional unicast flow analysis. We call it the *unicastFFA* and derive its steps in unprecedented detail:

unicastFFA Step 1: Cross-traffic Arrival Bounding. The first unicastFFA step abstracts from the feed-forward network to the foi's path – a tandem of servers that can be analyzed with one of the existing procedures. In detail, this step proceeds as follows:

(i) Starting at the locations of interference with the foi, cross-flows are backtracked to their sources. This procedure derives the dependencies between the foi, its cross-flows, their cross-flows, etc., in a recursive fashion. A new instance of this sub-step is started for any cross-flow of the current cross-flow under consideration. Due to the network's feed-forward property, the recursion is guaranteed to terminate.

(ii) Next, the dependencies are converted into equations, i.e., a sequence of algebraic operations for each location of interference with the foi. They capture the worst-case transformation of flow arrivals towards foi.

(iii) Finally, the equations are solved to obtain the bounds on cross-traffic arrivals.

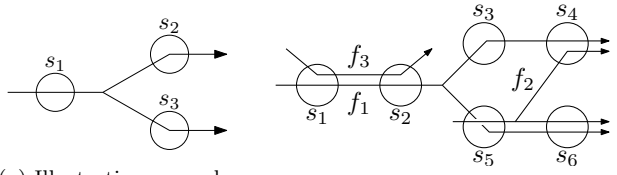
After these substeps, all cross-flows' arrivals are bounded with arrival curves (arrival bounds).

unicastFFA Step 2: foi Performance Bounding. Given the cross-traffic arrival bounds from step 1, step 2 does not need to consider the part of the network traversed by these flows nor the potentially complex interference patterns they are subject to. The foi's end-to-end delay bound in the feed-forward network is derived with a tandem analysis.

Note, that this step provides information required in the previous one. It defines the flow of interest and thus its cross-flows as well as their locations of interference used in step 1(i). This step is strongly based on the tandem analysis that, in turn, is derived from the aim to analyze a unicast flow from end to end. It is not directly applicable to the analysis of multicast flows and thus needs generalization.

2.4 Multicast Flows

As defined at the beginning of this section, flows and network analysis in network calculus have been mostly focused on the modeling of unidirectional and unicast communications. Such a model is not directly applicable to multicast network protocols, where packets are duplicated at certain points of the network in order to provide one-to-many communications as illustrated in Figure 1.



(a) Illustrative example. (b) Running example.

Figure 1: Multicast networks.

We define the following terms for describing parts of a multicast flow:

DEFINITION 5 (TRAJECTORY AND FORK). *A trajectory of a given source-sink pair corresponds to the equivalent unicast flow going from the source to the sink. A fork corresponds to a server where packets are duplicated.*

In the following, we will provide an illustration as well as a running example. They are meant to serve two distinct purposes: On the one hand, we depict the basic idea behind the approaches to handle multicast flows with the minimal network of Figure 1a. Additionally, we will analyze the network of Figure 1b with the given approach. Analyzing flow f_2 makes this network minimal w.r.t. covering all effects relevant to DNC and multicast flows: There one multicast flow in each step of the analysis, cross-traffic arrival bounding (f_1) as well as flow of interest analysis (f_2). Moreover, a unicast flow is present and this network allows us to observe the impact of different flow analyses described earlier in Section 2.3.1 (TFA, SFA with the PBOO effect, PMOO).

3. RELATED WORK

In this related work section, we provide two DNC approaches to the analysis of multicast flows. For both, we focus on how these approaches enable the unicastFFA of the previous section to analyze networks with multicast flows.

unicastFFA Transformation: A Set of Unicast Flows

A first approach to circumvent this issue is to transform a multicast flow to a set of unicast flows. It was mentioned as a possibility to cope with multicast flows in [6]. Each trajectory will become one independent unicast flow, as illustrated in our two sample networks (Figures 2a and 2b).

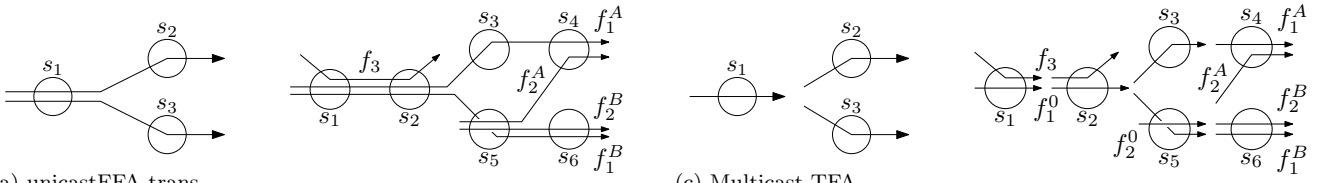
From a procedural point of view, the unicast transformation does not integrate into the unicastFFA. It only enables for using it by a preceding step that transforms the network. This step is static, i.e., it does not consider the unicastFFA's information like the flow(s) that are under analysis.

