PERFORMANCE MEASUREMENTS FOR MUSUBI

NHR4CES Performance for CFD

Harald Klimach, DLR, Institute of Software Methods for Product Virtualisation

Preamble

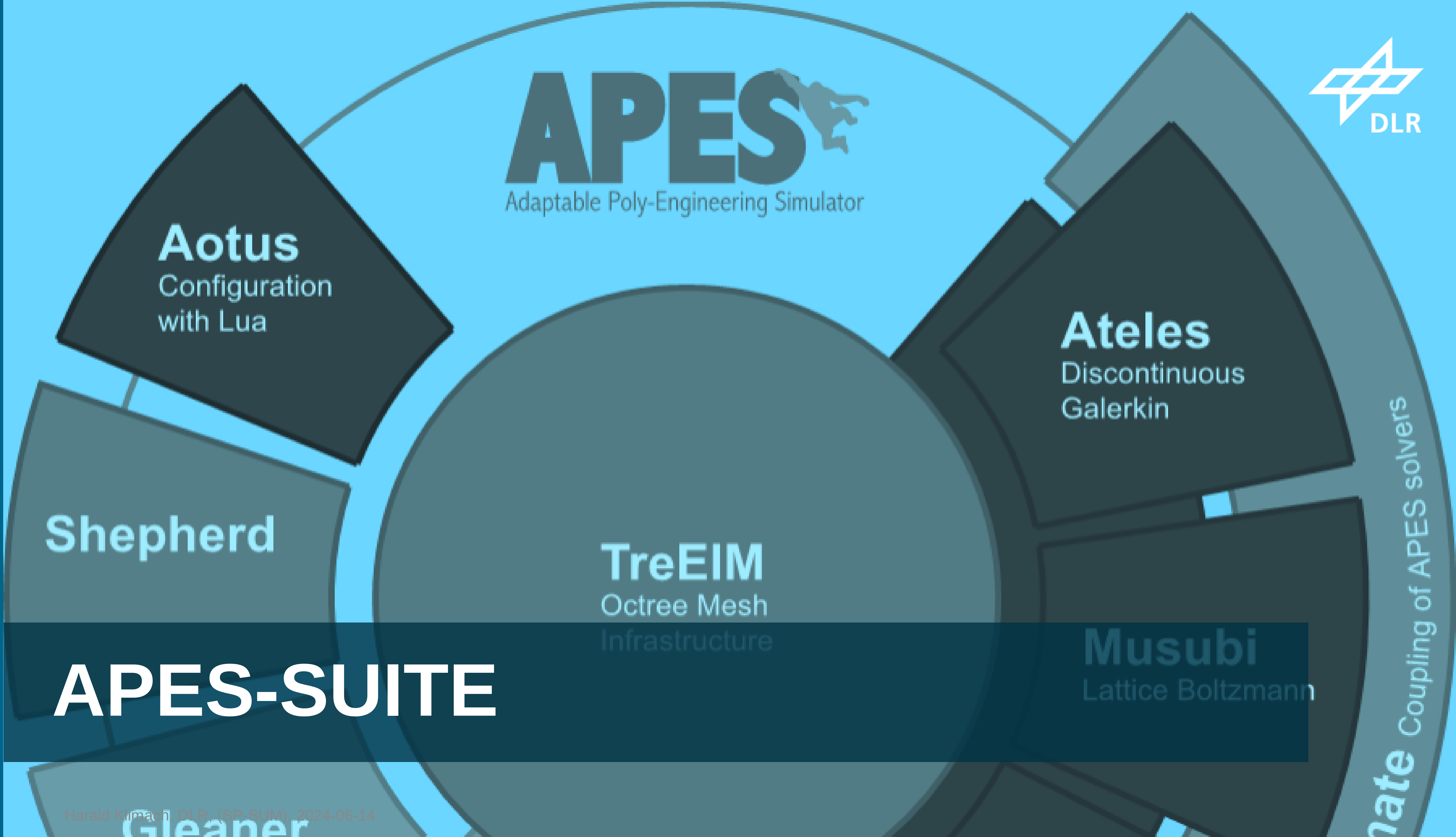
Most of the work I present has been done by my colleagues over the years:

- Manuel Hasert
- Kannan Masilamani
- Jiaxing Qi
- Simon Zimny
- Kartik Jain
- Jana Gericke
- Gregorio Gerardo Spinelli

Overview

- APES Framework
- Musubi Design
- Performance Measurements

APES-SUITE


APES
Adaptable Poly-Engineering Simulator
Aotus
Configuration
with Lua
Shepherd
TreElM
Octree Mesh
Infrastructure
Ateles
Discontinuous
Galerkin
Musubi
Lattice Boltzmann
Coupling of APES solvers
Gleaner

