Running Snails (GEMSim)

Aleksandr Saprykin
Marcello Marini
Christiaan Joubert

Mentors: Thibault Notargiacomo, Jeffrey Kelling

Agent-based Modelling

Agent-based modelling is a modelling approach to simulate actions of autonomous agents to assess their effects on the transportation network.

For each agent we provide a daily plan:

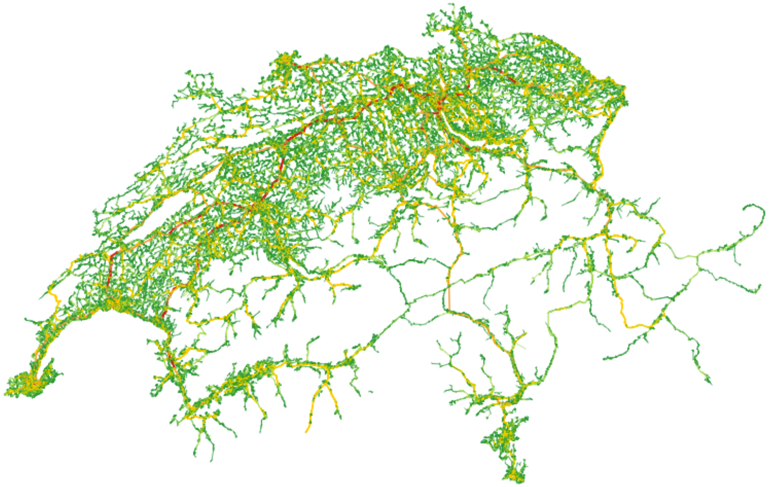
- Activities with place and time (i.e. home, work).
- Travelling mode (car, public transport) between activities.

GEMSim

- Part of EnerPol simulation framework, aimed to simulate energy, urban, and transportation infrastructure performance for future scenarios on continental scale

- GEMSim is a GPU-based mobility simulator for large-scale scenarios
  - Scenarios on large-scale networks with millions of agents
  - C++ code with use of CUDA toolkit for acceleration
  - Some code can run in parallel on CPU, some is only on GPU
  - Use Qt libraries for cross-platform development
  - No MPI nor multi-GPU support
  - Data structures: mostly SoA

