
# DSim

## A Distributed Message-Passing Mobsim Implementation

Janek Laudan, Christian Rakow, Marcel Rieser · Simunto GmbH
Steffen Axer, Hannes Rewald · Volkswagen AG

# Problem & Recap

# What is our problem?

MATSim modelers want to:

- Simulate larger model domains
- With greater detail

This leads to increased computational demands and long runtimes as:

- We can only run on a single machine
- The QSim's current parallelization architecture does not take full advantage of modern multicore CPU hardware

# What could be the solution?

We presented: *Distributed parallel Qsim* (with **Paul Heinrich** and **Kai Nagel**)

- A prototype implementation of a distributed mobsim in Rust
- MPI based distributed computation
- Aimed at High-Performance Computing Hardware
- Scalability up to 1000 CPU-Cores
- Open-Source Implementation

Published as *High-Performance Mobility Simulation: Implementation of a Parallel Distributed Message-Passing Algorithm for MATSim*[1]



[1] https://doi.org/10.3390/info16020116

# Why DSim?

# Why DSim?

Implement a message-passing mobility simulation in Java

1. Take advantage of modern multicore CPUs
2. Stay compatible with the existing MATSim-Codebase
3. Support distributed execution of MATSim-Simulations

# Why DSim – General Idea

Take the main ideas from Rust-Prototype

- Adopt a message-passing architecture for the mobsim
- Distribute simulation work by partitioning the simulated network

While staying compatible to the ecosystem

- The simulation must execute in the JVM
- Current Infrastructure should be compatible
- The new mobsim implementation should be usable as a drop-in replacement

# Architecture

# Architecture

Distribute Workload geographically

Domain Decomposition is a well-understood problem with plenty of algorithms, e.g., METIS

Estimate computational load for each vertice of the graph by estimating #vehicles crossing the node

METIS balances the computational load across partitions

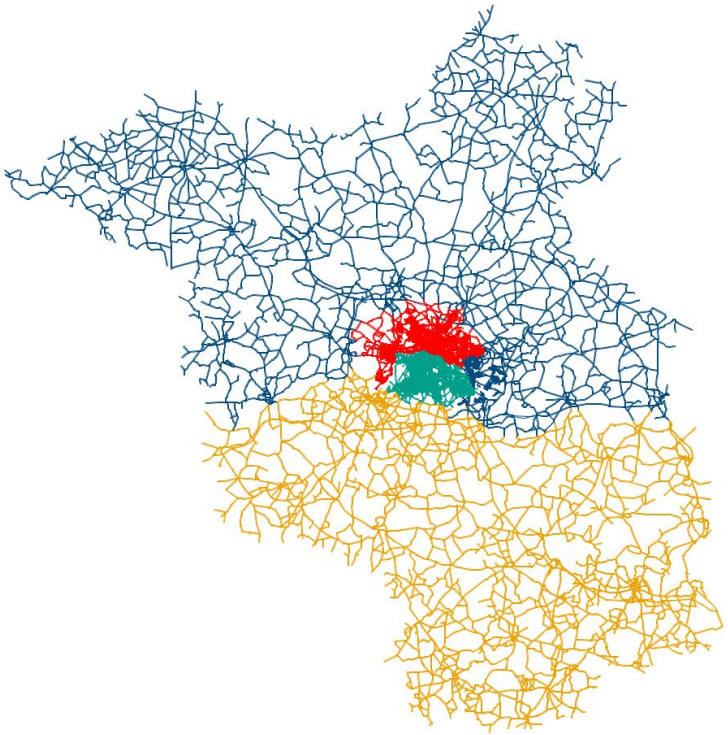
# Architecture

Distribute Workload geographically

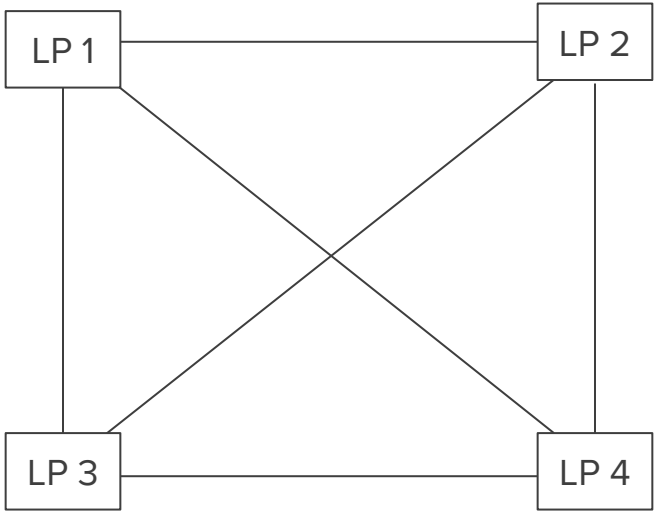
Domain Decomposition is a well-understood problem with plenty of algorithms, e.g., METIS

