DeepTMA: Predicting Effective Contention Models for Network Calculus using Graph Neural Networks

Fabien Geyer[1,2] and Steffen Bondorf[3]

INFOCOM 2019

Wednesday 1st May, 2019

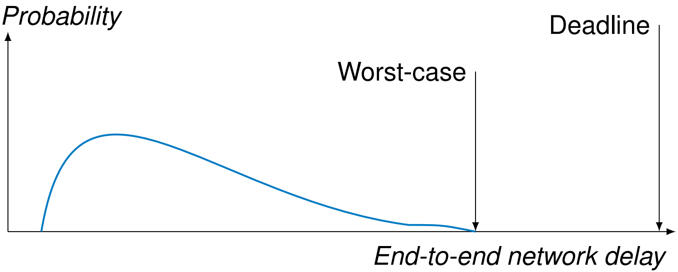
[1]Chair of Network Architectures and Services
Technical University of Munich (TUM)

[2]Airbus Central R&T
Munich

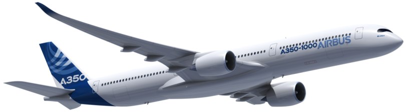
[3]Dept. of Information Security and Communication Technology
Norwegian University of Science and Technology (NTNU)

Motivation

Worst-Case End-to-End Performance Analysis

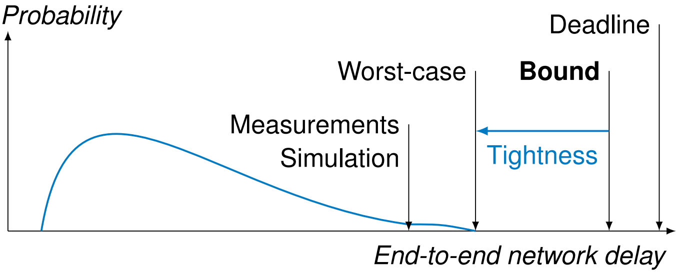


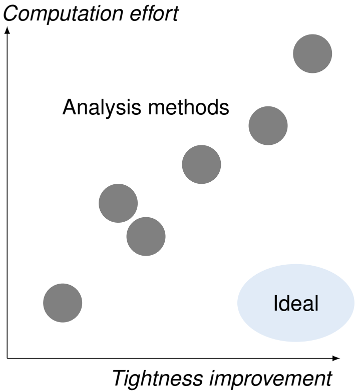
- Important for critical applications
- Need formal proof on network delay



Motivation

Worst-Case End-to-End Performance Analysis

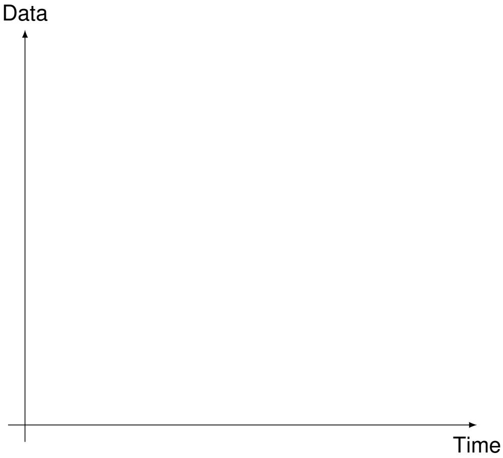




- Trade-off between computational effort and tightness
- **This talk: network analysis method with good tightness and fast execution**

Motivation

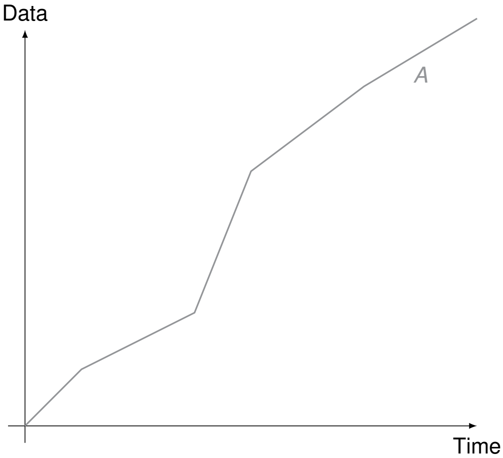
Network Calculus – Basics



Basis: Cumulative arrivals and services [Cruz, 1991a]

Motivation

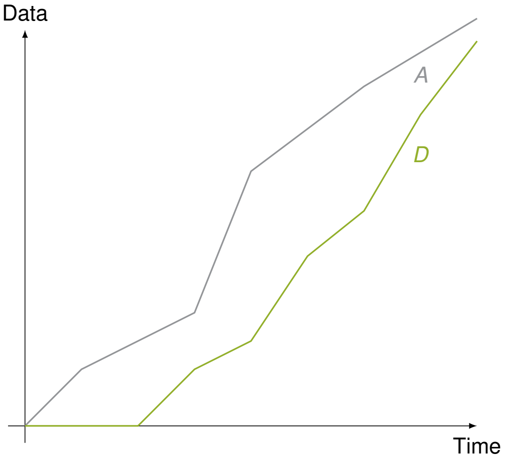
Network Calculus – Basics



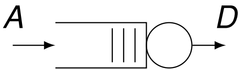
Basis: Cumulative arrivals and services [Cruz, 1991a]

Motivation

Network Calculus – Basics

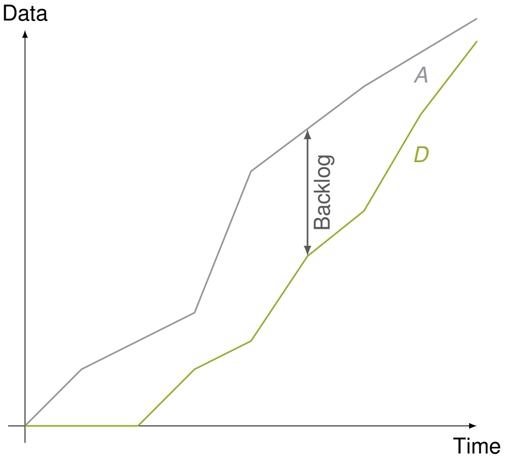


Basis: Cumulative arrivals and services [Cruz, 1991a]

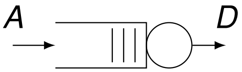


Motivation

Network Calculus – Basics

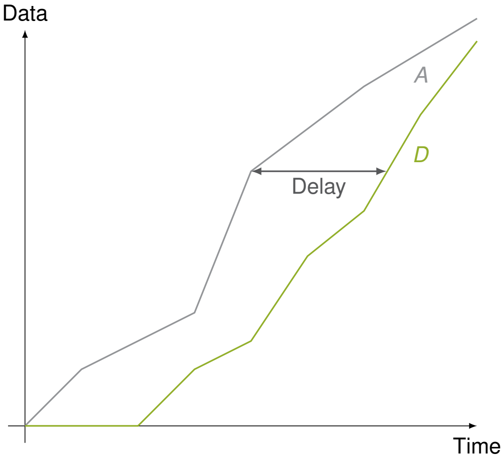


Basis: Cumulative arrivals and services [Cruz, 1991a]