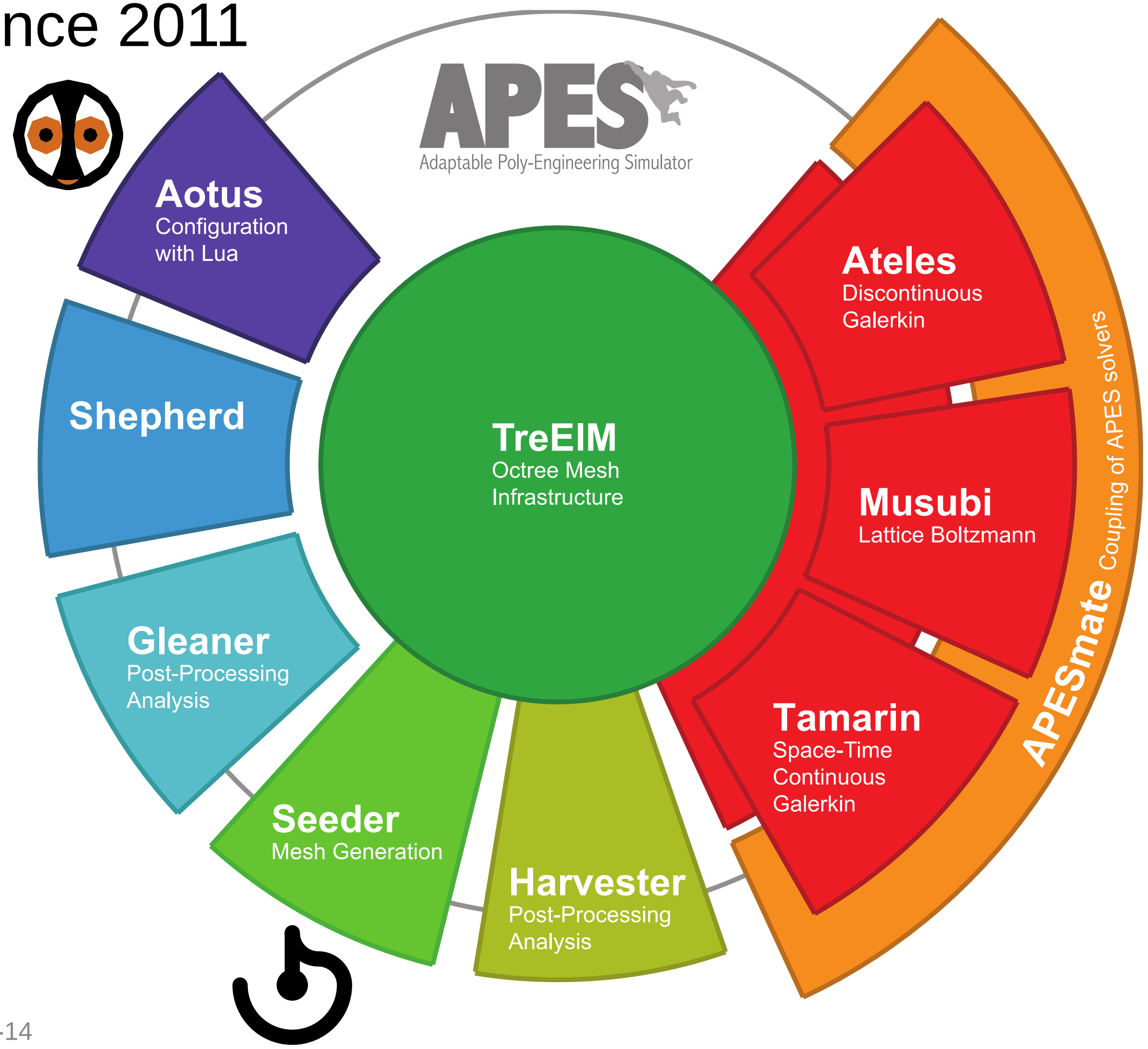
APES-Suite Objectives

- Common infrastructure for stencil-based solvers
- Complete distributed parallelism across the process chain
  - Octree mesh: implicit information on topology
  - Space-Filling curve: Simple partitioning allows distributed reading of data
- High portability on HPC system
  - Few dependencies
- Sources (F2003) available at:

https://github.com/apes-suite

APES-Suite

In development since 2011



Typical Processing Pipeline

- Example on HLRS Hermit (Cray XE System with AMD Opteron 6276 processors; 2012 - 2014)

Seeder

- ~520 mil. Elem.
- Pre-/Post nodes
- Memory: ~70 GB
- *Wall clock: ~3 h*

Musubi

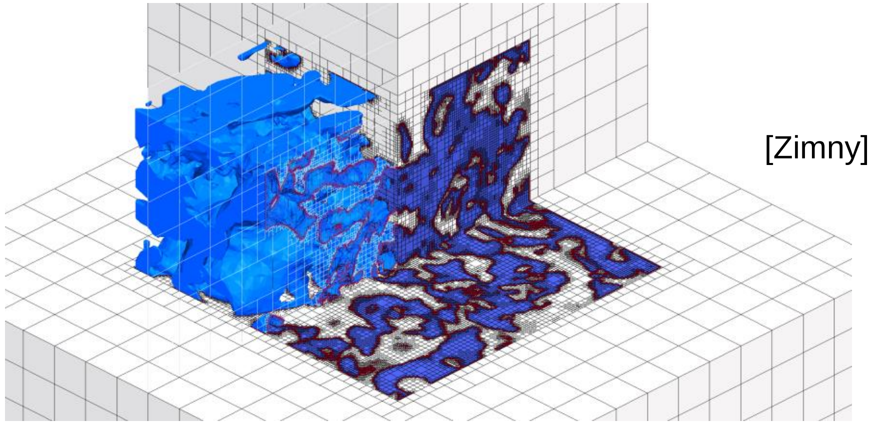
- Sim. Time: ~20 s
- Compute nodes
- 64 nodes
- *Wall clock: ~40 h*

Harvesting

- ~75 mil. Elem.
- Compute nodes
- 16 nodes
- *Wall clock: ~2 h*

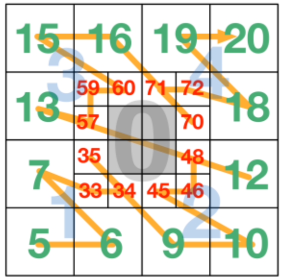
STL → Mesh → Restart → VTK

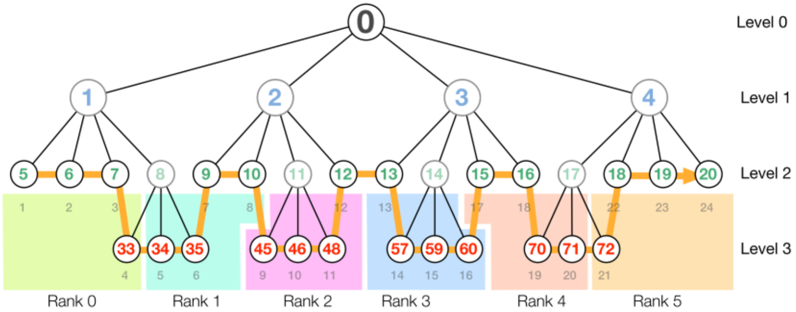
Treelm



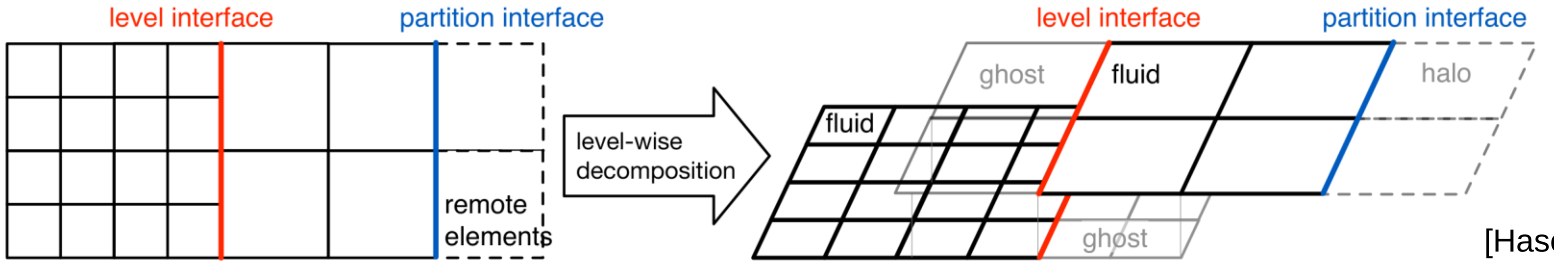
[Zimny]