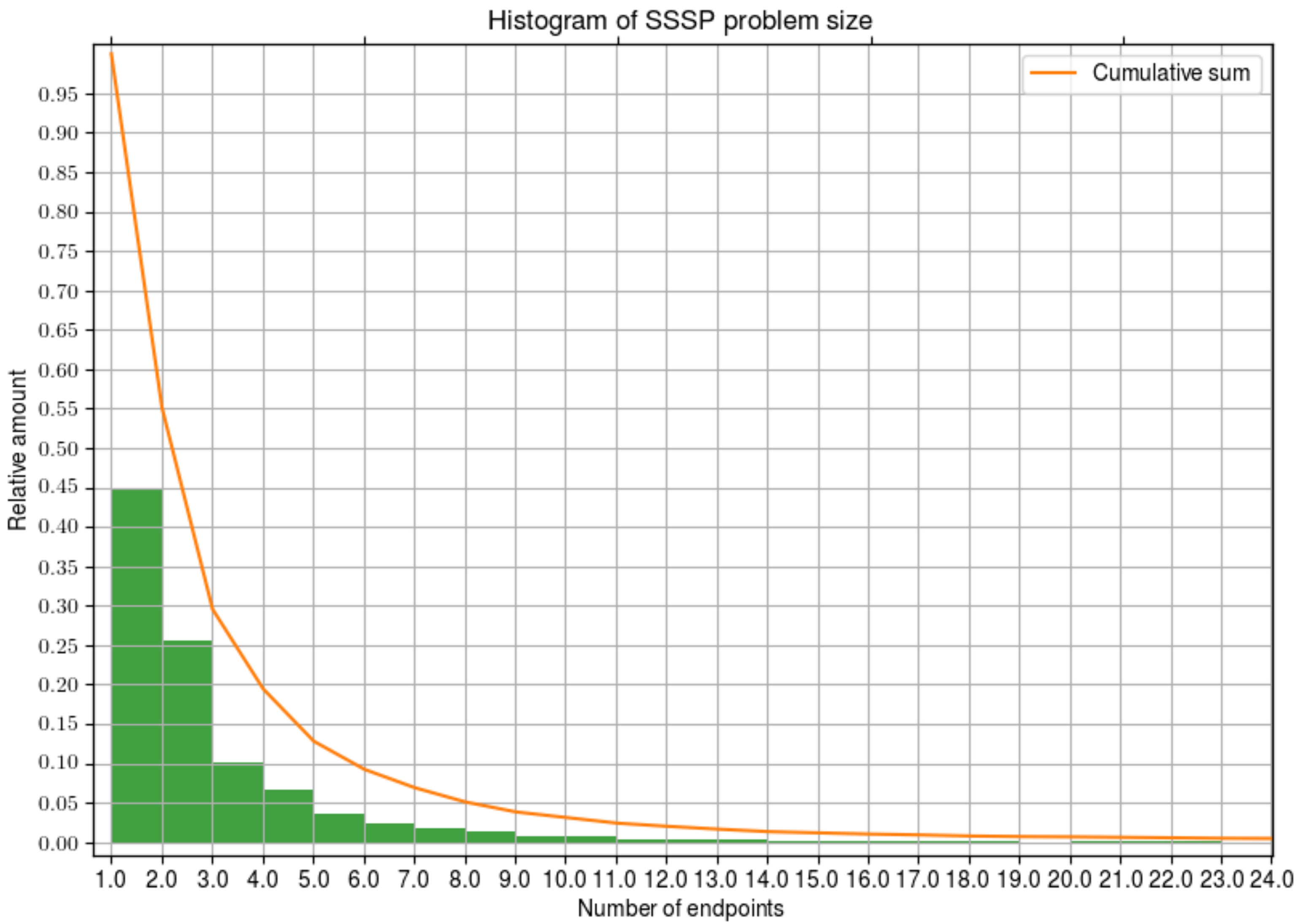
Mobility simulation from city-wide to country-wide