Motivation

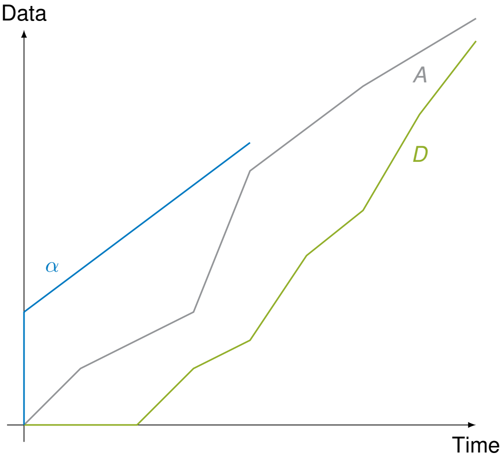
Network Calculus – Basics



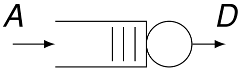
Basis: Cumulative arrivals and services [Cruz, 1991a]

Motivation

Network Calculus – Basics



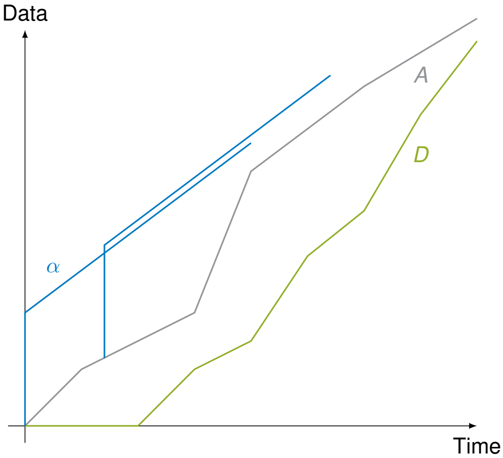
Basis: Cumulative arrivals and services [Cruz, 1991a]



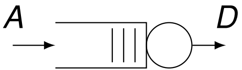
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Motivation

Network Calculus – Basics



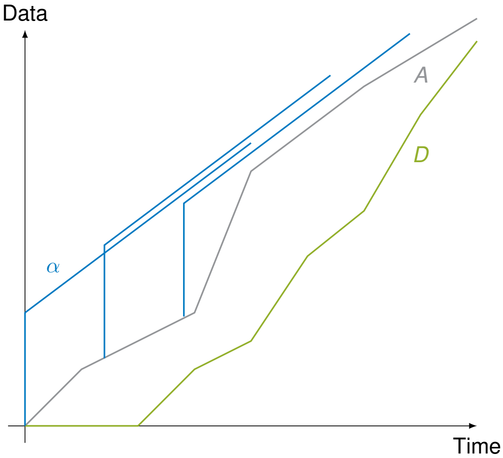
Basis: Cumulative arrivals and services [Cruz, 1991a]



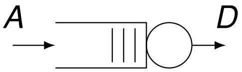
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Motivation

Network Calculus – Basics



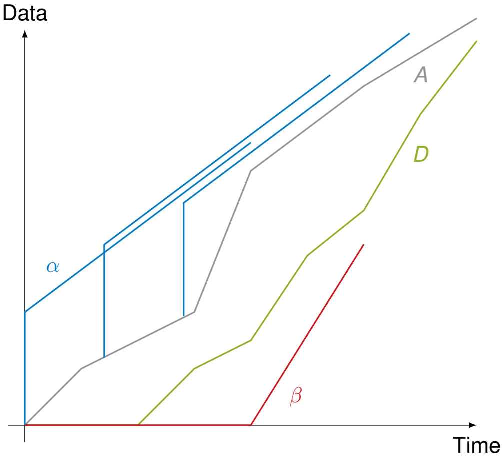
Basis: Cumulative arrivals and services [Cruz, 1991a]



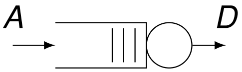
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Motivation

Network Calculus – Basics



Basis: Cumulative arrivals and services [Cruz, 1991a]

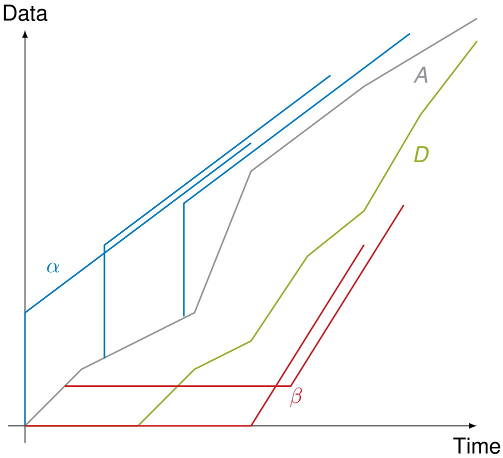


Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

Motivation

Network Calculus – Basics



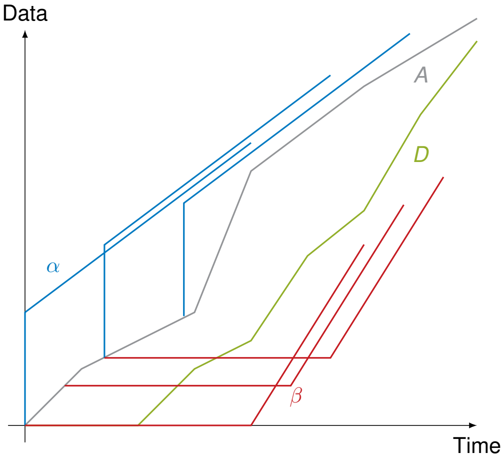
Basis: Cumulative arrivals and services [Cruz, 1991a]

Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

Motivation

Network Calculus – Basics



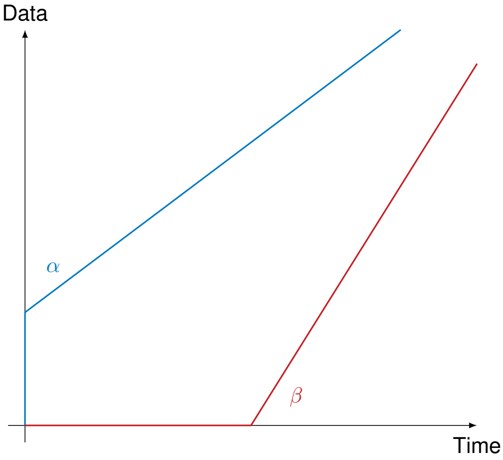
Basis: Cumulative arrivals and services [Cruz, 1991a]

Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

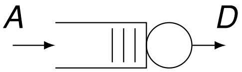
Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

Motivation

Network Calculus – Basics

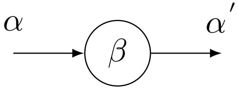


Basis: Cumulative arrivals and services [Cruz, 1991a]