The foremost problem of this approach is its overly pessimistic assumption about resource demand of multicast flows. On common sub-paths of a multicast flows' trajectories, i.e., the servers before a fork, multiple unicast flows compete for resources. The unicastFFA thus models the worst case with mutual interference between these flows that are not present in the original network model.

On the other hand, this approach allows for the PBOO and the PMOO principle in the unicastFFA.

Multicast TFA

Grieu [11] proposes a procedure to apply the TFA presented in Section 2.3.1 in the analysis of multicast flows. It is tailored to the TFA and shares its inherent isolation of servers.



(a) unicastFFA transformation illustration.

(b) unicastFFA transformation applied to the running example.

(c) Multicast TFA illustration.

(d) Multicast TFA applied to the running example.

Figure 2: Existing DNC approaches to the multicast analysis applied to the networks presented in Figure 1.

Thus, it does not integrate into the unicastFFA for deriving delay bounds. Figures 2c and 2d depict this procedure on the illustration and the running example, respectively. Flows are cut between all servers, the arrivals are aggregated and a server-local delay bound is computed. In a second step, the server delay bounds on the trajectory of interest are summed up. As this last step is similar to the unicastFFA step 1, it inherits its decisive TFA shortcomings. I.e., neither the PMOO nor the PBOO principle are implemented and the delay bounds are known to be inaccurate.

Related Approaches

The existing DNC approaches both have significant drawbacks. Therefore, the literature created novel multicast analyses based on the DNC system description.

The *Trajectory Approach* (TA) is an adaptation to the study of network delays of the holistic approach [19]. It was originally developed to give bounds on the scheduling of tasks on a processor. The approach was initially proposed in [16] and later extended to FIFO systems in [15]. [1] applied TA to the study of avionic networks with multicast flows and showed, via numerical evaluations, that it outperforms the multicast TFA.

The *Forward End-To-End Delay Approach* (FA) has been proposed more recently in [13]. It addresses the shortcomings of the TA. Similarly to the TA, FA is also an adaptation of the holistic approach to the case of FIFO networks. [13] and [12] applied the FA to the performance evaluation of avionic networks with multicast flows and showed that this approach outperforms the multicast TFA as well.

Although FA sets its focus on the end-to-end analysis – similar to the DNC tandem analyses – neither FA nor TA have been benchmarked against a modern DNC that implements PBOO or PMOO. This can be attributed to the lack of such an analysis for multicast flows. In this paper, we will generalize multicast TFA as well as the unicastFFA in order to provide such DNC solutions and benchmarking results.

4. EXPLICIT INTERMEDIATE BOUNDS

Explicit Intermediate Bounds (EIB) analysis is the generalization of the multicast TFA analysis. We do not create an entirely new analysis (unlike TA and FA) but adapt the model such that we can analyze it with our NC tools. EIB combines the strengths of the two related approaches from above. Like the TFA, it follows the foi's trajectory of interest without additional cross-traffic assumptions. Moreover, it proceeds in tandems like the set of unicast flows. Therefore, it can benefit from both the PBOO and the PMOO principle on these tandems – unlike the server-local approach responsible for TFA's general inferiority w.r.t. delay bounds.

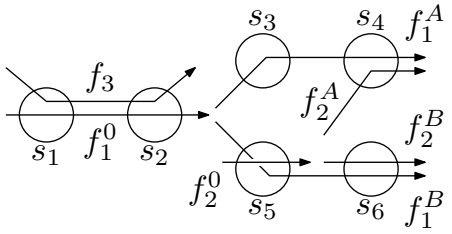


Figure 3: Application of the EIB analysis to the studied scenario: multicast flows are cut into unicast sub-flows.

We achieve this by an approach consisting of two parts:

1. A static, preceding step takes the network description and transforms it for analysis. In contrast to the multicast TFA, we cut multicast flows after forking locations only; not after every server. These are the locations of explicit intermediate bounds. In our illustrative example, the result is equal to the multicast TFA (Figure 2c). However, we transform the network into tandems that are crossed by parts of the multicast flows. This is revealed by our running example. For instance, compare the tandems $\langle 1, 2\rangle$ and $\langle 3, 4\rangle$ of multicast TFA (Figure 2d) with EIB (Figure 3).

