Erlang-based Desynchronized Urban Traffic Simulation for High-Performance Computing Systems

Wojciech Turek

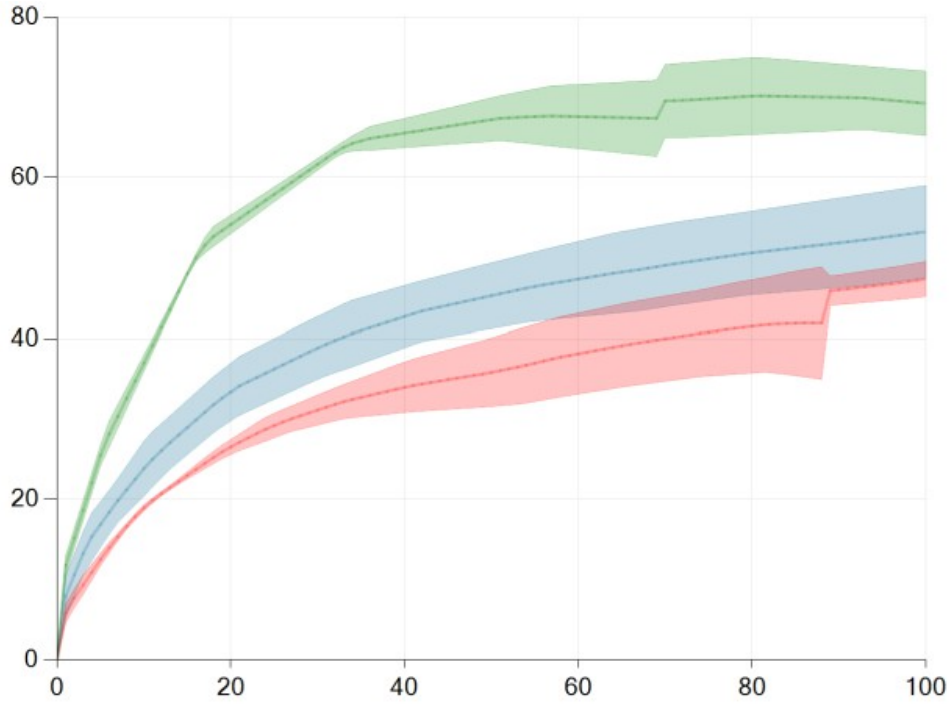
AGH University of Science and Technology
Krakow, Poland

AGH

- Micro-scale Traffic Simulation
- Parallel implementation
- Synchronization
- HPC system
- Results

Micro-scale Traffic Simulation

- Different abstraction levels:
  - Macro
  - Meso
  - Micro
- Individual cars make difference!
- More details
- Large cities
- Results ASAP



