# Meta-Scheduling for MultiPath-TCP with NeuroEvolution of Augmenting Topologies

Master-Thesis Final Presentation

**Kay Luis Wallaschek**
kayluis.wallaschek@gmail.com

Boris Koldehofe, Amr Rizk

KOM – Multimedia Communications Lab
Technical University of Darmstadt
Prof. Dr.-Ing. Ralf Steinmetz (Director)
Dept. of Electrical Engineering and Information Technology
Dept. of Computer Science (adjunct Professor)
www.KOM.tu-darmstadt.de

Picture Reference

20. August 2020

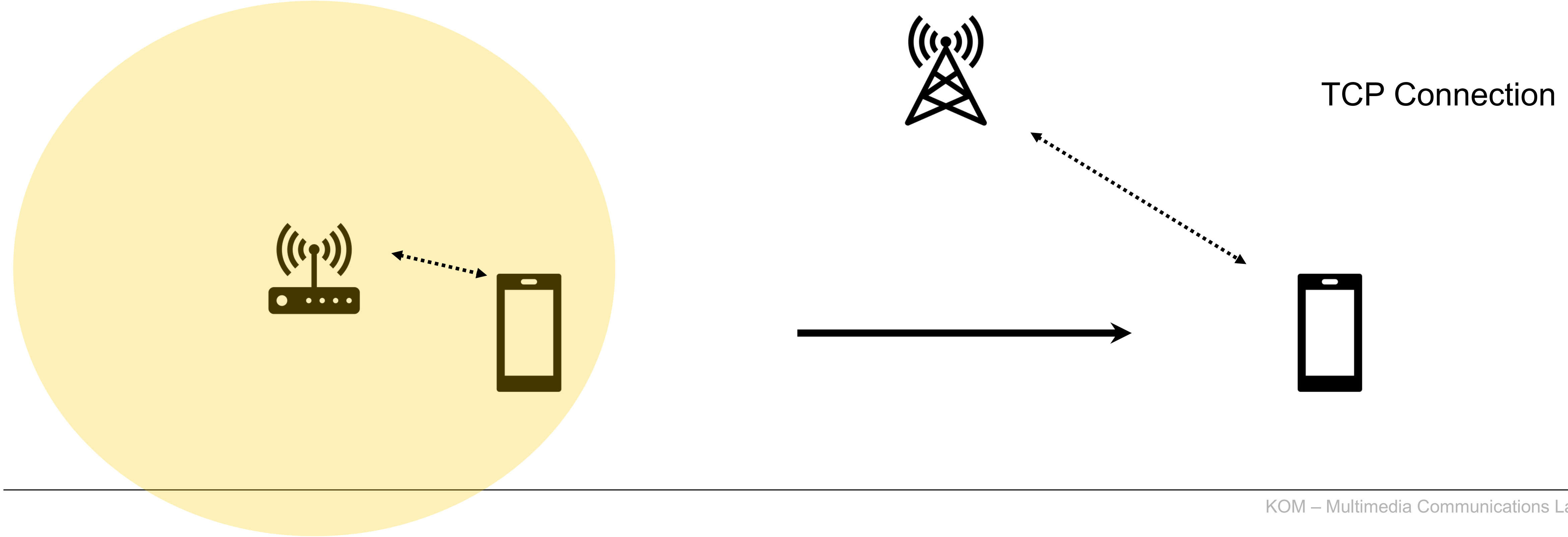
# Outline

- MultiPath-TCP
- Scheduling
- Related Work
- Goal / Meta-Scheduling
- Learning
- Evaluation
- Insights/Future Work

# MultiPath-TCP
## Background

### Motivation

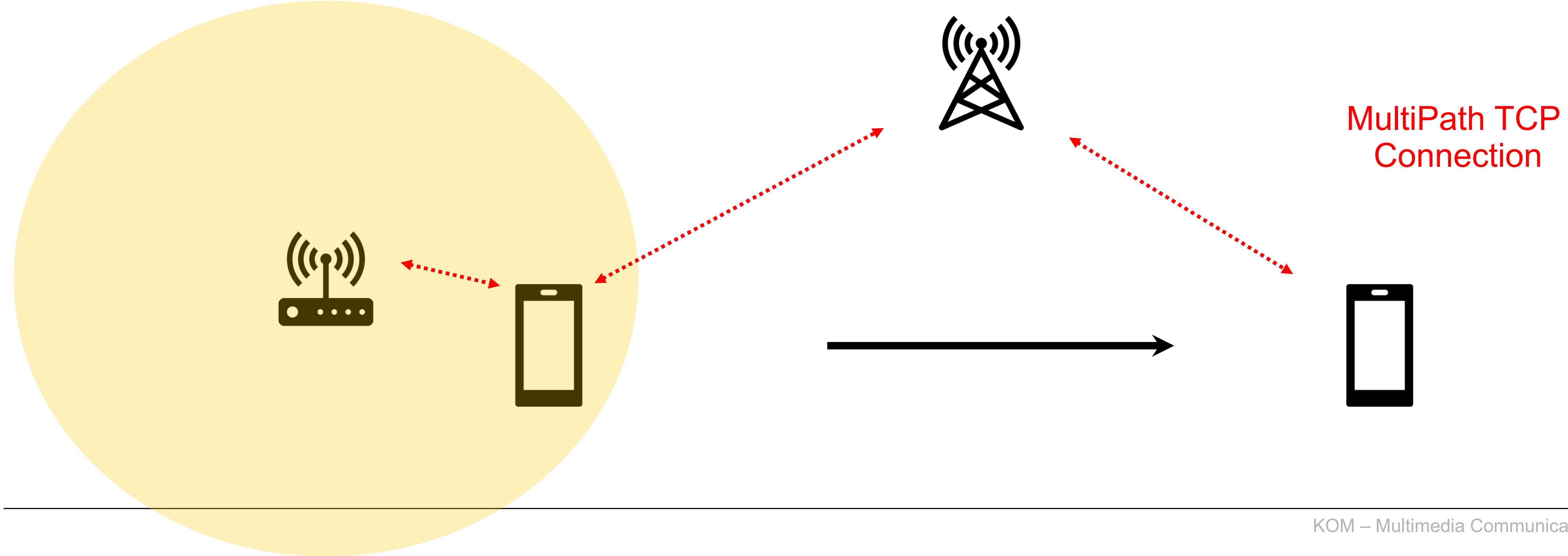
- Most devices have multiple network interfaces, that are not utilized by traditional TCP.
- Connections break down if the origin-interface breaks down. (Handover fails)
- Performance/Reliablity improvement



# MultiPath-TCP
## Background

### Motivation

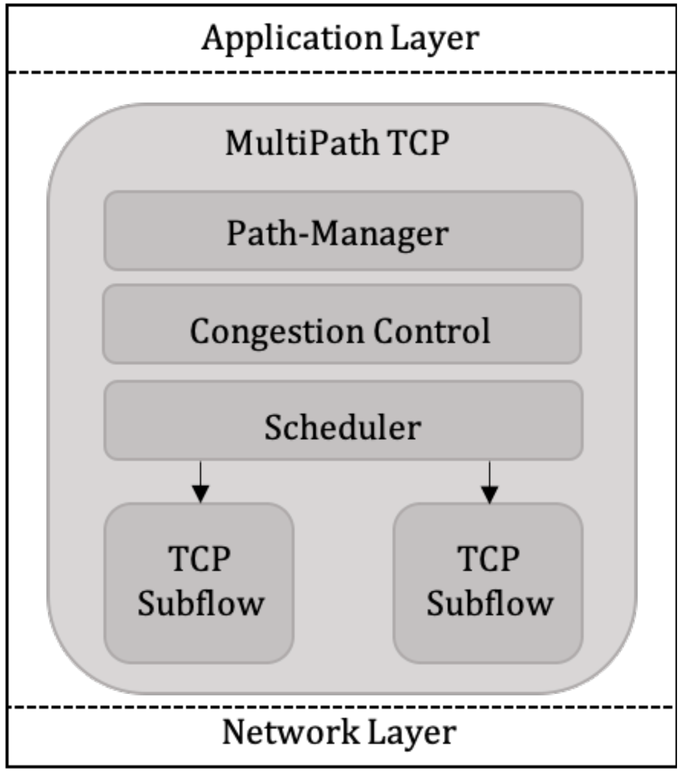
- Most devices have multiple network interfaces, that are not utilized by traditional TCP.
- Connections break down if the origin-interface breaks down. (Handover fails)
- Performance/Reliablity improvement



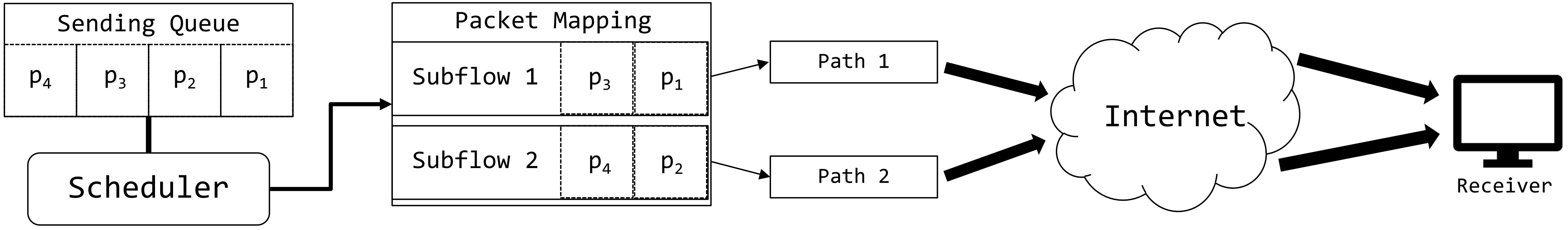
# MultiPath-TCP - Components
## Background

### Components

- Path Manager
  - Management of Path Creation.
- Congestion Control
- Scheduler
  - When and over which path should a packet be sent?



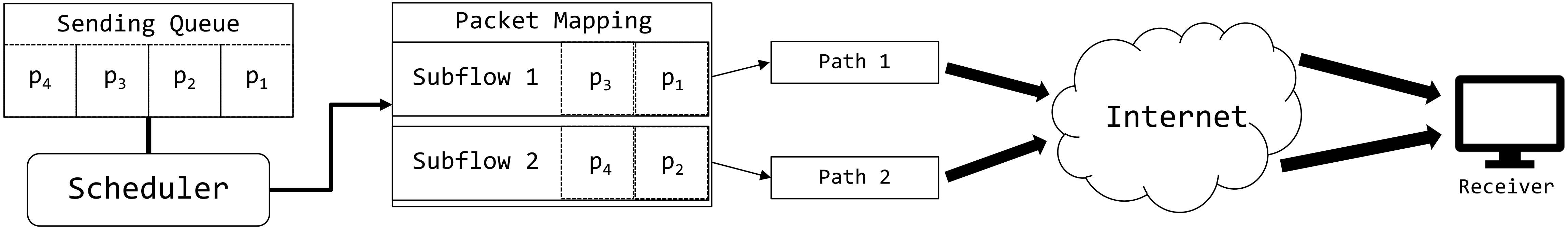
# MultiPath-TCP
## Scheduling




## Questions in Scheduling

- According to what algorithm should the packets in the sending queue be mapped?
- What if paths are heterogeneous?
- Is the scheduling decision optimal?