2. The locations of cuts is static, i.e., only defined by the network, not the foi. The EIB analysis can be viewed to create a global worst case for all flows, independent of the foi. Afterwards it runs a unicastFFA in its second part.

These two steps are, however, not as separated as in the literature's multicast TFA. We integrate the EIB idea into the unicastFFA is as follows:

- Step 1: An adaptation of the cross-traffic arrival bounding is required at the locations of explicit intermediate bounds. Previously, tandems for analysis were defined by the common path of flow aggregates [6]. Using EIB, we have an additional restriction for the tandem lengths as we derive the output bound at the server a multicast flow forks. I.e., when the backtracking reaches such a server, the search for a tandem to operate on terminates. A new instance of unicastFFA step 1 is started with this server being the last one for the next tandem in the analysis.

- Step 2: The flow of interest analysis actually becomes a "trajectory of interest" analysis. Yet, for multicast flows it is not able to analyze the trajectory in an end-to-end fashion with a single left-over service curve.

A multicast flow's entire trajectory will consist of at least two tandems as it has at least one fork location to cut at. Thus, the PMOO principle cannot be implemented entirely; similar to the FIFO multiplexing tandem analysis LUDB [2].

We call this integrated procedure EIB unicastFFA; the name emphasizes the application of this specific way to obtain results with the EIB idea. Regarding the results themselves, EIB unicastFFA and EIB are synonymous as both of the above procedures yield the same performance bounds.

Analysis of the Running Example

As mentioned above, the difference between multicast TFA and EIB unicastFFA becomes apparent in more involved networks. We derive the left-over service curves for multicast flow f_2 of our running example (Figure 3) to depict the new analysis. We follow the EIB unicastFFA procedure that does not reveal a preceding EIB step.

EIB unicastFFA step 1. Bounding the delay of f_2 requires to bound the interference of multicast cross-flow f_1. The first location of interference with f_2 is at s_5 where we need to backtrack f_1^B. EIB enforces a cut after f_1's fork at server s_2 that we need to consider. We get $\alpha_5^{f_1^B} = \alpha_5^{f_1^0}$, i.e., the backtracking is stopped there. In a second instance of EIB unicastFFA step 1, f_1^0 is backtracked in order to derive its explicit intermediate bound at the output of s_2 / at the input of server s_5:

$$\alpha_5^{f_1^B} = \alpha^{f_1^0} \oslash \beta_{\langle 1,2\rangle}^{\text{l.o.}f_1^0}$$

In contrast to the multicast TFA, we can benefit from either SFA/PBOO or PMOO in the derivation of $\beta_{\langle 1,2\rangle}^{\text{l.o.}f_1^0}$. The according derivation that applies both alternatives is called aggregate arrival bounding, aggrAB, [6]:

$$\begin{aligned}\alpha^{f_1^0} \oslash \beta_{\langle 1,2\rangle}^{\text{l.o.}f_1^0} &= \left(\alpha^{f_1^0} \oslash \left(\left(\beta_1 \ominus \alpha^{f_3}\right) \otimes \left(\beta_2 \ominus \alpha_2^{f_3}\right)\right)\right) \\ &\quad \wedge \left(\alpha^{f_1^0} \oslash \left((\beta_1 \otimes \beta_2) \ominus \alpha^{f_3}\right)\right)\end{aligned}$$

where $\alpha_2^{f_3}$ corresponds to the arrival bound of f_3 at s_2.

At the second location of interference between f_1 and f_2, server s_4, we need to backtrack trajectory f_1^A. This yields

$$\begin{aligned}\alpha_4^{f_1^A} &= \alpha_5^{f_1^B} \oslash \beta_3 \\ &= \left(\alpha^{f_1^0} \oslash \beta_{\langle 1,2\rangle}^{\text{l.o.}f_1^0}\right) \oslash \beta_3 = \alpha^{f_1^0} \oslash \left(\beta_{\langle 1,2\rangle}^{\text{l.o.}f_1^0} \otimes \beta_3\right)\end{aligned}$$

Note, that we cannot derive a left-over service curve $\beta_{\langle 1,2,3\rangle}^{\text{l.o.}f_1^A}$ due to the cut enforced by EIB. In general, this prevents the implementing of the PMOO principle.

