Performance Analysis of ~~Load Imbalance~~ in SPH-EXA Simulations

SKACH Winter Meeting
Basel, Switzerland
January 12, 2023

Florina M. Ciorba

Universität Basel

Florina Ciorba (PI) *florina.ciorba@unibas.ch*
Ruben Cabezon (Co-PI) *ruben.cabezon@unibas.ch*
Osman Seckin Simsek *osman.simsek@unibas.ch*
Ahmed Eleliemy *ahmed.eleliemy@unibas.ch*
Lukas Schmidt *luke.schmidt@unibas.ch*

Universität Zürich

Lucio Mayer (Co-PI) *lmayer@physik.uzh.ch*
Noah Kubli *noah.kubli@uzh.ch*
Darren Reed *darren.reed@uzh.ch*

CSCS

Sebastian Keller *sebastian.keller@cscs.ch*
Jean-Guillaume Piccinali *jgp@cscs.ch*
Jean Favre *jean.favre@cscs.ch*
Benjamin Cumming *benjamin.cumming@cscs.ch*
John Biddiscombe *john.biddiscombe@cscs.ch*

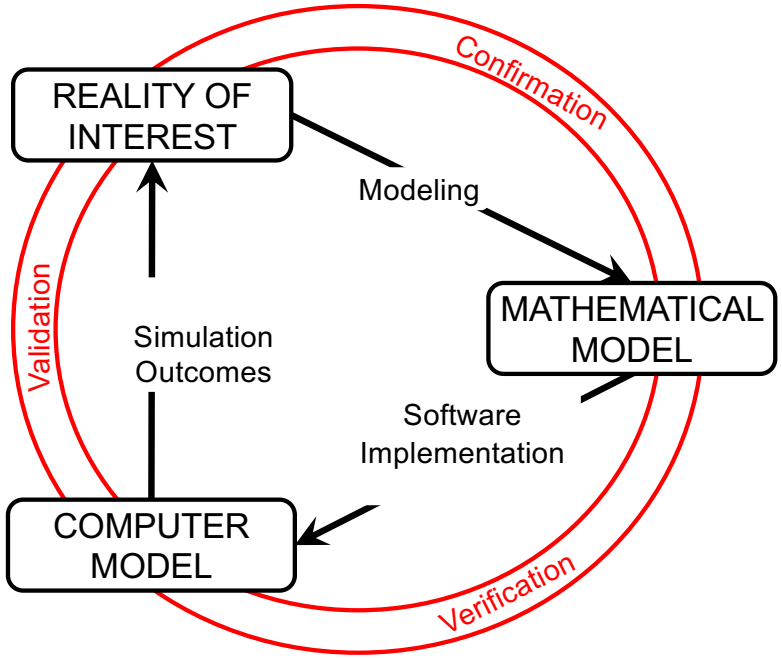
José Escartin (ESO)
Axel Sanz (UPC)
Joseph Touzet (Paris-Saclay)

SPH-EXA SKACH PASC Platform for Advanced Scientific Computing

https://github.com/unibas-dmi-hpc/SPH-EXA

SPH-EXA is a *scalable* and *fault tolerant* Smoothed Particle Hydrodynamics simulation framework co-designed between computational physicists and computer scientists to exploit Exascale supercomputers.

Classical Modeling and Simulation Philosophy

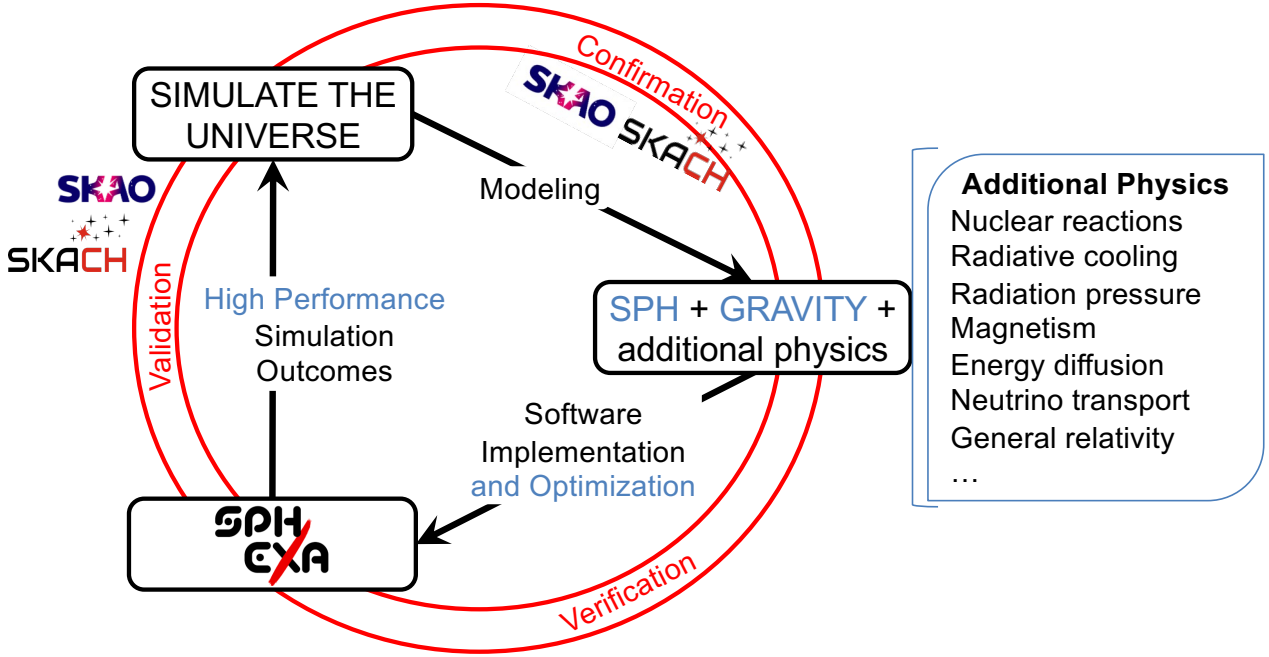


Modeling and Simulation Activities

Iterative Assessment Activities

[Adapted from: Schlesinger, S., "Terminology for Model Credibility," Simulation, Vol. 32, No. 3, 1979.]

SPH-EXA Philosophy: Modeling, Simulation, and Optimization through Interdisciplinary Co-Design



Modeling, Simulation, and Optimization Activities

Iterative Assessment Activities

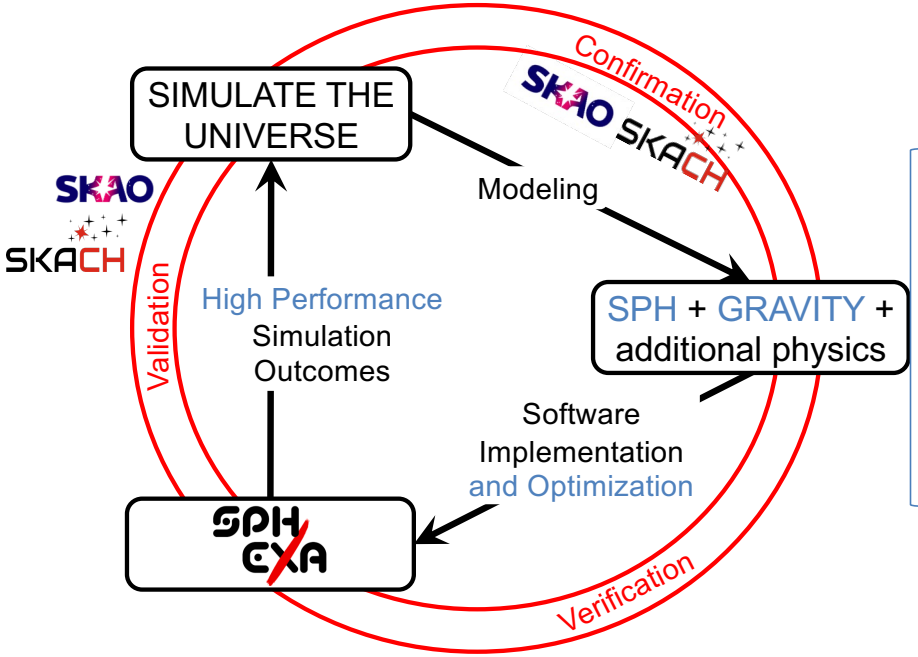
[Adapted from: Schlesinger, S., "Terminology for Model Credibility," Simulation, Vol. 32, No. 3, 1979.]

SPH-EXA Philosophy: Modeling, Simulation, and Optimization through Interdisciplinary Co-Design

Parallelization
Domain decomposition
Multiprocess
Multithreaded
Acceleration
Vectorization
Overlapping comm. & comp.
Scheduling and load balancing
Parallel I/O
…

Portability + Scalability
Mixed precision
Multiple accelerator models
Minimal SW dependencies
Cross-platform compilation, testing, benchmarking
Fault tolerance
Reproducibility
…

Analysis + Visualization
Performance profiling, tracing, visualization
Scientific visualization via snapshots and in-situ
…



Additional Physics
Nuclear reactions
Radiative cooling
Radiation pressure
Magnetism
Energy diffusion
Neutrino transport
General relativity
…

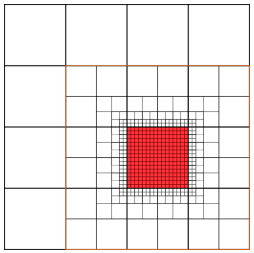
Modeling, Simulation, and Optimization Activities
Iterative Assessment Activities

[Adapted from: Schlesinger, S., "Terminology for Model Credibility," Simulation, Vol. 32, No. 3, 1979.]