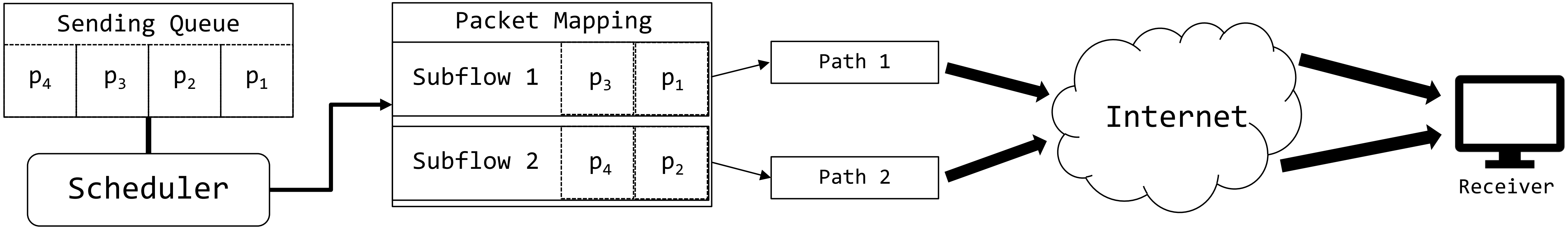
# MultiPath-TCP
## Naïve Scheduler




### RoundRobin

- Spread the packets evenly over all available paths.
- What if the paths are not homogeneous?
- What if one path has 10x the throughput of another one?

# MultiPath-TCP
## Default Scheduler




### MinRTT

- Saturate the flow with the lowest RTT.
  - Then take the next best flow.

- What if the path with the lowest RTT has bad performance?

# Related Work

## Rule based

- Use „simple“ rules to schedule packets.
- Usually developed for a specific use case.
  - Video Streaming, Load-time reduction of websites, ...

## Machine Learning based (Deep Neural Networks)

- Learn scheduling in operation.
- Flexible, but needs to relearn if the topology changes

# Meta-Scheduling: The MAKI-Scheduler

## Assumption

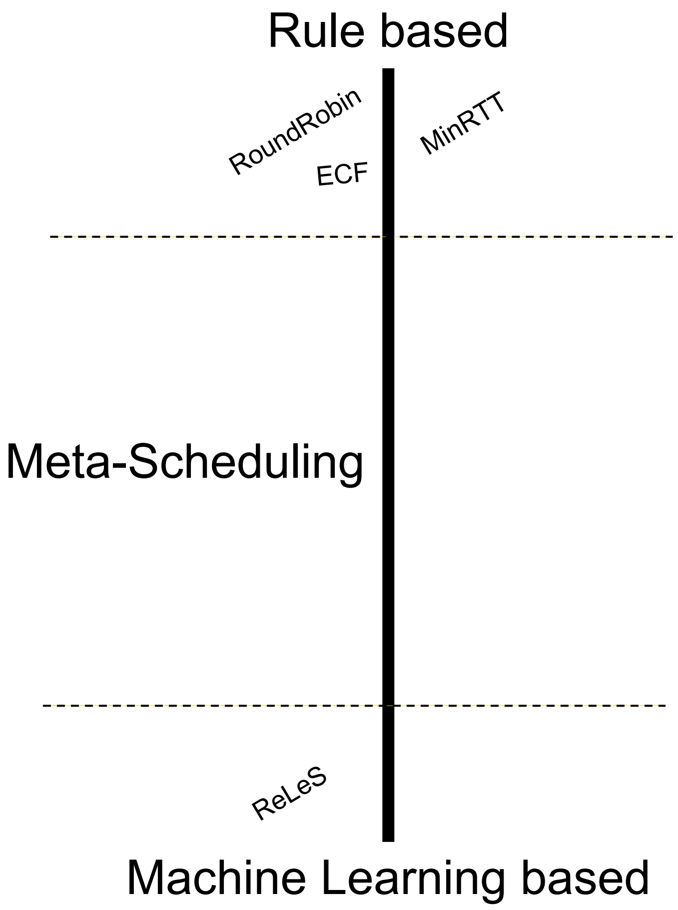
- There is no scheduler that performs optimal in every topology,
- That is also low-overhead, fast and simple.

## Schedule Schedulers

- Instead of using only one scheduler, switch between multiple schedulers.
- Use the scheduler that performs optimal to the current topology.

## Goal

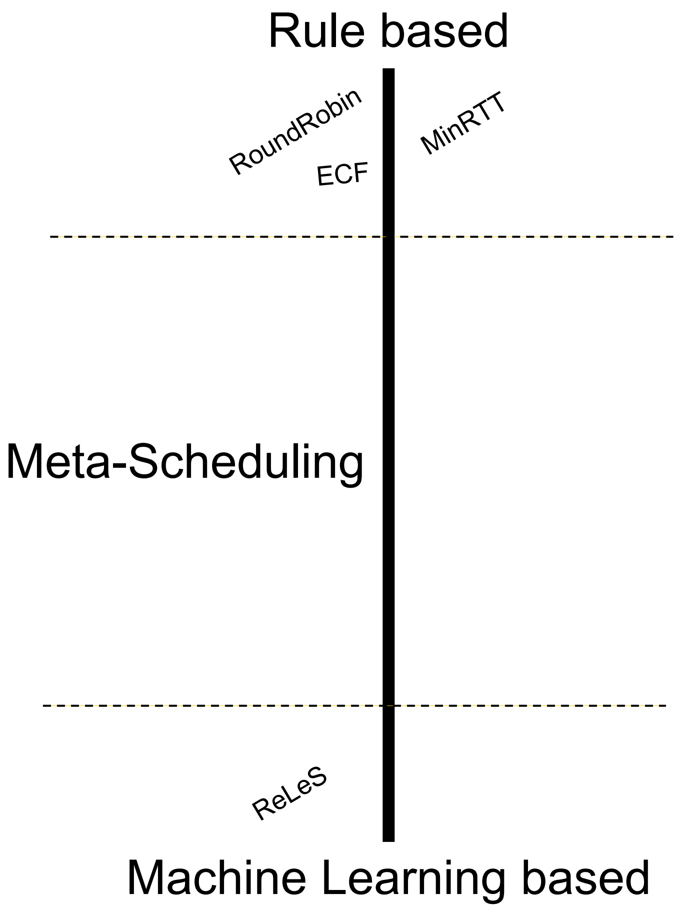
- Combine machine learning and rule based schedulers to create a flexible scheduler for a wide range of topolgies without the overhead of relearning.



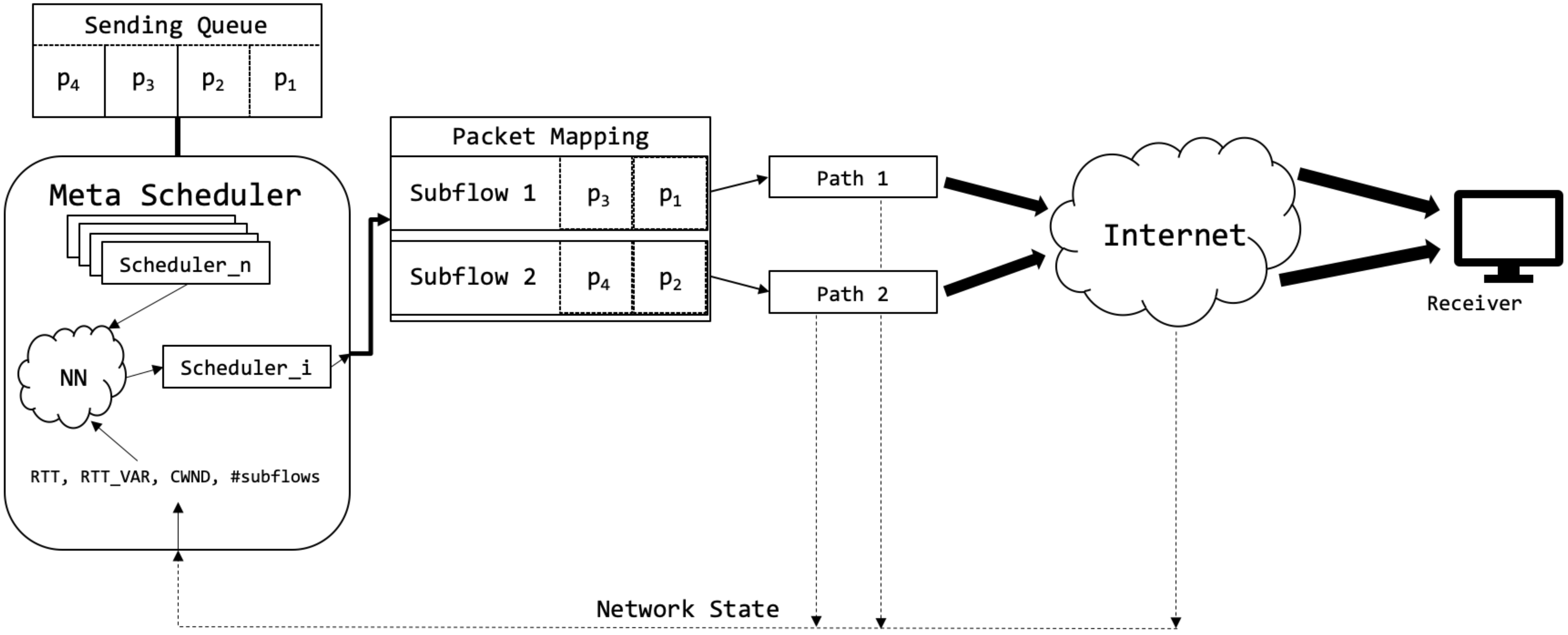
# Meta-Scheduling: The MAKI-Scheduler

## Model

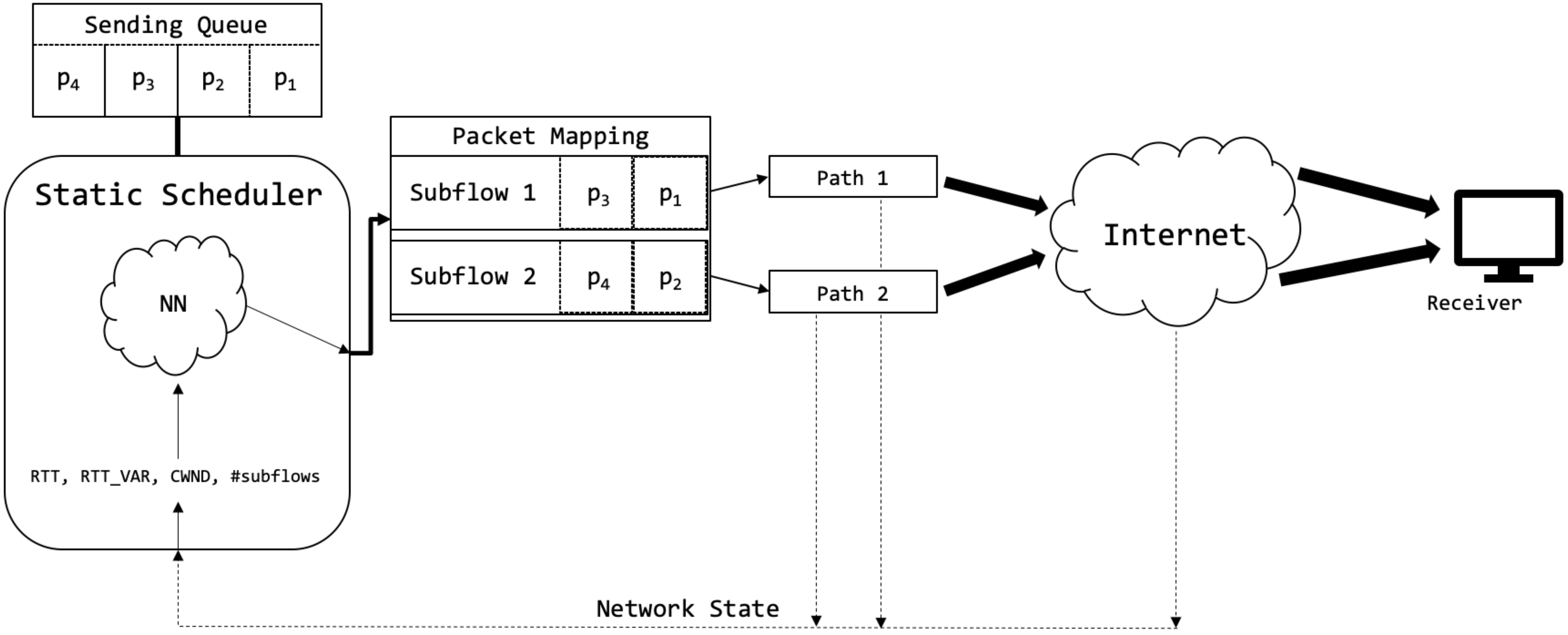
- Create a neural network that chooses schedulers according to the current condition.
- The neural network should be small and low-overhead.
  - Also for easy kernel integration with ProgMP.
- Online learning should be avoided.