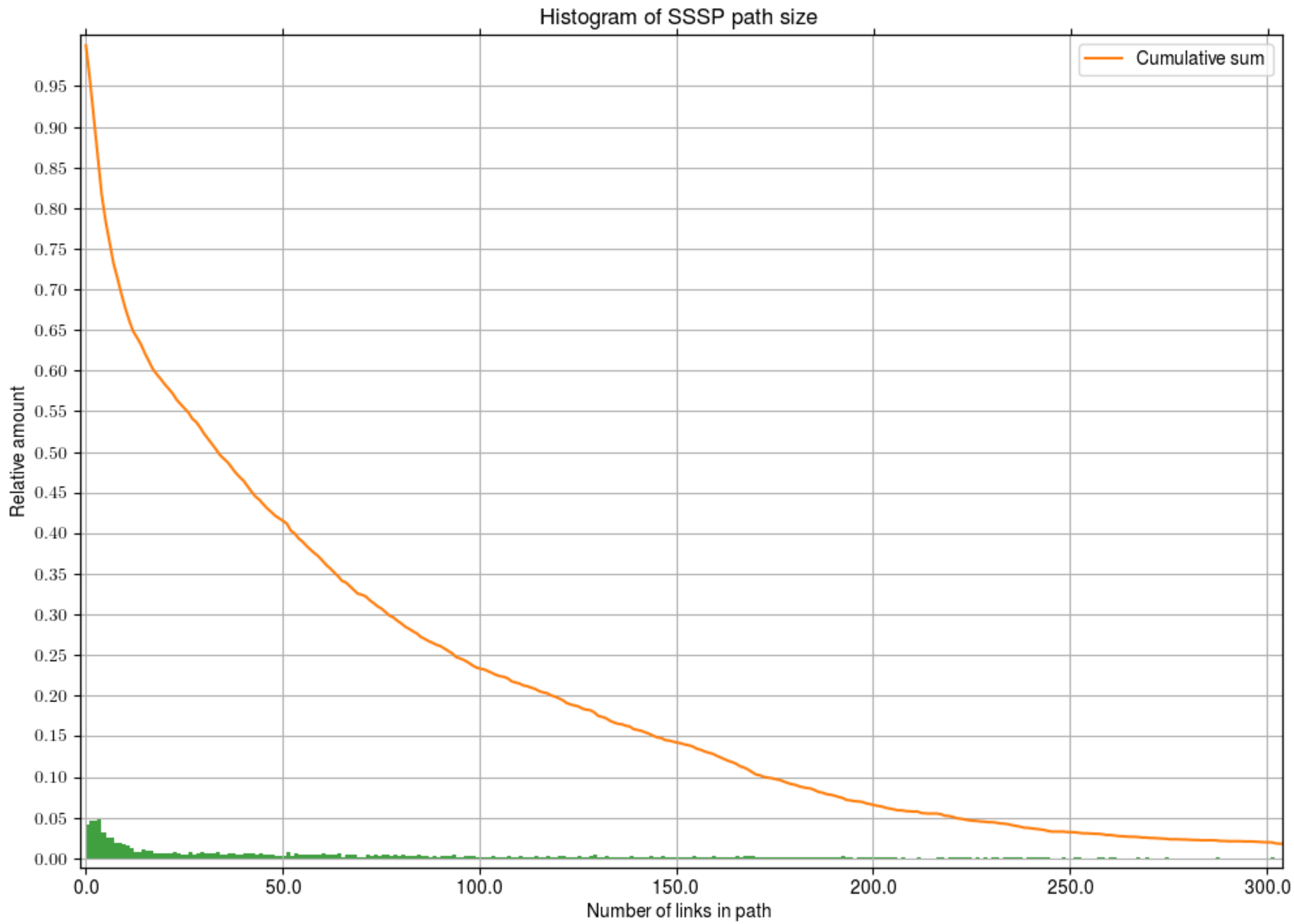
Initial Profile

- What parts where you focusing on?
  - Accelerating the initial routing solver (Dijkstra shortest path) = Single Source Shortest Path (SSSP) problem
  - Network: > 0.5 mil vertices, > 1 mil edges
  - 0.5 mil of agents
- What’s the algorithmic motif?
  - Graph exploration following a Breadth-first search pattern