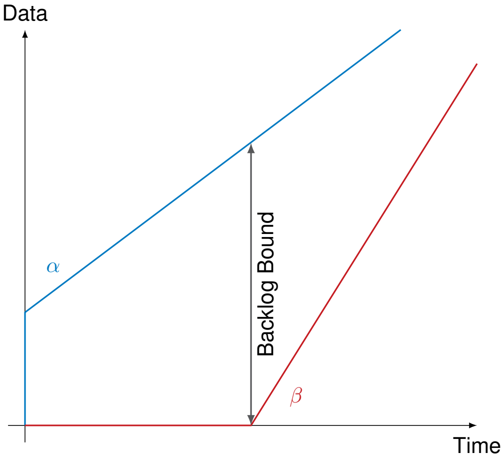
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

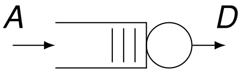


Motivation

Network Calculus – Basics

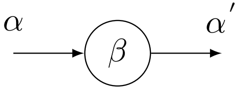


Basis: Cumulative arrivals and services [Cruz, 1991a]



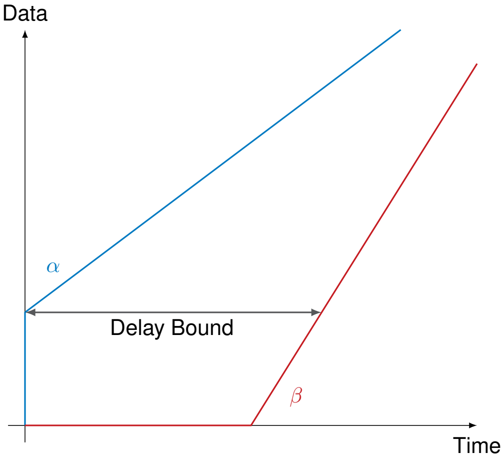
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

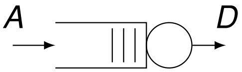


Motivation

Network Calculus – Basics

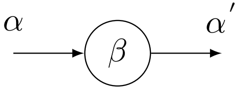


Basis: Cumulative arrivals and services [Cruz, 1991a]



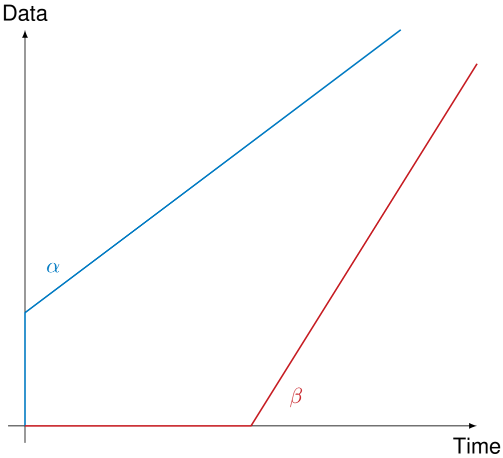
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$



Motivation

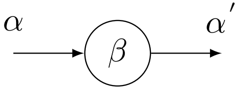
Network Calculus – Basics



Basis: Cumulative arrivals and services [Cruz, 1991a]

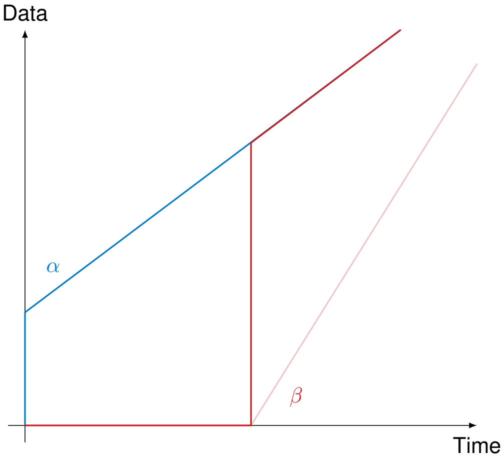
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

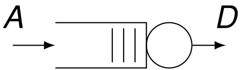


Motivation

Network Calculus – Basics

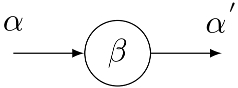


Basis: Cumulative arrivals and services [Cruz, 1991a]



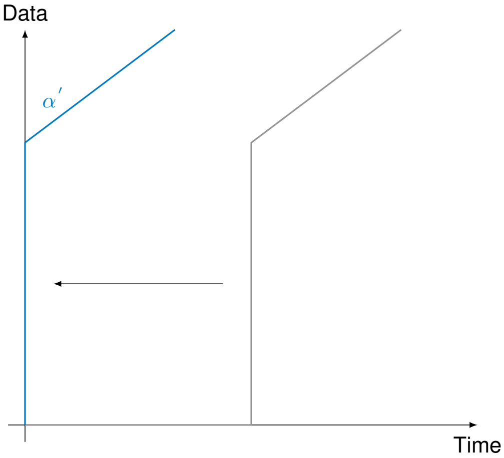
Arrival curve α: $A(t) - A(t-s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$

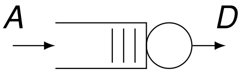


Motivation

Network Calculus – Basics

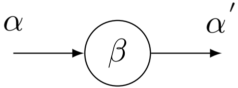


Basis: Cumulative arrivals and services [Cruz, 1991a]



Arrival curve α: $A(t) - A(t - s) \leq \alpha(s), \forall t \leq s$

Service curve β: a server is said to offer a strict service curve β if, during any backlogged period of duration u, the output of the system is at least equal to $\beta(u)$



MODELING AND ANALYSIS OF NETWORK INFRASTRUCTURE IN CYBER-PHYSICAL SYSTEMS

LIANG CHENG (LEHIGH UNIVERSITY, BETHLEHEM, USA)

STEFFEN BONDORF (NTNU TRONDHEIM, NORWAY)

ACM SIGCOMM 2019 TUTORIALS

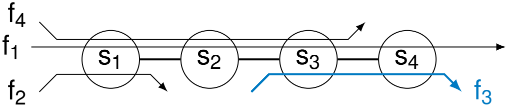
2019-08-23

BEIJING, CHINA

Motivation

Network Calculus – Network Analysis

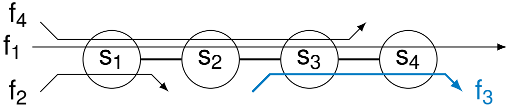
How to compute end-to-end performance?



Motivation

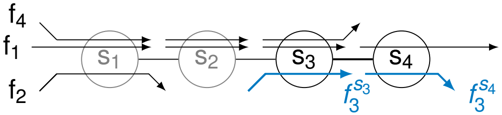
Network Calculus – Network Analysis

How to compute end-to-end performance?