# Model – Meta Scheduler



# Model – Static Scheduler



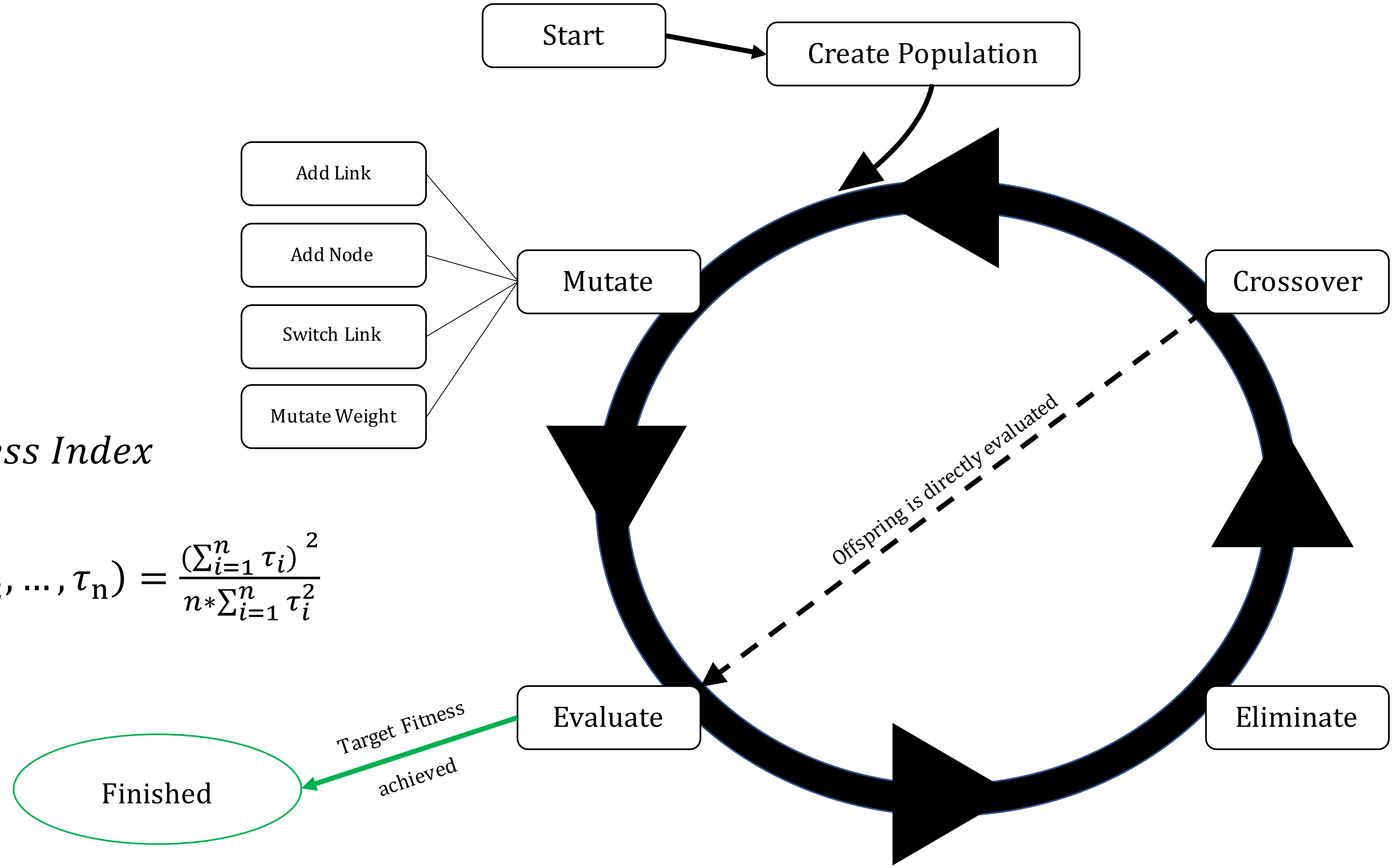
# Learning – NEAT

TECHNISCHE UNIVERSITÄT DARMSTADT

## How to create the neural networks?

- Use genetic algorithm NEAT
  - Creates minimal sized neural networks
  - by gradually adding topology to the neural networks.
- Optimize according to a fitness function
  - (Scheduling performance)
- NEAT
  - Starts with an “empty” neural network (without hidden structure)
  - Mutates the neural network and optimizes with “survival of the fittest”

# Learning – NEAT




## Fitness Function

- $\frac{Total\ Throughput}{Avg(MeanDelay)} * Jain's\ Fairness\ Index$

- Jain‘s Fairness Index: $J(\tau_1, \tau_2, \ldots, \tau_{\text{n}}) = \frac{\left(\sum_{i=1}^{n} \tau_i\right)^2}{n * \sum_{i=1}^{n} \tau_i^2}$

# Created Schedulers

**Static Scheduler:**

- *static_neat*

**Meta Scheduler:**

- *meta2*
  - Access to MinRTT and RoundRobin.
- *meta_triple*
  - Access to MinRTT, RoundRobin and *static_neat.*

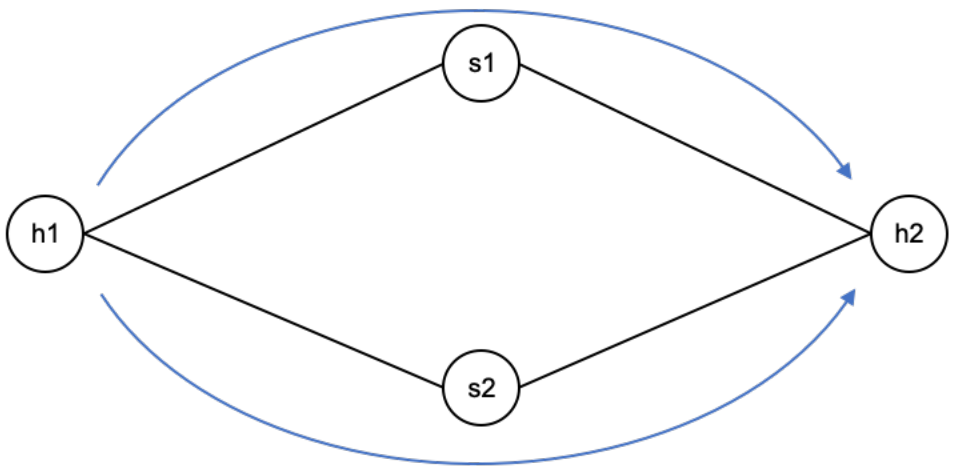
# Evaluation – Simple Two Flow – Setup

**Configuration:**

- 10 Mbit/s
- 5 ms / 1 ms
- 2 TCP Flows h1 -> h2
  (via s1 and s2)
- 1 MPTCP Connection h1 -> h2

