RUHR-UNIVERSITÄT BOCHUM

EFFICIENT GRADIENT-BASED NETWORK CALCULUS FOR SCALABLE SYNTHESIS OF NETWORK CONFIGURATIONS

Fabien Geyer (Airbus Central Research & Technology) and **Steffen Bondorf** (Ruhr University Bochum)

Outline aka Bisecting the Paper Title

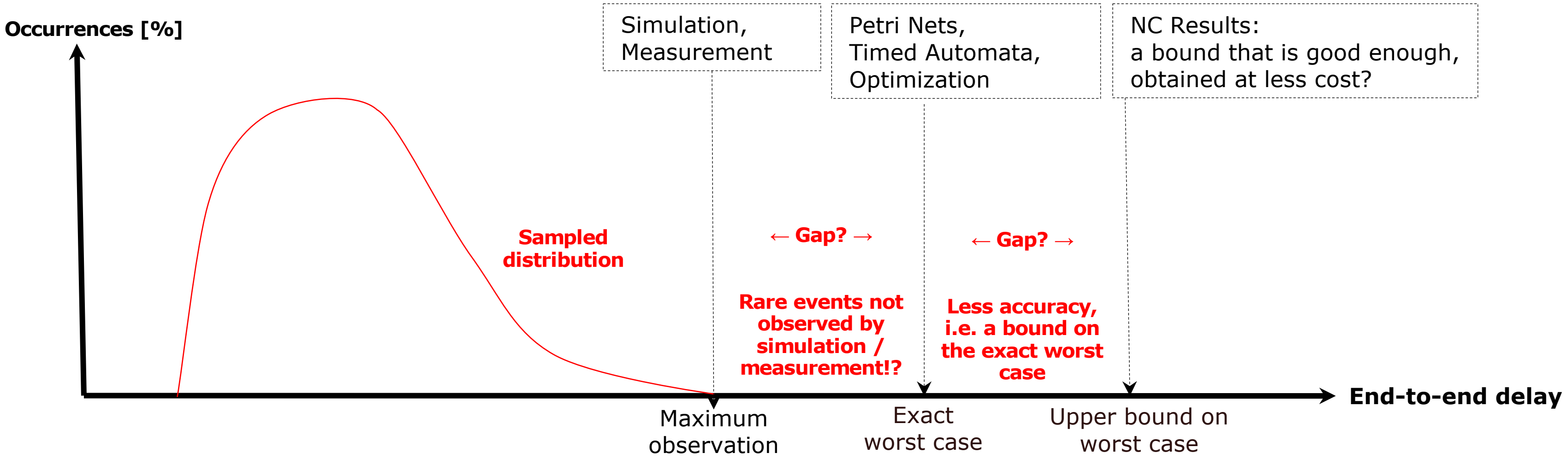
(1) Efficient Gradient-based (non-linear optimization)
(2) Network Calculus
(3) for Scalable Synthesis of Network Configurations

becomes

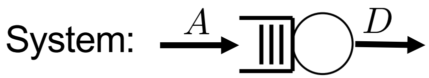
I.	**(2)**	**Network Calculus (NC)**
II.	**(1, 2)**	**the NC extension “DiffNC”**
III.	**(1, 2, 3)**	**Challenge: Flow Paths and Priorities**
IV.	**(3)**	**Evaluation**

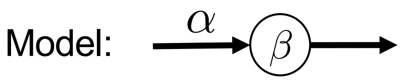
NC Motivation and Basics

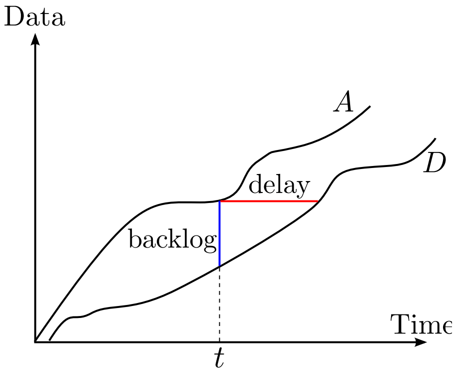
- **Theory of deterministic queueing systems [Cruz91]**
  - Metric: end-to-end communication delay of a data flow crossing a network
  - NC: a worst-case bound on the end-to-end delay of a specific data flow