EIB unicastFFA step 2. Next, we bound the delay of trajectory f_2^A. Again, it cannot be done with a single, PMOO left-over service curve due to EIB's cut between s_5 and s_6. In this case, the cut means SFA is the only analysis option:

$$\begin{aligned}\beta^{\text{l.o.}f_2^A} &= \beta_{\langle 5,4\rangle}^{\text{l.o.}f_2^A} \\ &\text{(cut enforced by EIB, no single-tandem analysis)} \\ &= \beta_5^{\text{l.o.}f_2^A} \otimes \beta_4^{\text{l.o.}f_2^A} \\ &= \left(\beta_5 \ominus \alpha_5^{f_1^B}\right) \otimes \left(\beta_4 \ominus \alpha_4^{f_1^A}\right)\end{aligned}$$

where $\alpha_5^{f_1^B}$ and $\alpha_4^{f_1^A}$ have been derived in EIB unicastFFA step 1, the cross-traffic arrival bounding.

In the analysis of f_2, the delay bound for trajectory f_2^B remains to be bounded. Again, we need to derive the according left-over service curve that illustrates the proceedings of EIB unicastFFA:

$$\begin{aligned}\beta^{\text{l.o.}f_2^B} &= \beta_{\langle 5,6\rangle}^{\text{l.o.}f_2^B} \\ &\text{(cut enforced by EIB, no single-tandem analysis)} \\ &= \beta_5^{\text{l.o.}f_2^B} \otimes \beta_6^{\text{l.o.}f_2^B} \\ &= \left(\beta_5 \ominus \alpha_5^{f_1^B}\right) \otimes \left(\beta_6 \ominus \alpha_6^{f_1^B}\right)\end{aligned}$$

Most notably, f_2^B sees multi-hop interference by f_1^B but the left-over service curve derivation cannot make use of the PMOO principle. EIB inhibits an end-to-end analysis in this unicastFFA step 2, only SFA/PBOO can be applied

Theoretical Evaluation

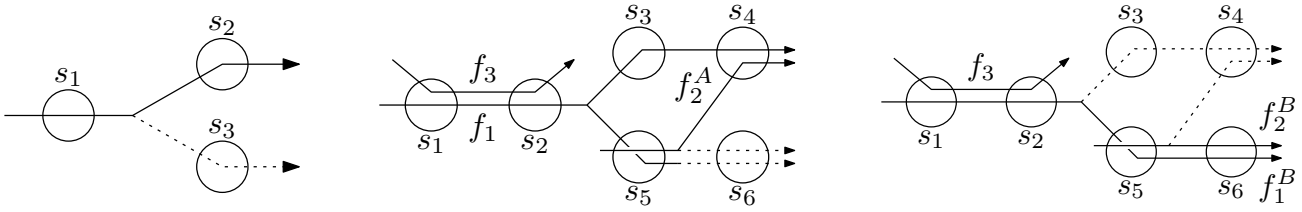
We conclude this section by a theoretical evaluation of the EIB approach against the previous DNC approaches:

- *Relation to multicast TFA:*
  The integration into the unicastFFA constitutes a generalization of the multicast TFA. On every tandem to analyze in either of the two unicastFFA steps, we can apply the TFA depicted in Section 2.3. Then, an additional intermediate step that separates the individual servers is executed and the per-server results are composed to the respective tandem result. The multicast TFA delay bounds cannot outperform those of EIB with either SFA or PMOO analysis on all tandems.
- *Relation to unicastFFA transformation:*
  In contrast to this section's EIB, the unicastFFA transformation allows for a single end-to-end left-over service curve for every trajectory of interest. I.e., it can fully benefit from the PMOO principle. However, the cuts enforced by EIB can also have a positive effect. They break the $\otimes$'s commutativity, allowing to better exploit service rates on the tandem. This is a variant of the problem shown in [17], but with a handicapped PMOO. None of the alternatives is strictly superior.

5. A MULTICAST FEED-FORWARD ANALYSIS PROCEDURE

In this section, we generalize the unicastFFA presented in Section 2.3.2 to networks with multicast flows. We call this generalized method *multicast Feed-Forward Analysis*, or mcastFFA. This allows us to make use of the knowledge only available in the unicastFFA itself. In contrast to the existing DNC approaches and the EIB analysis, no network transformation is amended to the analysis. We do not create a network-wide worst-cast setting for all flows before executing the unicastFFA. Instead, our generalization solely constructs a single flow of interest's worst case during analysis – a less pessimistic setting than the network-wide one for all flows at once. With this approach, the mcastFFA analysis obtains best results by exploiting the PMOO and PBOO principles to a larger extent than the EIB.