**Schedulers optimized for it:**

- *static_neat, meta_triple*



# Evaluation – Double Dumbbell (1) – Setup

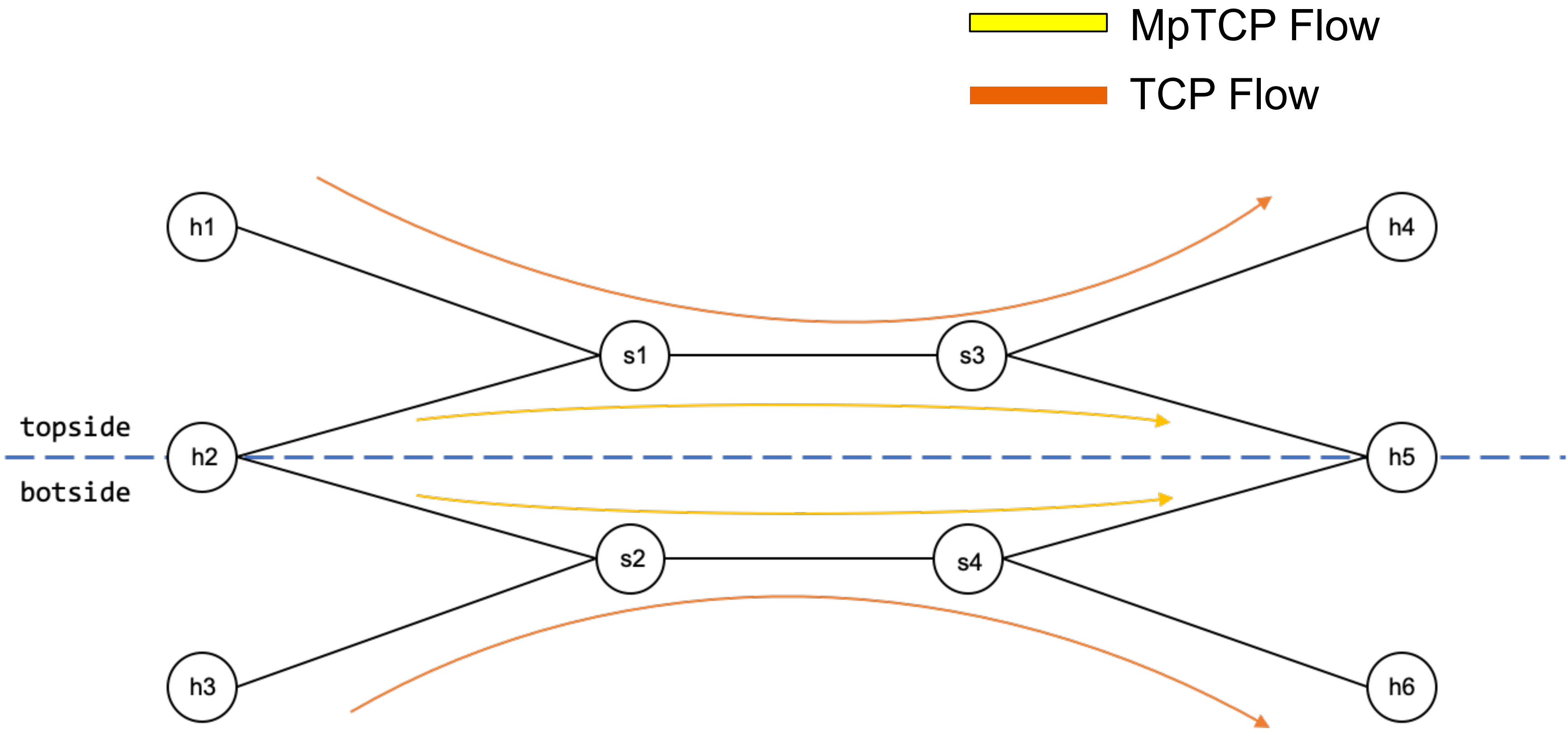
TECHNISCHE UNIVERSITÄT DARMSTADT

## Configuration:

- topside: 10 Mbit/s,  botside: 1 Mbit/s
- 2 ms / 20 ms
- heterogeneous

## Schedulers optimized for it:

- *meta2, meta_triple*



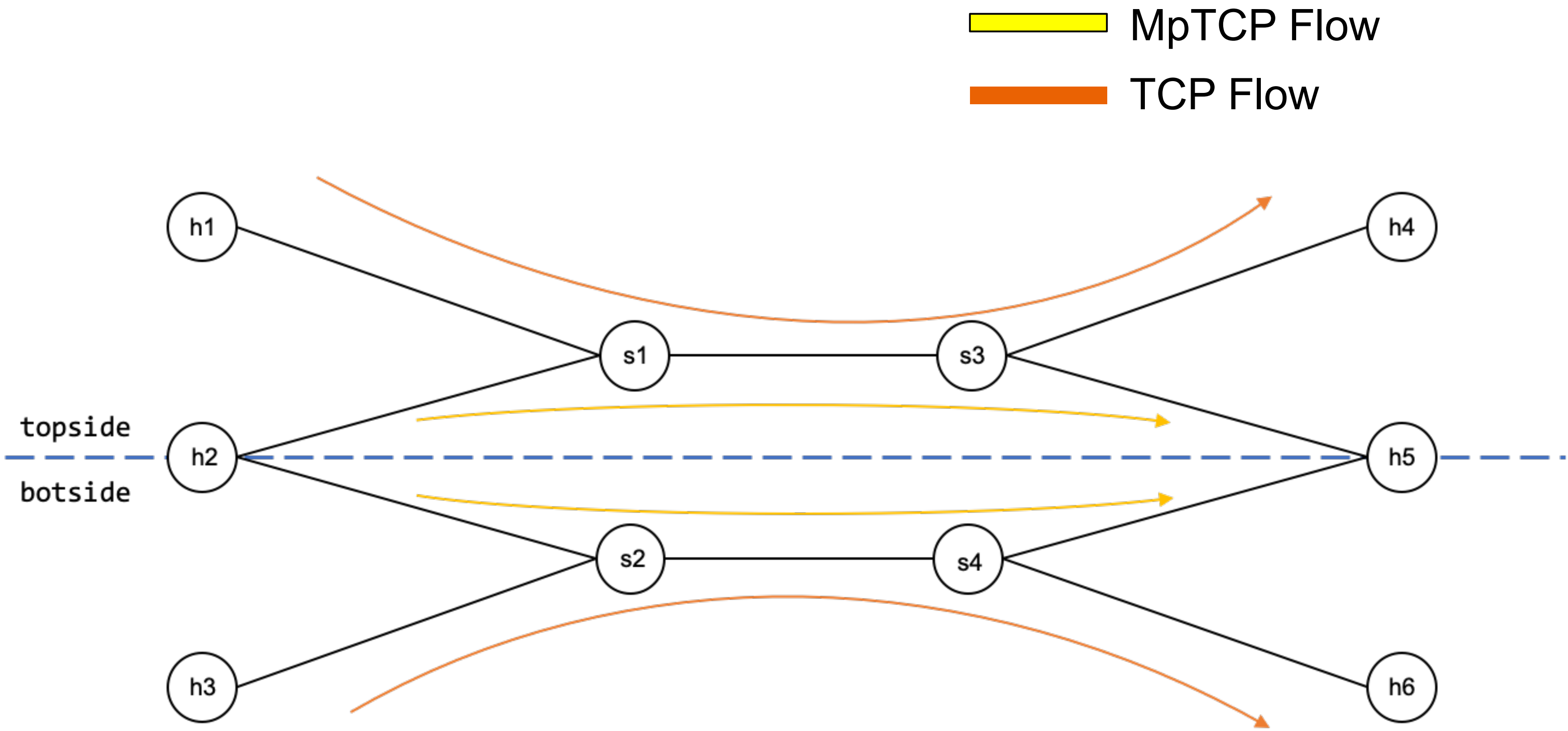
# Evaluation – Double Dumbbell (2) – Setup

**Configuration:**

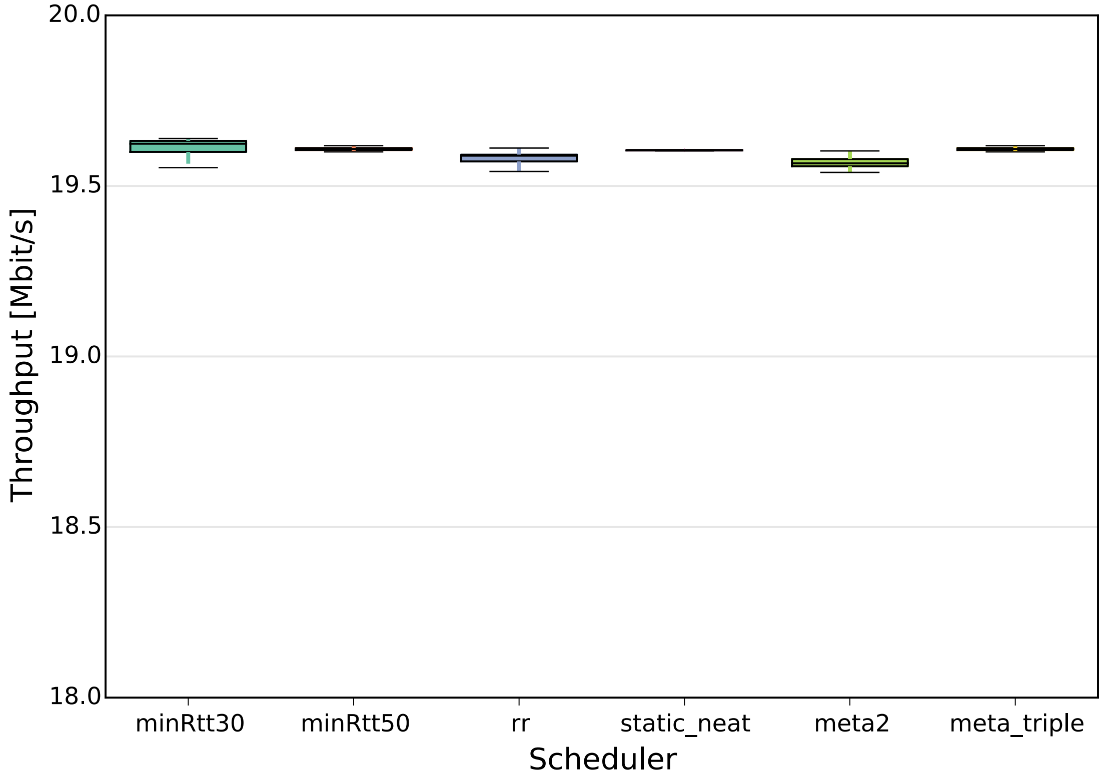
- topside: 10 Mbit/s, botside: 10 Mbit/s
- 5 ms / 1 ms
- homogeneous