- Sparse octree mesh
- Distributed loading
  - Partitioning via Z-curve

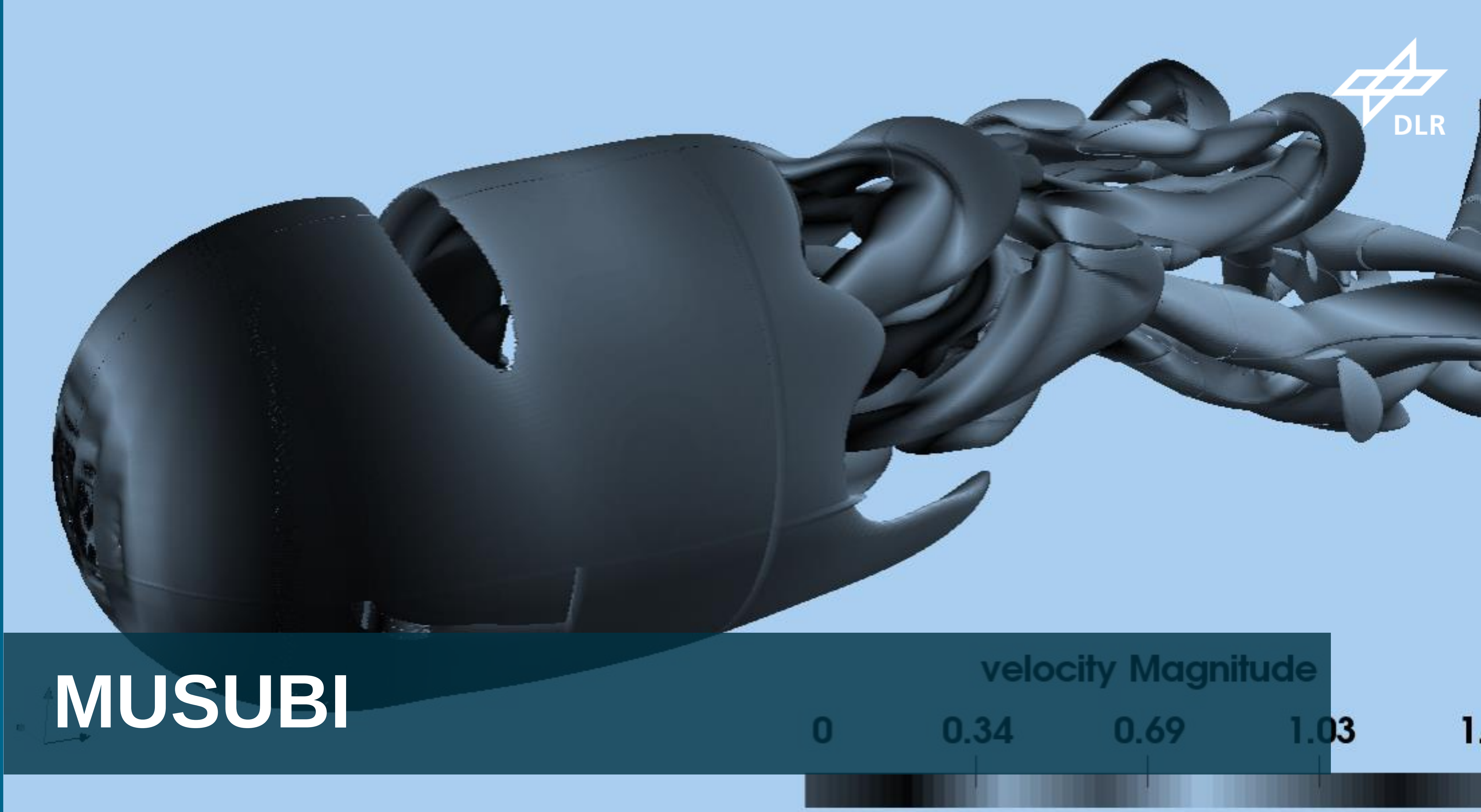




- Neighborhood information generated after loading on distributed system:
  - Adds Halo elements on each partition for communication
  - Adds „Ghost“ elements in each partition for interpolation



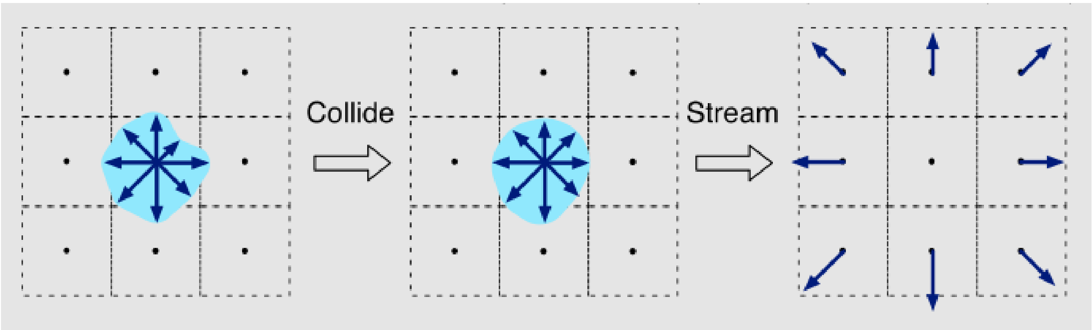
[Hasert]



MUSUBI

Lattice-Boltzmann Method (LBM)