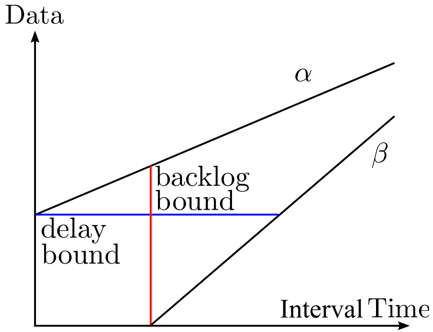


NC Modeling: Bounding Curves in Interval Time







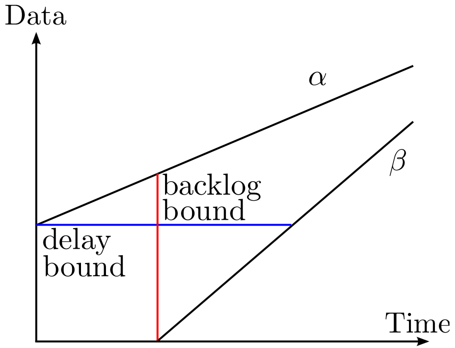


- **Arrival Curve** $\alpha(d)$ $\forall\, 0 \leq d \leq t\ :\ A(t) - A(t-d) \leq \alpha(d)$ **(derived from traffic regulation)**
- **Service Surve** $\beta(d)$ $\forall t : A'(t) \geq \inf_{0 \leq d \leq t} \{A(t-d) + \beta(d)\}$ **(derived from scheduler)**

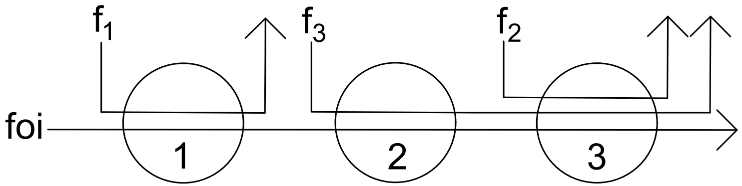
NC Analysis: A (min,plus)-algebraic Term

- **(min,plus) Operations (complexity depends on curve shapes)**

- Concatenation of servers $\beta_1 \otimes \beta_2 = \beta_{1,2}$
- Output bound $\alpha'(t) = \alpha \oslash \beta(t) := \sup_{u \geq 0}\{\alpha(t+u) - \beta(u)\}$
- Delay bound $hdev(\alpha, \beta) = \inf\{d \geq 0 : (\alpha \oslash \beta)(-d) \leq 0\}$
- Left-over service $\beta(t) \ominus \alpha(t) = \max\{0, \beta(t) - \alpha(t)\}$
  (fixed priorities and arbitrary multiplexing)



- **Example: end-to-end delay bound for data of the flow of interest (foi) crossing 3 servers.**
  Priorities (ascending): foi, f1, f2, f3



$$\mathrm{hdev}(\alpha_{\mathrm{foi}}, (\beta_1 \ominus \alpha_1) \otimes ((\beta_2 \otimes (\beta_3 \ominus \alpha_2)) \ominus \alpha_3))$$

RUHR UNIVERSITÄT BOCHUM RUB

NC Intro Wrap-Up

- **The Good**
  - a quite powerful methodolody for worst-case modeling and analysis
  - has found application in the industry (certification of Airbus AFDX network)

- **The Bad**
  - analysis of non-feedforward networks is not yet as advanced (not part of this paper)

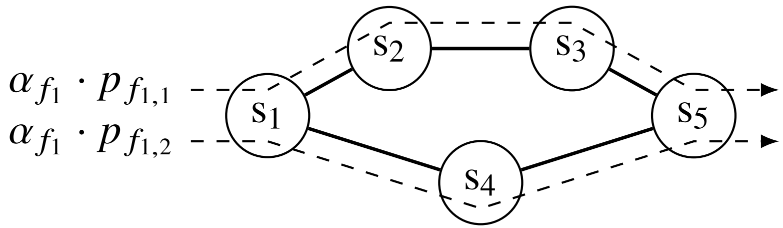
- **The „Ugly“**
  - NC is foremost a tool for analysis, not for synthesis,
    i.e., you need a fully specified model, you cannot optimize for open parameters,
    you can only sample your network design space