Micro-scale Traffic Simulation

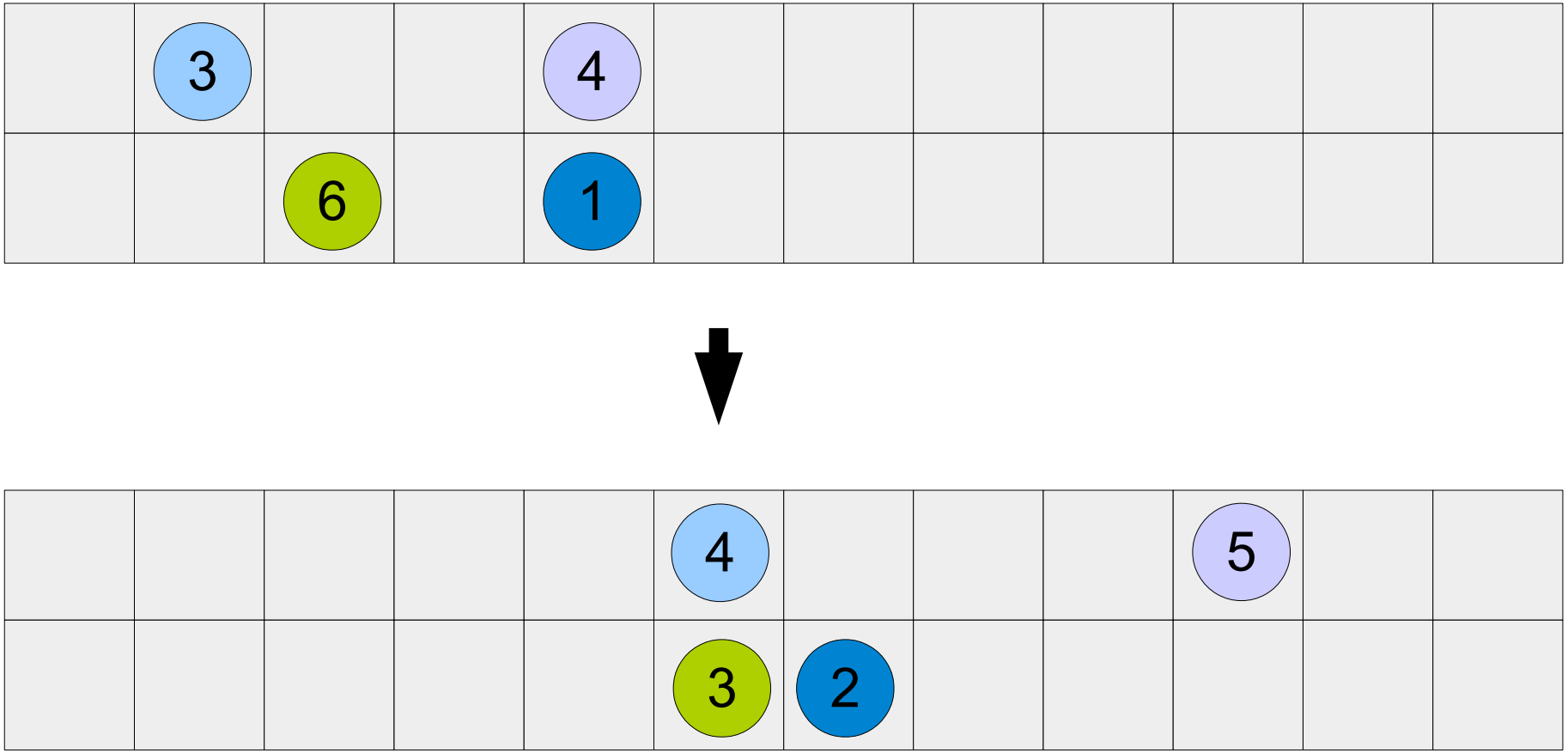
AGH

INTELLIGENT INFORMATION SYSTEMS GROUP

- More details
- Large cities
- Results ASAP

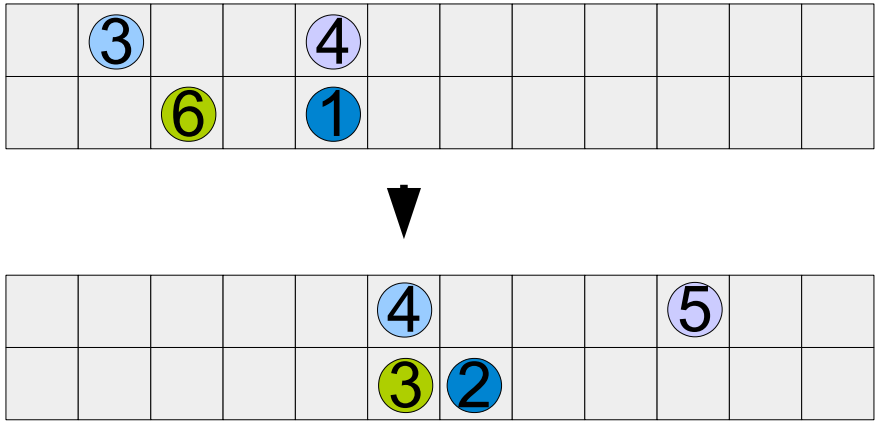
HPC

Nagel-Schreckenberg

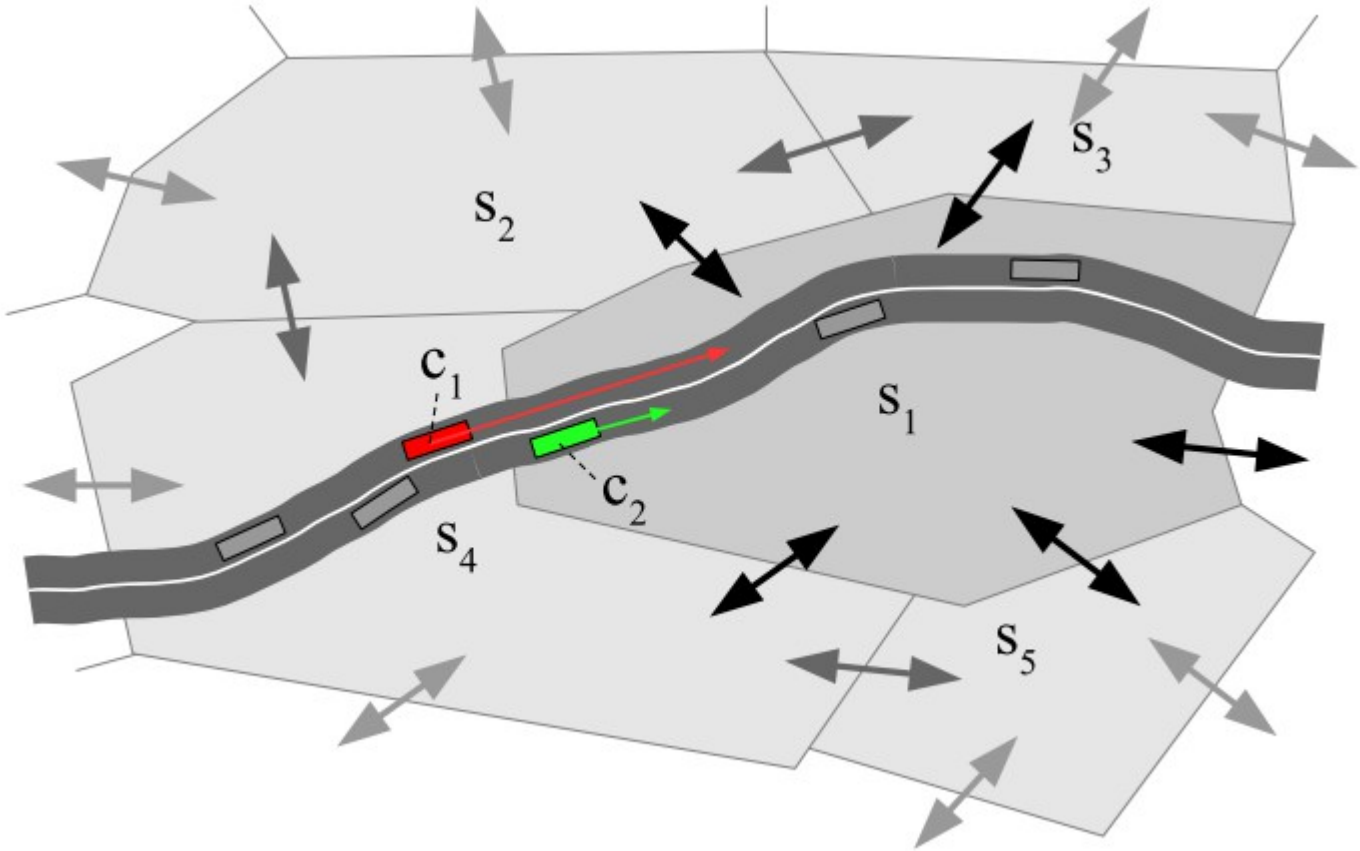


Nagel-Schreckenberg

- Extensions:
  - Lane changes
  - Crossroads
  - Traffic lights
  - Individual features