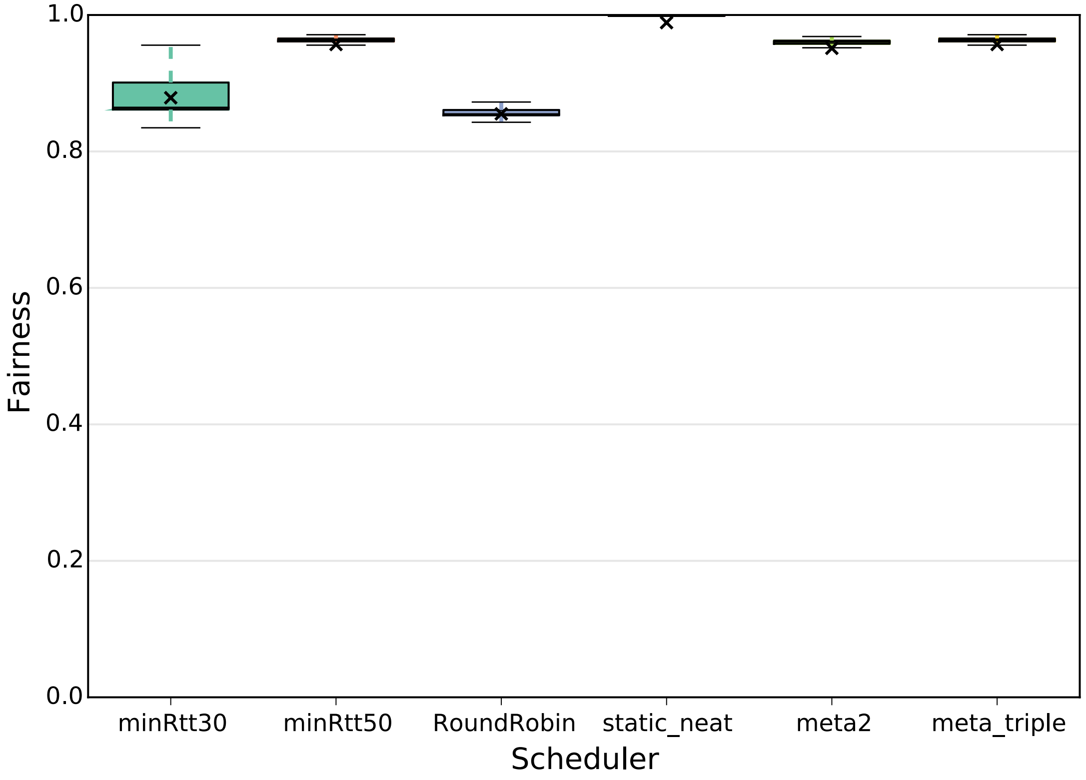
**Schedulers optimized for it:**

- *meta2, meta_triple*

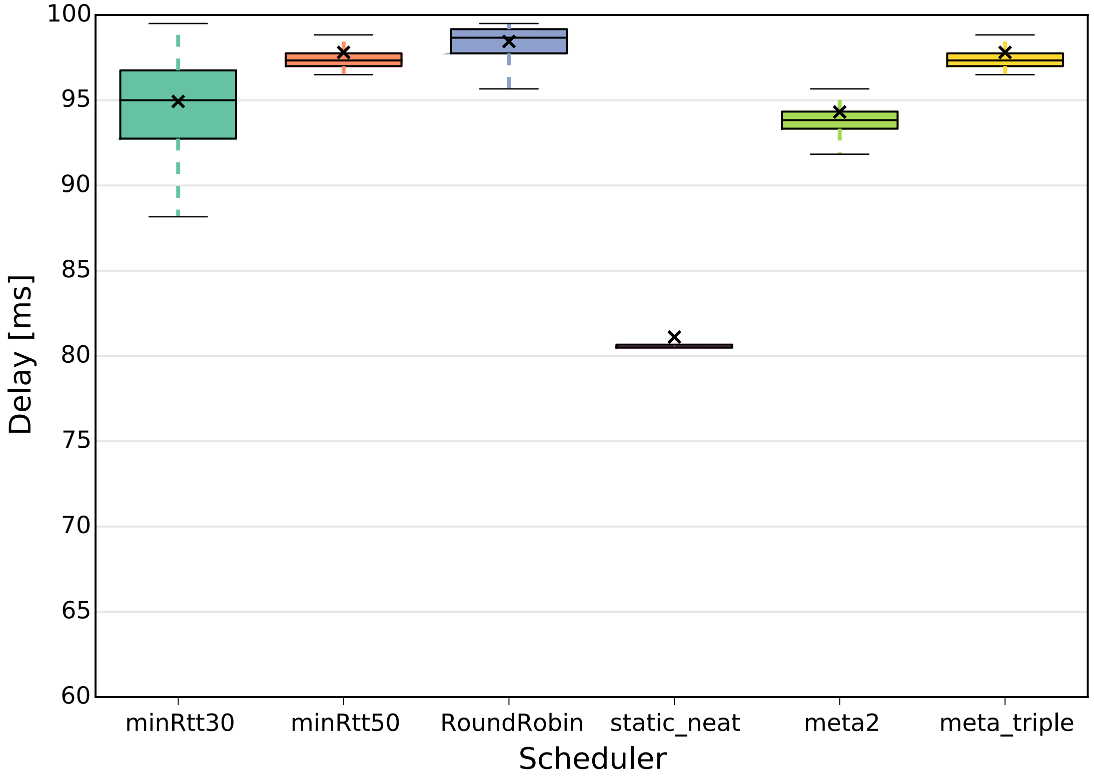


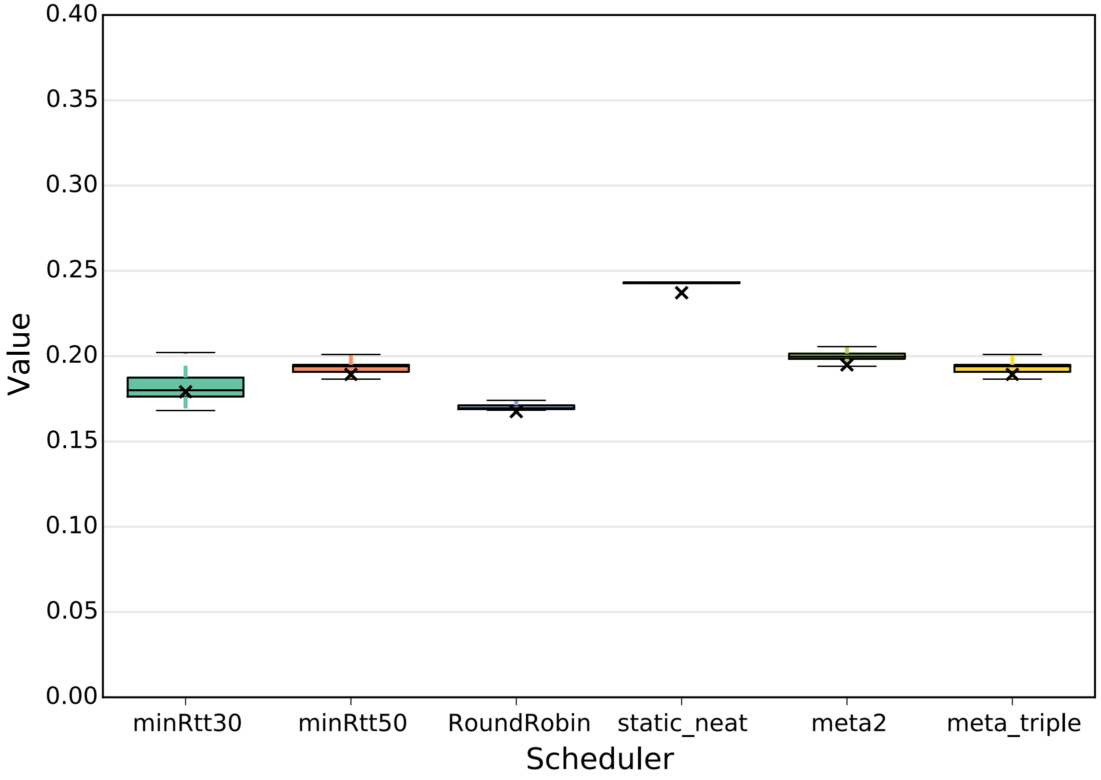
# Evaluation – Simple Two Flow – Results



# Evaluation – Simple Two Flow – Results



# Evaluation – Simple Two Flow – Results

| | MinRtt30 | MinRtt50 | RoundRobin | static_neat | meta2 | meta_triple |
|---|---|---|---|---|---|---|
| Throughput | 0.00% | -0.03% | -0.17% | -0.06% | -0.25% | -0.03% |
| Delay | 0.00% | 3.03% | 3.71% | -14.55% | -0.65% | 3.03% |
| Fairness | 0.00% | 8.88% | -2.68% | 12.53% | 8.25% | 8.88% |
| Value | 0.00% | 5.70% | -6.48% | 32.43% | 8.80% | 5.70% |

# Evaluation – Double Dumbbell (1) – Results

| | MinRtt30 | MinRtt50 | RoundRobin | static_neat | meta2 | meta_triple |
|---|---|---|---|---|---|---|
| Throughput | 0.00% | 0.18% | 1.56% | 2.92% | 0.03% | 0.35% |
| Delay | 0.00% | 0.45% | 24.12% | 27.17% | -0.61% | -0.47% |
| Fairness | 0.00% | 2.43% | 13.58% | -48.58% | 3.19% | 0.43% |
| Value | 0.00% | 2.49% | -6.60% | -57.18% | 3.60% | 1.27% |

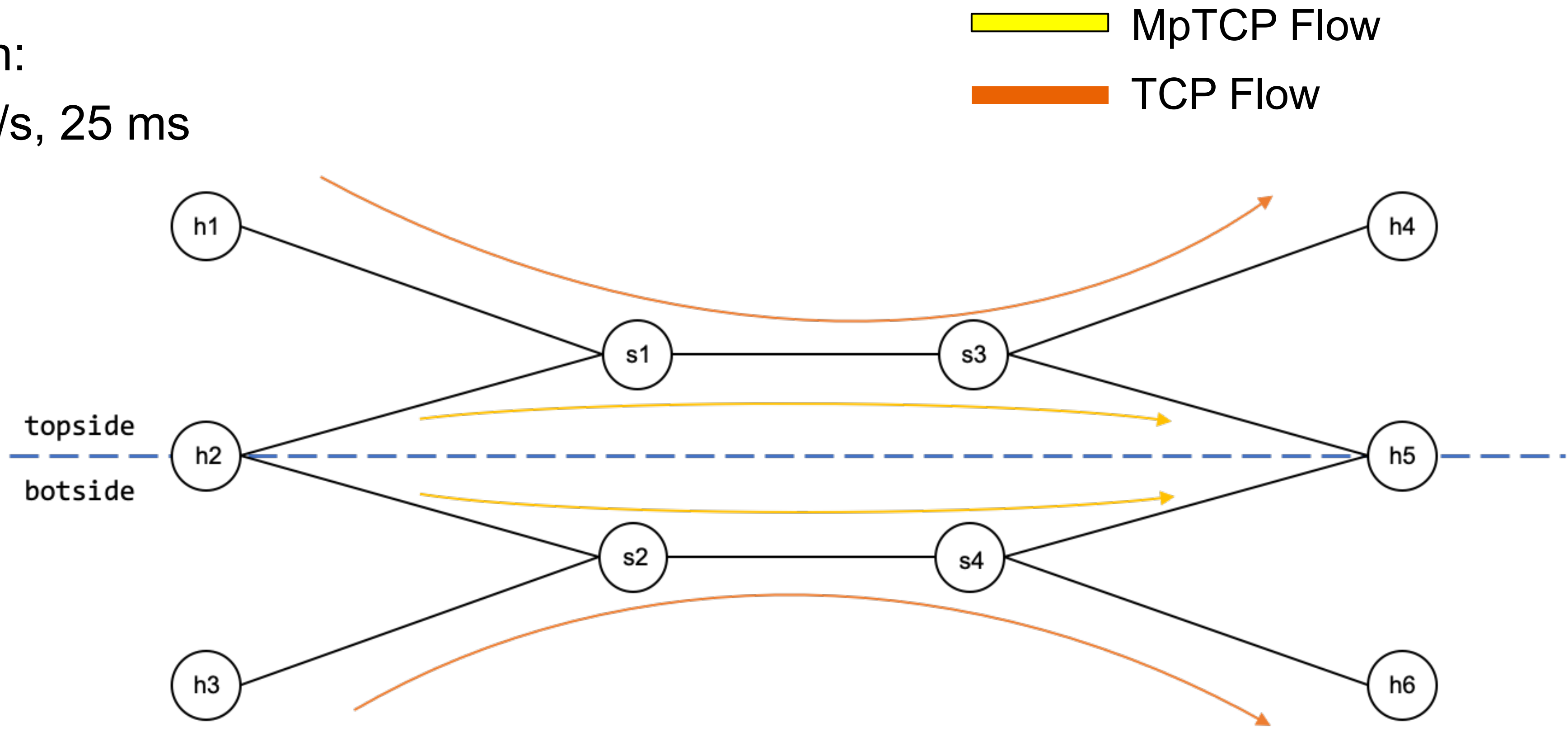
# Evaluation – Double Dumbbell (2) – Results

| | MinRtt30 | MinRtt50 | RoundRobin | static_neat | meta2 | meta_triple |
|---|---|---|---|---|---|---|
| Throughput | 0.00% | -0.30% | 1.92% | 6.37% | 2.30% | -0.30% |
| Delay | 0.00% | -0.03% | -1.29% | 0.01% | -2.09% | -0.86% |
| Fairness | 0.00% | -5.17% | 19.21% | -10.79% | 17.11% | -2.81% |
| Value | 0.00% | -5.46% | 22.08% | -5.45% | 22.14% | -2.36% |

# Evaluation – Double Dumbbell (3) – Setup

**Configuration:**

- topside/botside: 10 Mbit/s
- 5 ms / 1 ms
- Halfway in Simulation:
  botside: 3 Mbit/s, 25 ms



# Evaluation – Double Dumbbell (3) – Results

TECHNISCHE UNIVERSITÄT DARMSTADT

| | MinRtt30 | MinRtt50 | RoundRobin | static_neat | meta2 | meta_triple |
|---|---|---|---|---|---|---|
| Throughput | 0.00% | -1.20% | 4.36% | 9.49% | 4.37% | -1.26% |
| Delay | 0.00% | -0.93% | 1.04% | 1.58% | -2.10% | -0.97% |
| Fairness | 0.00% | -6.03% | 12.03% | -29.22% | 11.36% | -4.93% |
| Value | 0.00% | -6.46% | 15.23% | -24.31% | 18.15% | -5.14% |

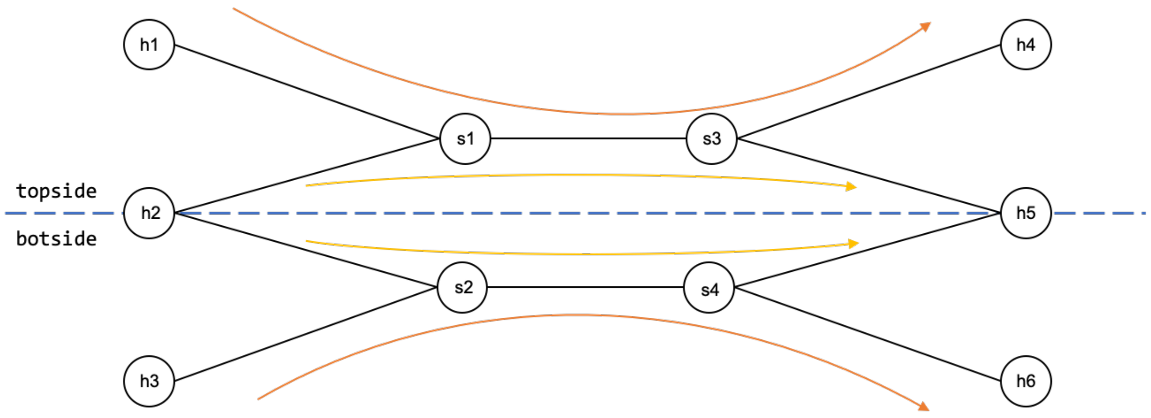
# Evaluation – Mininet

**To check if the created schedulers also work outside of simulations.**

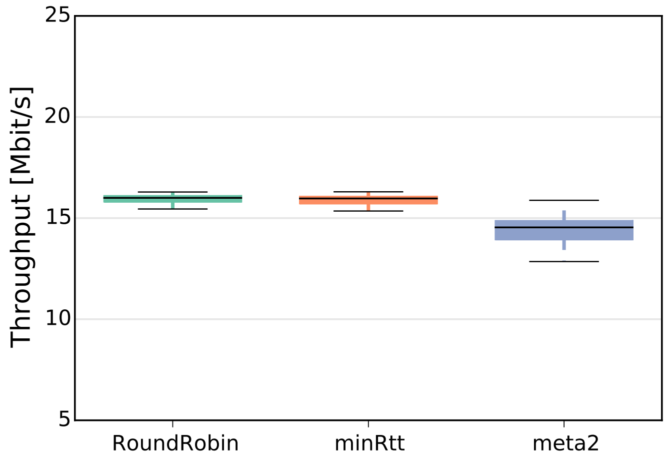
**Create comparable topology to the one of ns-3 in mininet**