SPH-EXA Framework Components

SPH-EXA application front-end

Parallel I/O and test case setup (ICs)

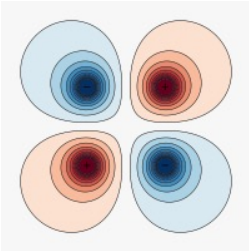


Cornerstone

Octree and domain decomposition framework

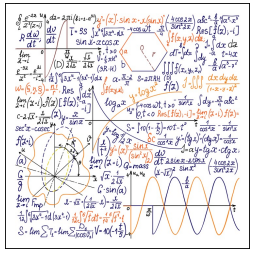
SPH

Hydrodynamics solver



Ryoanji

N-body gravity solver



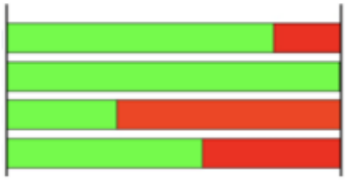
Physics modules

Radiative cooling (GRACKLE)
Nuclear reactions
Star formation
Stellar feedback (in development)
…

SPH-EXA Performance Optimization Strategy

Scalability

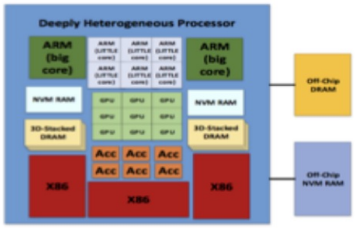
weak, strong



Scheduling & Load-balancing

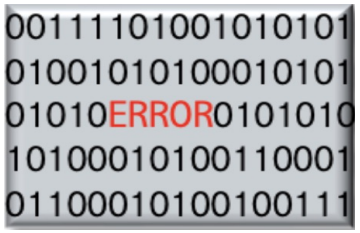
dynamic, adaptive

asynchronous execution



Heterogeneity

portability on various CPU and GPU architectures

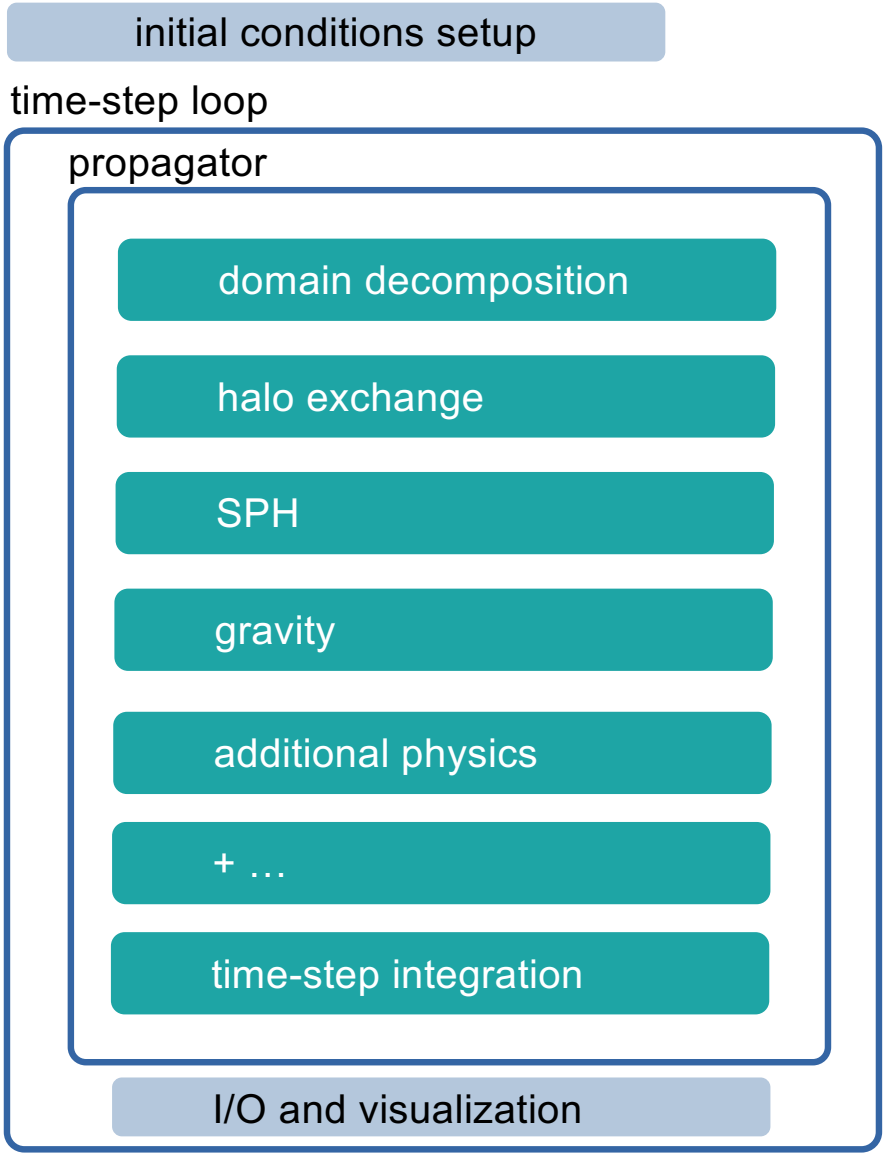


Fault-tolerance

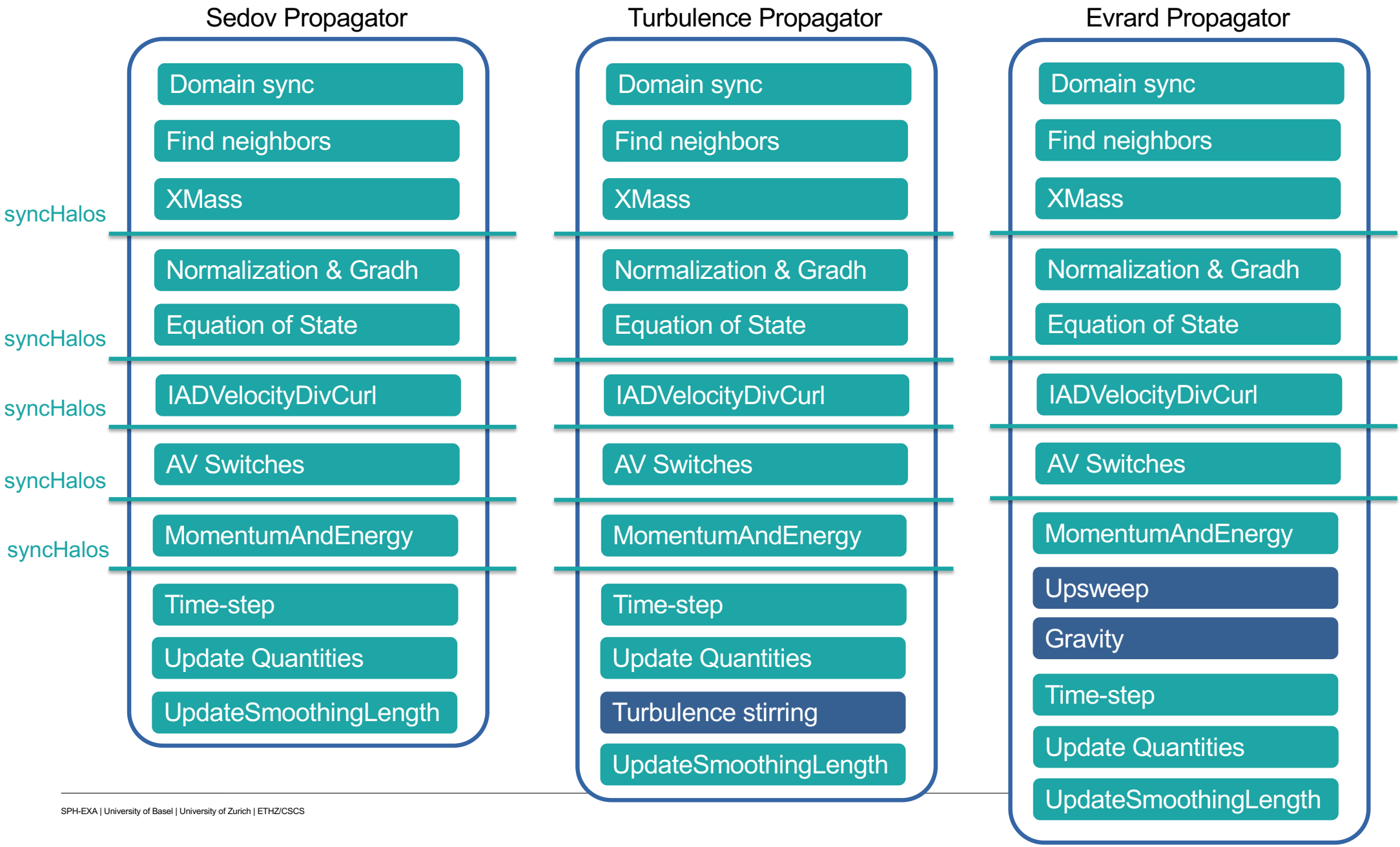
detection of and recovery from silent errors

SPH-EXA: Modular Software Design

- Each simulation has its own propagator and (if needed) an embedded initial conditions generator.
- Propagator may be components added / removed as needed.
- Disjoint API for domain-decomposition and halo exchange propagator components achieves separation of concerns between *communication* of particle data and *implementation* of local and short-ranged (gravity) forces.