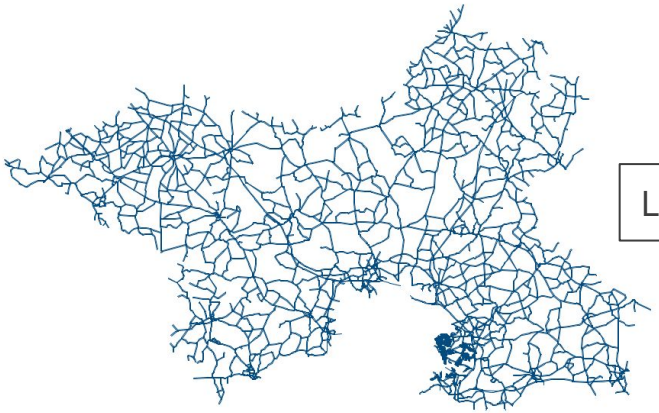
Estimate computational load for each vertice of the graph by estimating #vehicles crossing the node

METIS balances the computational load across partitions



# Architecture

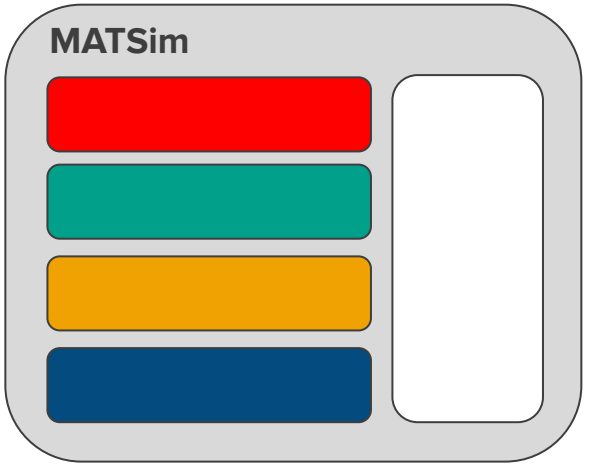


# Architecture

One MATSim process is started per JVM

- Multiple simulation processes
- Multiple events handlers
- One Communicator
- One Events Manager
- One Task Scheduler