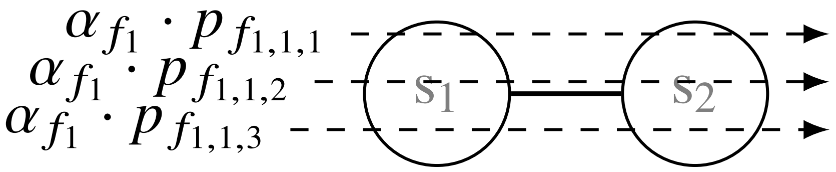
DiffNC [Geyer22]: NC-based Parameter Synthesis

Idea

- derive the delay-boudning NC term with some NC analysis
- leave certain parameters open and/or add binary variables for design alternatives
- differentiate w.r.t. these parameters (off the shelf automatic differentiation AD tool)
- let a solver do the heavy lifting (NLopt library)

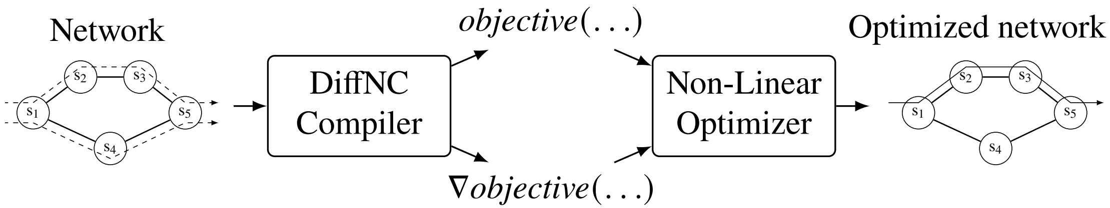
Modeling Example: Encode alternative paths and (global) priorities





DiffNC [Geyer22]: NC-based Parameter Synthesis

The original tool chain



$$\min_{p_{f_{i,j}}, \forall f_i \in \mathcal{F}, j \in \mathcal{P}_{f_i}} \frac{1}{|\mathcal{F}|} \sum_{i,j} delay\ bound(f_{i,j}) \cdot p_{f_{i,j}}$$

$$\text{s.t.} \quad 0 \leq p_{f_{i,j}} \leq 1, \forall f_i \in \mathcal{F}, j \in \mathcal{P}_{f_i}$$

$$\sum_{j \in \mathcal{P}_{f_i}} p_{f_{i,j}} = 1, \forall f_i \in \mathcal{F}$$

$$\sum_{i \in T(k)} r_i \cdot p_{f_{i,j}} \leq R_k, \forall k \in \mathcal{S}$$

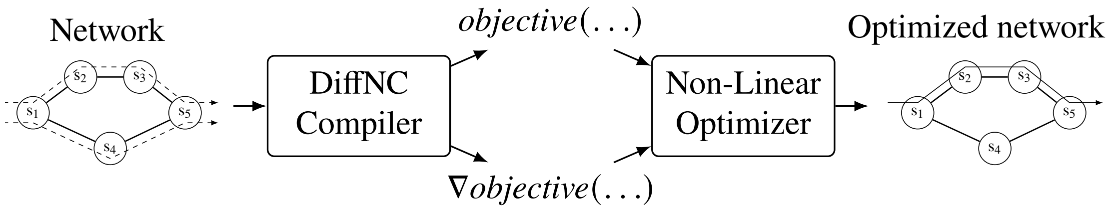
- **Steps in reverse**
  - get optimization result
  - from a solver (i.e., an NLP algorithm), provided
    - objective term and differentiated objective
    - constraints and differentiated constraints
  - have the model be transformed into gradient-based NLP formulation.
    - objective == Network Calculus analysis term
  - take the extended model with open parameters

- **Challenges identified**
  - the network calculus analysis term is in (min,plus) algebra
  - off the shelf AD tools work with (plus,times) algebra
  - off the shelf solvers work with (plus,times) algebra