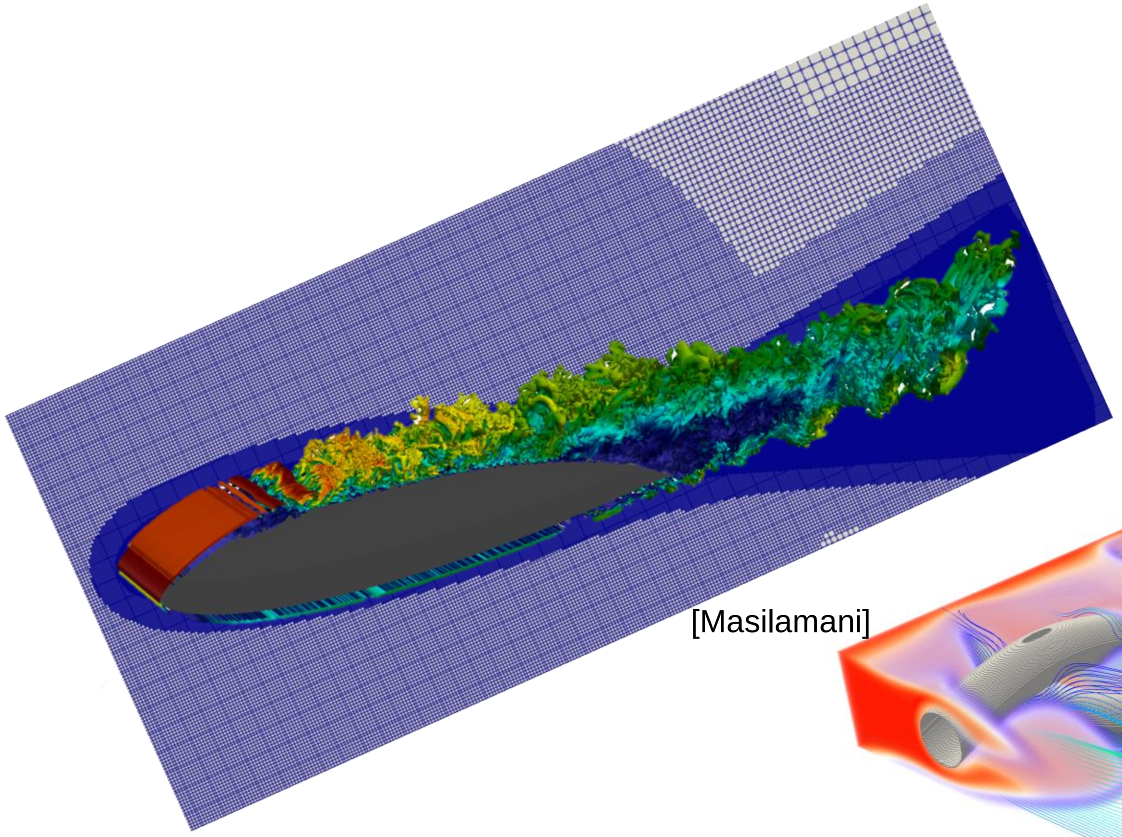
- Considering the Boltzmann Equation but only in a discrete velocity space, Mesoscopic scale
  - Simple geometric representation
- Stencil-based with two foundational algorithmic steps: stream & collide



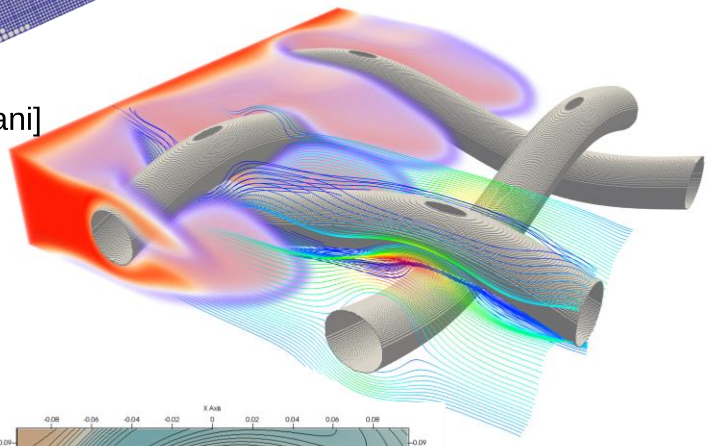
- Yields an explicit time stepping scheme for weakly compressible flows
- Few floating point operations
  - Typically memory-bound

Musubi

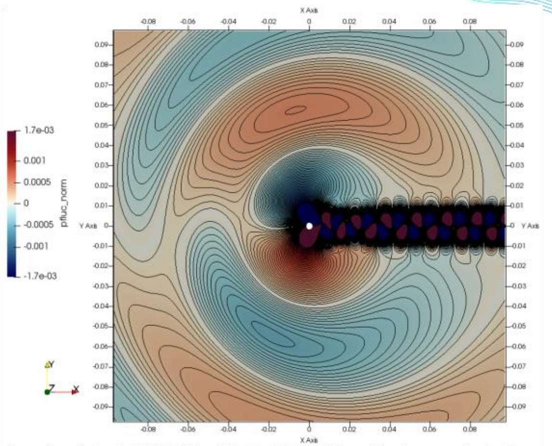
- LBM solver in Apes-Suite
- Multiple scales
- Multiple species
- Arbitrary stencil definitions



[Masilamani]



- Levelwise kernels:
  - Kernel implemented as in single-level, serial implementation
  - Communication in Halo elements outside kernel
  - Interpolation in Ghost elements outside kernel