TFA – Total Flow Analysis [Cruz, 1991b]

Step 1: Compute delay at each server on the path

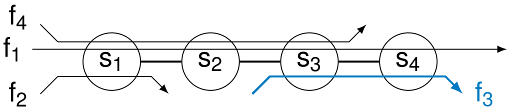


Step 2: Sum delays

Motivation

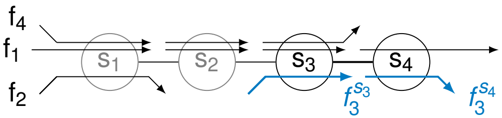
Network Calculus – Network Analysis

How to compute end-to-end performance?



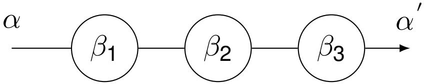
TFA – Total Flow Analysis [Cruz, 1991b]

Step 1: Compute delay at each server on the path

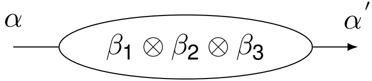


Step 2: Sum delays

Server concatenation [Le Boudec and Thiran, 2001]



(min, +) algebra gives us:



→ Pay Bursts Only Once principle

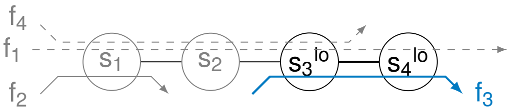
Motivation

Network Calculus – Network Analysis

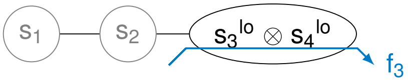
SFA – Separate Flow Analysis

[Le Boudec and Thiran, 2001]

Step 1: Compute per-server residual service



Step 2: Concatenate the servers



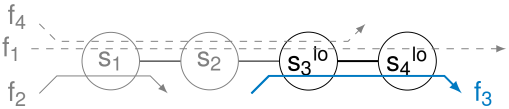
Step 3: Compute delay over concatenated server

Motivation

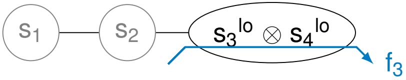
Network Calculus – Network Analysis

SFA – Separate Flow Analysis
[Le Boudec and Thiran, 2001]

Step 1: Compute per-server residual service



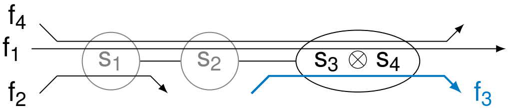
Step 2: Concatenate the servers



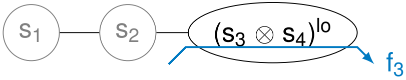
Step 3: Compute delay over concatenated server

PMOO – Pay Multiplexing Only Once
[Schmitt et al., 2008b]

Step 1: Concatenate the servers



Step 2: Compute residual service



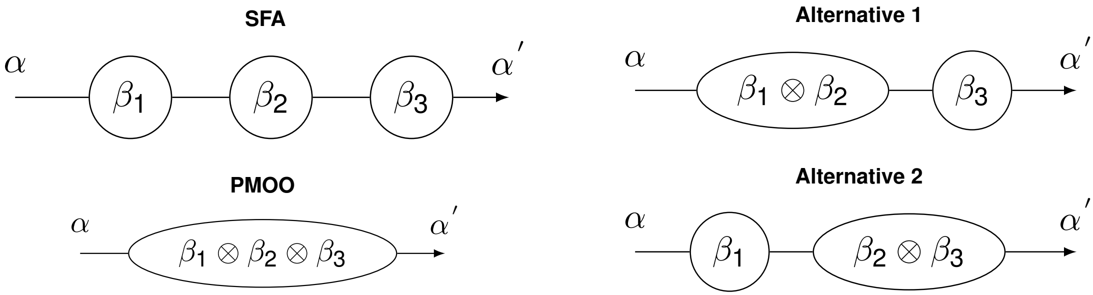
Step 3: Compute delay over concatenated server

Motivation

Network Calculus – TMA

TMA – Tandem Matching Analysis [Bondorf et al., 2017]

- Main concept: apply concatenation only for some servers
- Exhaustive search to find which concatenations will result in the tightest end-to-end delay $\rightarrow \mathcal{O}\left(2^{n-1}\right)$

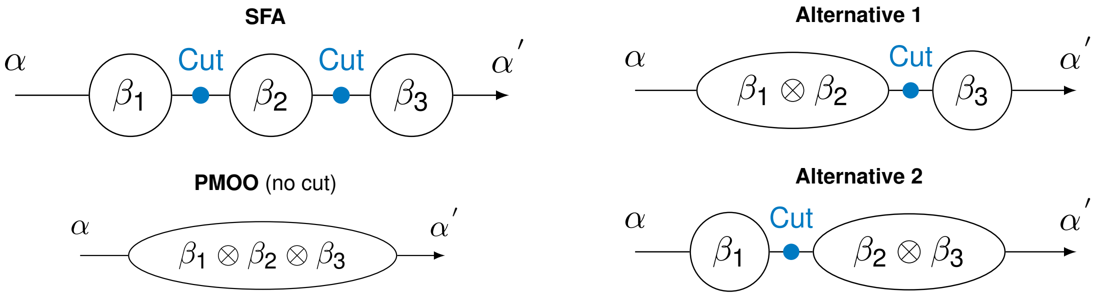


Motivation

Network Calculus – TMA

TMA – Tandem Matching Analysis [Bondorf et al., 2017]

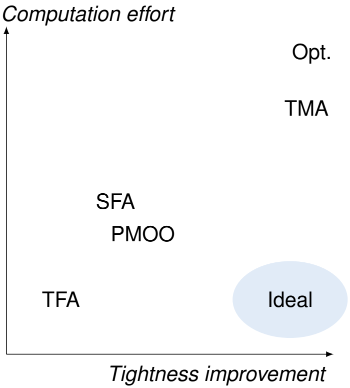
- Main concept: apply concatenation only for some servers
- Exhaustive search to find which concatenations will result in the tightest end-to-end delay $\rightarrow \mathcal{O}\left(2^{n-1}\right)$



Motivation

Network Calculus – DeepTMA

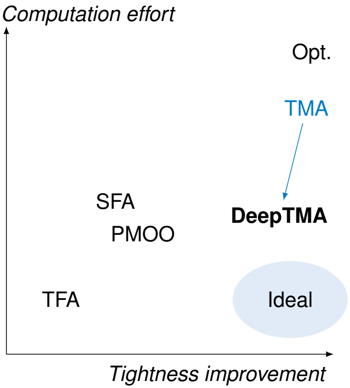
Question: Can we avoid TMA's exhaustive search?



Opt.: [Schmitt et al., 2008a][Bouillard et al., 2010]

Motivation

Network Calculus – DeepTMA



Opt.: [Schmitt et al., 2008a][Bouillard et al., 2010]

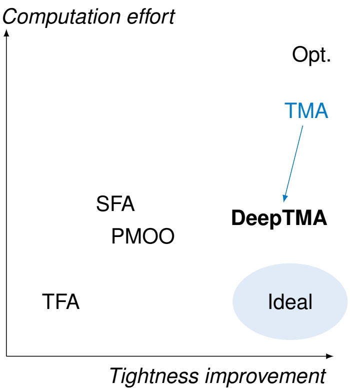
Question: Can we avoid TMA's exhaustive search?