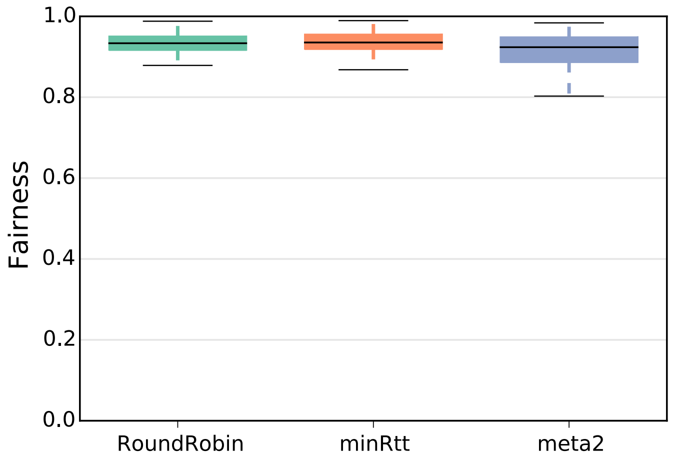
- Different applications
  - BulkSendApplication / iperf3

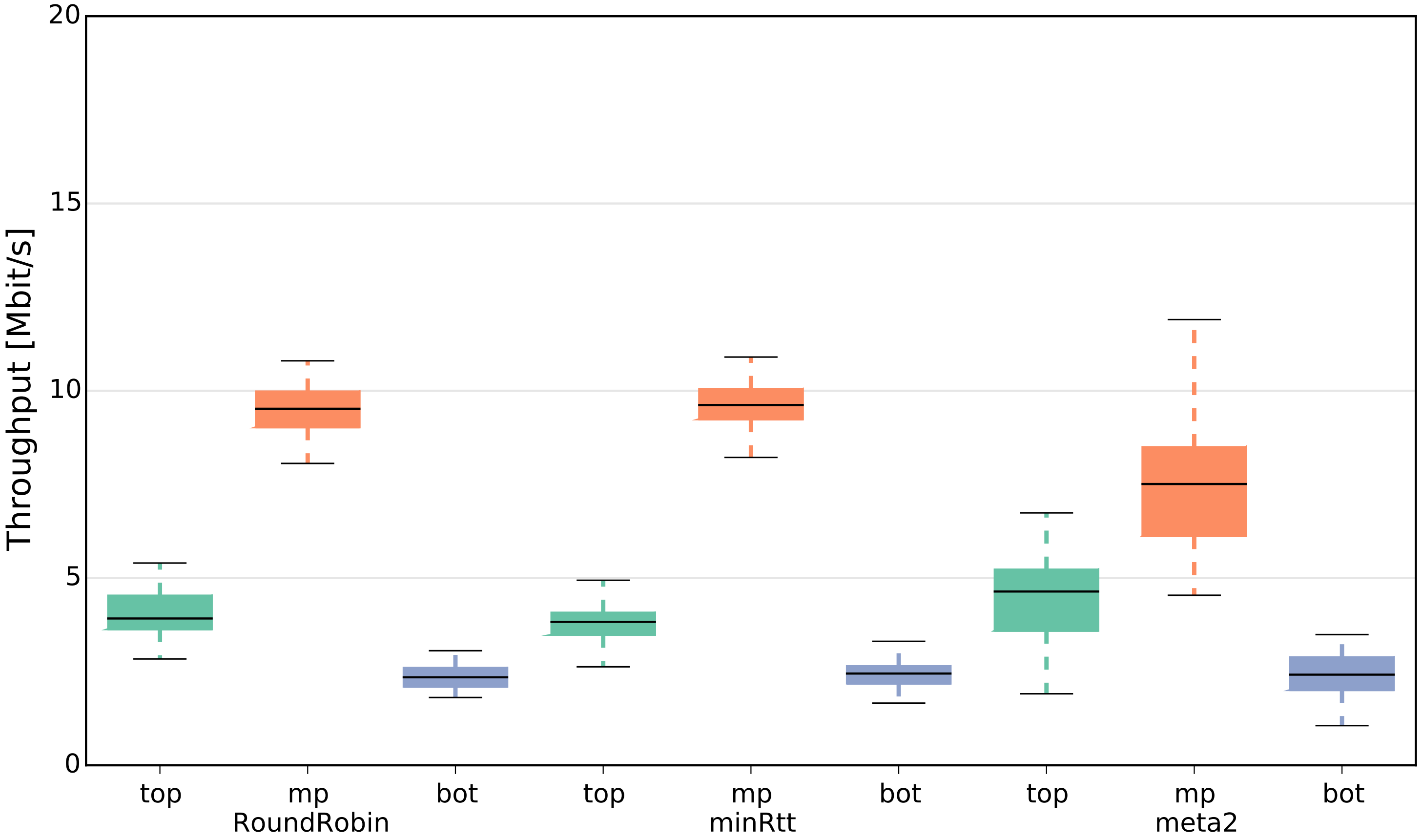
- Use Double Dumbbell (3) as it is unseen for the schedulers in ns-3 and mininet



# Evaluation – Mininet – Results



# Conclusions – Insights

**It is possible to create static and meta schedulers with the presented approach.**

**It is of benefit to be able to change the scheduler within a single data transmission, even though the topology is constant.**

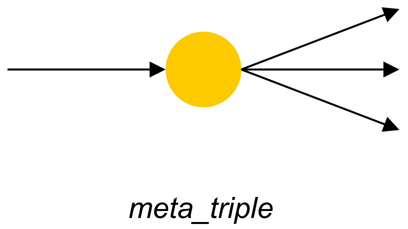
**The created schedulers do not yet perform satisfactorily in mininet with ProgMP.**

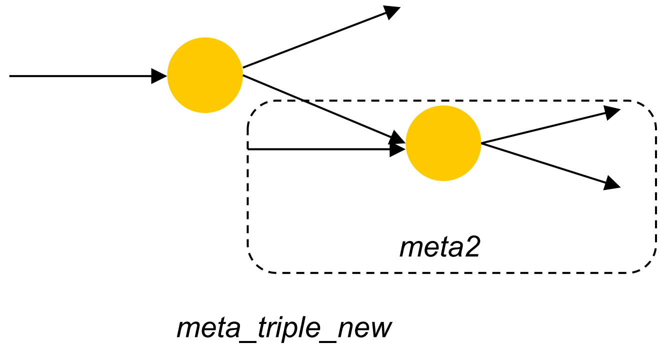
- Could have various reasons:
  - Implementation in ns-3 and linux too different. (Scheduler handling different)
  - BulkSendApplication and iperf3 could be not comparable.
  - Simulation vs. Emulation