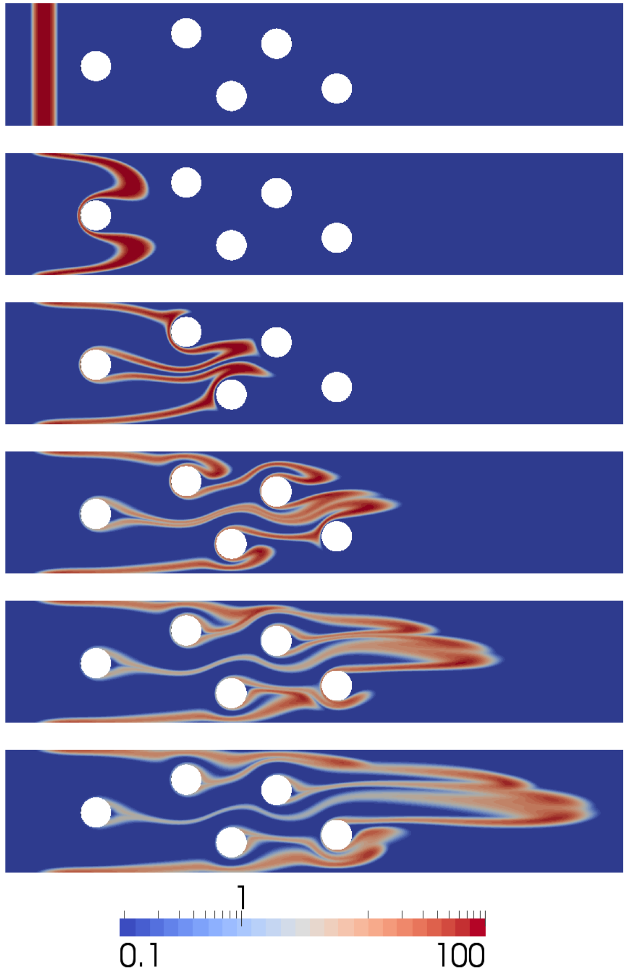


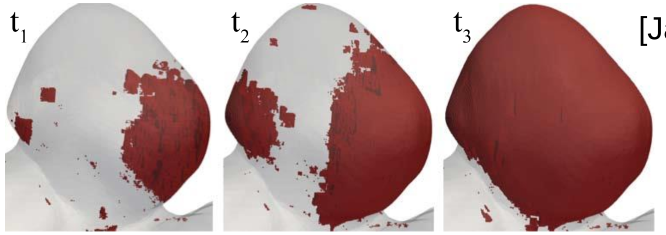
Acoustic cylinder: L=H=300D, Re=150, Ma=0.2, BGK, Normalized pressure fluctuations

Musubi Diversity

- Different Physics
  - Navier-Stokes flow
  - Maxwell-Stefan diffusion
  - Poisson equation
- Different Collision Operators
  - BGK
  - MRT
  - TRT
  - Cumulant
- Different Stencils
  - D2Q9
  - D3Q19, D3Q27

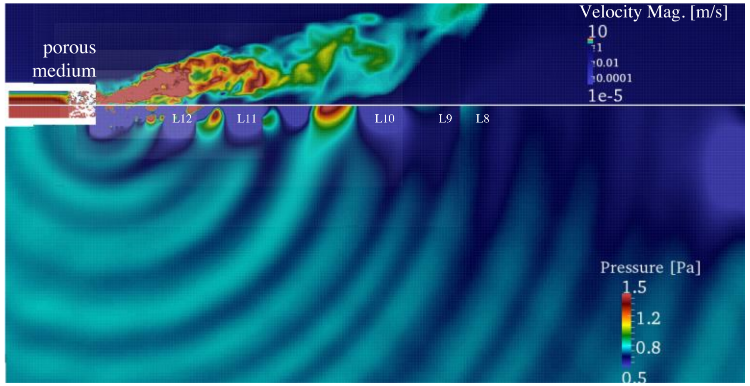
[Masilamani]





[Jain]

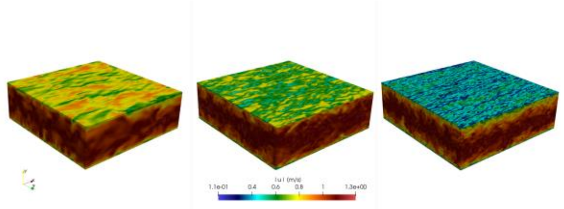
(a) Clotting process inside an aneurysm

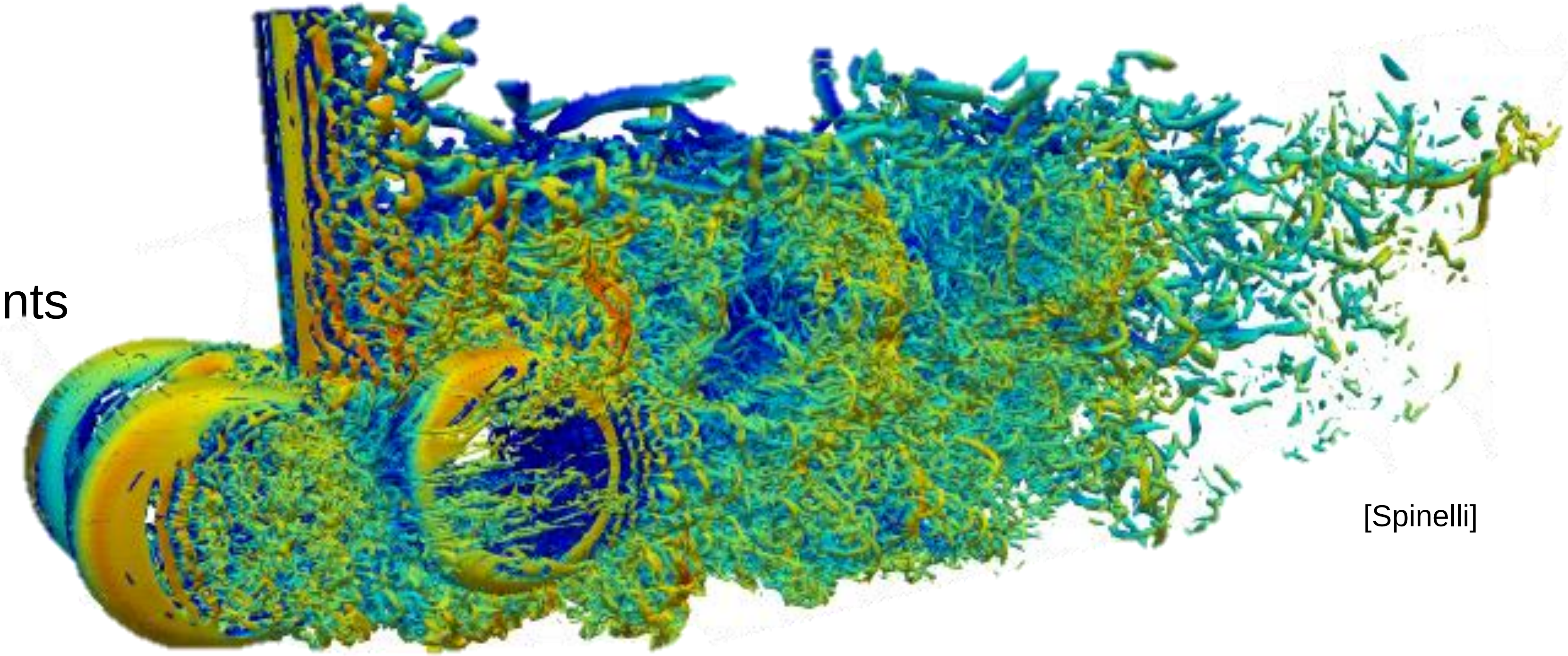


[Hasert]