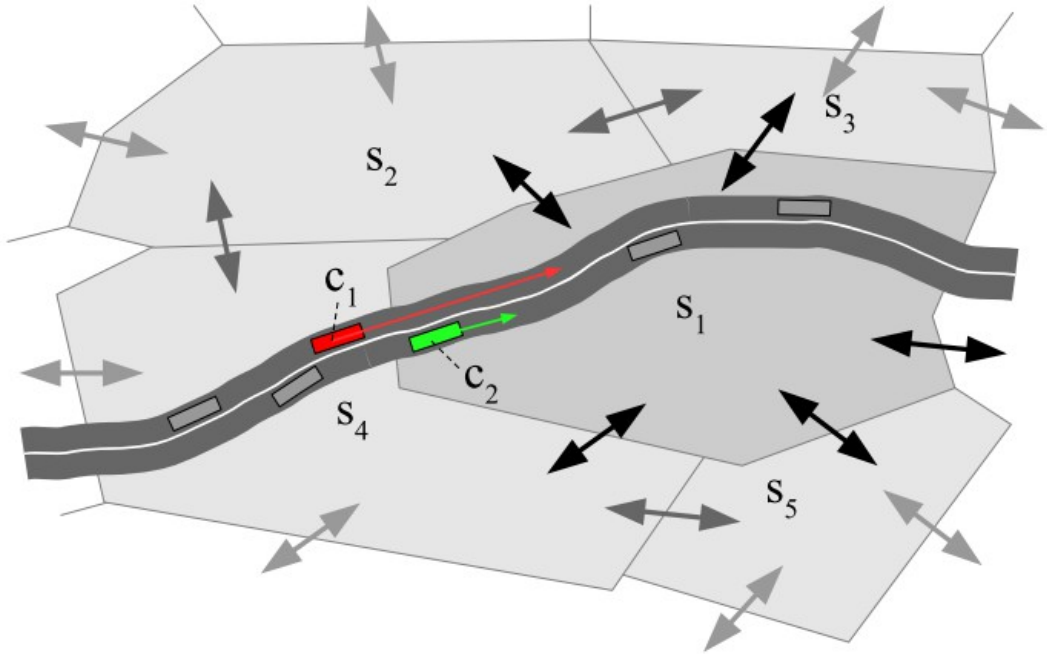


Parallel processing



Parallel processing

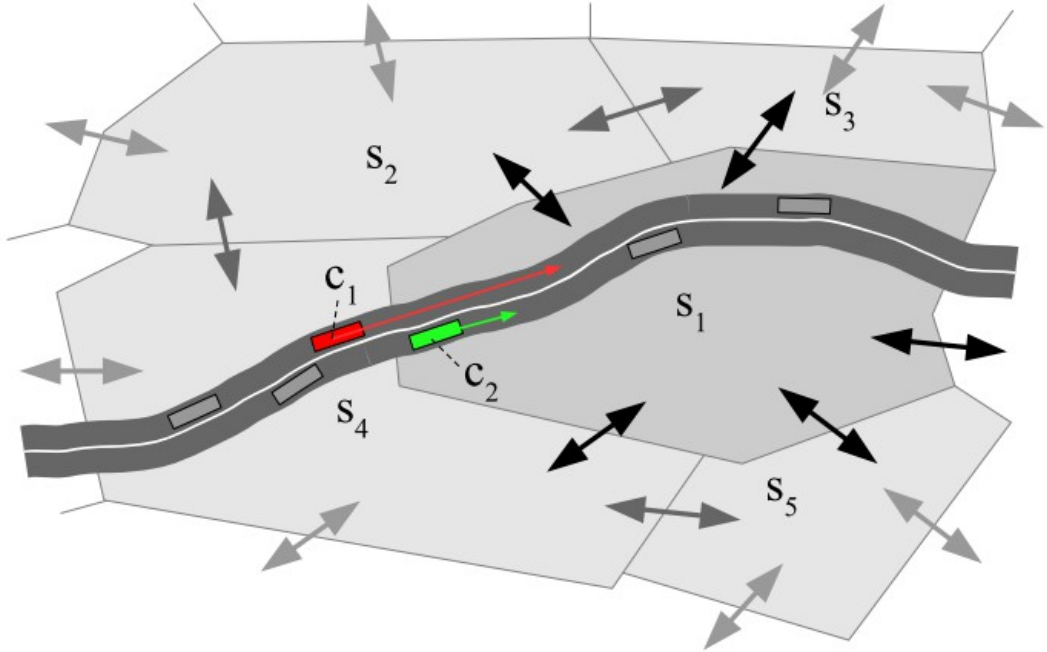
$$STATE_{t_k}^{s_i} = sim(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_k}^{s_{N_i}})$$



Parallel processing

$$STATE_{t_k}^{s_i} = sim(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_k}^{s_{N_i}})$$

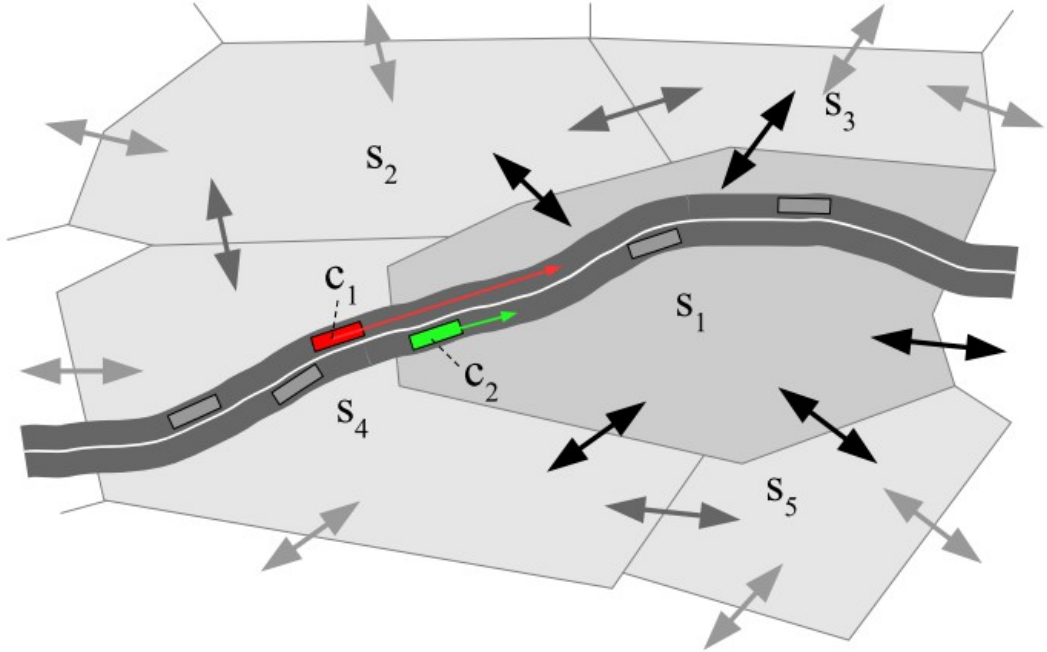
Deadlock!