Figure 4a illustrates the basic idea behind our solution: If we analyze this multicast flow's trajectory crossing s_2,



(a) mcastFFA illustration. (b) Running example, trajectory f_2^A. (c) Running example, trajectory f_2^B.

Figure 4: Application of mcastFFA. The dashed lines depict parts of flows that are not considered in the current analysis.

the other trajectory crossing s_3 becomes irrelevant for the delay bound computation. We neither need to add an entire cross-flow for it nor do we require the output bound from s_1. Thus, mcastFFA can treat the multicast trajectory (or unicast flow) of interest in an end-to-end fashion and apply, for instance, the PMOO principle.

The main challenge of this approach is to reduce the network to relevant servers as well as (partial) flows and multicast flow trajectories. This may constitute considerable effort in large networks. Therefore, we present a solution that generalizes the unicastFFA analysis in order to gain from its efficiency [3]. I.e., deriving the sub-network relevant to a specific foi is part of the analysis proceedings, not covered by a separate step. Our mcastFFA solution is mainly based on the newly derived sub-step 1(i) of the unicastFFA: backtracking of dependencies.

In this step, dependencies of a flow on others are derived by traversing the network in the opposite direction of links [5]. The entire unicastFFA starts this procedure with the flow of interest. Our mcastFFA will iterate over all n trajectory of interest and execute separate analyses. In case of a unicast flow, we get $n = 1$; for multicast flows n equals the amount of trajectories (source/sink-pairs). Multicast cross-flows are traversed backwards, too, such that their fork locations do not enforce to cut the tandem to analyze; the relevant trajectory of the cross-flow is known and can be treated similar to a unicast cross-flow. Namely, the mcastFFA is a generalization of the known unicastFFA and thus it operates on longer tandems than EIB.

Whereas the EIB required to explicitly consider each location a multicast flow forks, the mcastFFA implicitly restricts the analysis to the trajectory relevant for the analysis. After the backtracking, we know the entire sub-network whose servers and (partial) flows appear in the analysis equation of unicastFFA step 1(ii). I.e., we rely on the detailed understanding of unicastFFA that we derived in Section 2.3.2.

Analysis of the Running Example

We will derive the left-over service curves for f_2's trajectories in order to compare them against the EIB unicastFFA. For brevity, we restrict the depiction to f_2^A's cross-traffic arrival bounding (mcastFFA step 1, Figure 4b) and f_2^B's delay bounding (mcastFFA step 2, Figure 4c). These derivations depict the crucial improvement of mcastFFA's proceedings in both of the analysis steps. They point out the reduction of the network and the increased tandem lengths.

mcastFFA step 1. We consider f_2^A's cross-traffic arrival bounding. Backtracking will be "local" to a single trajectory of a multicast cross-flow. In our example, we finally have established the possibility to apply the PMOO-principle when computing f_1^A's aggregate arrival bound aggrAB at server s_4 [6]. See $\alpha_4^{f_1^A}$ in the follwoing left-over service curve derivation we require to bound cross-traffic arrivals:

$$\begin{aligned}
\beta^{\text{l.o.}f_2^A} &= \beta^{\text{l.o.}f_2^A}_{\langle 5,4\rangle} \\
&\text{(only single-hop interference so cutting is fine)} \\
&= \beta_5^{\text{l.o.}f_2^A} \otimes \beta_4^{\text{l.o.}f_2^A} \\
&= \left(\beta_5 \ominus \alpha_5^{f_1^B}\right) \otimes \left(\beta_4 \ominus \alpha_4^{f_1^A}\right) \\
&= \left(\beta_5 \ominus \left(\alpha^{f_1} \oslash \beta^{\text{l.o.}f_1}_{\langle 1,2\rangle}\right)\right) \otimes \left(\beta_4 \ominus \left(\alpha^{f_1} \oslash \beta^{\text{l.o.}f_1^A}_{\langle 1,2,3\rangle}\right)\right)
\end{aligned}$$

A cut of $\beta^{\text{l.o.}f_1^A}_{\langle 1,2,3\rangle}$ into $\beta^{\text{l.o.}f_1}_{\langle 1,2\rangle} \otimes \beta_3^{\text{l.o.}f_1^A}$ was needed in the EIB analysis, meaning that PMOO could not be implemented.