Route Statistics for Population



Path Length Variability



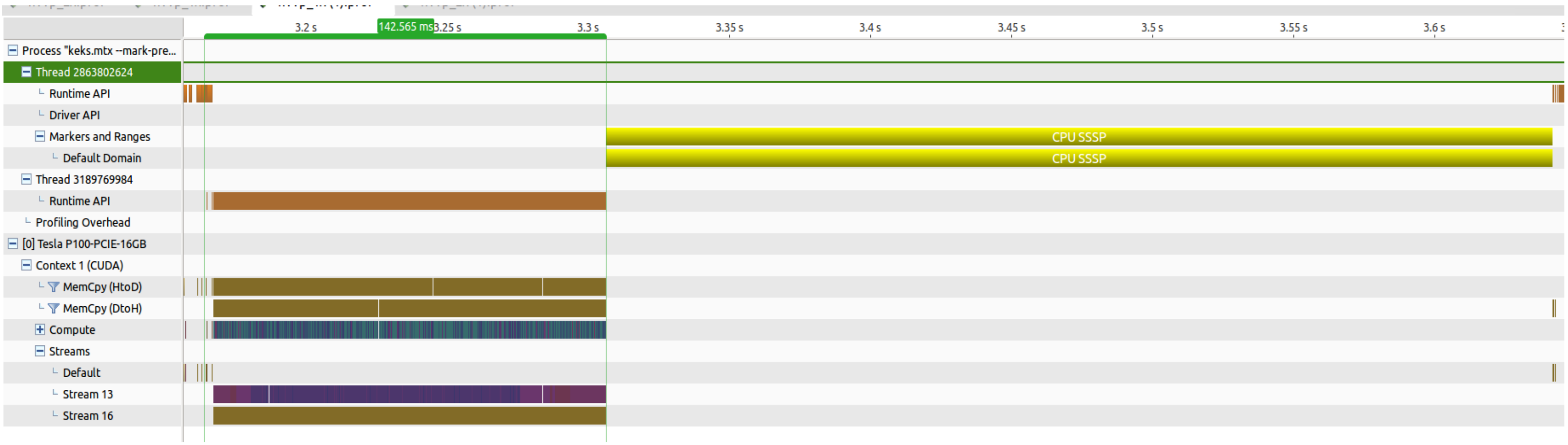
Evolution and Strategy

- Evaluate and bring graph routing algorithms on GPU, add support for MPI and multi-GPU
- Initial strategy: use existing libraries (nvGRAPH) for GPU acceleration
- At the end, more focus on MPI support

Results and Final Profile

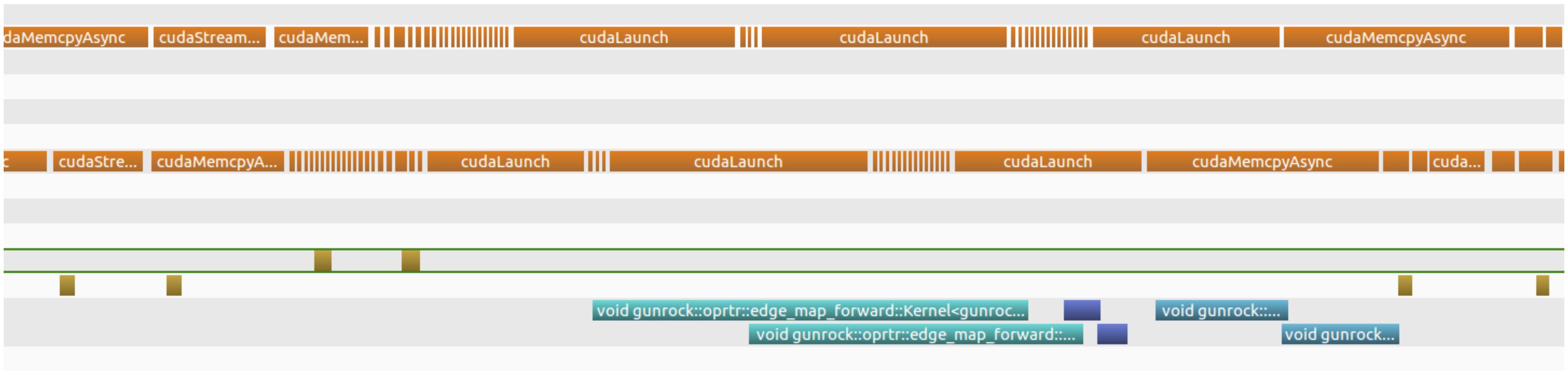
- What were you able to accomplish
  - Speed with Gunrock GPU-based library is ~2x comparing to a single CPU core
  - GPU is not well suited for Dijkstra algorithm on the given graph
  - MPI support provides speed up over single-node solution