→ **DeepTMA:**

- **Main idea: use fast heuristic for predicting best cuts**
- Even if the heuristic is wrong, the bounds are still valid

Motivation

Network Calculus – DeepTMA



Opt.: [Schmitt et al., 2008a][Bouillard et al., 2010]

Question: Can we avoid TMA's exhaustive search?

→ **DeepTMA:**

- **Main idea: use fast heuristic for predicting best cuts**
- Even if the heuristic is wrong, the bounds are still valid

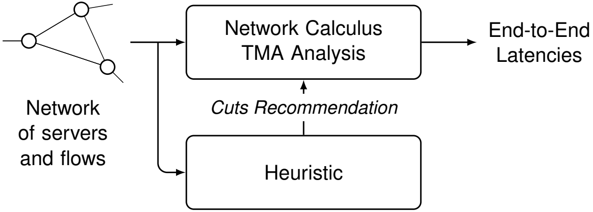


Figure 1: Approach

Outline

Heuristic based on Graph Neural Networks

Numerical evaluation

Conclusion

Heuristic based on Graph Neural Networks

Introduction

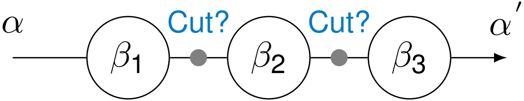
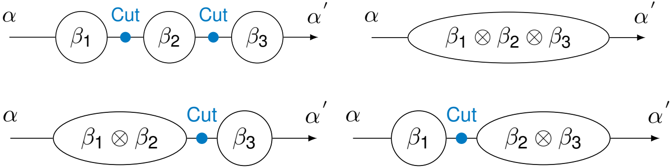


Figure 2: Classification problem

Heuristic

- **Use Graph Neural Network**
- **Classification problem for cuts**



Heuristic based on Graph Neural Networks

Introduction

Heuristic

- **Use Graph Neural Network**
- **Classification problem for cuts**

Graph formulation

- Nodes: flows, servers, cuts
- Edges: relationships between elements
- Prediction if cut is applied or not

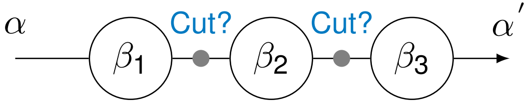


Figure 2: Classification problem

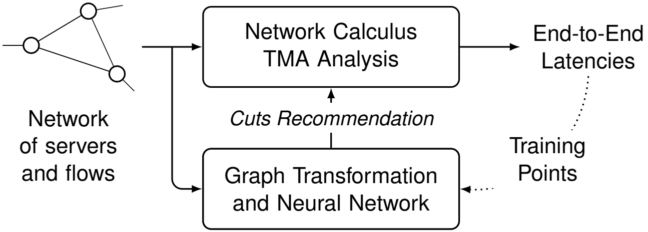
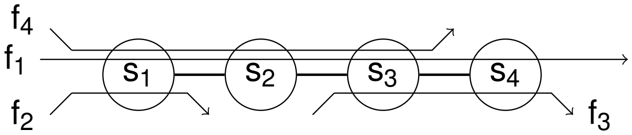


Figure 3: Approach

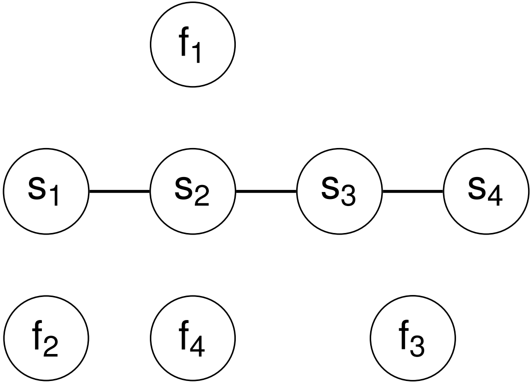
Heuristic based on Graph Neural Networks

Problem formulation as graph



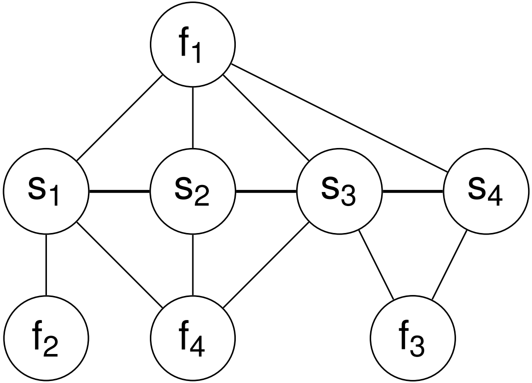
Heuristic based on Graph Neural Networks

Problem formulation as graph



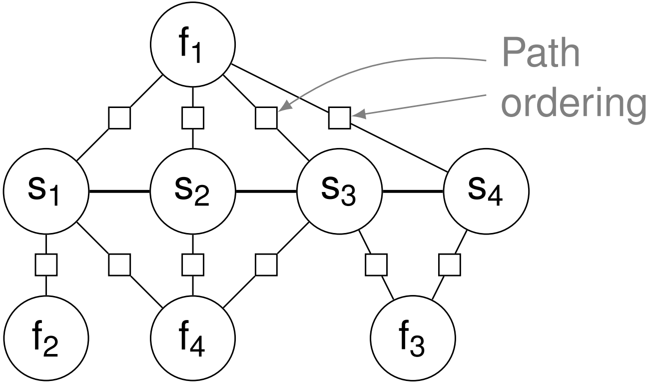
Heuristic based on Graph Neural Networks

Problem formulation as graph



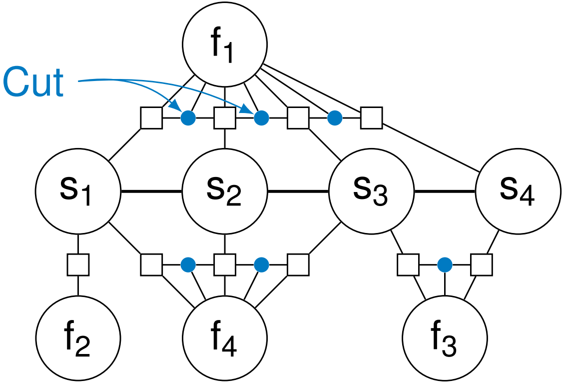
Heuristic based on Graph Neural Networks