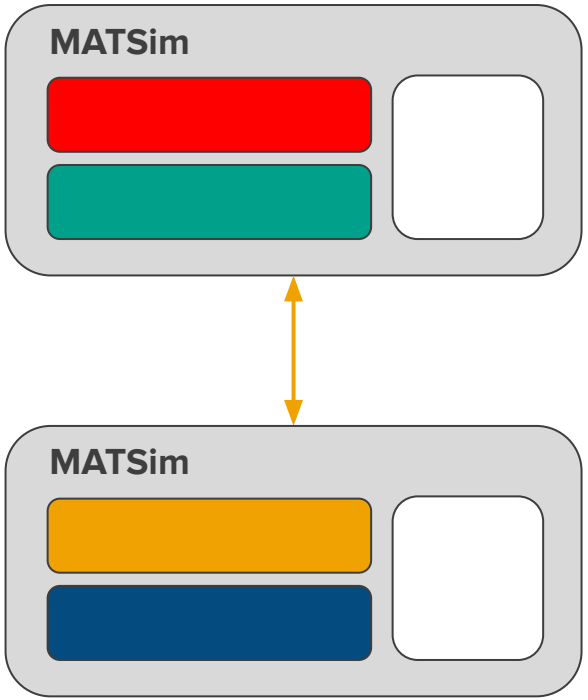


# Architecture

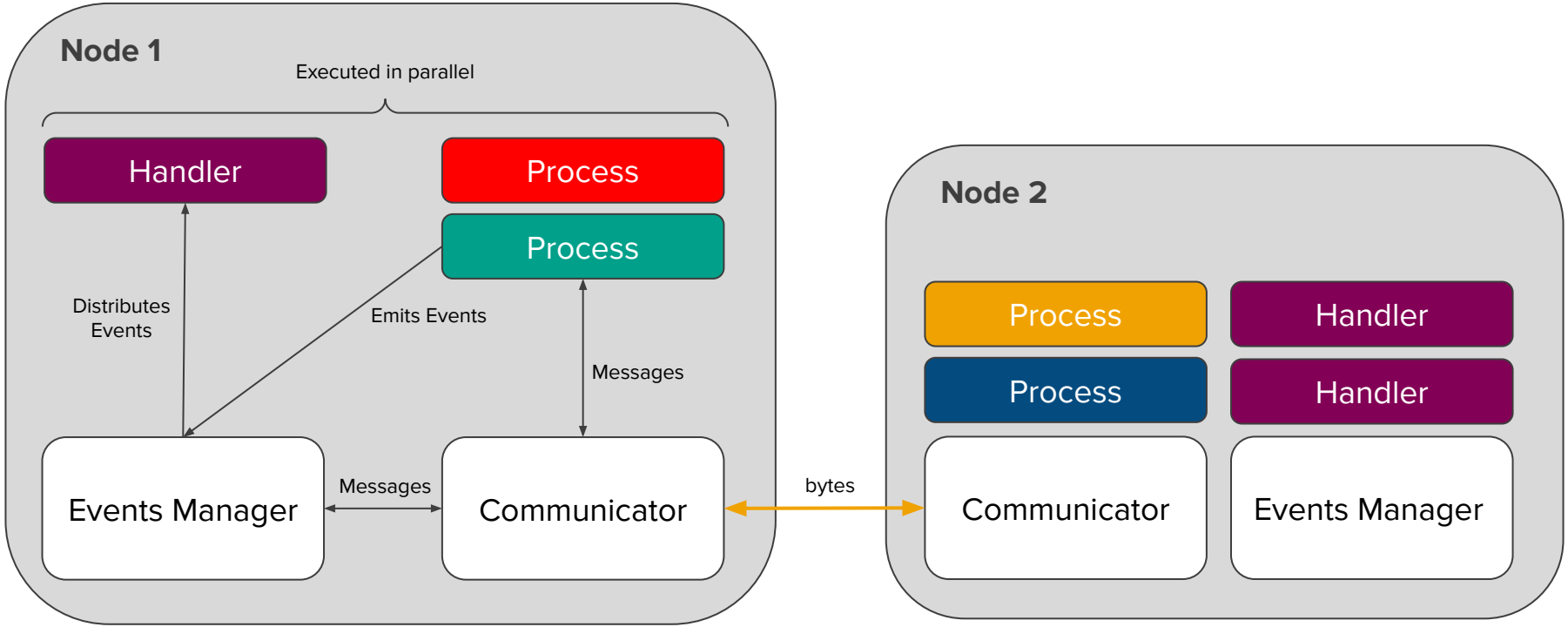
One MATSim process is started per JVM

- Multiple simulation processes
- Multiple events handlers
- One Communicator
- One Events Manager
- One Task Scheduler

Multiple processes communicate via the network layer



# Architecture



# Architecture – Gotchas

Unified Parallelization Strategy

- Mobsim and event handling are part of the same executor
- Parallelization on the ‘outer’ loop
- Simulation process is single threaded
- Event Handlers are single threaded
- Well defined synchronization via messages