Parallel processing

$$STATE_{t_k}^{s_i} = sim(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_k}^{s_{N_i}})$$

$$STATE_{t_k}^{s_i} = sim_1(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_{k-\Delta t}}^{s_{N_i}})$$



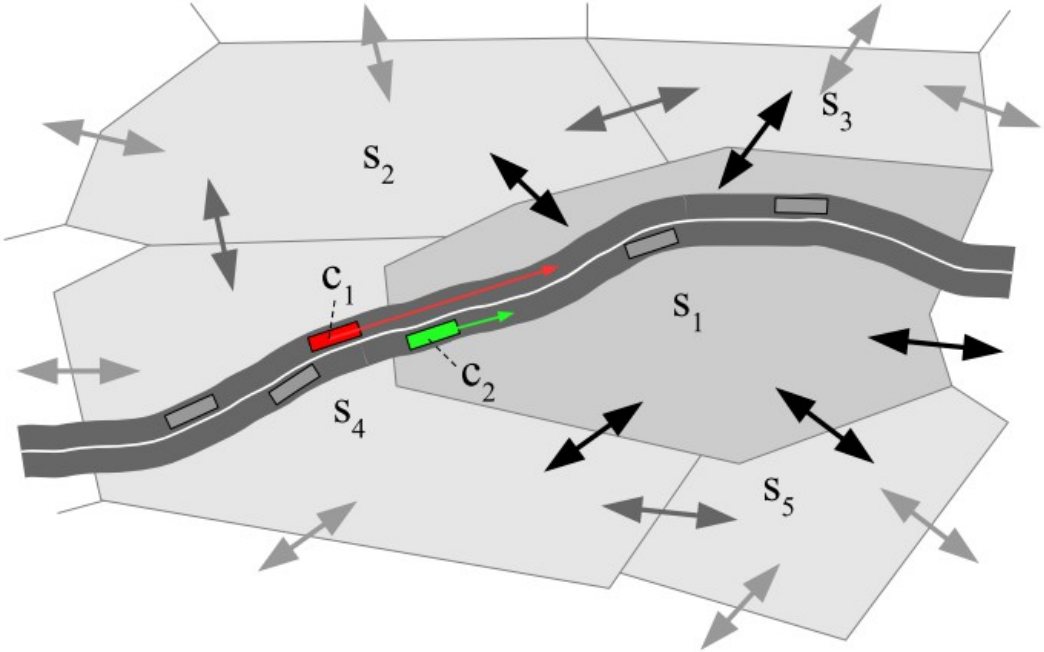
Parallel processing

$$STATE_{t_k}^{s_i} = sim(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_k}^{s_{N_i}})$$

$$STATE_{t_k}^{s_i} = sim_1(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_{k-\Delta t}}^{s_{N_i}})$$

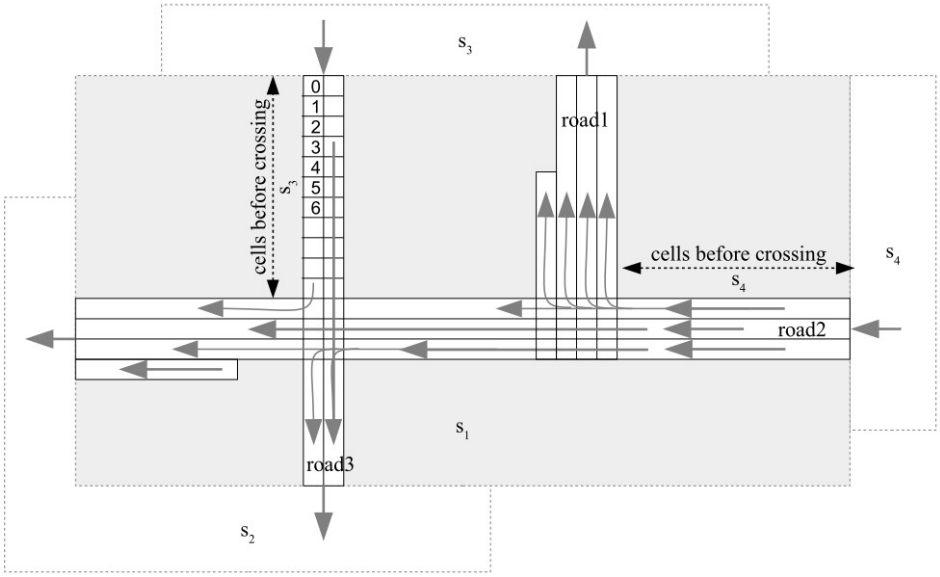
$$STATE_{t_k}^{s_i} = sim(STATE_{t_{k-\Delta t}}^{s_i}, STATE_{t_{k-d\Delta t}}^{s_{N_i}})$$

***d* – desynchronization level**



Desynchronization

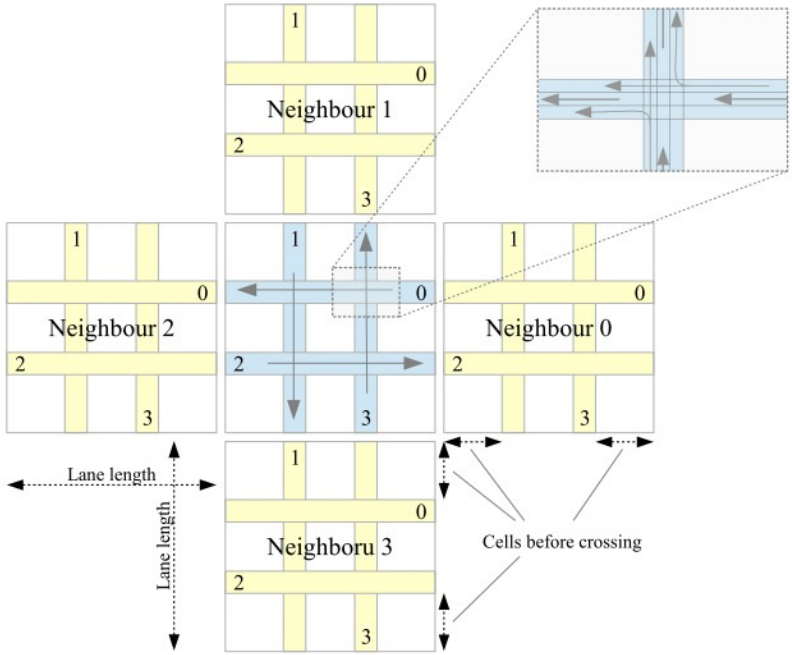
- Calculate state changes without full knowledge about incoming cars
- Correct the state after the information appears
- Limit *d* – till first crossroad
- Publish available space
- Forbid lane changes at *d*Vmax*