Musubi Recently

- Focus on flows with high Reynolds number and curved walls
- Introducing and validating more collision operators like recursively regularized ones
- Turbulence modelling
  - Need to deal with gradients
- Turbulent wall models





[Spinelli]

PERFORMANCE

Performance Metric: MLUPS

- **L**attice **U**pdates **P**er **S**econd (LUPS) have become a common metric to express the performance for LBM

- Akin to FLOPS, as there is a given number of floating point operations per lattice update

Collision Scheme	FLOP/LUP
(plain) BGK (Q19)	160
MRT (Q19)	205

- Memory access, double buffering, indirect access (at least)

Stencil	Count	=	Bytes/LUP
D3Q19	19 x 8 + 18 x 4	=	224
D3Q27	27 x 8 + 26 x 4	=	320

Usually

Balance > 1 Byte/FLOP

Machine Balance

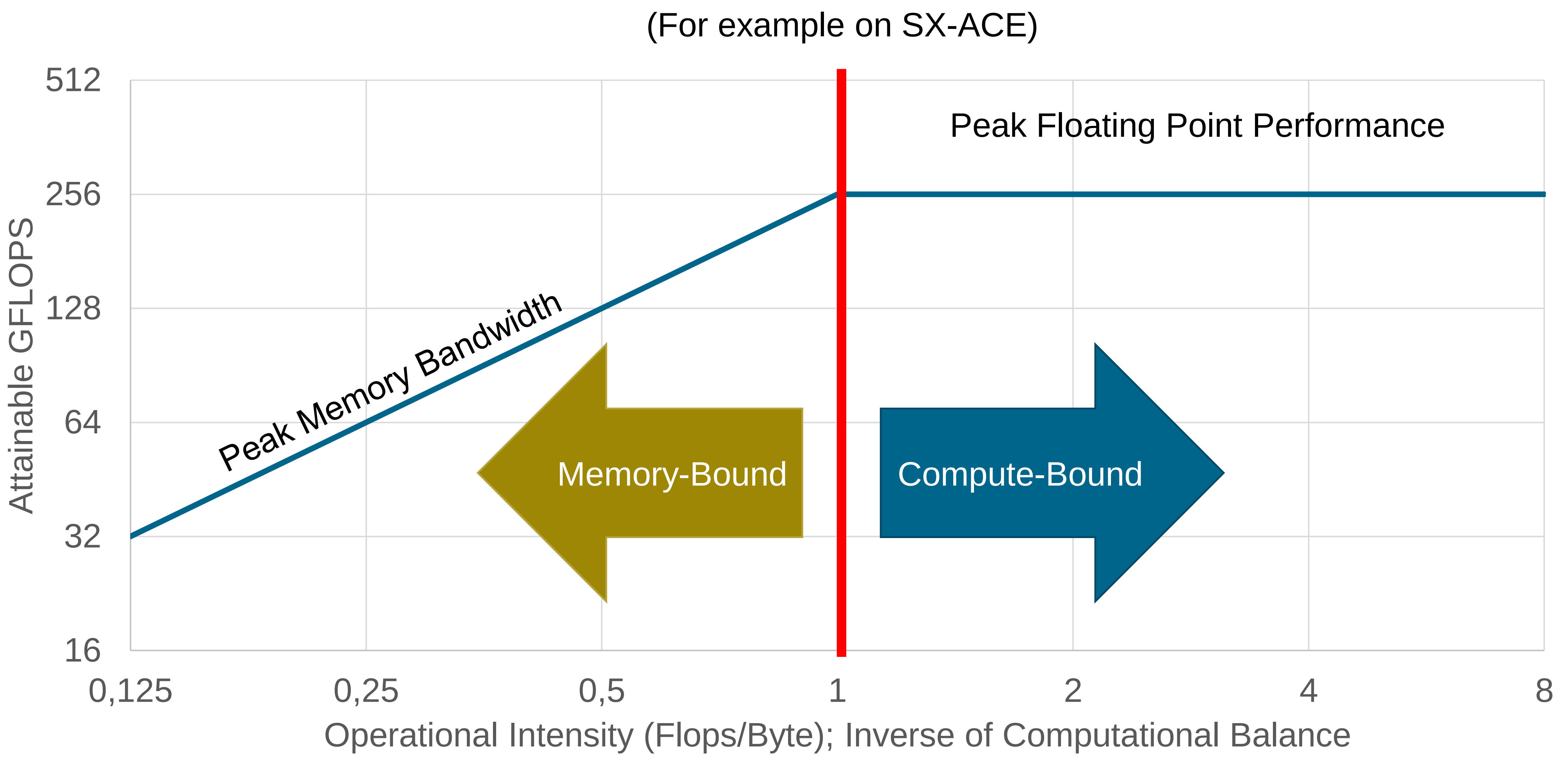
- Computational balance of the machine given by