SPH-EXA: Test Cases Propagators



Design of Factorial Experiments For Performance Analysis (1/2)

Factors	Values	Properties
SPH-EXA Test cases	-n 32|64|128|256M particles/MPI rank -s 0.0787	Sedov explosion (SPH), starting at 0.0787 s physical time for 5 time-steps
	-n 32|64|128|256M particles/MPI rank -s 200	Turbulence (SPH), starting at simulation time-step 200 for 5 time-steps
	-n 32|64|128|256M particles/MPI rank -s 1.0	Evrard collapse (SPH+gravity), starting at 1.0 s of physical time for 5 time-steps
Computing Nodes	2|4|8 LUMI-G nodes, 8 AMD Mi250X/node	1 AMD Epyc 7603 CPU, 8 AMD Mi250X GPUs half cards, Cray Slingshot interconnect
	2|4|8|16 Piz Daint nodes, 1 NVidia P100/node	1 Intel Xeon E5-2690v3 CPUs, 1 Nvidia P100 GPU, Cray Aries routing and communications ASIC, and Dragonfly network topology
Parallelization	MPI + HIP (LUMI-G)	1 MPI rank associated with 1 GPU device
	MPI + CUDA (Piz Daint)	
Strong Scaling	64M particles / 2|4|8 LUMI-G nodes	
	64M particles / 2|4|8|16 Piz Daint nodes	
Weak Scaling	2M particles/GPU LUMI-G (16M particles/node)	
	16M particles/GPU Piz Daint (16M particles/node)	

Design of Factorial Experiments For Performance Analysis (2/2)

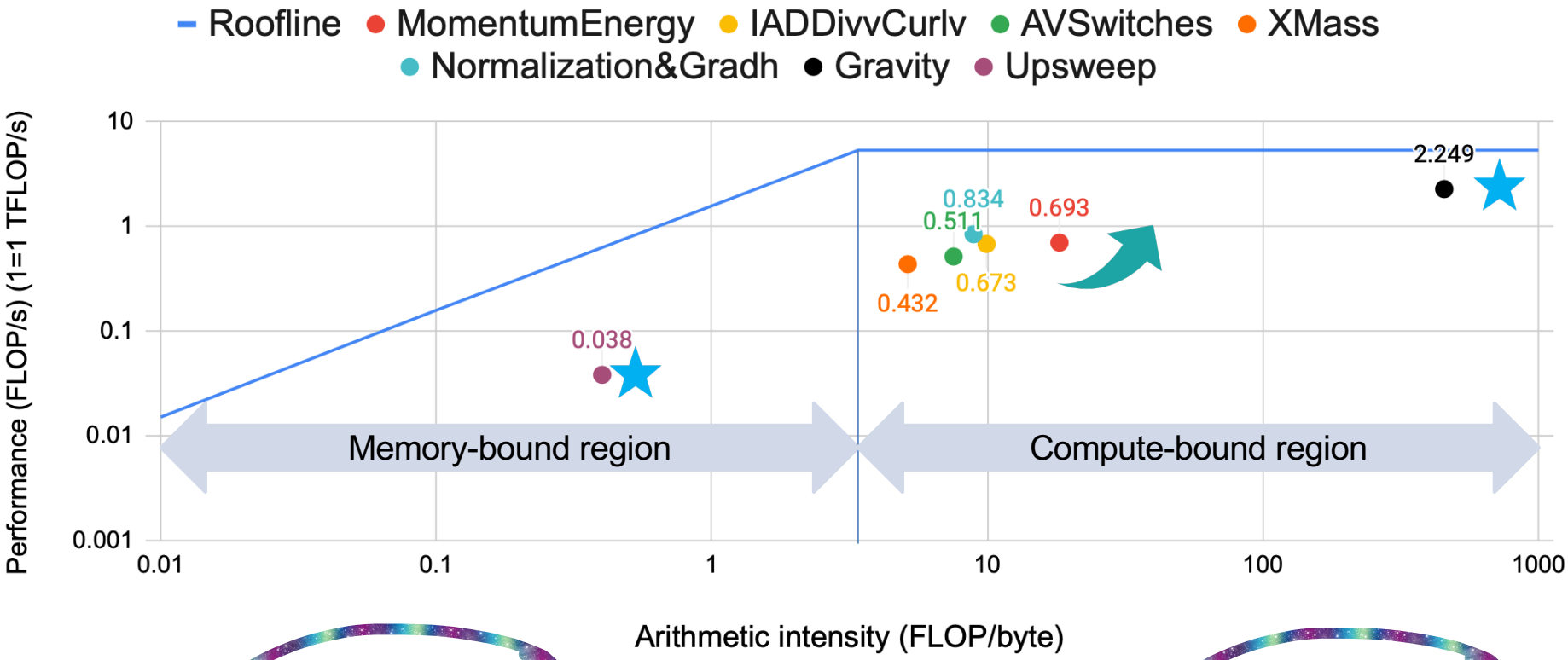
Factors	Values	Properties	
Performance Metrics	Arithmetic intensity [FLOP/Byte] vs. Speed [GFLOP/s]	Arithmetic intensity vs. compute speed w.r.t. memory and compute HW roofs	
	T = max ($T_{\text{MPI ranks}}$)	Time-to-solution (across all MPI ranks)	
	t_p of MPI rank/per GPU: across all | individual nodes	Finishing times of MPI ranks across all nodes | within individual nodes	
Load Imbalance Metrics	$\lambda = (1 - \boldsymbol{\mu} / \max) \times 100$ [%]	Percent load imbalance	$\boldsymbol{\mu}$ of t_p for each MPI rank/per GPU: across all | individual nodes Ignores statistical properties of load distribution
	c.o.v. = $\boldsymbol{\sigma} / \boldsymbol{\mu} < 1$ [unitless]	Coefficient of variation	$\boldsymbol{\sigma}$ of t_p for each MPI rank/per GPU: across all | individual nodes Indicate either few highly loaded outlier processes or many slightly imbalanced processes. Cannot capture info. about application domain.
	$g_1 = {}^{n_p}/_{(n_p-1)(n_p-2)} \sum ({}^{t_p-\mu}/_{\sigma})^3$ [s]	Skewness	
	$g_2 = \left\{ {}^{n_p(n_p+1)}/_{(n_p-1)(n_p-2)(n_p-3)} \sum ({}^{t_p-\mu}/_{\sigma})^4 \right\} - {}^{3(n_p-1)^2}/_{(n_p-2)(n_p-3)}$ [s]	Kurtosis	
	$\vec{I} = \begin{pmatrix} I_0 \\ \cdot \\ \cdot \\ \cdot \\ I_{P-1} \end{pmatrix}$, $I_p = 1 - \frac{t_p}{\boldsymbol{\mu}}$, $\lvert\vec{I}\rvert_2 = \sqrt{I_0^2 + \ldots + I_{P-1}^2}$ [unitless]	* Spatial imbalance per processing element	Captures where load imbalance ($\vec{I}$) occurs & how high (I_p) it is.
Sustainability Metrics	$E = T \times (n_p \times P_p \times u_p + n_m \times P_m) \times PUE$ [J = W × s] Energy consumption [J]	Total energy consumed over the course of a simulation n_p #processing elements, P_p power draw (W) of a processing element, u_p usage factor, n_m mem (GB), P_m power draw from memory (W/GB), PUE Power Usage Efficiency of the system Average energy cost of the simulation	
	Power consumption = E/T [W = J/s] Power efficiency [FLOP/W]	Average power needed for the simulation (energy used/time-to-solution) Efficiency of consuming the power	
	Carbon footprint = E × carbon intensity [gCO_2e]		

* Joint work with Ahmed Eleliemy (UniBas)

Performance

Performance: Roofline Analysis for SPH-EXA

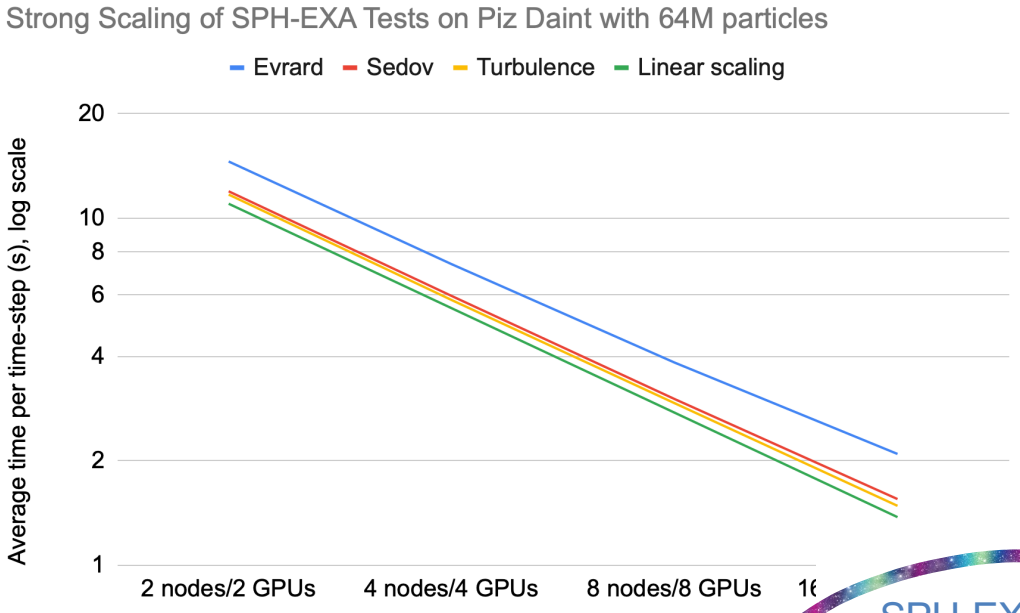
Roofline performance of the gravity (Gravity + Upsweep) and SPH-related (all others) functions for **Evrard collapse** with 64M particles executed on a single Nvidia A100 GPU (miniHPC).