# Conclusions – Future Work

## Accelerate Evolutions

## Improvements on *meta_triple*

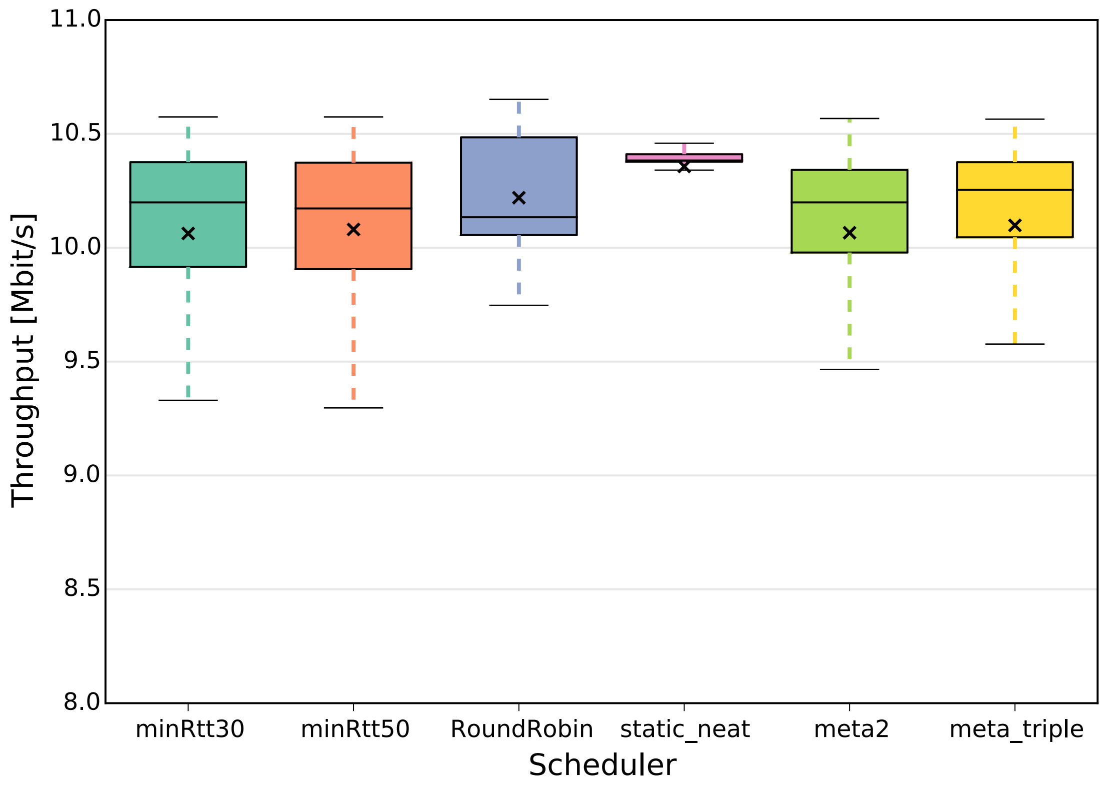


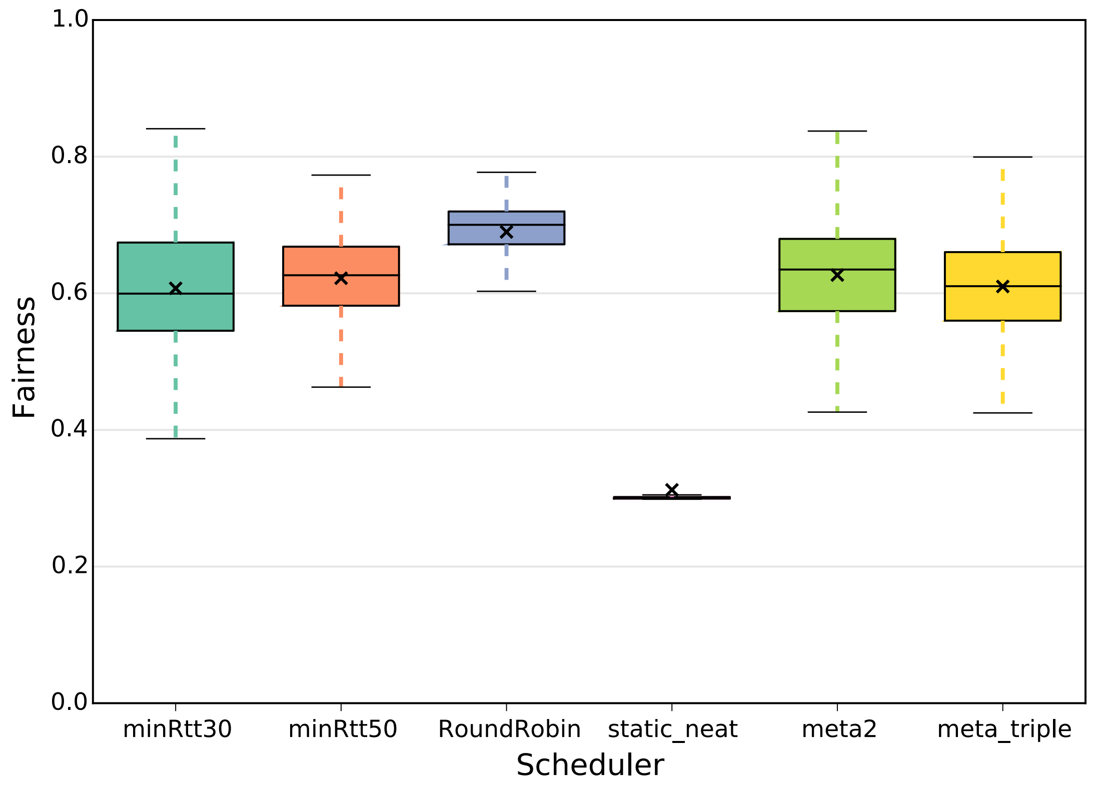




## Use Different Schedulers.

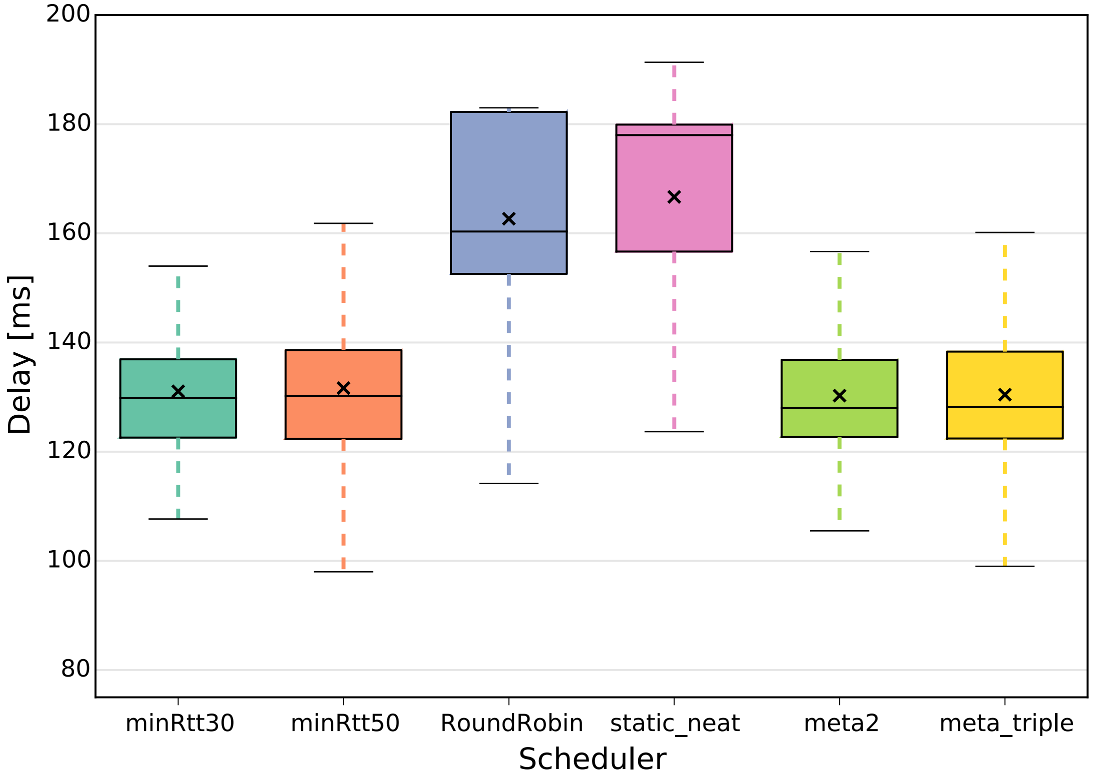
## Extension to MultiPath TCP in ns-3

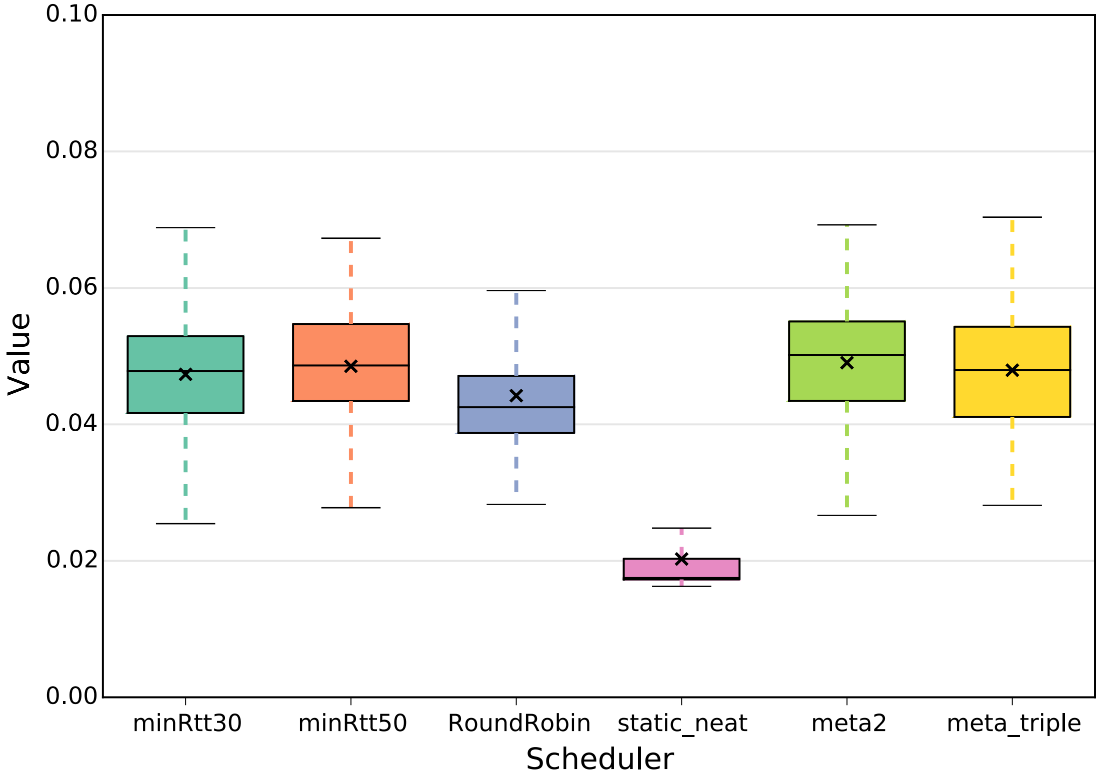
# Evaluation – Double Dumbbell (1) – Results



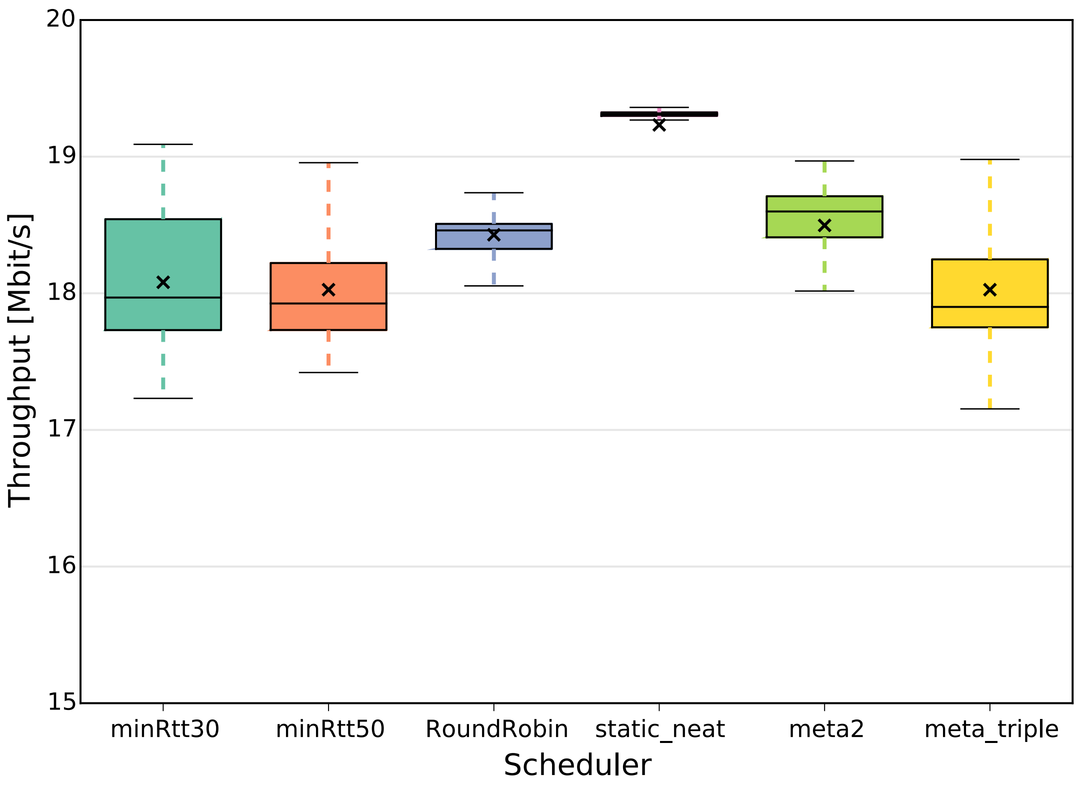



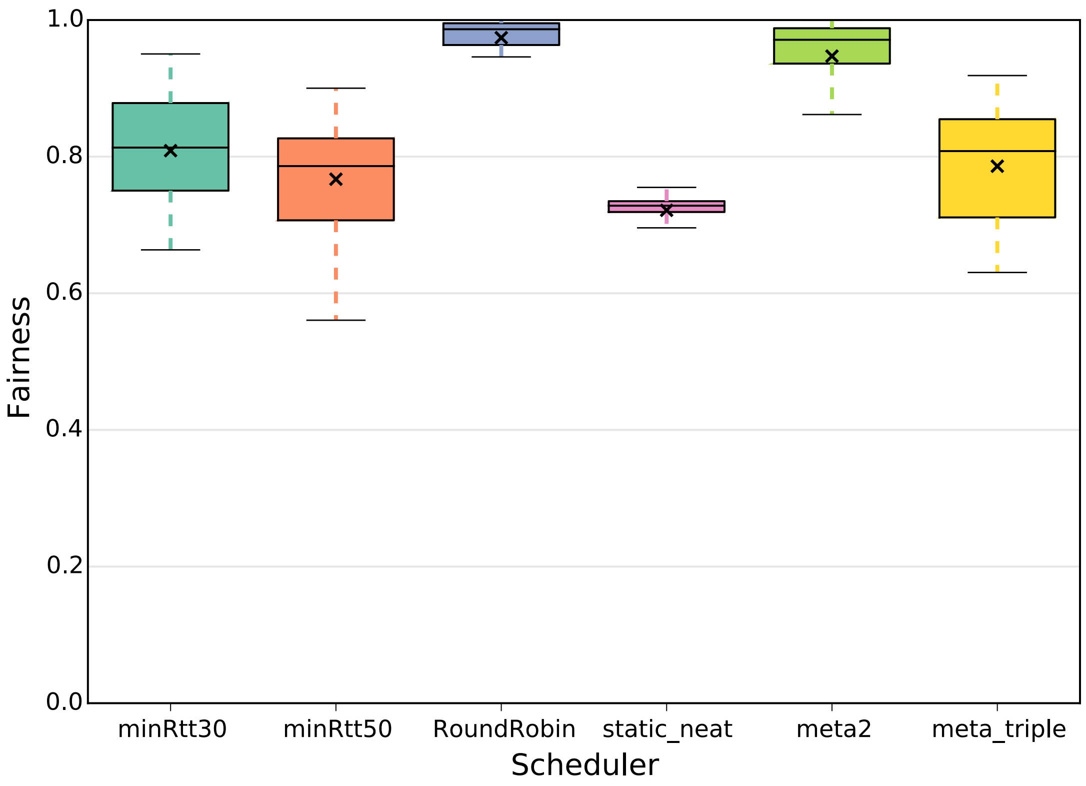
# Evaluation – Double Dumbbell (1) – Results