$$\frac{\text{Maximum bandwidth}}{\text{Peak performance}}$$

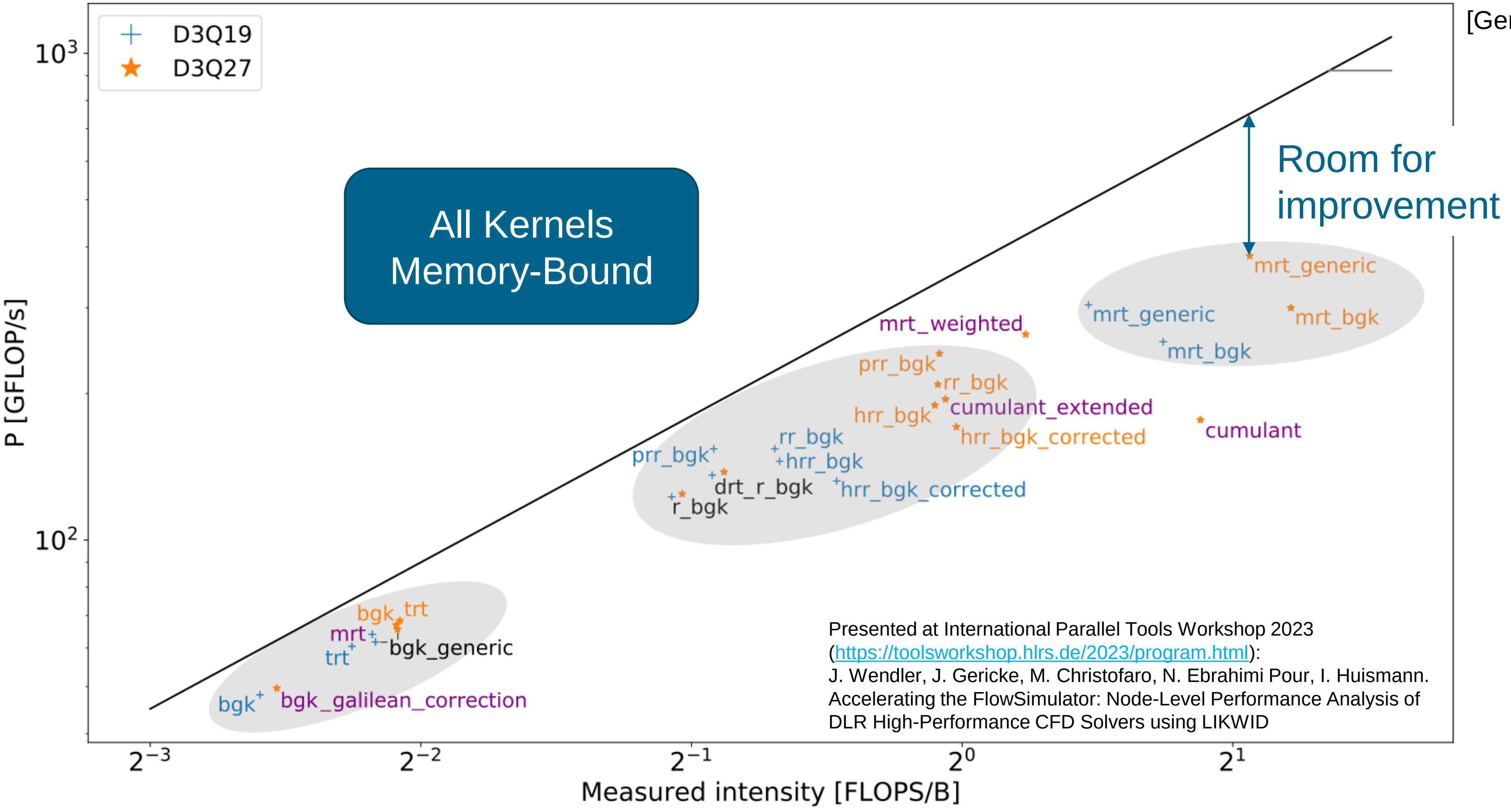
- Examples:
  - NEC SX-ACE: 256 GB/s **/** 256 GFLOPS = 1 Byte/FLOP
  - AMD Opteron 6276 (Hermit): 51.2 GB/s **/** 147.2 GFLOPS = 0.35 Byte/FLOP
  - AMD EPYC 7742 (Hawk): 190 GB/s **/** 2300 GFLOPS = 0.08 Byte/FLOP

Usually → Balance < 1 Byte/FLOP

Roofline Model



Measurement with LIKWID

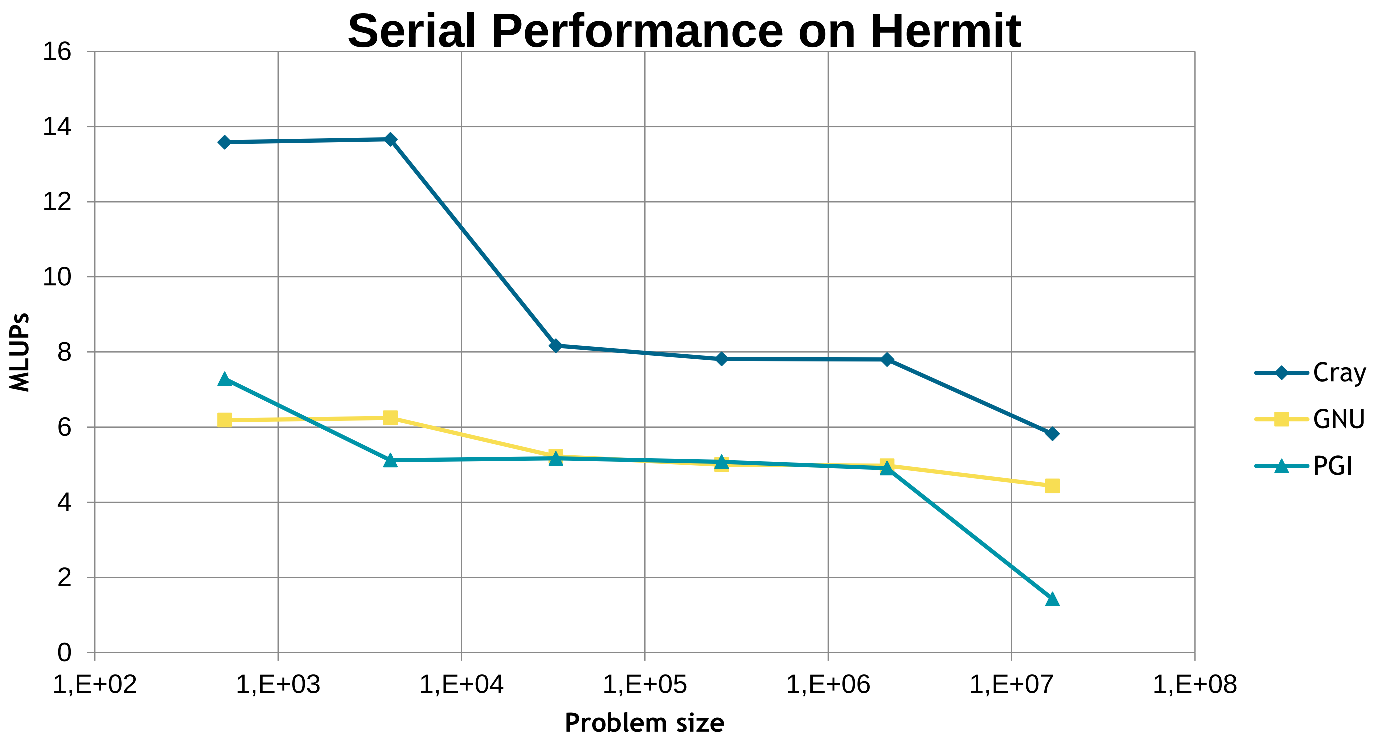


[Gericke]