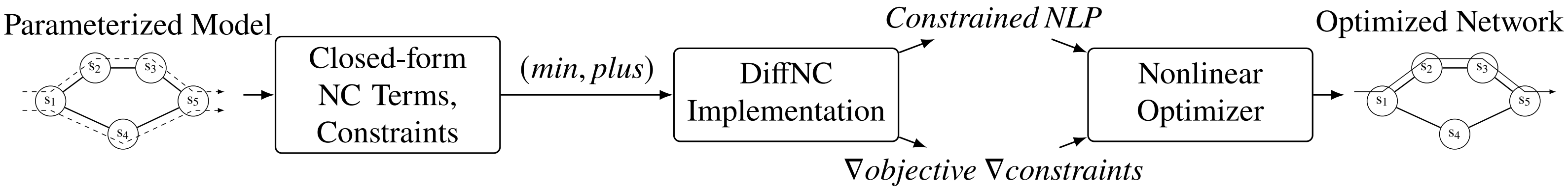
Previous solution:
Convert between algebras

Contribution: A new, better scaling tool chain

The original tool chain



The new tool chain



- **new AD tool highly specialized for Network Calculus min,plus) operations**
- **new implementation of a gradient-descent NLP algorithm: Frank-Wolfe**

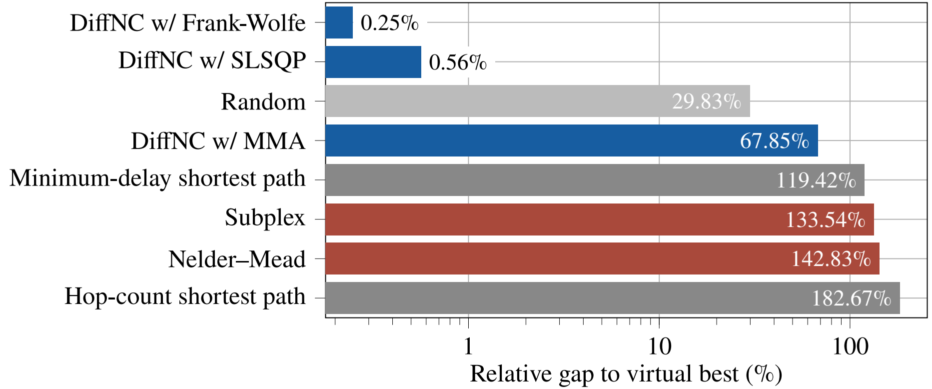
Evaluation

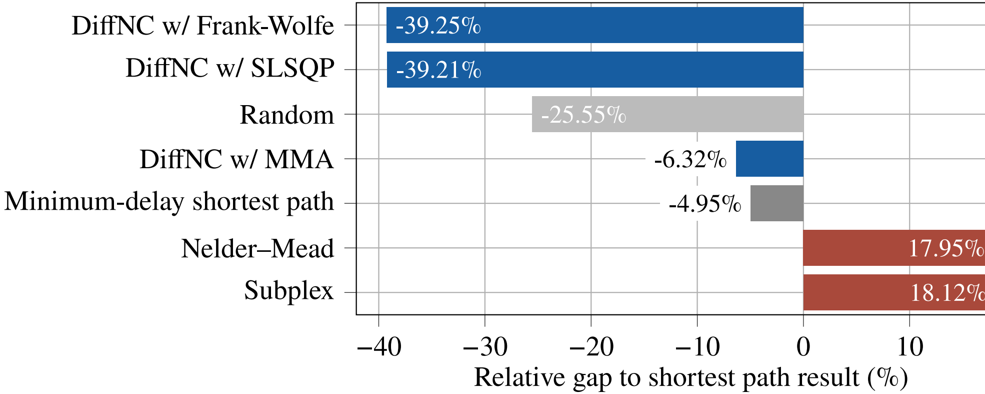
- **Networks to be analyzed (prio + path)**
  - Random networks of industrial size, see table
  - Airbus A350 with ~1100 flows
- **NLP algorithms (max 500 repetitions)**
  - Frank-Wolfe (new, own implementation)
  - Sequential Least Squares Programming SLSQP (best in [Geyer22], NLopt library)
  - Method of Moving Asymptotes MMA (good performance in [Geyer22], NLopt library)
  - Subplex and Nelder Mead (both based on simplex method, gradient not considered)
    - Yet, Subplex performed very well in [Herll25]
  - (Weighted) Shortest Path (weight: lower bound on delay, neglecting queueing effects)
  - Random (500 combinations uniformly at random)