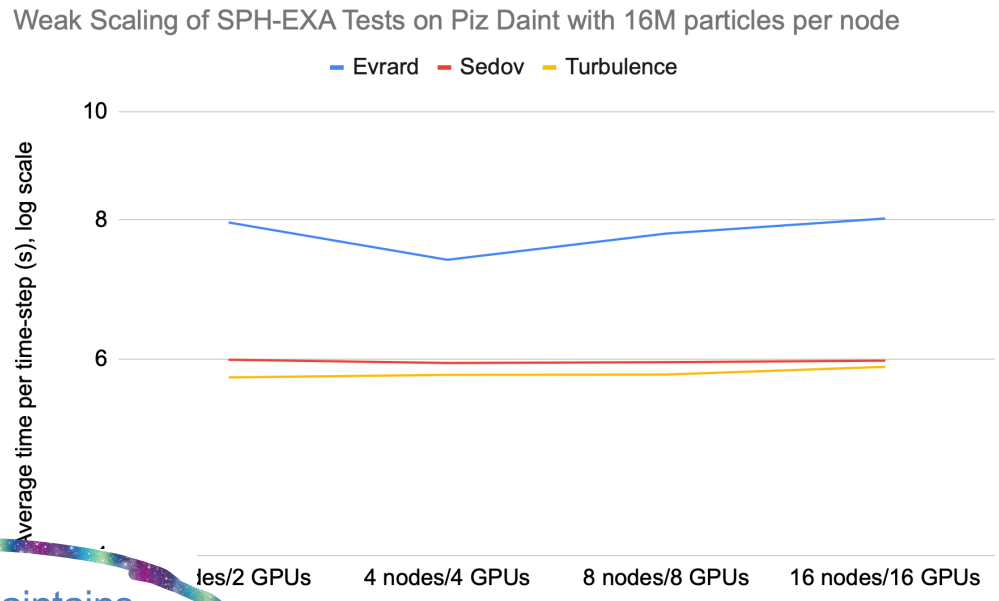


Gravity is highly optimized!

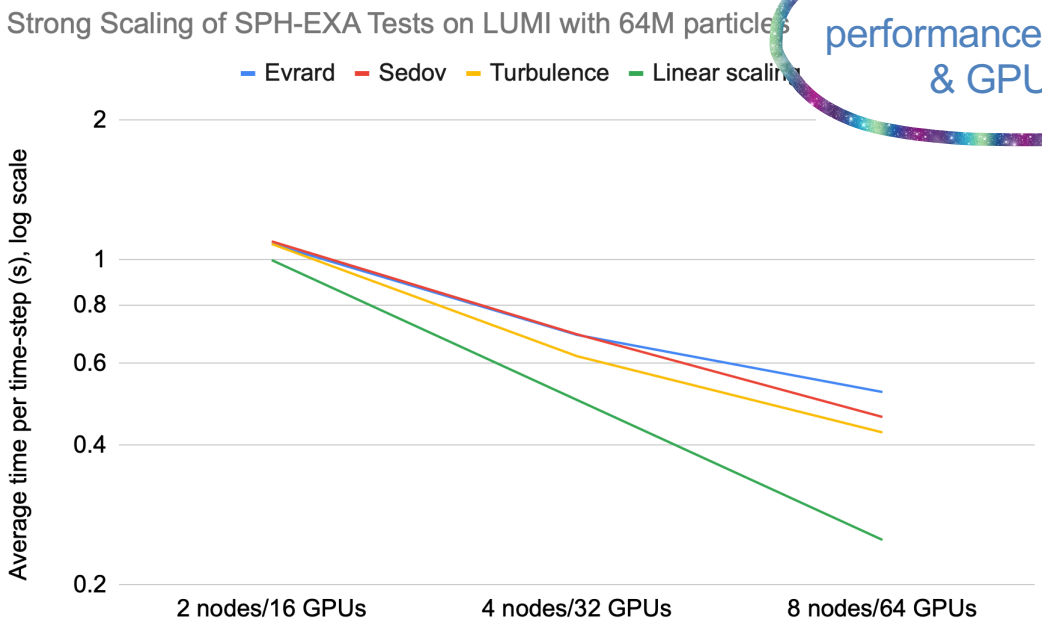
There is room for improvement for SPH

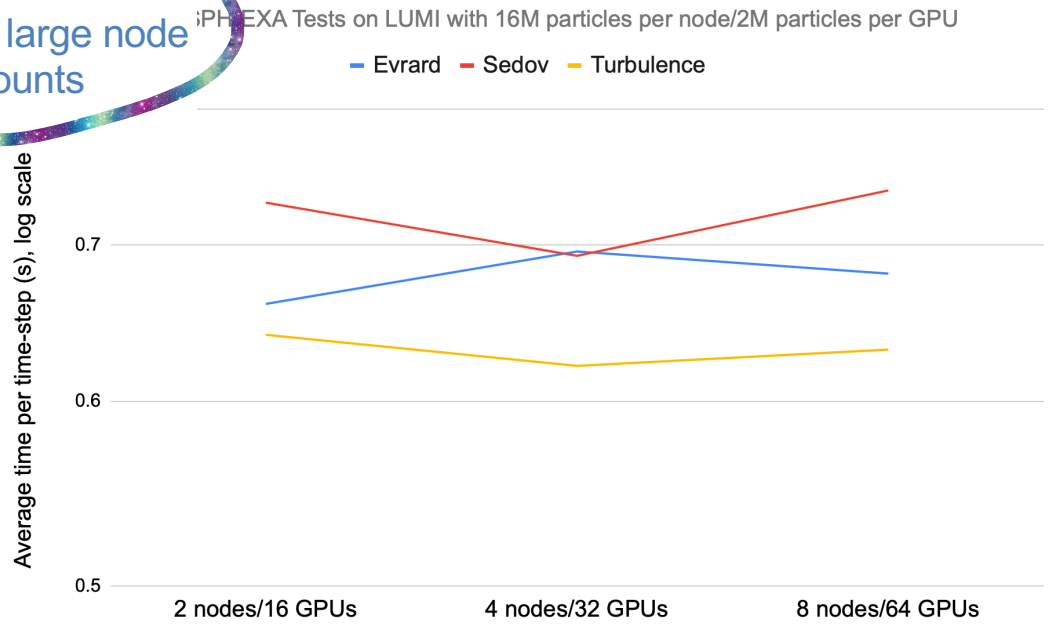
Performance: Scaling of SPH-EXA Test Cases





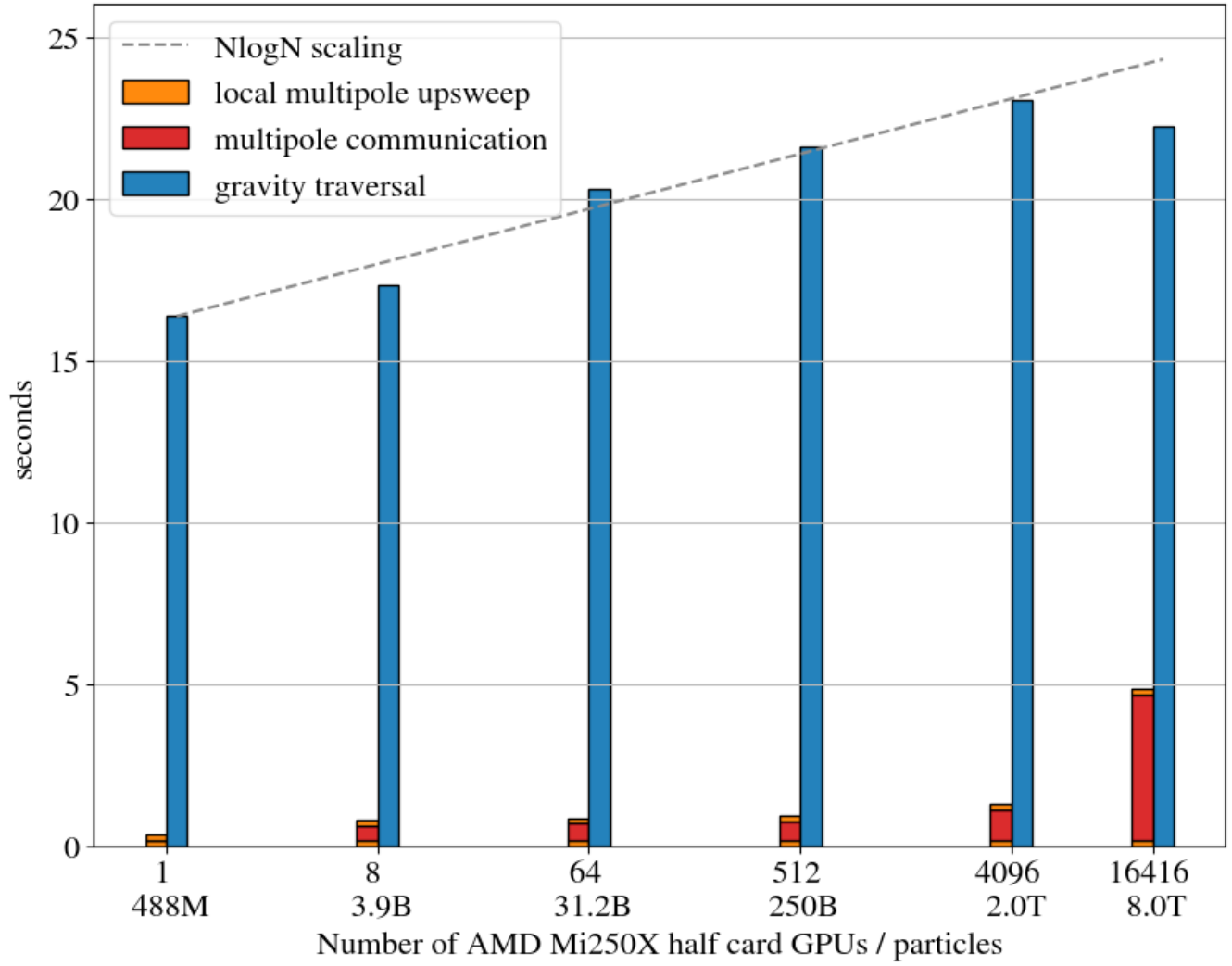
SPH-EXA maintains performance at large node & GPU counts