This advantage is also depicted in Figure 4b where f_1 retains its multicast shape in the mcastFFA's point of view.

mcastFFA step 2. For the second trajectory of f_2, f_2^B, our mcastFFA derives $\beta^{\text{l.o.}f_2^B} = \beta^{\text{l.o.}f_2^B}_{\langle 5,6\rangle}$. Again, we are not enforced to cut this trajectory's path (see Figure 4c) and in contrast to EIB we can apply alternative tandem analyses:

- SFA/PBOO:

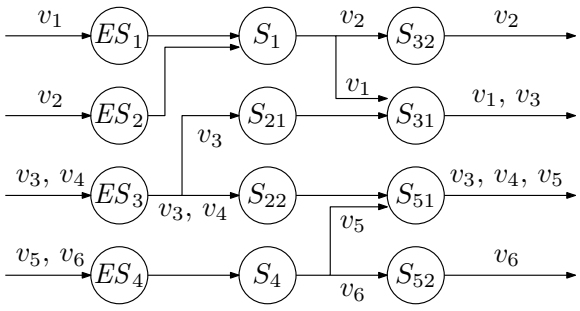
$$\begin{aligned}
\beta^{\text{l.o.}f_2^B} &= \beta^{\text{l.o.}f_2^B}_{\langle 5,6\rangle} \\
&\text{(cut enforced by SFA, no single-tandem analysis)} \\
&= \beta_5^{\text{l.o.}f_2^B} \otimes \beta_6^{\text{l.o.}f_2^B} \\
&= \left(\beta_5 \ominus \alpha_5^{f_1^B}\right) \otimes \left(\beta_6 \ominus \alpha_6^{f_1^B}\right) \\
&= \left(\beta_5 \ominus \left(\alpha^{f_1} \oslash \beta^{\text{l.o.}f_1}_{\langle 1,2\rangle}\right)\right) \otimes \left(\beta_6 \ominus \left(\alpha^{f_1} \oslash \beta^{\text{l.o.}f_1^B}_{\langle 1,2,5\rangle}\right)\right)
\end{aligned}$$

  Note, that the actual trajectory of the cross-flow, f_1 or f_1^B, was automatically chosen correctly by the backtracking. Moreover, note the contrast to EIB: We can derive f_1^B's arrivals at s_6 with an end-to-end left-over service curve that, in turn, can make use of aggrAB.

- PMOO:

$$\begin{aligned}
\beta^{\text{l.o.}f_2^B} &= \beta^{\text{l.o.}f_2^B}_{\langle 5,6\rangle} \\
&\text{(there is no enforced cut)} \\
&= (\beta_5 \otimes \beta_6) \ominus \alpha_5^{f_1} \\
&= (\beta_5 \otimes \beta_6) \ominus \left(\alpha^{f_1} \oslash \beta^{\text{l.o.}f_1}_{\langle 1,2\rangle}\right)
\end{aligned}$$

  where $\beta^{\text{l.o.}f_1}_{\langle 1,2\rangle}$ can be computed either applying the left-over service curve derivation of SFA/PBOO or PMOO.



(a) AFDX network.

Flow	From [13] TA	From [13] FA	u. trans. PMOO[1]	EIB TFA[2]	EIB SFA	EIB PMOO	mcastFFA SFA	mcastFFA PMOO
v_1	**82**	**82**	**82**	**82**	**82**	**82**	**82**	**82**
v_2	**72**	**72**	**72**	**72**	**72**	**72**	**72**	**72**
$v_{3(S_{31})}$	**82**	**82**	92	**82**	**82**	**82**	**82**	**82**
$v_{3(S_{51})}$	**82**	112	92	102	102	92	102	**82**
v_4	**82**	112	92	102	102	92	102	**82**
v_5	-	112	92	102	102	92	102	**92**
v_6	**72**	82	**72**	82	82	72	82	**72**

(b) Delay bounds (values given in μs, least bounds in bold).

Figure 5: Simple AFDX network evaluation of [13], extended with DNC's EIB and mcastFFA delay bounds.

This derivation is illustrated in Figure 4c. Compared to Figure 3, we indeed notice the longer tandem for the second trajectory of f_2.

Theoretical Evaluation

We conclude this section by a theoretical evaluation of mcastFFA against the related DNC approaches:

- *Relation to unicastFFA (Section 2.3.2):* The mcastFFA is a generalization of the unicastFFA. Analysis of unicast flows in either of the two steps remains unaffected (see f_3 in the running example).
- *Relation to unicastFFA transformation (Section 3):* Like the unicastFFA transformation, the mcastFFA is able to derive a PMOO end-to-end left-over service curve. However, it does so without the additional cross-traffic assumptions introduced by the unicastFFA transformation. I.e., there are less cross-flows to consider in the analysis, left-over service curves will be larger and delay bounds will be smaller. Thus, mcastFFA outperforms unicastFFA transformation.
- *Relation to EIB unicastFFA:* In comparison to EIB, we gained the ability to operate on end-to-end tandems. This constitutes increased flexibility to cut this tandem during the analysis: Our mcastFFA is compatible with SFA/PBOO, PMOO, aggrAB, or [3] for best attainable left-over service curves. This best solution to cut a tandem and combine sub-tandem results might coincide with EIB's enforced alternative, i.e., mcastFFA is indeed a generalization of EIB unicastFFA.