Number of	Min	Mean	Median	Max
Servers	8	17.08	16	31
Flows	5	170.67	164	1001
Virtual flows	9	355.22	343	1884
Path combinations	$10^{1.08}$	$10^{46.04}$	$10^{44.10}$	$10^{229.08}$
Path + priority comb.	$10^{2.58}$	$10^{97.41}$	$10^{94.28}$	$10^{530.41}$

Evaluation I (random networks): Delay Bounds

- **Metric: Deviation from best result and from shortest path (hop count only, no weights)**
- **Remember: NLP algorithms do not guarantee to find the optimum**

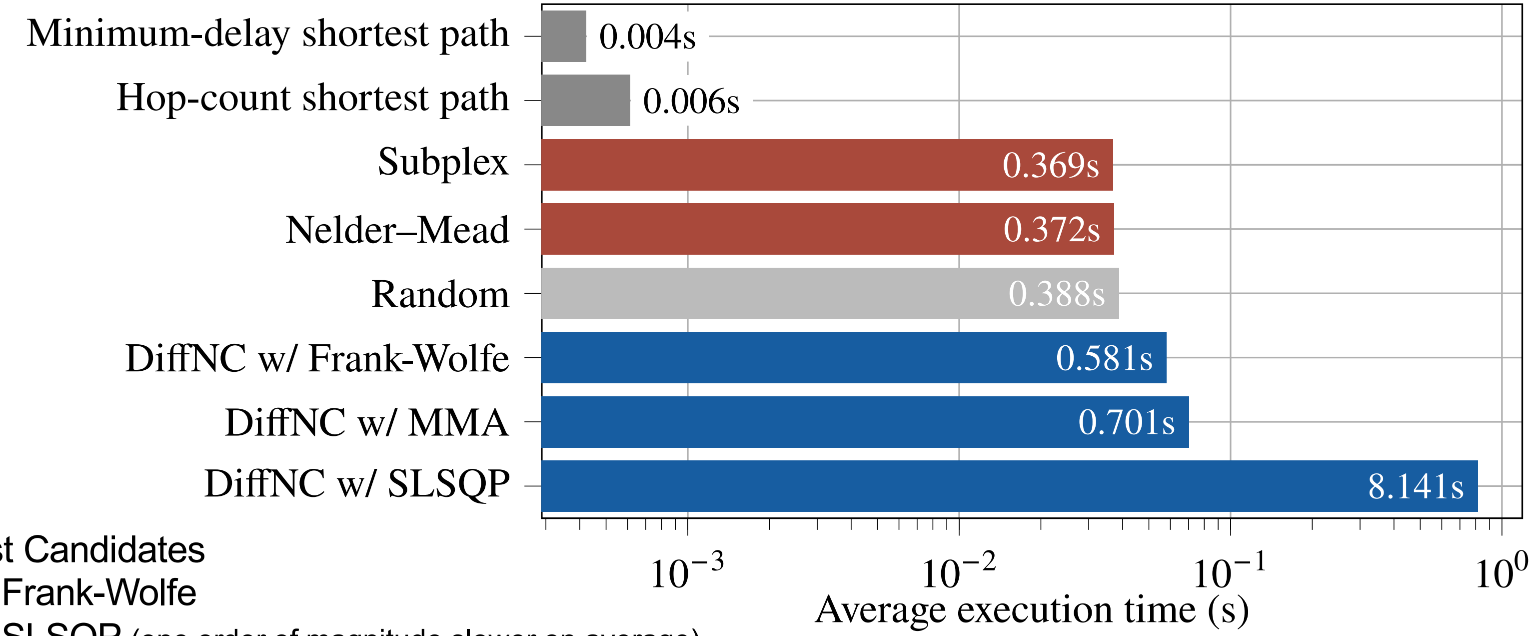




Best Candidates

- Frank-Wolfe
- SLSQP

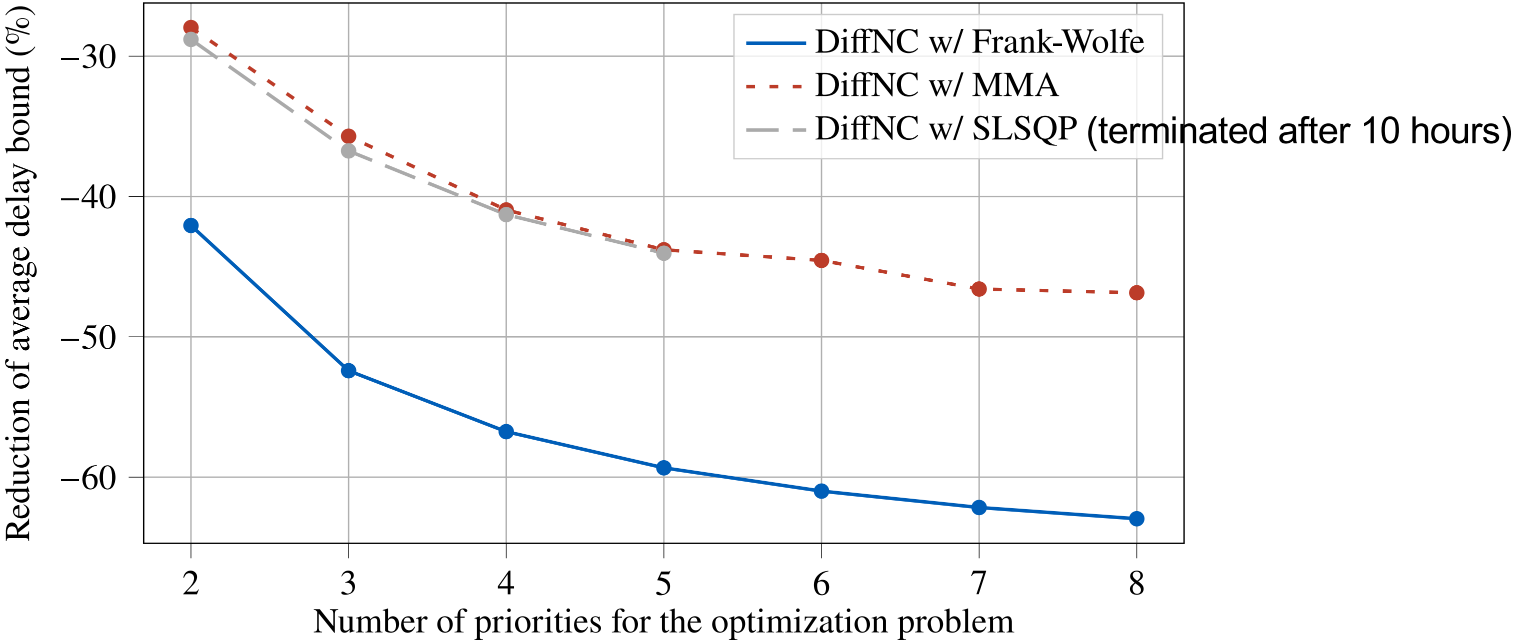
Evaluation I (random networks): Execution Times



Best Candidates
- Frank-Wolfe
- ~~SLSQP~~ (one order of magnitude slower on average)