Possibility for distributed traffic simulation

Possibility for distributed event processing

# Results

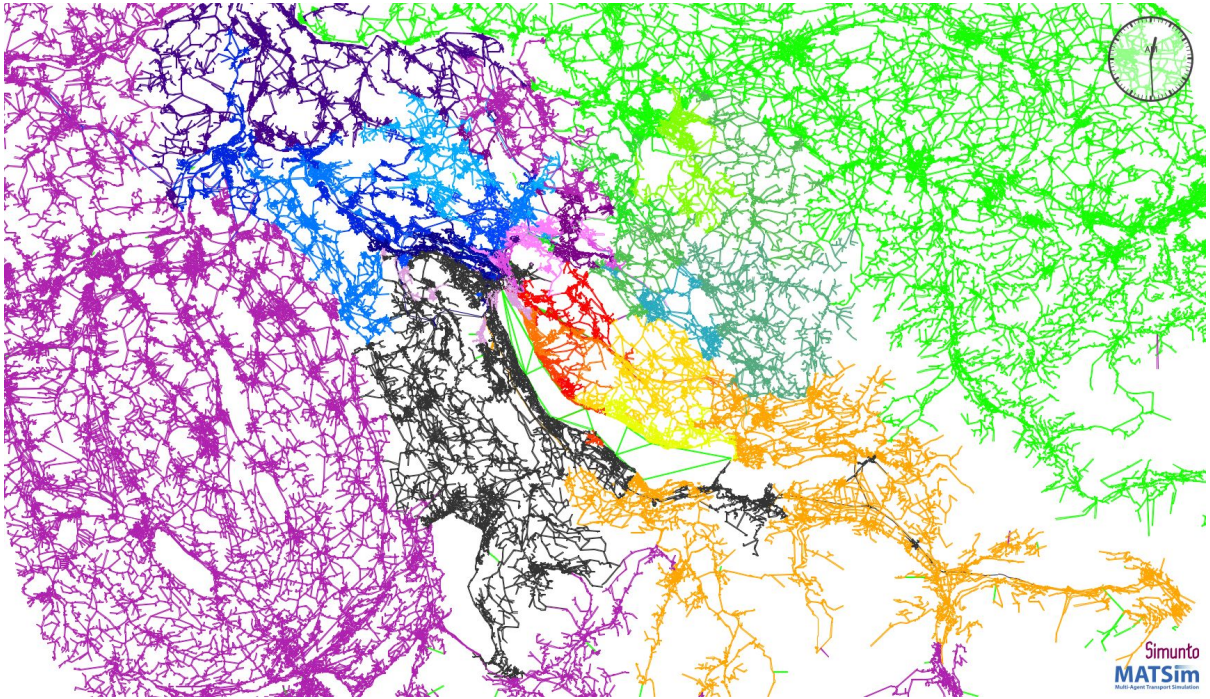
# Benchmark Scenario

Kanton Zürich with 1.6 million synthetic persons

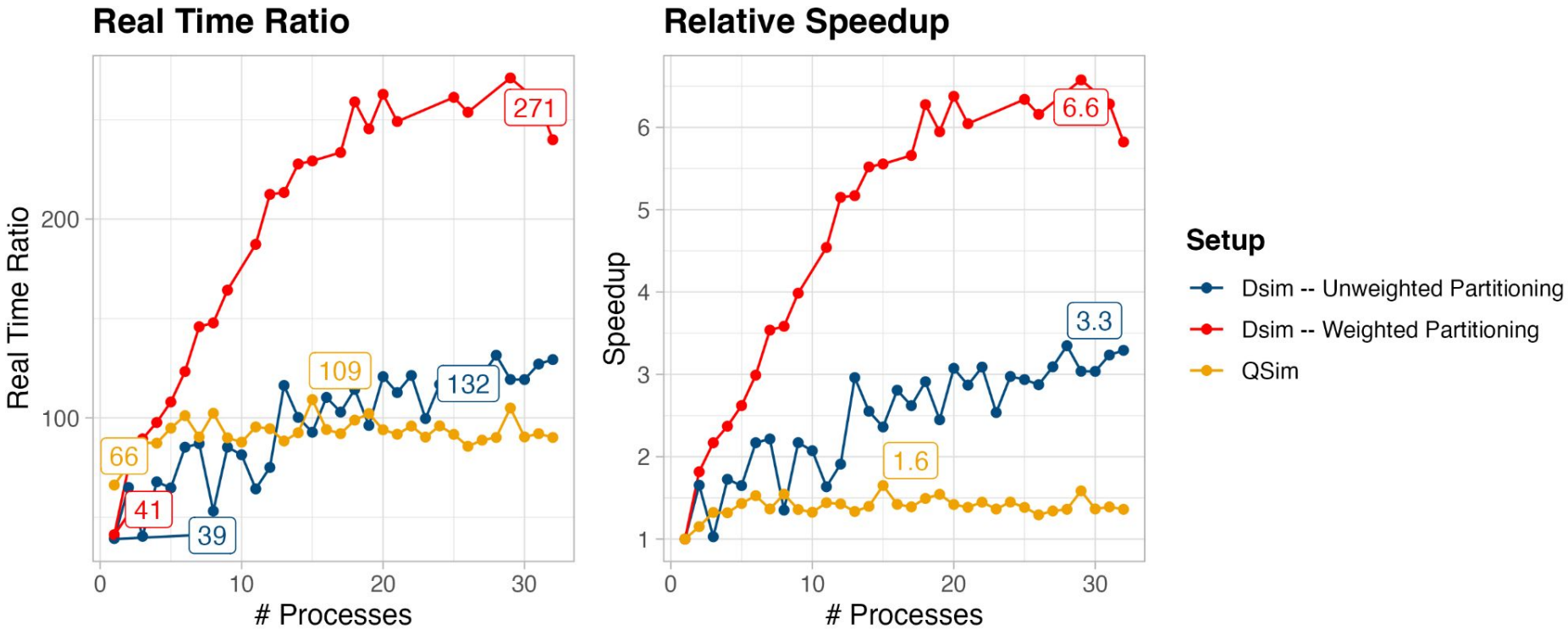