Implementation in Erlang

- *Map* as a basic data structure, Erlang 18
- Each crossroad in a separate process
- Messaging: publish available space and leaving cars
- Calculate *d* steps forward

- Distributed Erlang
- Hidden nodes only!

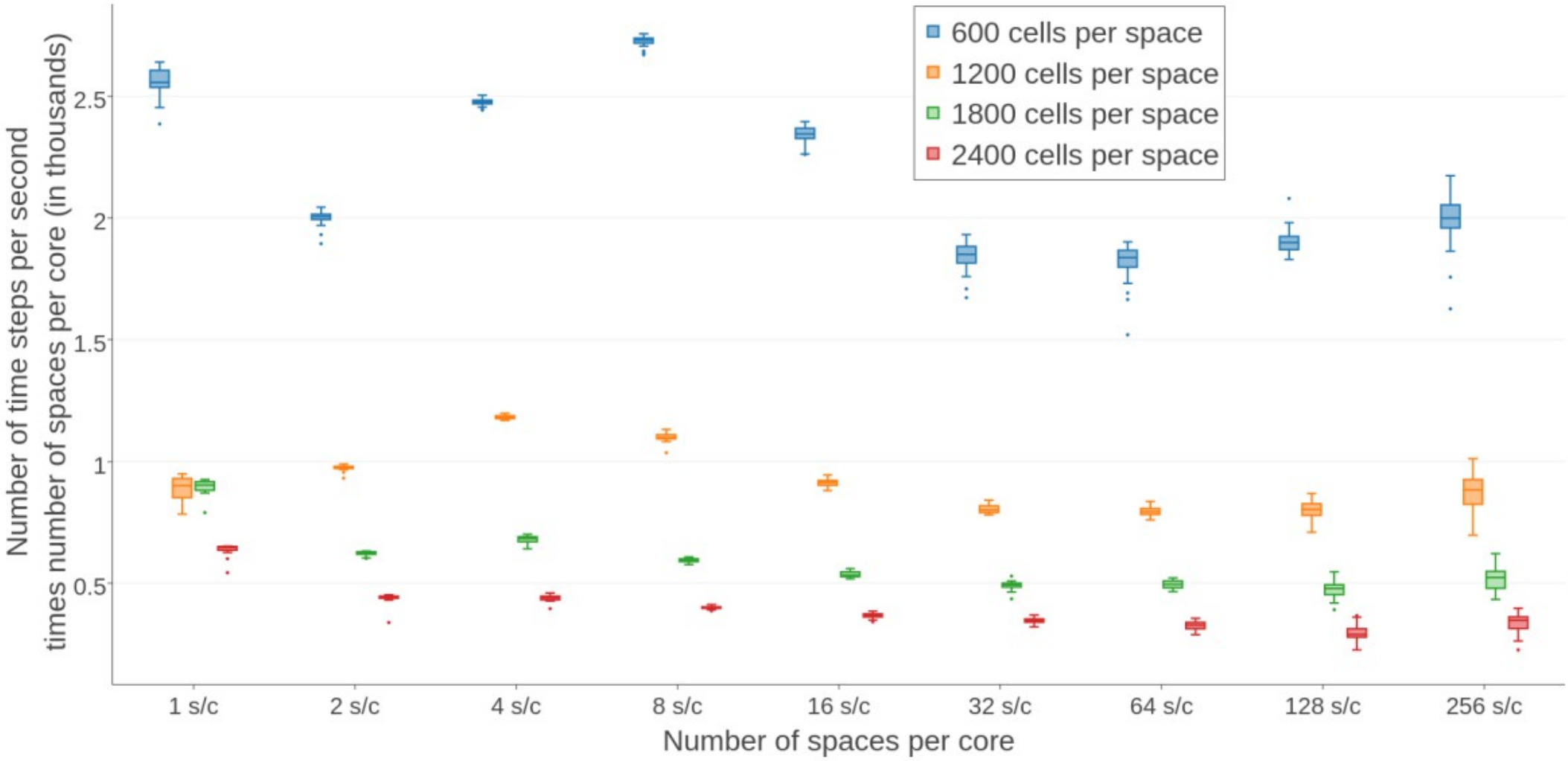


The computer

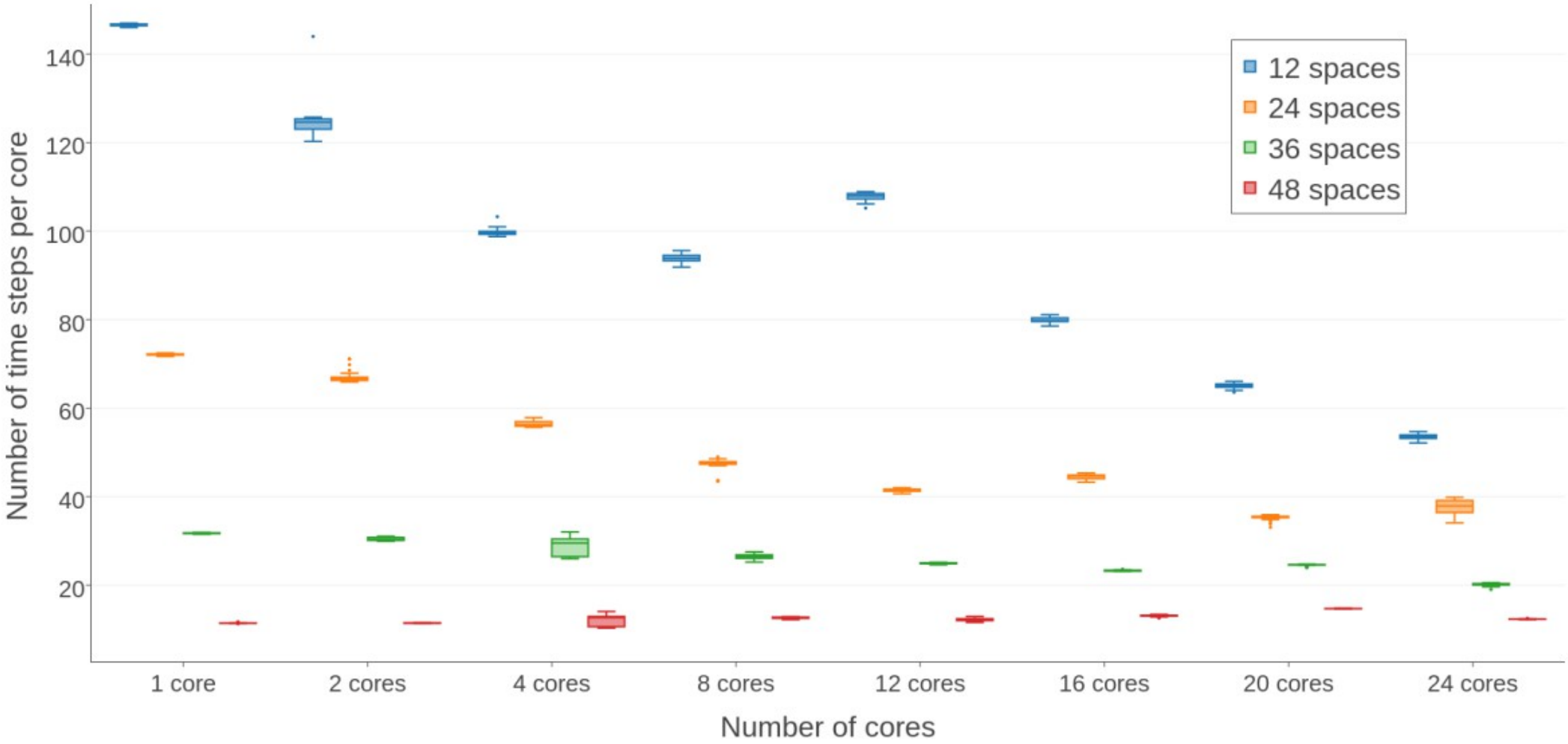
- Prometheus @ ACK Cyfronet
- 1728 servers of the HP Apollo 8000 platform
- InfiniBand network with 56 Gbit/s
- 41,472 cores
- 216 TB RAM