Problem formulation as graph



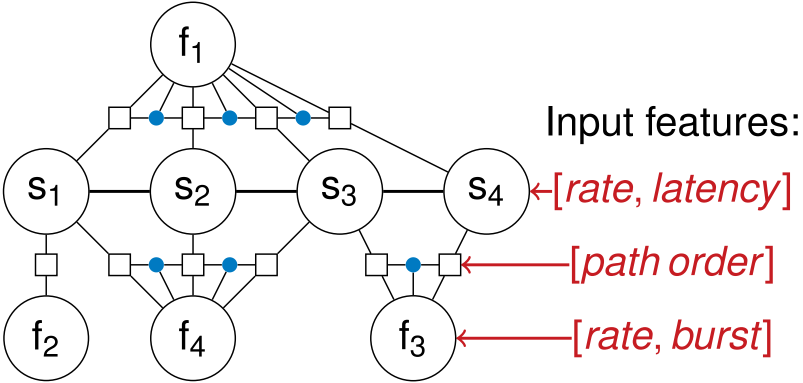
Heuristic based on Graph Neural Networks

Problem formulation as graph



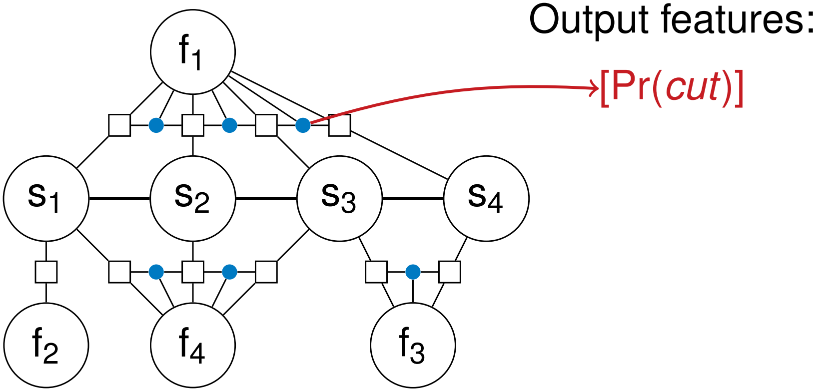
Heuristic based on Graph Neural Networks

Problem formulation as graph



Heuristic based on Graph Neural Networks

Problem formulation as graph



Heuristic based on Graph Neural Networks

Graph Neural Networks – Introduction

Graph Neural Networks [Scarselli et al., 2009] and related architectures are able to process general graphs and predict feature of nodes $\mathbf{o}_v$

Principle

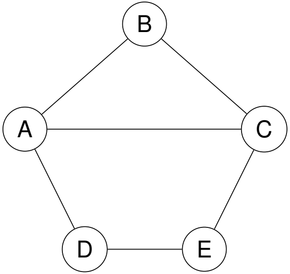
- Each node has a *hidden* vector $\mathbf{h}_v \in \mathbb{R}^k$
- . . . computed according to the vector of its neighbors
- . . . and are propagated through the graph

Algorithm

- Initialize $\mathbf{h}_v^{(0)}$ according to features of nodes
- for $t = 1, \dots, T$ do
  - $\mathbf{a}_v^{(t)} = AGGREGATE\left(\left\{\mathbf{h}_u^{(t-1)} \mid u \in Nbr(v)\right\}\right)$
  - $\mathbf{h}_v^{(t)} = COMBINE\left(\mathbf{h}_v^{(t-1)}, \mathbf{a}_v^{(t)}\right)$
- return $READOUT\left(\mathbf{h}_v^{(T)}\right)$

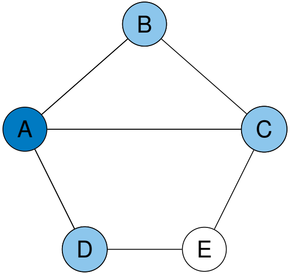
Heuristic based on Graph Neural Networks

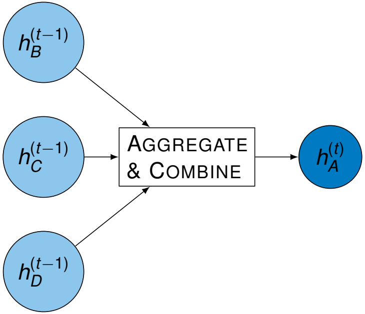
Graph Neural Networks – Illustration



Heuristic based on Graph Neural Networks

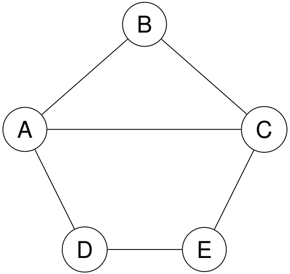
Graph Neural Networks – Illustration

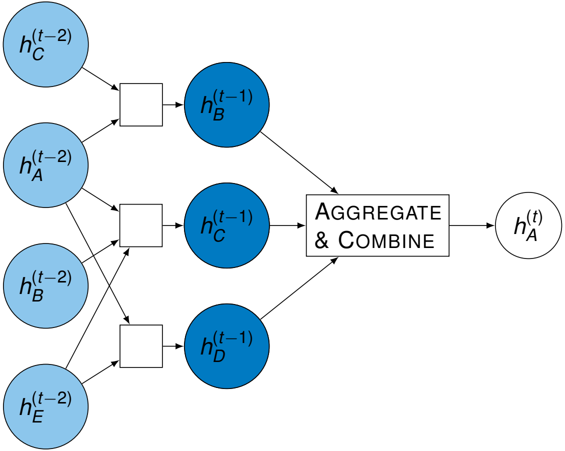




Heuristic based on Graph Neural Networks

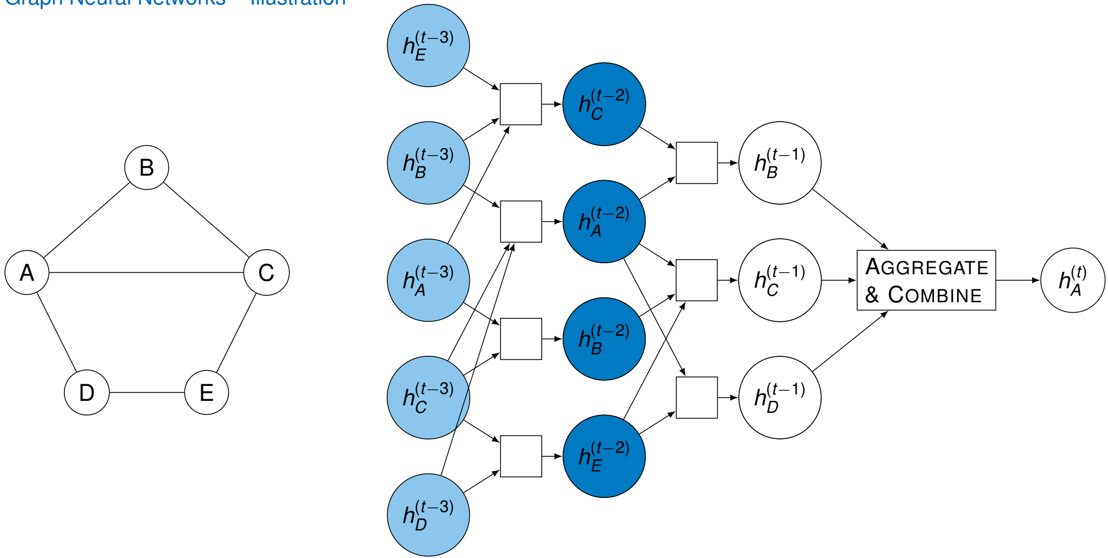
Graph Neural Networks – Illustration





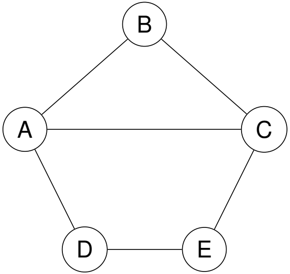
Heuristic based on Graph Neural Networks