Demand Generation with Creario

Partitioning with METIS based on expected node weights

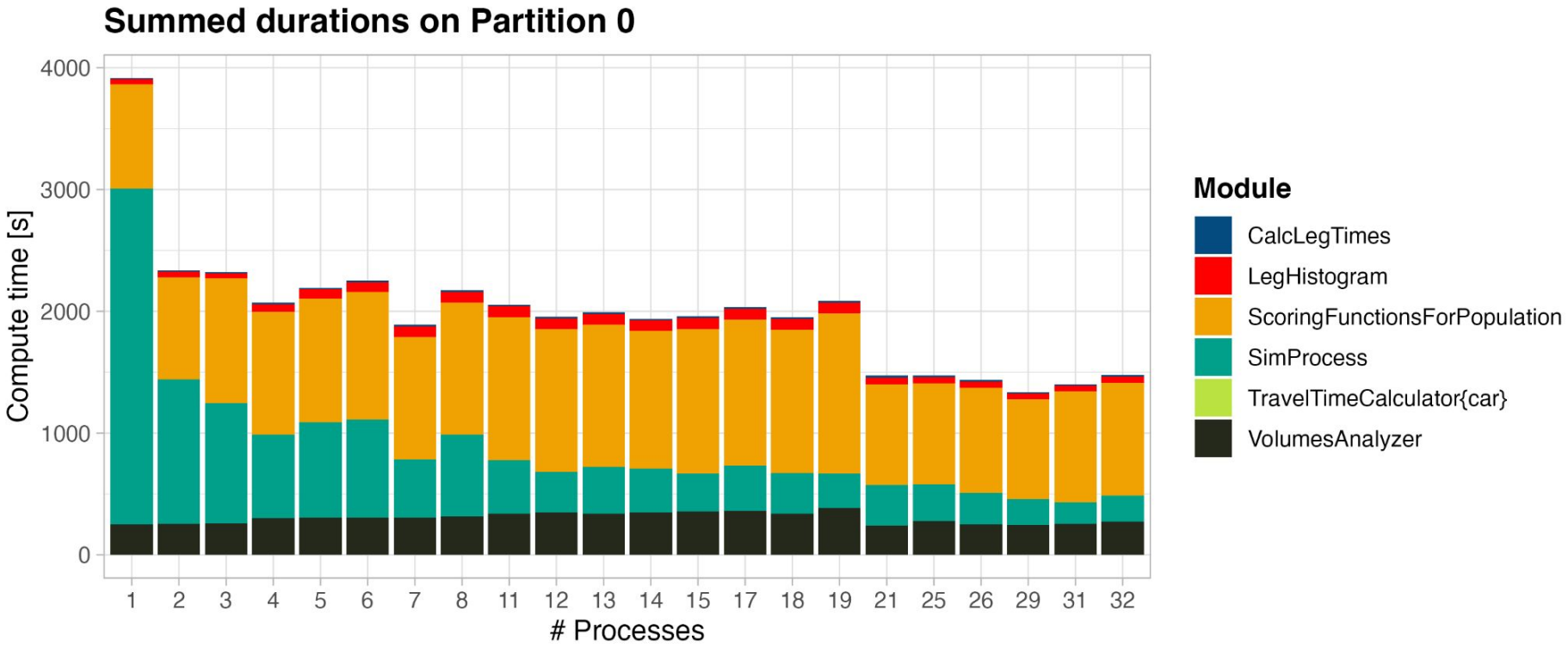
Test Environment: High-Performance Cluster with AMD EPYC™ 7313 (2x16 cores) and 240GB RAM