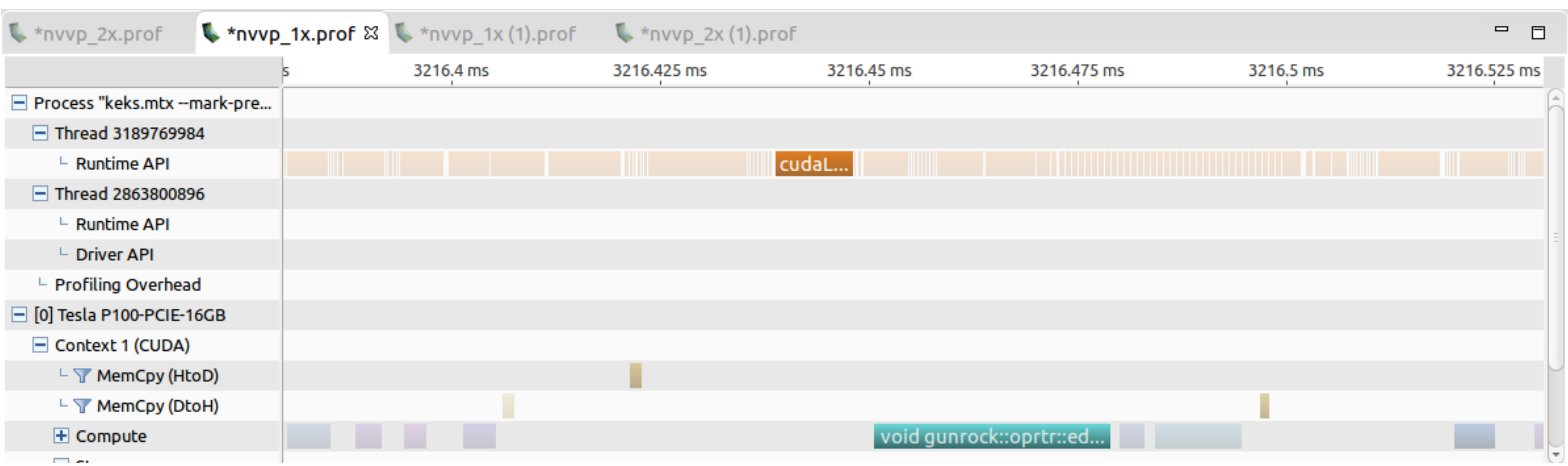
GPU vs CPU



3.2 s
142.565 ms
3.25 s
3.3 s
3.35 s
3.4 s
3.45 s
3.5 s
3.55 s
3.6 s
Process "keks.mtx --mark-pre...
Thread 2863802624
Runtime API
Driver API
Markers and Ranges
Default Domain
Thread 3189769984
Runtime API
Profiling Overhead
[0] Tesla P100-PCIE-16GB
Context 1 (CUDA)
MemCpy (HtoD)
MemCpy (DtoH)
Compute
Streams
Default
Stream 13
Stream 16
CPU SSSP
CPU SSSP

Gunrock Library Profiling (1)