Performance: Scaling of SPH-EXA Tree Code

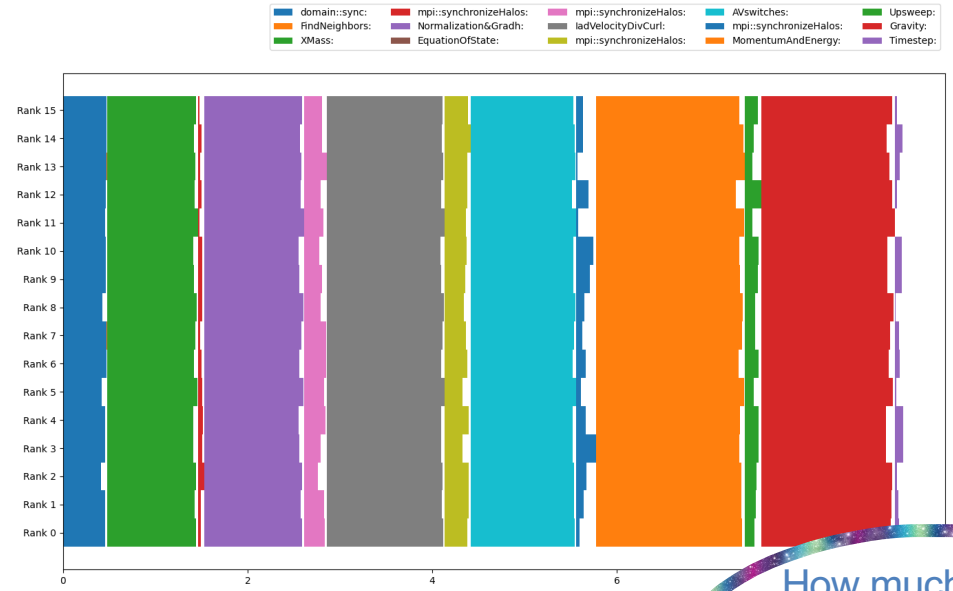
Weak scaling of Barnes-Hut tree traversal, particle-particle, and multipole-particle interactions on LUMI-G with 488M particles/GPU and **8T particles in total** on 2'052 computing nodes (16'416 GPU half cards).



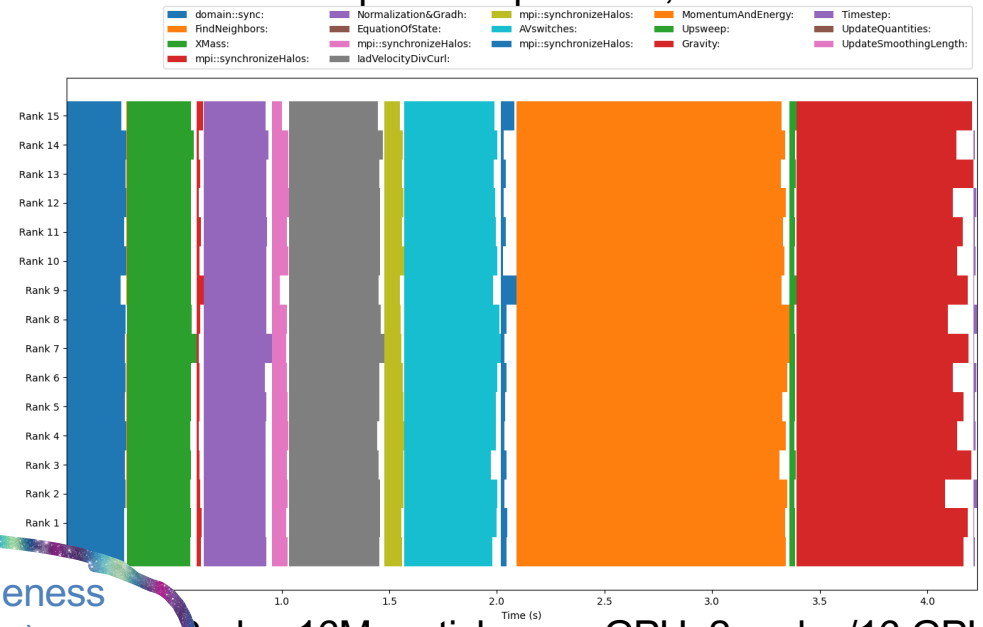
Almost perfect logarithmic weak scaling.

Performance: Profile of Propagator Functions in SPH-EXA Tests

Piz Daint Evrard 16M particles per GPU, 2 nodes/2 GPUs

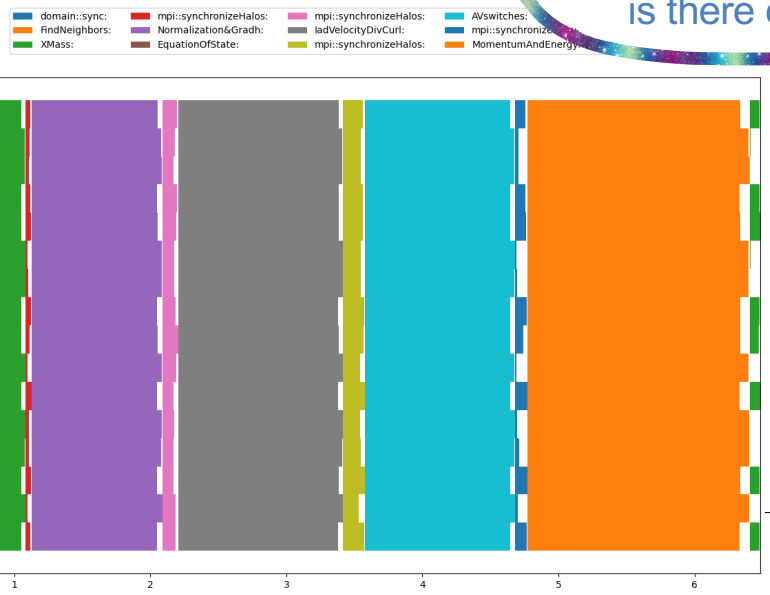


LUMI Evrard 16M particles per GPU, 2 nodes/16 GPUs

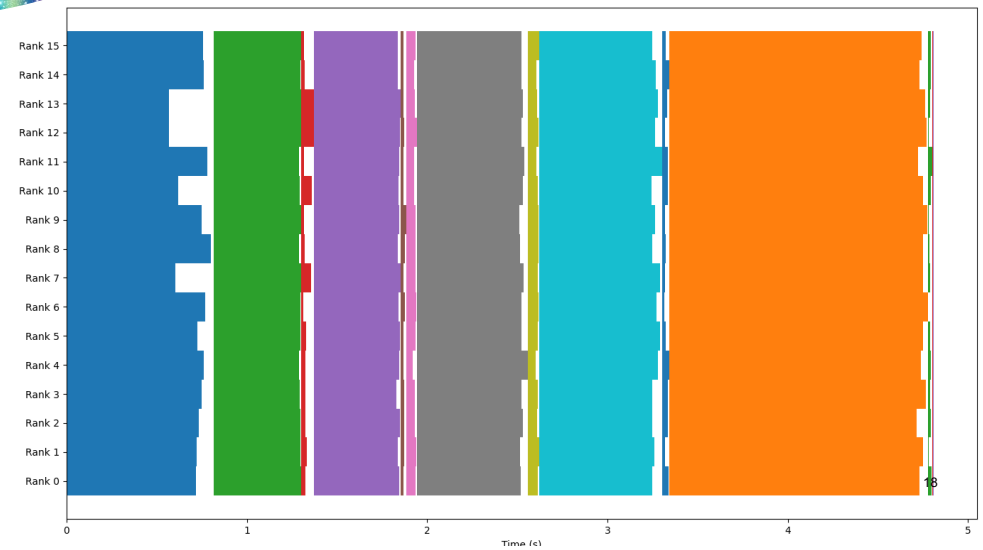


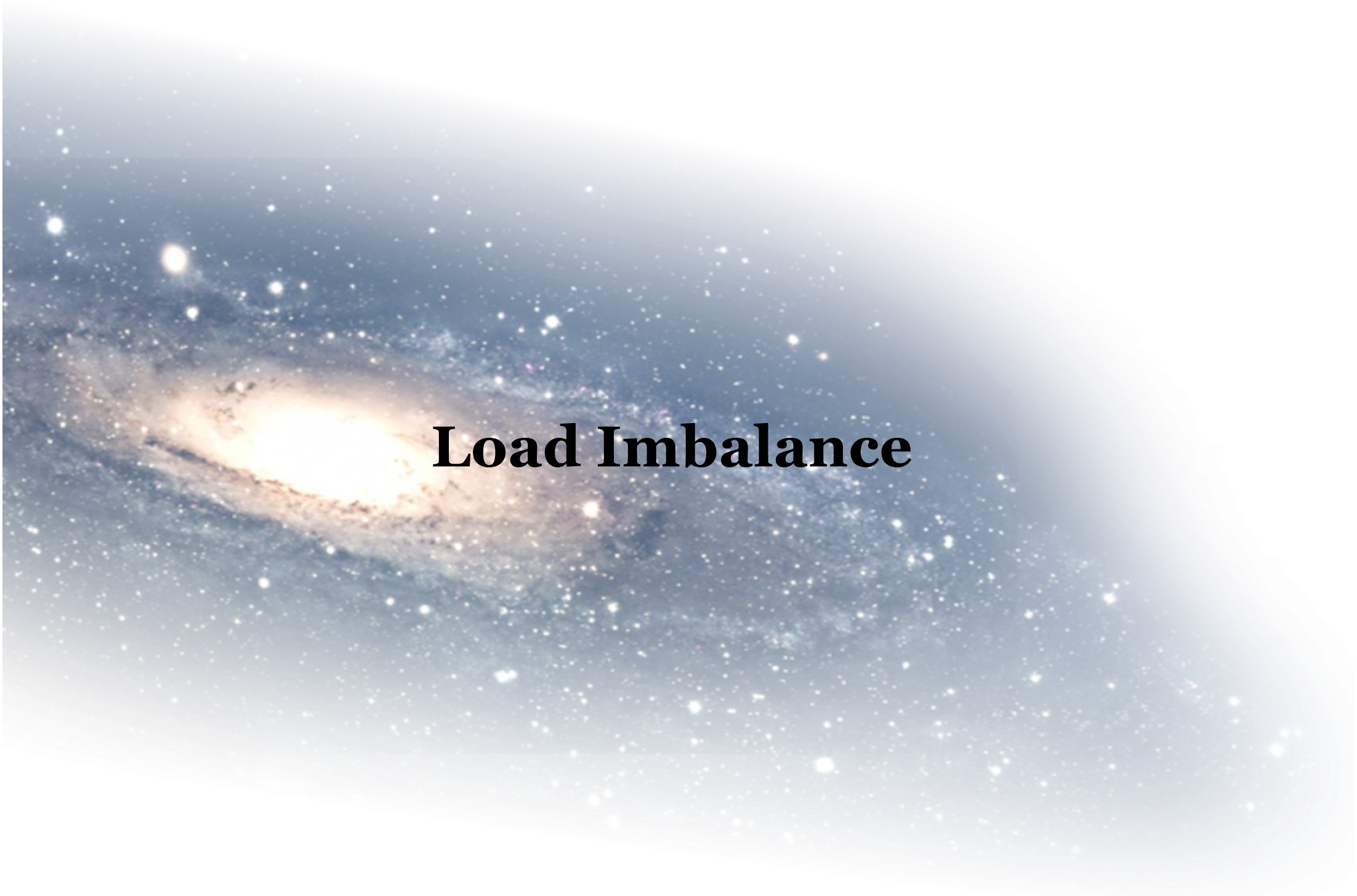
How much idleness
(aka waste)
is there overall?