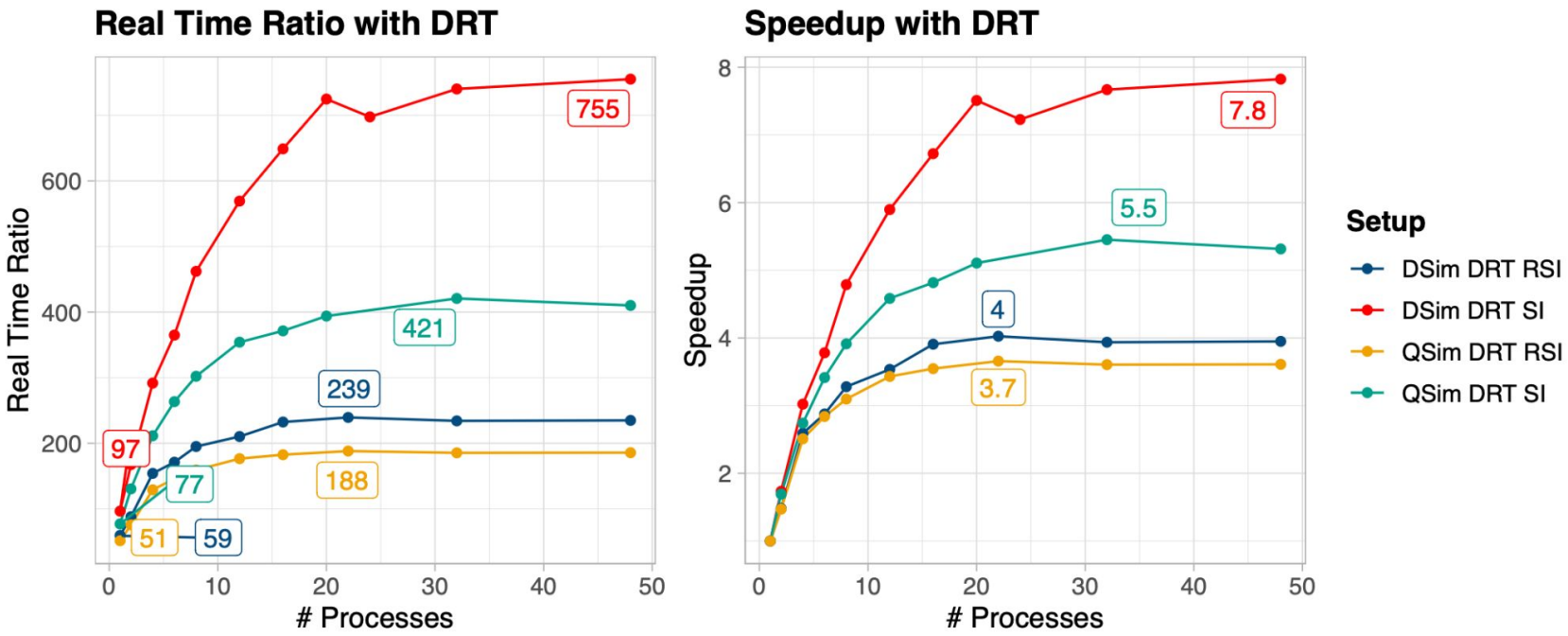
# Benchmark Kanton Zürich



# Benchmark Kanton Zürich



# Benchmark DRT (Berlin Scenrio)



# Conclusion

1. Architecture for distributed computing in MATSim implemented
2. Improved scalability compared to QSim
3. Better reasoning about bottlenecks
4. Truly distributed execution needs further testing and improved stability

# Outlook

Help and Support appreciated

Adapt more parts of MATSim to take advantage of new architecture

- Event handling must be switched to a distributed approach
- Develop strategy for services such as DRT

DSim will be merged into the main repository in the upcoming weeks

# Closing

Janek Laudan
Software Consultant @ Simunto
laudan@simunto.com
https://simunto.com

Simunto GmbH
Riedgrabenweg 49
8050 Zürich
Schweiz
mail@simunto.com

Thanks to Steffen Axer's group at Volkswagen for supporting this work financially!