Before evaluating our contributions, let us briefly clarify their impact on the server backlog bound Q presented in Theorem 3. Deriving these bounds requires the arrival bounds of all flows at a server. I.e., in the DNC analysis procedures, (EIB) unicastFFA and mcastFFA, step 1 is crucial for the result accuracy; step 2 is not required. As shown with the running example, we improved of cross-traffic arrival bounding in case there are multicast flows present. Thus, backlog bounds are also improved by our contribution.

6. NUMERICAL EVALUATION

We have shown in Sections 4 and 5 that our proposed approaches are superior to the previous network calculus ones, presented as related work in Section 3. We investigate now in this section the gains in terms of accuracy of end-to-end delay bounds via a numerical evaluation. We study the two AFDX networks presented in [13] and [12]. This allows us to benchmark our proposed approaches with the TA and FA since numerical results are given in the literature. Note, that [12] extends the TA and FA by a grouping property that accounts for serialization of packetized flows when crossing links. We leave its implementation in the generalized DNC solutions, based on [8], to future work and restrict our comparison to the non-serialized results. This also allows us to use the range of established SFA/PBOO and PMOO $\beta^{\text{l.o.}}$ derivations implemented in the DiscoDNC tool [4] as well as the aggrAB arrival bounding. I.e., we inherit the DiscoDNC assumptions of a fluid model and curves that can be decomposed into a set of either token buckets or rate latencies.

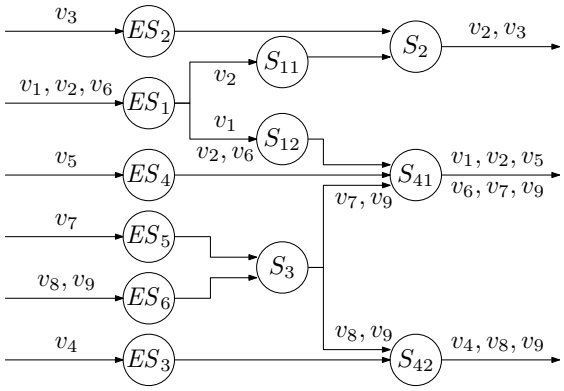
The first network, illustrated in Figure 5a, is a simple AFDX scenario with only one multicast flow (v_3) and a simple flow interference pattern. The second network, illustrated in Figure 6a, is a more complex AFDX scenario with two multicast flows (v_2 and v_9). Numerical results on the end-to-end delay bounds of the different flows are shown in Figures 5b and 6b, respectively. Key observations w.r.t. the performance of DNC analyses confirm our conjectures:

- mcastFFA with PMOO produces gains of 8.74% and 13.08% respectively compared to the multicast TFA.
- mcastFFA produces more accurate bounds than the EIB analysis, since it can operate on longer tandems.
- For some flows, all results are equal. These are simplistic cases that become less in the larger network.

Next, we compare mcastFFA to TA and FA. We observe that mcastFFA results are never inferior to these contenders. Moreover, cases of equal results often coincide with the simplistic ones where even mutlicast TFA is competitive. This is especially visible in the second scenario with less competitive TA and FA bounds. A maximum gain of 5.86% compared to TA and 18.58% compared to FA is achieved in this small AFDX scenario. AFDX networks as deployed in existing Airbus aircraft are already far bigger and more involved than the ones of Figures 5a and 6a. They consist of ∼1000 multicast flows (virtual links, VLs) that have an average of ∼6.5 trajectories per VL. Therefore, the improvements we achieve with DNC's PMOO in conjunction with mcastFFA is expected to be considerably larger in practice.

[1]unicastFFA transformation approach with the stated PMOO end-to-end left-over service curve derivation.

[2]Remember, that EIB with TFA corresponds to the multicast TFA analysis presented as related work in Section 3.



(a) AFDX Network.