Piz Daint Sedov 16M particles per GPU,



Sedov 16M particles per GPU, 2 nodes/16 GPUs





Load Imbalance

Performance: Load Imbalance

Scalar load imbalance metrics do not show **where** waste occurs.

Piz Daint Evrard 16M particles per GPU, 16 nodes/16 GPUs

Function	*mu*	*sigma*	**lambda** [%]	**c.o.v.**	**g1** [s]	**g2** [s]	**|I|_2**
domain::sync:	0.4544	0.0186	3.9624	0.0410	-1.3987	0.8548	0.1588
FindNeighbors:	**0.0007**	**0.0002**	**48.4458**	**0.3340**	**2.3951**	**4.8466**	**1.2921**
XMass:	0.9569	0.0168	3.1129	0.0180	0.3558	-0.9587	0.0679
mpi::synchronizeHalos:	**0.0382**	**0.0143**	**43.6148**	**0.3750**	**0.1105**	**-0.3168**	**1.4521**
Normalization&Gradh:	1.0484	0.0184	2.8483	0.0180	0.2402	-1.1403	0.0680
EquationOfState:	0.0019	0.0000	1.4756	0.0080	-0.0570	-0.4037	0.0307
mpi::synchronizeHalos:	**0.2051**	**0.0278**	**16.6786**	**0.1350**	**-0.1870**	**-0.8325**	**0.5245**
IadVelocityDivCurl:	1.2535	0.0140	1.9769	0.0110	0.5395	-0.6951	0.0433
mpi::synchronizeHalos:	**0.2381**	**0.0247**	**15.6283**	**0.1040**	**-0.5184**	**0.0030**	**0.4010**
AVswitches:	1.1198	0.0135	1.7800	0.0120	0.1789	-1.3259	0.0468
mpi::synchronizeHalos:	**0.0967**	**0.0551**	**55.1535**	**0.5690**	**0.6885**	**0.1569**	**2.2041**
MomentumAndEnergy:	1.5756	0.0257	2.4390	0.0160	-0.2716	-0.1550	0.0631
Upsweep:	**0.1245**	**0.0293**	**31.4672**	**0.2350**	**0.1052**	**-0.7422**	**0.9121**
Gravity:	1.3914	0.0307	3.5412	0.0220	-0.0849	-1.1102	0.0855
Timestep:	**0.0490**	**0.0293**	**46.8381**	**0.5980**	**0.0960**	**-1.1385**	**2.3168**
UpdateQuantities:	**0.0000**	**0.0000**	**14.1647**	**0.1300**	**-3.2292**	**12.3751**	**0.5047**
UpdateSmoothingLength:	0.0000	0.0000	4.4964	0.0300	-0.3021	-0.4046	0.1162

Performance: Load Imbalance

Load imbalance occurs arbitrarily on ranks.
Why?

Piz Daint Evrard 16M particles per GPU, 16 nodes/16 GPUs

Spatial imbalance per processing element, measured with $|\vec{I}|_2$

Function	|I|_2	Rank 0	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	Rank 6	Rank 7	Rank 8	Rank 9	Rank 10	Rank 11	Rank 12	Rank 13	Rank 14	Rank 15
domain::sync:	0.1588	-0.0068	-0.0044	0.0890	-0.0055	0.0003	0.0820	-0.0275	-0.0286	0.0614	-0.0131	-0.0238	0.0041	-0.0160	-0.0334	-0.0365	-0.0413
FindNeighbors:	**1.2921**	**0.1356**	**0.1511**	**0.1817**	**0.0362**	**0.1290**	**0.1914**	**0.1761**	**-0.9397**	**0.1956**	**0.0073**	**0.0166**	**0.1856**	**0.1029**	**-0.7251**	**0.0272**	**0.1287**
XMass:	0.0679	-0.0106	0.0042	-0.0079	0.0179	0.0220	-0.0270	0.0167	-0.0049	-0.0211	0.0178	0.0226	-0.0321	0.0061	-0.0053	0.0114	-0.0095
mpi::synchronizeHalos:	**1.4521**	**0.2585**	**0.1369**	**-0.7735**	**-0.2748**	**-0.4039**	**-0.2099**	**-0.0334**	**0.3919**	**-0.2073**	**-0.2619**	**-0.2313**	**0.5897**	**-0.0130**	**0.3955**	**0.0781**	**0.5584**
Normalization&Gradh:	0.0680	-0.0125	0.0046	-0.0098	0.0193	0.0215	-0.0271	0.0162	-0.0046	-0.0222	0.0169	0.0244	-0.0293	0.0012	-0.0015	0.0128	-0.0099
EquationOfState:	0.0307	0.0002	0.0008	0.0111	0.0085	-0.0070	-0.0017	-0.0038	0.0135	-0.0024	-0.0150	0.0037	-0.0107	0.0010	-0.0045	-0.0040	0.0102
mpi::synchronizeHalos:	**0.5245**	**-0.0877**	**-0.0499**	**0.2565**	**0.1231**	**-0.1240**	**-0.0760**	**-0.1588**	**-0.1853**	**0.1119**	**0.0317**	**0.1825**	**-0.0258**	**0.1055**	**-0.2002**	**0.0472**	**0.0493**
IadVelocityDivCurl:	0.0433	-0.0060	0.0030	-0.0016	0.0115	0.0123	-0.0198	0.0117	0.0001	-0.0125	0.0077	0.0150	-0.0202	0.0077	-0.0101	0.0019	-0.0008
mpi::synchronizeHalos:	**0.4010**	**-0.0439**	**-0.0876**	**-0.0819**	**0.1912**	**-0.0765**	**0.1918**	**-0.0348**	**0.0452**	**0.1208**	**0.0674**	**0.0095**	**0.0253**	**-0.0333**	**-0.0316**	**-0.1852**	**-0.0763**
AVswitches:	0.0468	-0.0081	-0.0002	-0.0017	0.0142	0.0132	-0.0181	0.0139	0.0012	-0.0137	0.0064	0.0093	-0.0176	0.0173	-0.0154	-0.0058	0.0051
mpi::synchronizeHalos:	**2.2041**	**0.5682**	**0.1449**	**-0.2141**	**-1.2298**	**-0.0856**	**0.4400**	**-0.0916**	**0.2951**	**0.0856**	**-0.5520**	**-0.9366**	**0.7595**	**-0.4070**	**0.7968**	**0.2451**	**0.1814**
MomentumAndEnergy:	0.0631	-0.0069	-0.0004	-0.0007	0.0111	0.0094	-0.0213	0.0157	-0.0053	-0.0099	0.0078	0.0134	-0.0208	0.0351	-0.0250	-0.0145	0.0122
Upsweep:	**0.9121**	**0.1709**	**0.0475**	**-0.0095**	**-0.2108**	**-0.1900**	**0.2409**	**-0.2099**	**0.0930**	**0.1188**	**-0.1412**	**-0.2202**	**0.3431**	**-0.4592**	**0.3701**	**0.2007**	**-0.1443**
Gravity:	0.0855	-0.0043	-0.0095	-0.0146	0.0321	0.0326	-0.0208	0.0075	0.0013	-0.0301	0.0200	0.0221	-0.0367	-0.0157	0.0030	0.0280	-0.0150
Timestep:	**2.3168**	**0.0417**	**0.1873**	**0.4282**	**-0.8678**	**-0.8811**	**0.6072**	**-0.1801**	**0.0026**	**0.7568**	**-0.5586**	**-0.6161**	**0.9990**	**0.4265**	**-0.0436**	**-0.7532**	**0.4511**
UpdateQuantities:	**0.5047**	**-0.0179**	**-0.0597**	**-0.0206**	**-0.0224**	**-0.0553**	**-0.0425**	**-0.0002**	**-0.1650**	**0.0024**	**0.0075**	**-0.0150**	**0.4626**	**-0.0570**	**0.0008**	**-0.0185**	**0.0009**
UpdateSmoothingLength:	0.1162	-0.0175	-0.0143	0.0177	-0.0002	-0.0247	-0.0061	0.0570	-0.0471	-0.0059	-0.0033	0.0146	0.0452	0.0160	-0.0455	-0.0233	0.0374