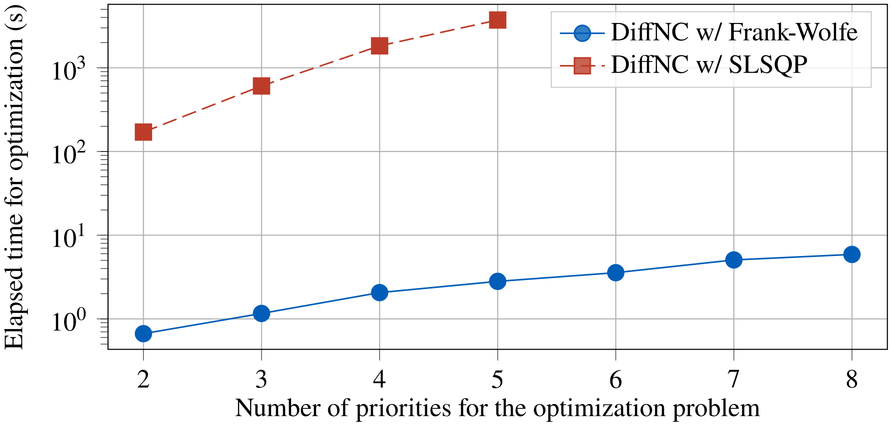
Evaluation II (A350): **Scaling** of Delay Bounds

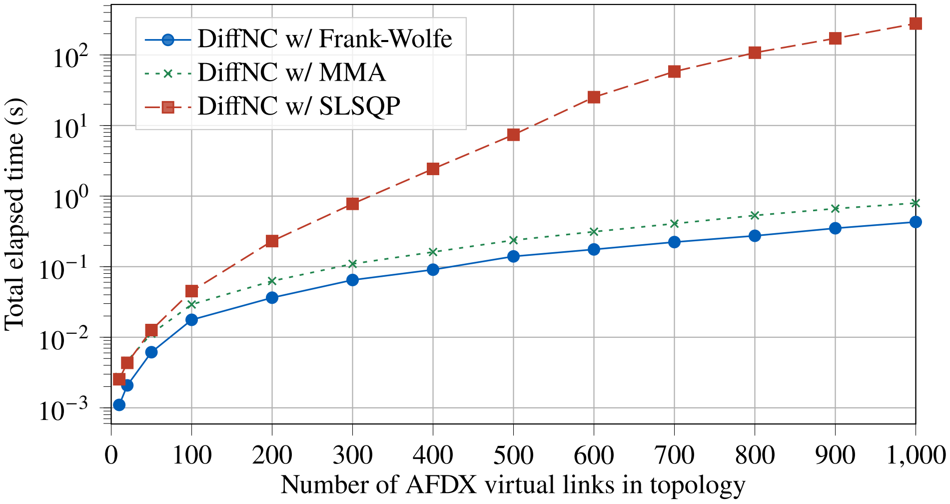
- **Idea: Increase the number of available priority levels and synthesize a configuration**



Evaluation II (A350): **Scaling** of Execution Times

- **Idea(left): Increase the number of available priority levels**
- **Idea (right): flow paths within (so-called virtual links in AFDX)**





RUHR UNIVERSITÄT BOCHUM RUB

Conclusion

- **(Gradient-based) NLP optimization can scale to large, complex problems**
  - Sophisticated tools are key
    - No detour from Network Calculus (min,plus) algebra to “regular” (plus, times) algebra
    - Choosing the right NLP algorithm for more accurate results in shorter times

Future Work

- [Theoretical] Prove convexity of the problem
- [Practical] Extend to more Network Calculus analyses
  - Currently: fixed priorities and SFA under arbitrary multiplexing [Bondorf16]
  - Related stream of work: FIFO multiplexing with another set of open parameters [Herll25]

References

[Geyer22] F. Geyer and S. Bondorf. *Network Synthesis under Delay Constraints: The Power of Network Calculus Differentiability.*
In Proc. of INFOCOM, 2022.

[Cruz91] R. L. Cruz. *A Calculus for Network Delay, Part I: Network Elements in Isolation* and *A Calculus for Network Delay, Part II: Network Analysis.*
In IEEE Transactions on Information Theory, 1991.

[Herll25] Lukas Herll and Steffen Bondorf. *Non-linear Programming for the Network Calculus Analysis of FIFO Feedforward Networks.*
In Proc of ACM/SPEC ICPE 2025

[Bondorf16] Steffen Bondorf and Fabien Geyer. *Generalizing Network Calculus Analysis to Derive Performance Guarantees for Multicast Flows.*
In Proc of ValueTools 2016