- You can visit him tomorrow

Single node – growing task size



Single node – more cores

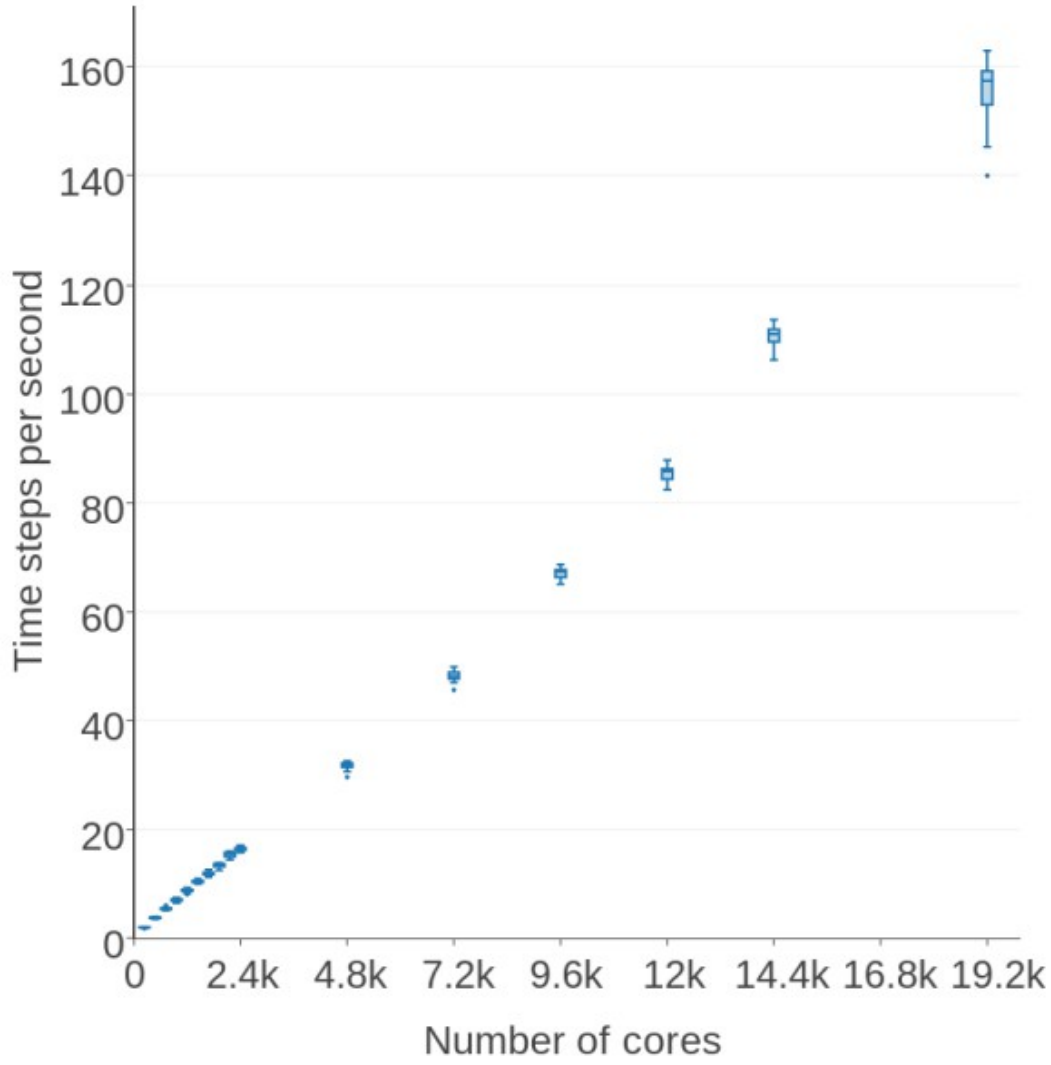


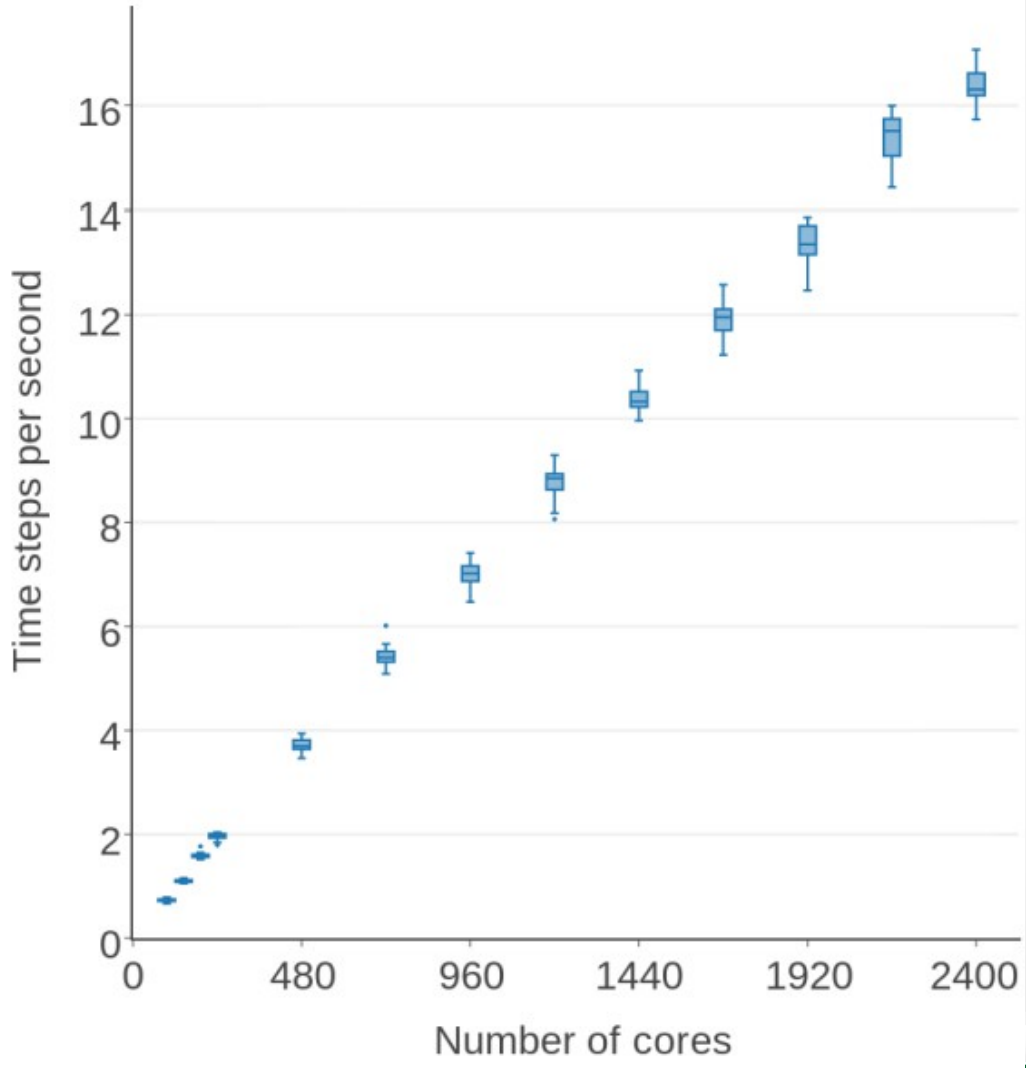
More and more cores

AGH

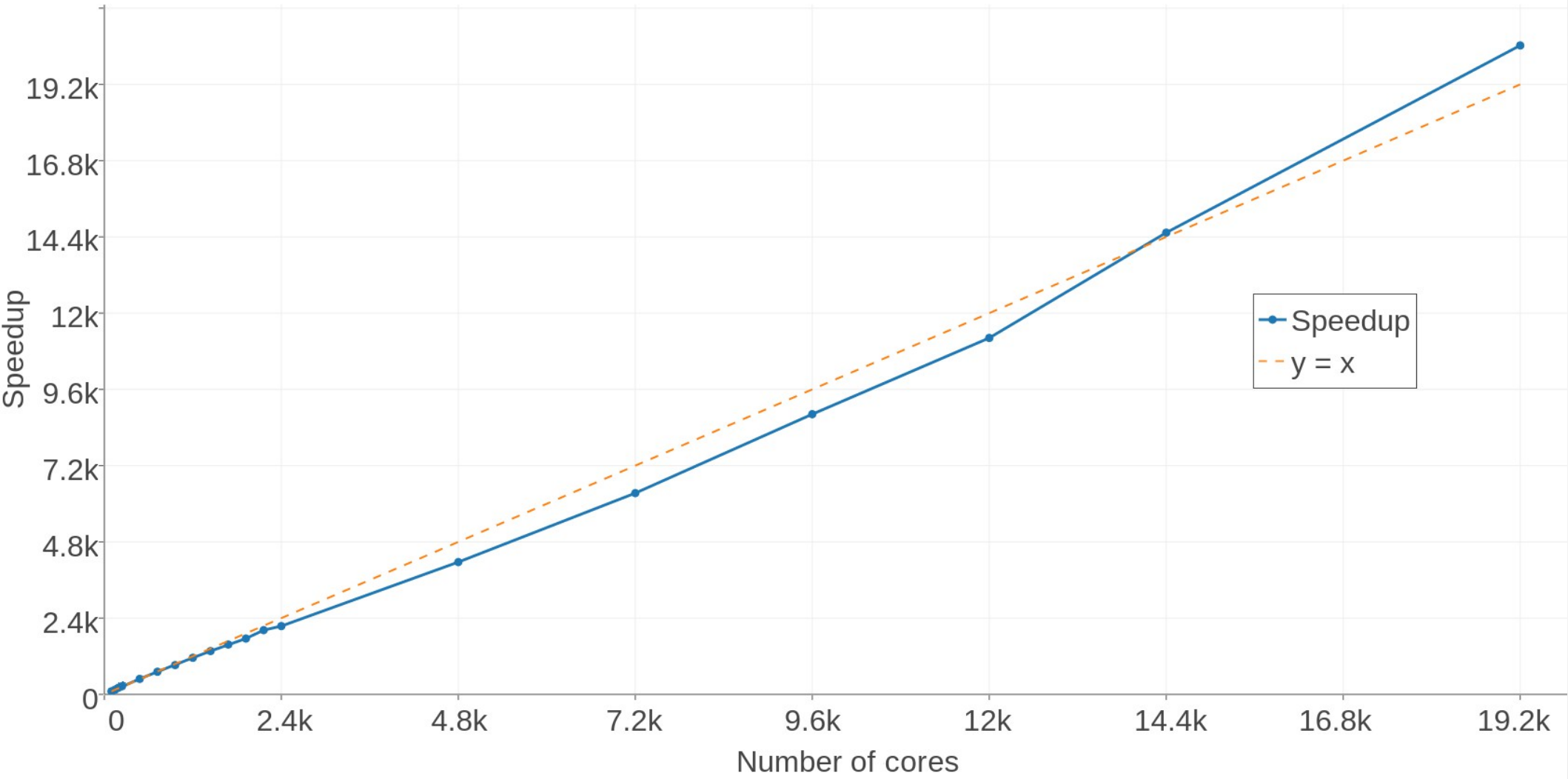