Flow	From [12] TA	From [12] FA	u. trans. PMOO[1]	EIB TFA[2]	EIB SFA	EIB PMOO	mcastFFA SFA	mcastFFA PMOO
v_1	142	192	142	182	182	142	182	**122**
$v_{2(S_2)}$	**122**	**122**	142	**122**	**122**	**122**	**122**	**122**
$v_{2(S_{41})}$	**142**	192	**142**	182	182	162	182	**142**
v_3	66	**56**	**56**	**56**	**56**	**56**	**56**	**56**
v_4	**56**	66	**56**	**56**	**56**	**56**	**56**	**56**
v_5	106	106	**96**	**96**	**96**	**96**	**96**	**96**
v_6	142	192	142	182	182	142	182	**122**
v_7	-	152	142	142	142	142	142	**132**
v_8	**92**	122	102	112	112	102	112	**92**
$v_{9(S_{41})}$	-	162	142	152	152	142	152	**132**
$v_{9(S_{42})}$	**92**	122	102	112	112	102	112	**92**

(b) Delay bounds (values given in μs, least bounds in bold).

Figure 6: More complex AFDX network evaluation of [12], extended with DNC's EIB and mcastFFA delay bounds.

7. CONCLUSION

In this paper, we tackled the problem of analyzing multicast flows with deterministic network calculus. DNC was tailored to the analysis of unicast flows – a property that was assumed to invariantly hold. Therefore, previous approaches for the DNC analysis of multicast flows tried to adjust to this restriction by, e.g., pessimistic re-modeling of the network. This lead to inaccurate performance bounds and the development of alternative, non-DNC analyses to derive multicast flow guarantees. In contrast, we generalized DNC unicast feed-forward analysis to a multicast one.

We took two crucial steps to achieve this, both constituting an analysis approach of their own: the EIB analysis and the mcastFFA. In theoretical and numerical evaluations we showed that this paper contributes a single best DNC analysis for multicast flows, the mcastFFA. Not only does it outperform any other DNC approach, the evaluation of AFDX scenarios from the literature also shows that DNC achieves at least the results of competing analyses (Trajectory Approach and Forward Analysis). Moreover, the presented mcastFFA has the flexibility to be combined with any DNC tandem analysis and improvement thereof. For instance, [7], [3], FIFO multiplexing service analysis [2] or packetization [8] can tighten guarantees in AFDX networks.

8. REFERENCES

[1] H. Bauer, J. Scharbarg, and C. Fraboul. Applying and Optimizing Trajectory approach for performance evaluation of AFDX avionics network. In *Proc. IEEE ETFA*, 2009.

[2] L. Bisti, L. Lenzini, E. Mingozzi, and G. Stea. Numerical Analysis of Worst-Case End-to-End Delay Bounds in FIFO Tandem Networks. *Springer Real-Time Systems Journal*, 2012.

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

[4] S. Bondorf and J. B. Schmitt. The DiscoDNC v2 – A Comprehensive Tool for Deterministic Network Calculus. In *Proc. EAI ValueTools*, 2014.

[5] S. Bondorf and J. B. Schmitt. Boosting Sensor Network Calculus by Thoroughly Bounding Cross-Traffic. In *Proc. IEEE INFOCOM*, 2015.

[6] S. Bondorf and J. B. Schmitt. Calculating Accurate End-to-End Delay Bounds – You Better Know Your Cross-Traffic. In *Proc. EAI ValueTools*, 2015.

[7] 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.

[8] M. Boyer and P. Roux. A common framework embedding network calculus and event stream theory. *hal-01311502*, 2016. Working paper or preprint.

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

[10] F. Geyer and G. Carle. Network engineering for real-time networks: comparison of automotive and aeronautic industries approaches. *IEEE Communications Magazine*, 2016.

[11] J. Grieu. *Analyse et évaluation de techniques de commutation Ethernet pour l'interconnexion des systèmes avioniques*. PhD thesis, INPT, 2004.

[12] G. Kemayo, N. Benammar, F. Ridouard, H. Bauer, and P. Richard. Improving AFDX End-to-End delays analysis. In *Proc. of IEEE ETFA*, 2015.

[13] G. Kemayo, F. Ridouard, H. Bauer, and P. Richard. A Forward end-to-end delays Analysis for packet switched networks. In *Proc. of RTNS*, 2014.

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

[15] S. Martin and P. Minet. Schedulability analysis of flows scheduled with FIFO: application to the expedited forwarding class. In *Proc. of IPDPS*, 2006.

[16] J. Migge. *L'ordonnancement sous contraintes temps-réel un modèle à base de trajectoires*. PhD thesis, INRIA Sophia Antipolis, 1999.

[17] 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.

[18] 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.

[19] K. Tindell and J. Clark. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and Microprogramming*, 1994.