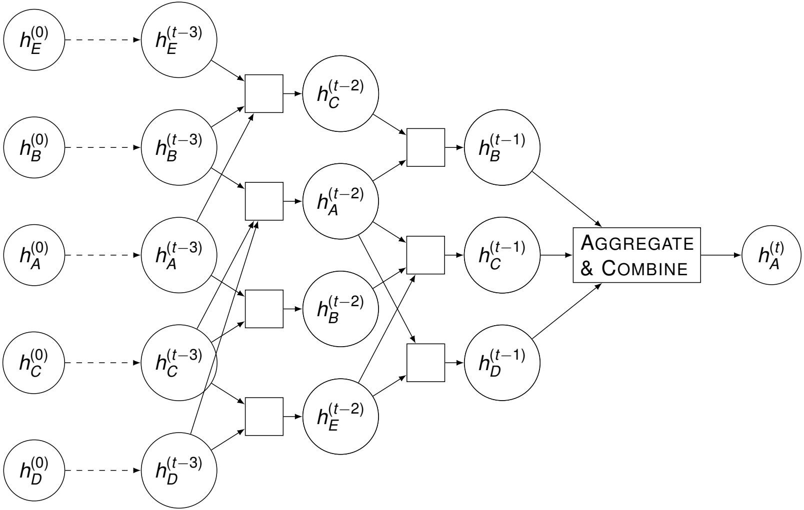
Graph Neural Networks – Illustration



Heuristic based on Graph Neural Networks

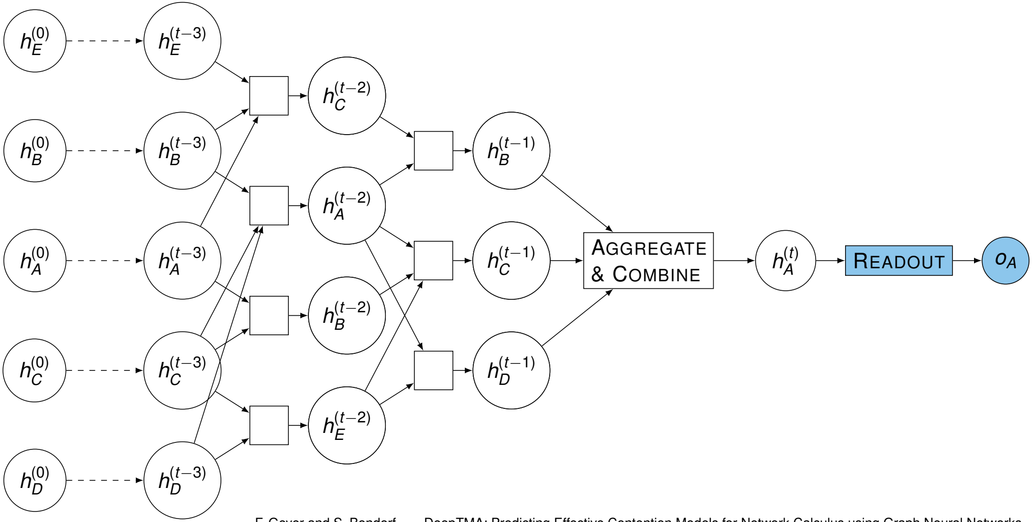
Graph Neural Networks – Illustration





Heuristic based on Graph Neural Networks

Graph Neural Networks – Illustration



Heuristic based on Graph Neural Networks

Graph Neural Networks – Implementation

Implementation (simplified)

- Initialize $\mathbf{h}_v^{(0)}$ according to features of nodes
- for $t = 1, \ldots, T$ do
  - *AGGREGATE* $\rightarrow$ $\mathbf{a}_v^{(t)} = \sum_{u \in Nbr(v)} \mathbf{h}_u^{(t-1)}$
  - *COMBINE* $\rightarrow$ $\mathbf{h}_v^{(t)} = \textit{Neural Network}\left(\mathbf{h}_v^{(t-1)}, \mathbf{a}_v^{(t)}\right)$
- *READOUT* $\rightarrow$ return $\textit{Neural Network}\left(\mathbf{h}_v^{(T)}\right)$

Training

- Using standard gradient descent techniques

Heuristic based on Graph Neural Networks

Graph Neural Networks – Implementation

Implementation (simplified)

- Initialize $\mathbf{h}_v^{(0)}$ according to features of nodes
- for $t = 1, \ldots, T$ do
  - *AGGREGATE* $\rightarrow$ $\mathbf{a}_v^{(t)} = \sum_{u \in Nbr(v)} \mathbf{h}_u^{(t-1)}$
  - *COMBINE* $\rightarrow$ $\mathbf{h}_v^{(t)} = \textit{Neural Network}\left(\mathbf{h}_v^{(t-1)}, \mathbf{a}_v^{(t)}\right)$
- *READOUT* $\rightarrow$ return *Neural Network* $\left(\mathbf{h}_v^{(T)}\right)$

Training

- Using standard gradient descent techniques

Different approaches

- **Gated-Graph Neural Network**
- Graph Convolution Network
- Graph Attention Networks
- Graph Spatial-Temporal Networks
- ...

$\rightarrow$ Hot area of research in the ML community

Numerical evaluation

Dataset generation

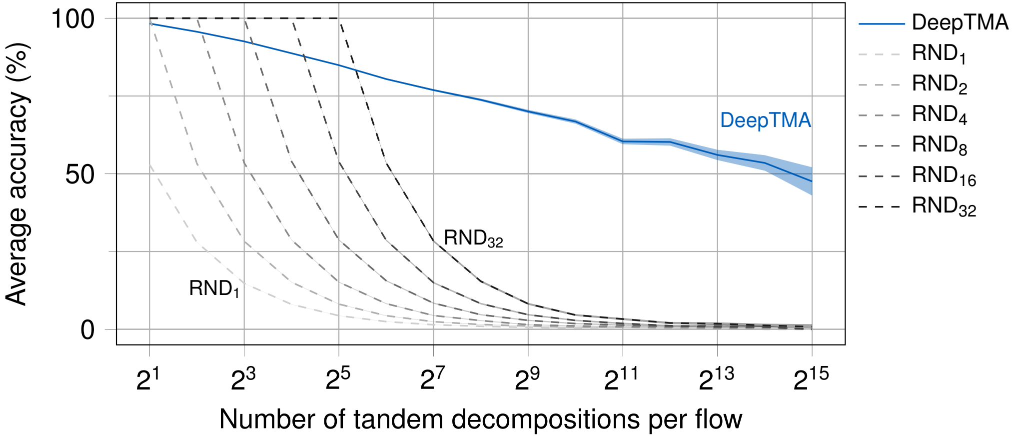
- Generation of 100 000 networks with tandem or tree topology
- Random generation of curve parameters for servers and flows
- Evaluation of each network using DiscoDNC and extract intermediary results of TMA
- Dataset available online: `https://github.com/fabgeyer/dataset-infocom2019`