INTELLIGENT INFORMATION SYSTEMS GROUP

240,000 spaces, 600 cells each, 11,520,000 cars

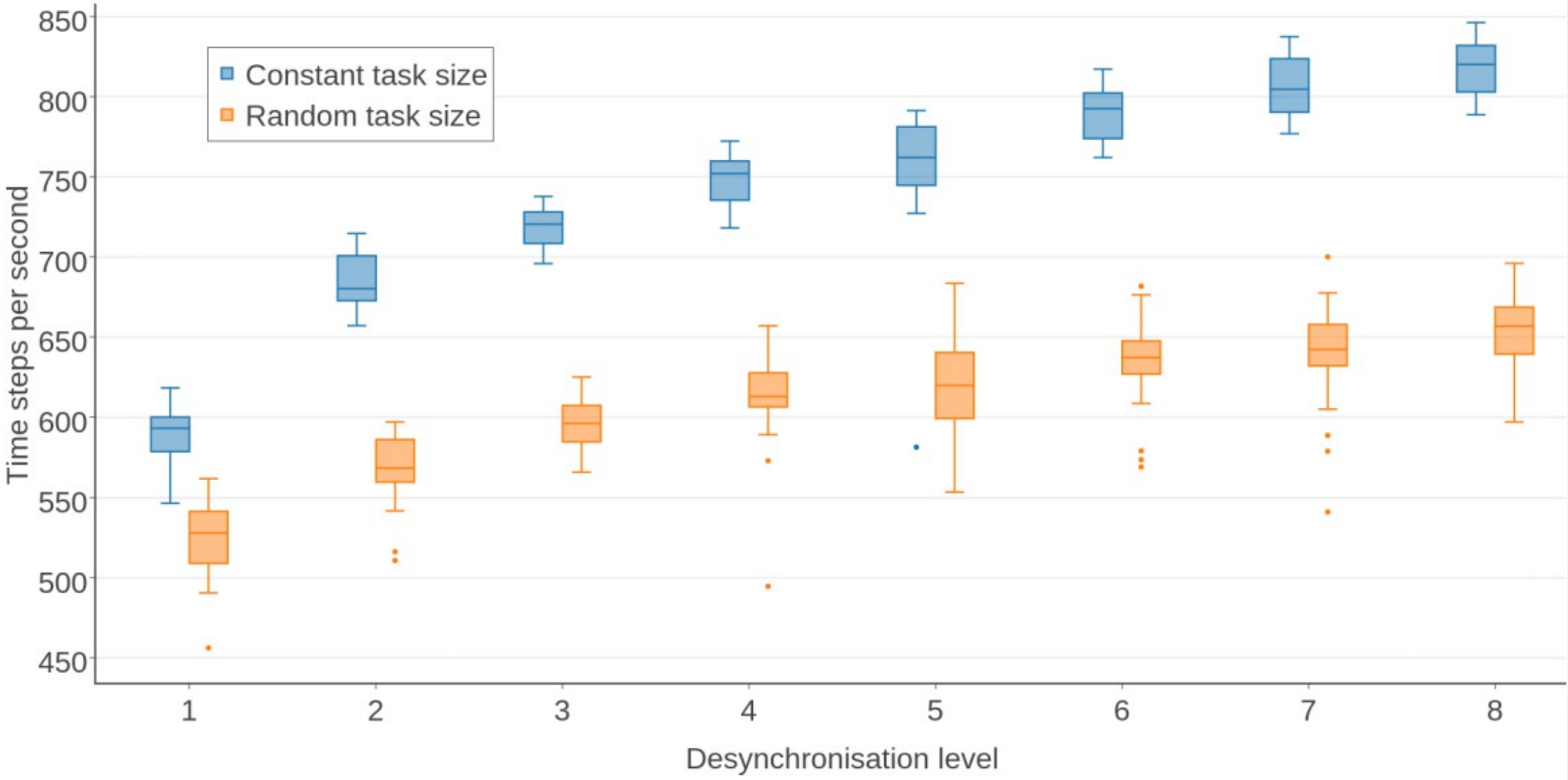




Speedup


Speedup
19.2k
16.8k
14.4k
12k
9.6k
7.2k
4.8k
2.4k
0
0
2.4k
4.8k
7.2k
9.6k
12k
14.4k
16.8k
19.2k
Number of cores
Speedup
y = x

Desynchronization



Further steps

- Compare with sequential version
- Compare with Java, C...
- Real-life scenarios
- More precise models
- Different problems

Thank you!