Sustainability

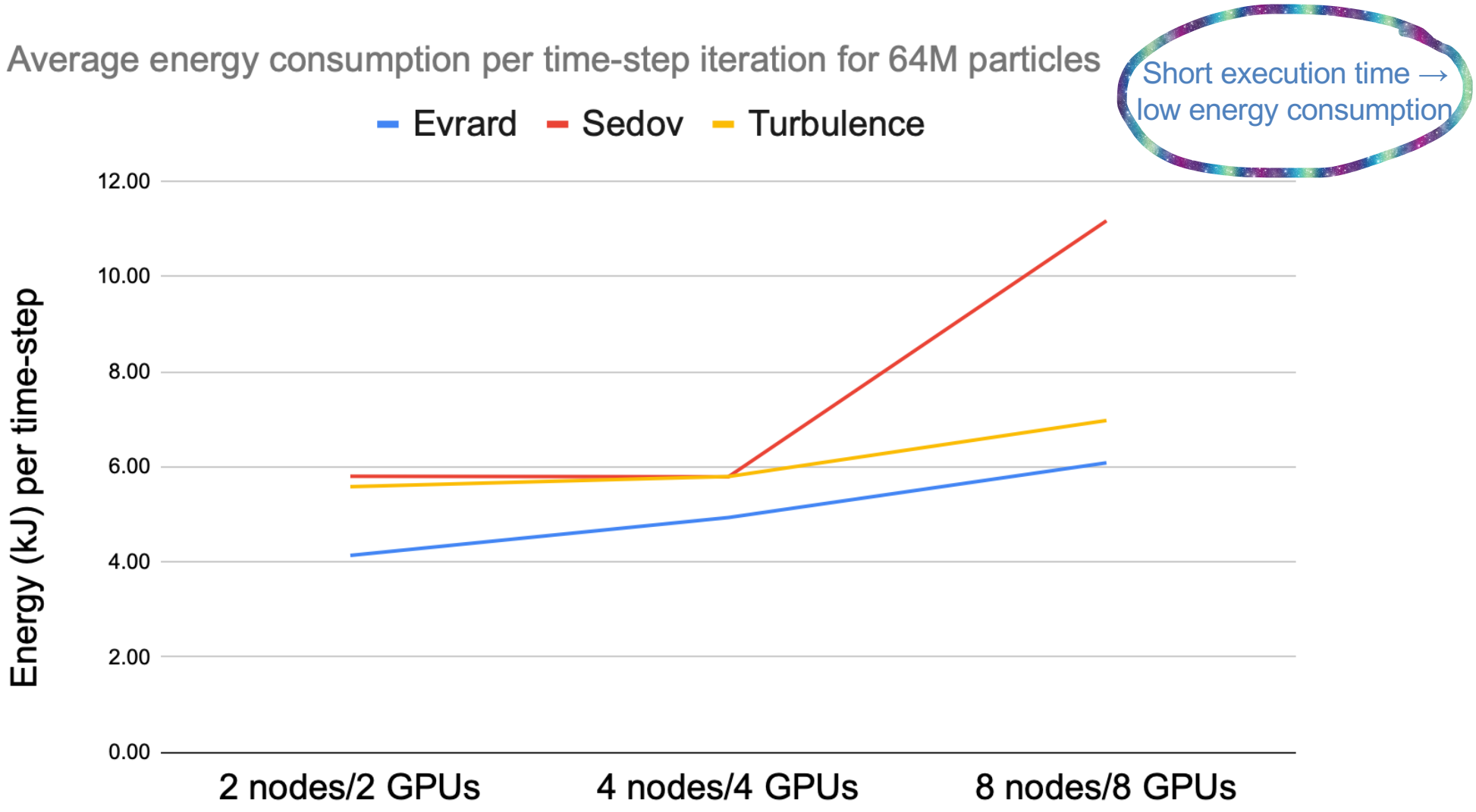
Sustainability

Information is available.
Let's use it!

Energy consumption (kJ) of a job and its resource usage factor (u_p) on Piz Daint

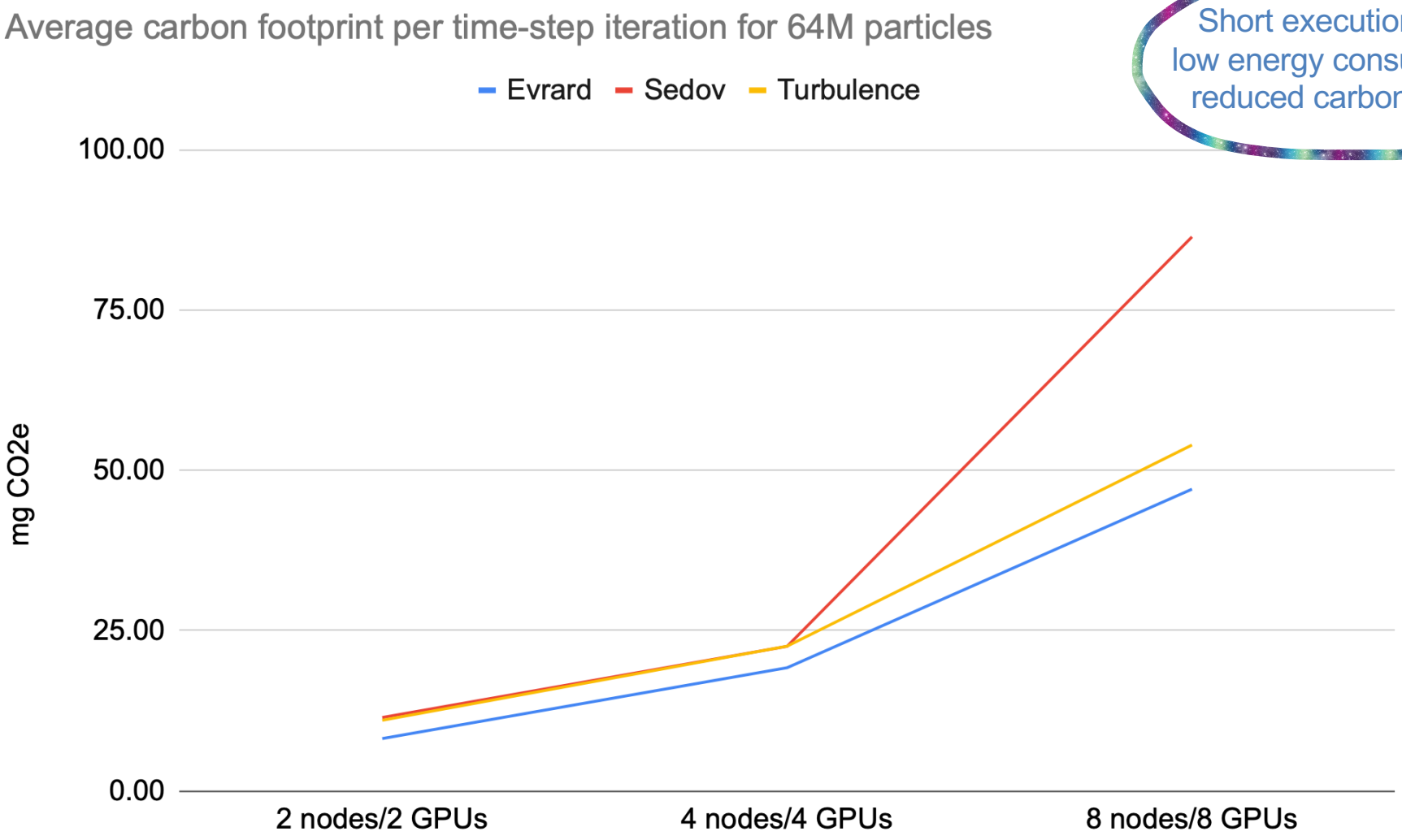
```
Job information (1/3)
------------------------------------------------------------------------------------------------------
             Submit            Eligible               Start                 End    Elapsed Time limit
------------------- ------------------- ------------------- ------------------- ---------- ----------
2023-01-02T10:27:56 2023-01-02T10:27:56 2023-01-02T10:27:59 2023-01-02T10:28:32   00:00:33   00:05:00
------------------------------------------------------------------------------------------------------

Job information (2/3)
-------------------------------------------------------------
    Username      Account    Partition   NNodes        Energy
------------ ------------ ------------ -------- -------------
     uname           acc        normal        8     34.861 kJ

Job information (3/3) - GPU utilization data
----------------------------------------------------
   Node name       Usage      Max mem Execution time
------------ ----------- ------------ --------------
    nid05013        62 %     2294 MiB       00:00:22
    nid05011        64 %     2294 MiB       00:00:22
    nid05010        62 %     2294 MiB       00:00:22
    nid05014        63 %     2294 MiB       00:00:22
    nid05012        63 %     2294 MiB       00:00:22
    nid05016        62 %     2294 MiB       00:00:22
    nid05015        62 %     2294 MiB       00:00:22
    nid05009        63 %     2294 MiB       00:00:22
```

Sustainability: SPH-EXA Energy Consumption on Piz Daint



Energy used = $\mathbf{T} \times (n_p \times P_p \times u_p + n_m \times P_m) \times \text{PUE}$ [kJ]

Sustainability: SPH-EXA Carbon Footprint on Piz Daint



Short execution time → low energy consumption → reduced carbon footprint

Carbon footprint = **energy used** x carbon intensity

Calculator: https://www.green-algorithms.org

Sustainability: Estimated Energy Needed for SPH-EXA on LUMI

"Hero" Runs on LUMI-G (Finland)