Parameter	**Min**	**Max**	**Mean**	**Median**
# of servers	2	41	14.2	12.0
# of flows	1	63	23.0	18.0
# of flows per server	1	44	5.8	4.6
# of tandem combinations	2	113 100	596.2	134.0
# of tandem combination per flow	2	32 768	25.9	4.0
# of nodes in analyzed graph	6	717	159.0	127.0

Table 1: Statistics about the generated dataset.

Numerical evaluation

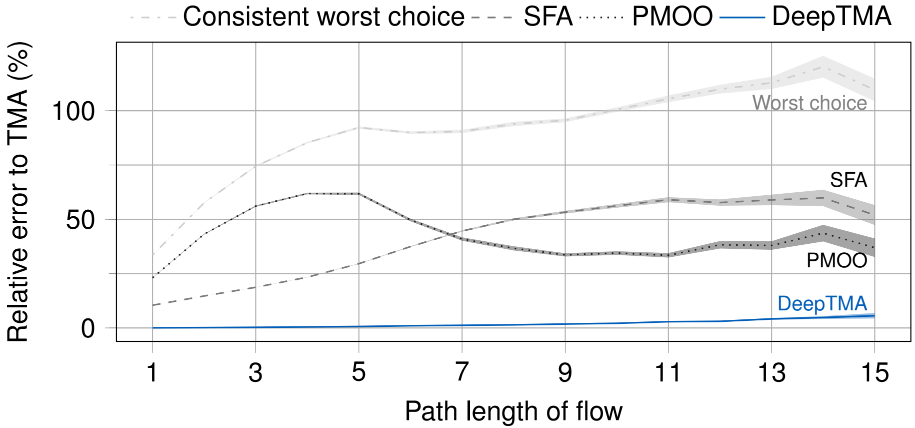
Prediction accuracy



Numerical evaluation

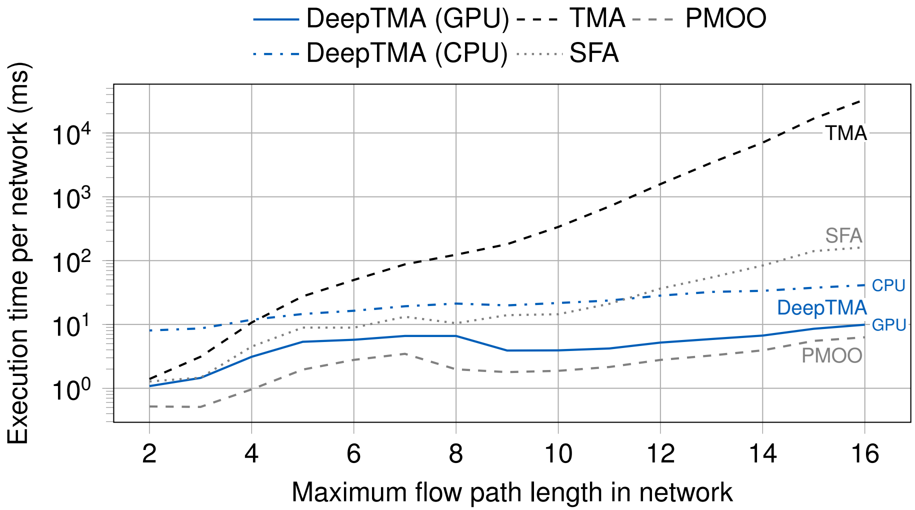
Tightness

The impact of these failures to predict the optimal decomposition only results in a relative error below 6%



Numerical evaluation

Runtime



Numerical evaluation

Additional results

Three other simpler heuristics defined in the paper

- Random Choice of Tandem Decomposition
- Path Length of Flows up to Location of Interference
- Hop Count Heuristic

Results

- DeepTMA better than random-based heuristics

Conclusion

Contributions

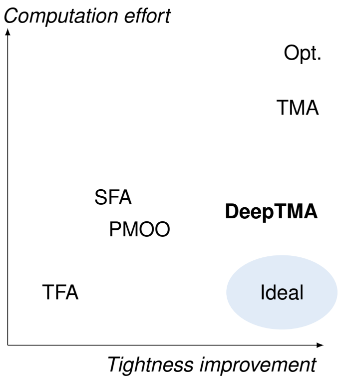
- **Framework combining network calculus and graph-based deep learning**
- **New NC analysis with fast execution times and good tightness**
- Dataset: `https://github.com/fabgeyer/dataset-infocom2019`

Future work

- Evaluation on more complex networks and curves
- Predictions for other NC analyses

Final thoughts

→ Graph Neural Networks are a promising paradigm for computer networks



Bibliography

[Bondorf et al., 2017] Bondorf, S., Nikolaus, P., and Schmitt, J. B. (2017).
Quality and cost of deterministic network calculus – design and evaluation of an accurate and fast analysis.
Proc. ACM Meas. Anal. Comput. Syst. (POMACS), 1(1):16:1–16:34.

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

[Bouillard et al., 2010] Bouillard, A., Jouhet, L., and Thierry, É. (2010).
Tight performance bounds in the worst-case analysis of feed-forward networks.
In *Proc. of IEEE INFOCOM*.

[Cruz, 1991a] Cruz, R. L. (1991a).
A calculus for network delay, part I: Network elements in isolation.
IEEE Trans. Inf. Theory, 37(1):114–131.

[Cruz, 1991b] Cruz, R. L. (1991b).
A calculus for network delay, part II: Network analysis.
IEEE Trans. Inf. Theory, 37(1):132–141.

[Scarselli et al., 2009] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009).
The graph neural network model.
IEEE Trans. Neural Netw., 20(1):61–80.

[Schmitt et al., 2008a] Schmitt, J. B., Zdarsky, F. A., and Fidler, M. (2008a).
Delay bounds under arbitrary multiplexing: When network calculus leaves you in the lurch. . . .
In *Proc. of IEEE INFOCOM*.

[Schmitt et al., 2008b] Schmitt, J. B., Zdarsky, F. A., and Martinovic, I. (2008b).
Improving performance bounds in feed-forward networks by paying multiplexing only once.
In *Proc. of GI/ITG MMB*.