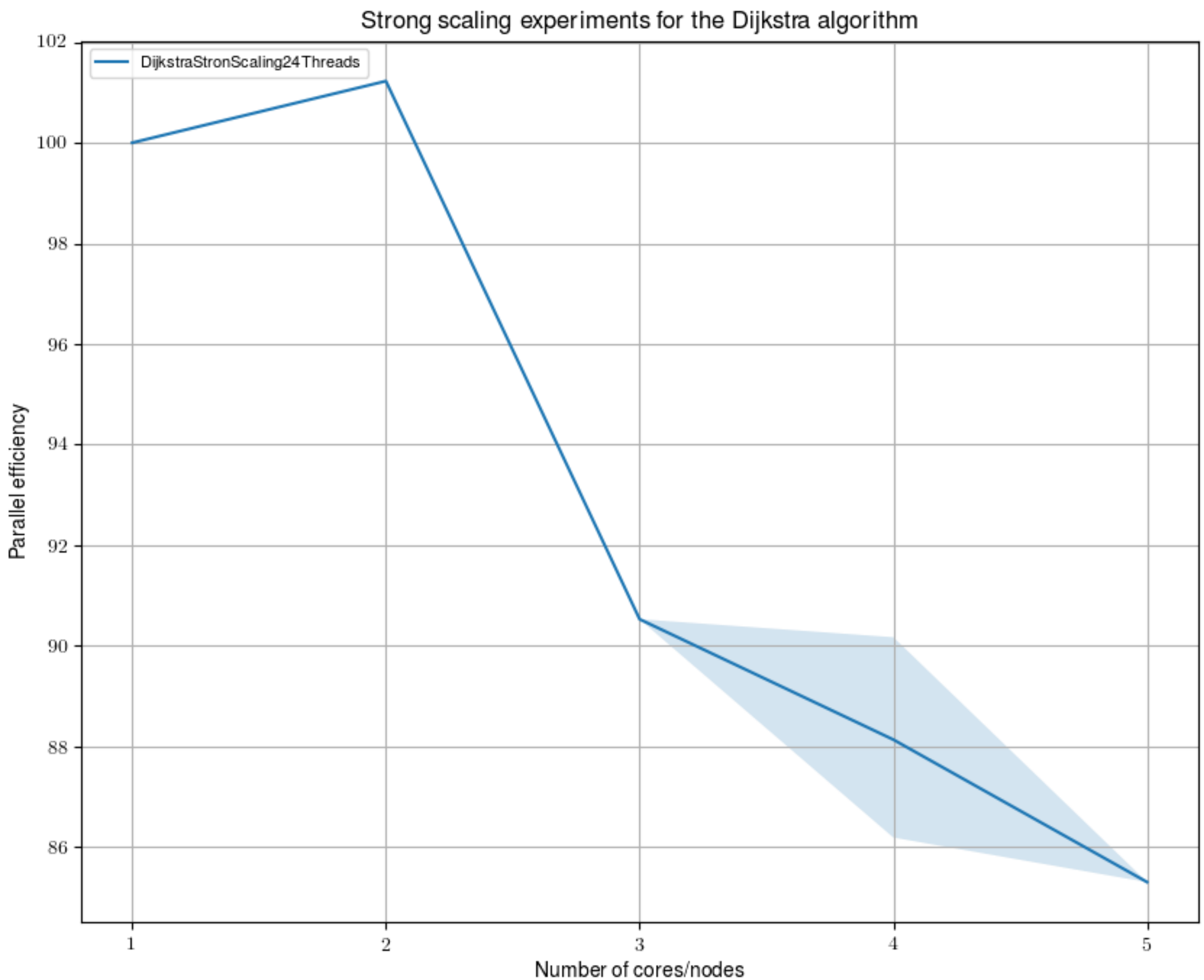
Gunrock Library Profiling (2)

- Compute
  - 46.2% void gunrock::oprtr:...
  - 21.2% void gunrock::oprtr:...
  - 9.1% void gunrock::oprtr::...
  - 6.0% void gunrock::app::A...
  - 5.9% void mgpu::KernelSc...
  - 4.7% void mgpu::KernelRe...
  - 4.3% void mgpu::KernelSc...
  - 1.1% void mgpu::KernelSc...
  - 0.9% void mgpu::KernelSc...
  - 0.6% void mgpu::KernelSc...
  - 0.0% void gunrock::util::M...
  - 0.0% void gunrock::util::M...
  - 0.0% void gunrock::util::M...
  - 0.0% void gunrock::util::M...

void gunrock::oprtr::edge_map_forward::Kernel<gunrock::oprtr::edge_

Start	3.216 s (3,216,450,592 ns)
End	3.216 s (3,216,478,912 ns)
Duration	28.32 μs
Stream	Stream 16
Grid Size	[448,1,1]
Block Size	[128,1,1]
Registers/Thread	50
Shared Memory/Block	5.875 KiB
▼ Occupancy	
Theoretical	56.2%
▼ Shared Memory Configuration	
Shared Memory Requested	64 KiB
Shared Memory Executed	64 KiB
Shared Memory Bank Size	4 B

MPI Strong Scaling



Wishlist

- What do you wish existed to make your life easier?
  - nvGRAPH library provided by CUDA is able to calculate distances between graph vertices, but doesn’t provide information to reconstruct the path.
  - nvGRAPH solves SSSP problem for the whole graph, probably we can speed up by stopping on target node.

Was it worth it?

- Was this worth it?
  - Absolutely YES!

- We will continue development of MPI part, as well as will look for further possibilities to utilize CUDA.