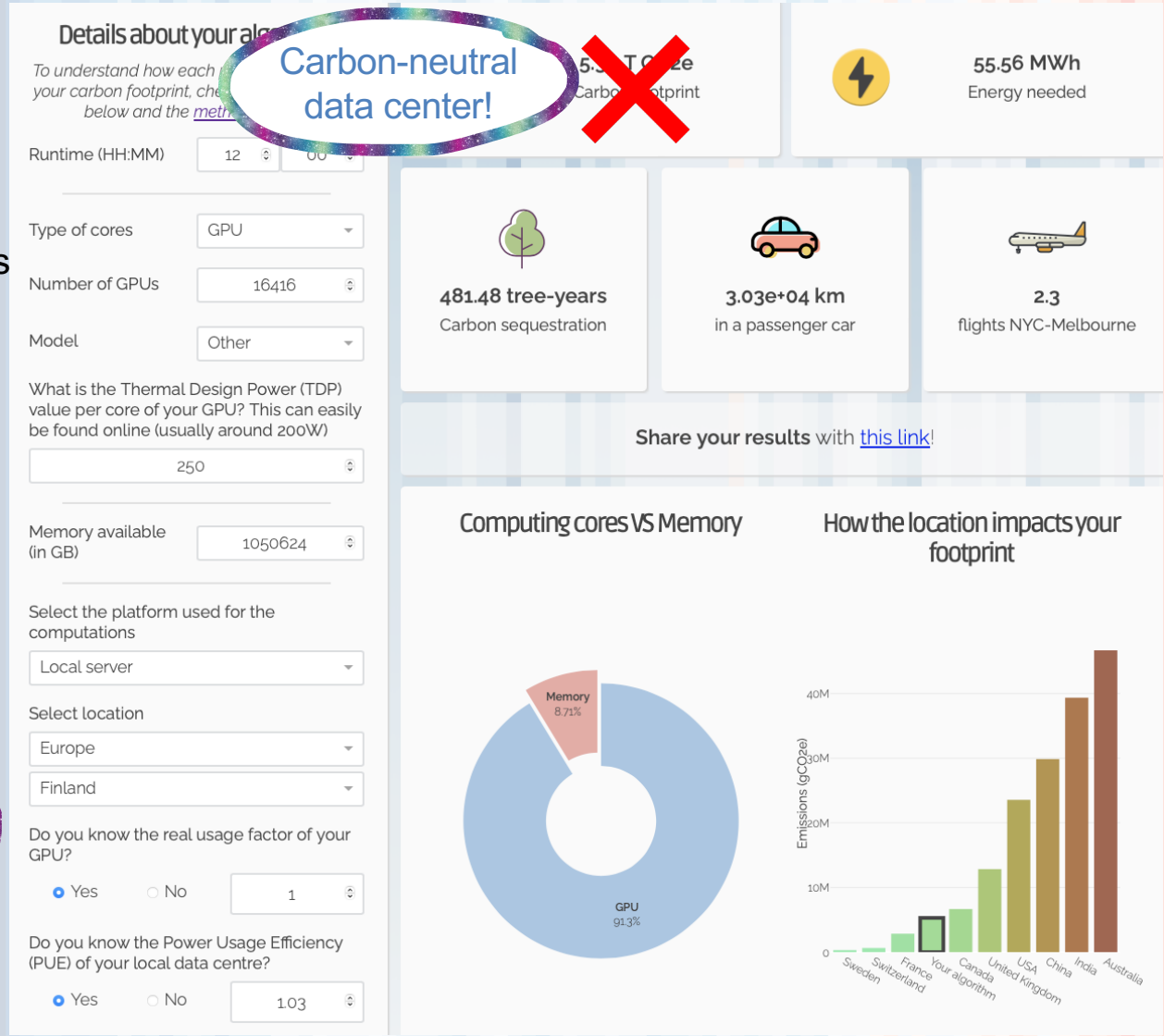
12 hours on Dec 19, 2022

2'052 AMD EPYC 7A53 CPUs
16'416 AMD Instinct MI250X 64GB GPUs

PUE 1.03

SPH-EXA Tests
- Turbulence simulation
- Pure gravity weak scaling (tree code on earlier slide)

On carbon-positive data centers (other than **LUMI**), parallelism would add up all the carbon footprint ☹
Need to take this into account!



Calculator: https://www.green-algorithms.org

Sustainability: Estimated Energy Needed for SPH-EXA on Piz Daint

"Hero" Runs on Piz Daint (Switzerland)

A hypothetical 12 hours run

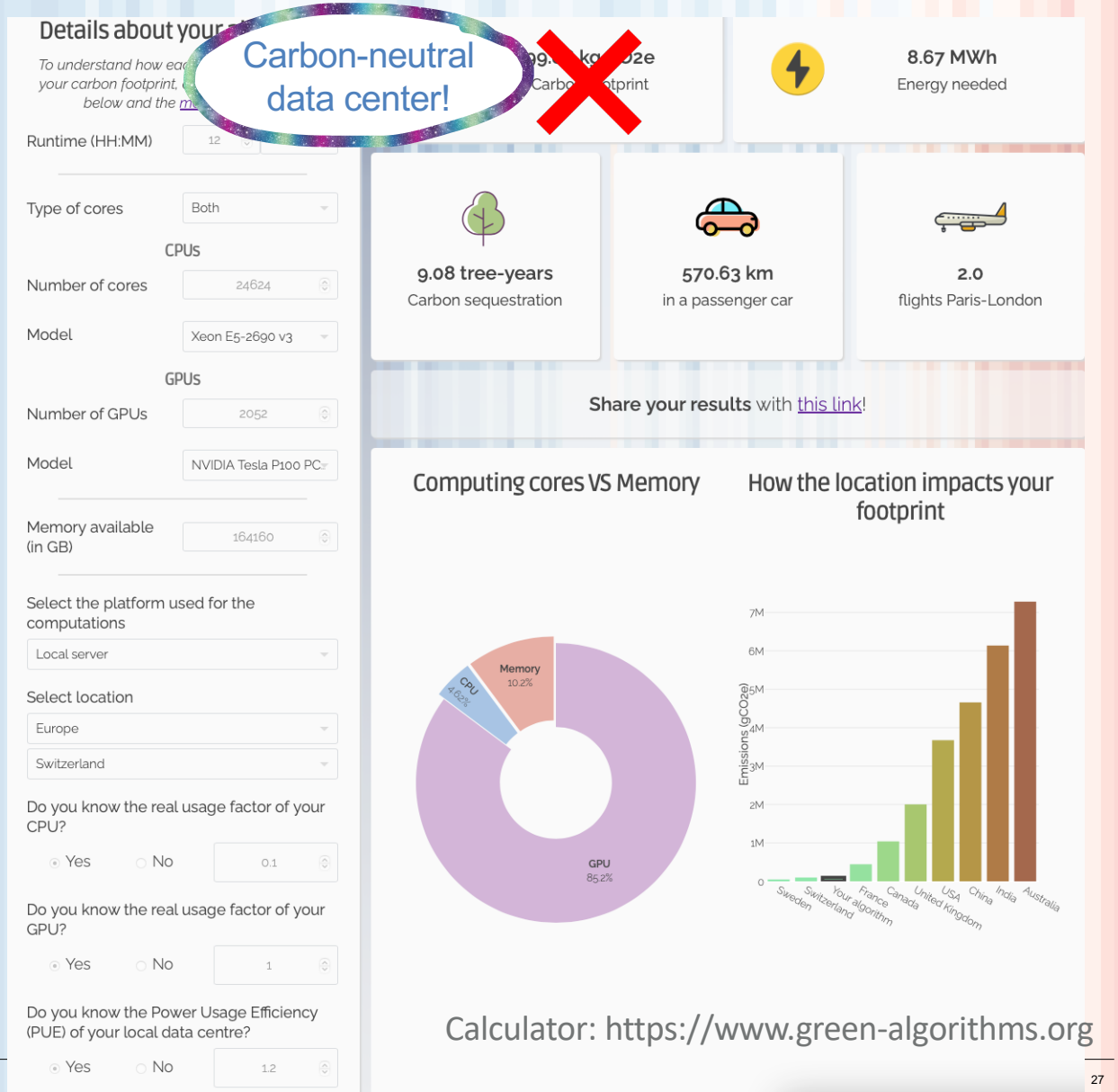
2'052 Intel® Xeon® E5-2690 v3 CPUs
2'052 NVIDIA® Tesla® P100 16GB GPUs

PUE 1.20

SPH-EXA Tests (smaller runs than LUMI)

- Turbulence simulation
- Pure gravity weak scaling (tree code on earlier slide)

On carbon-positive data centers (other than **Piz Daint**), parallelism would add up all the carbon footprint ☹
Need to take this into account!



Calculator: https://www.green-algorithms.org

What Have We Learnt Along the Way?

More nodes is not always better (performance and energy-wise) ...

… but more GPUs per node is.

Memory matters for scalability and energy consumption!

Gravity is optimized but SPH can still improve.

Load imbalance matters but needs to be measured where and when it arises.

Performance is **not** free. It costs time, energy, and has a carbon footprint.

Optimized algorithms, number of resources used, and their usage factor can minimize the environmental impact while still advancing scientific discovery.

What's Next?

Scale up this performance, load imbalance, sustainability analysis to drive further optimizations.

Optimize SPH implementation.

Incorporate additional physics
(for PHOEBOS and other breakthrough simulations).

Enable MPI-level load balancing in SPH-EXA for load balancing within time-steps.

Detect and predict when to rebalance among time-steps.

Exploit memory space by reducing
precision wherever possible.

Implement detection of and recovery from silent errors.

Report not only scalability but also energy and carbon footprint.
Be stingy with performance benchmarking and analysis.

Visualize performance in conjunction with scientific output (TB available!)

Performance Analysis of ~~Load Imbalance~~ in SPH-EXA Simulations

SKACH Winter Meeting
Basel, Switzerland
January 12, 2023

Florina M. Ciorba

Universität Basel

Florina Ciorba (PI) *florina.ciorba@unibas.ch*
Ruben Cabezon (Co-PI) *ruben.cabezon@unibas.ch*
Osman Seckin Simsek *osman.simsek@unibas.ch*
Ahmed Eleliemy *ahmed.eleliemy@unibas.ch*
Lukas Schmidt *luke.schmidt@unibas.ch*

Universität Zürich

Lucio Mayer (Co-PI) *lmayer@physik.uzh.ch*
Noah Kubli *noah.kubli@uzh.ch*
Darren Reed *darren.reed@uzh.ch*

CSCS

Sebastian Keller *sebastian.keller@cscs.ch*
Jean-Guillaume Piccinali *jgp@cscs.ch*
Jean Favre *jean.favre@cscs.ch*
Benjamin Cumming *benjamin.cumming@cscs.ch*
John Biddiscombe *john.biddiscombe@cscs.ch*

José Escartin (ESO)
Axel Sanz (UPC)
Joseph Touzet (Paris-Saclay)

SPH-EXA SKACH PASC Platform for Advanced Scientific Computing

https://github.com/unibas-dmi-hpc/SPH-EXA