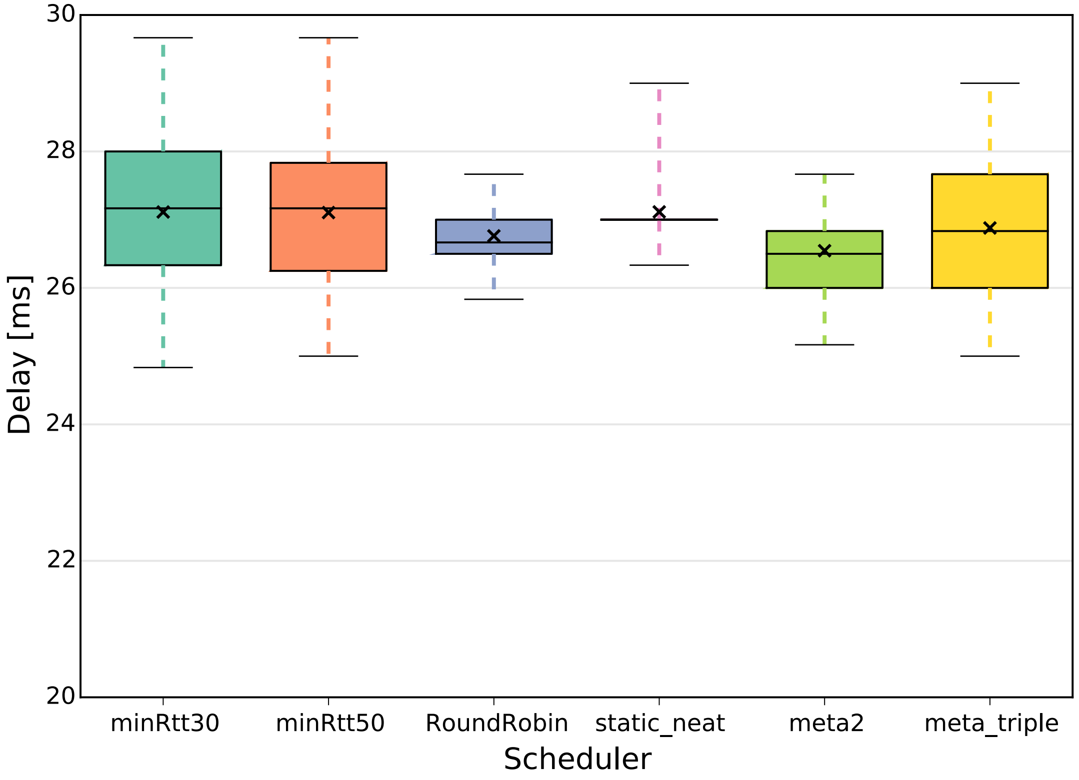



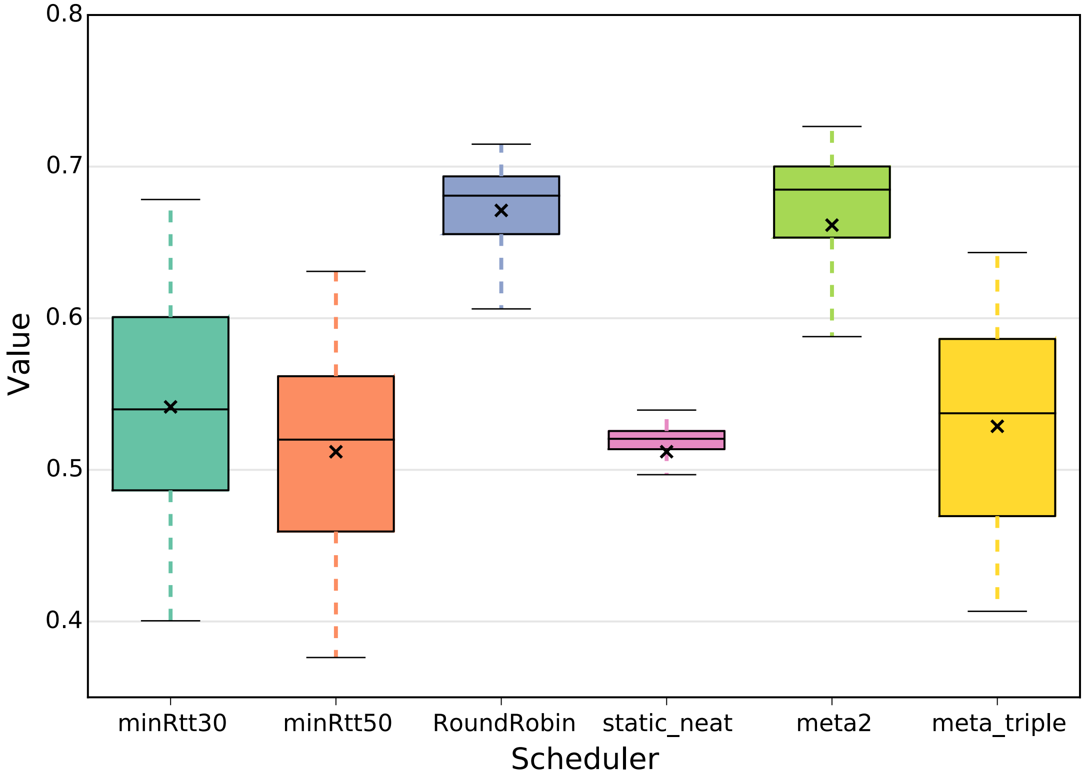
# Evaluation – Double Dumbbell (2) – Results



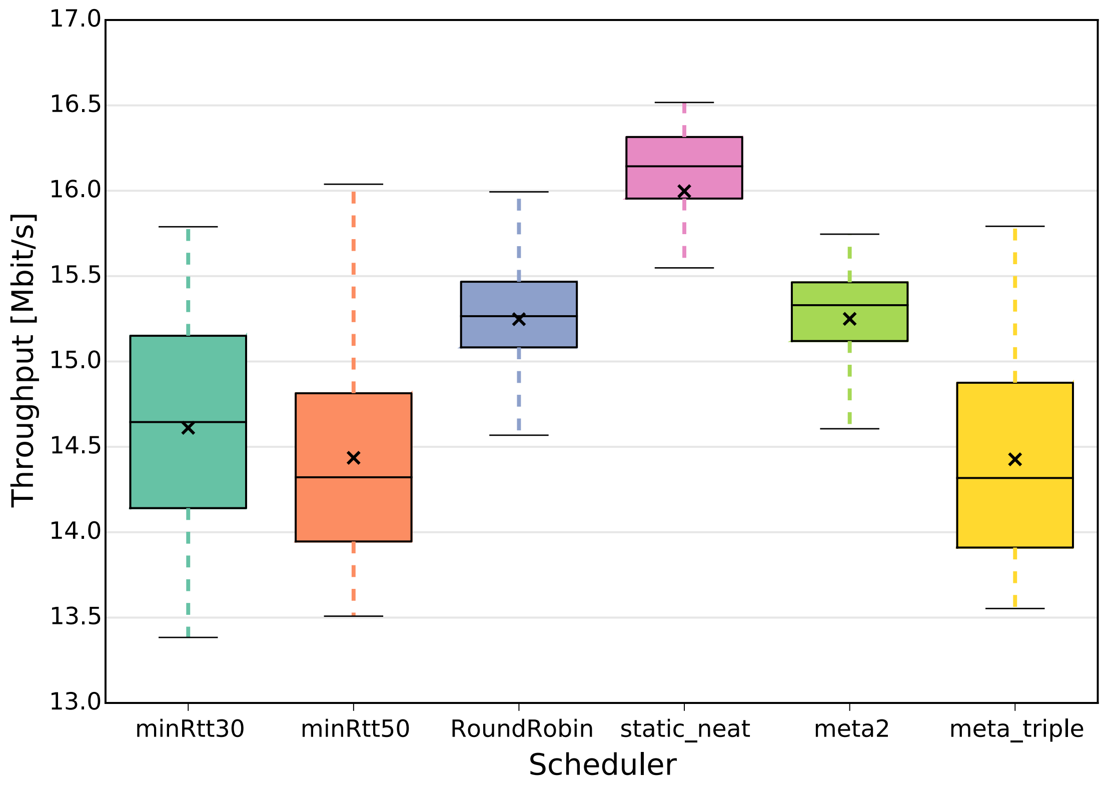



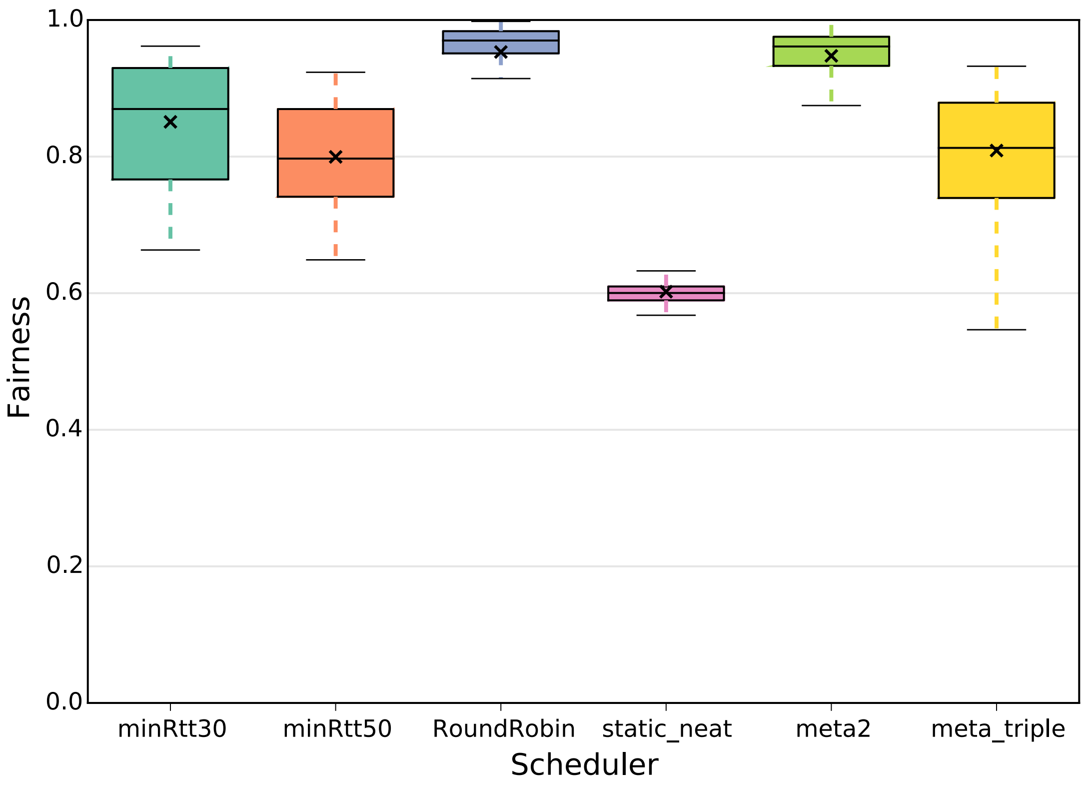
# Evaluation – Double Dumbbell (2) – Results



# Evaluation – Double Dumbbell (3) – Results



# Evaluation – Double Dumbbell (3) – Results