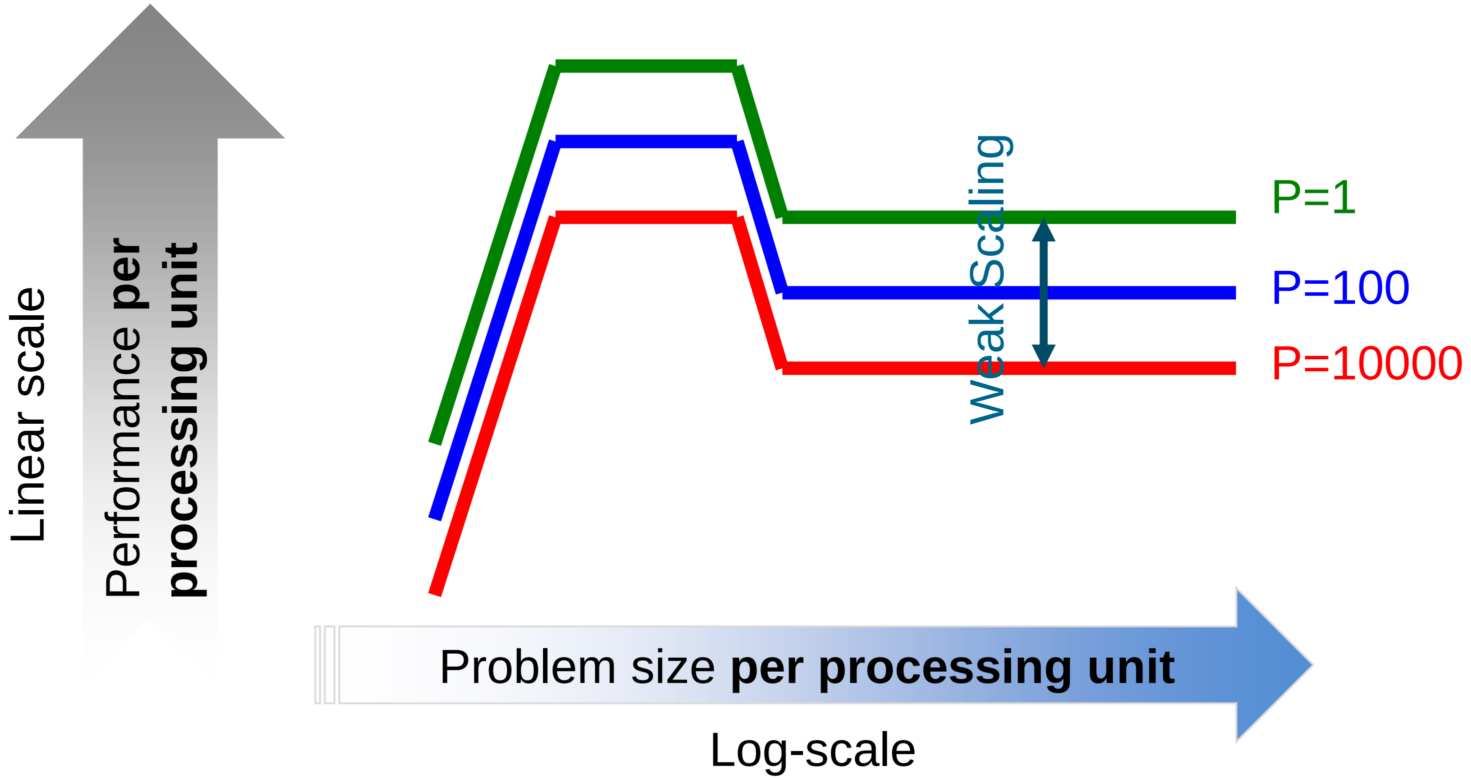
Presented at International Parallel Tools Workshop 2023 (https://toolsworkshop.hlrs.de/2023/program.html):
J. Wendler, J. Gericke, M. Christofaro, N. Ebrahimi Pour, I. Huismann. Accelerating the FlowSimulator: Node-Level Performance Analysis of DLR High-Performance CFD Solvers using LIKWID

Dependency on Problem Size

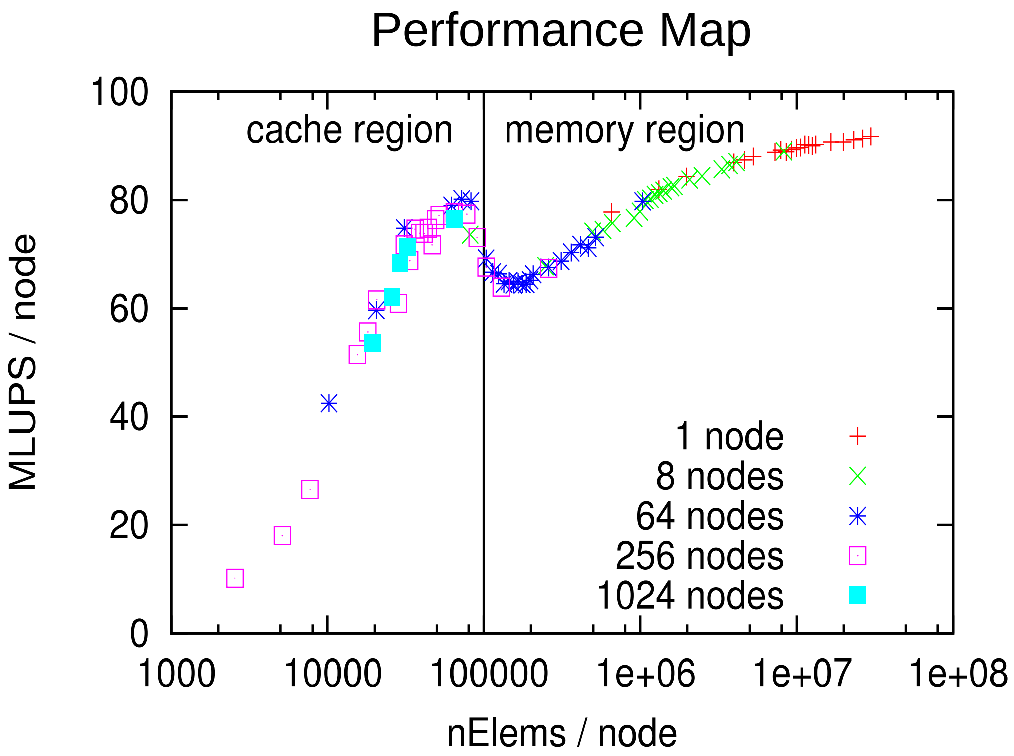
- Achieved performance usually also depends on the problem size
  - Memory hierarchy
  - Utilization of SIMD instructions, pipelines
  - Overheads
  - …

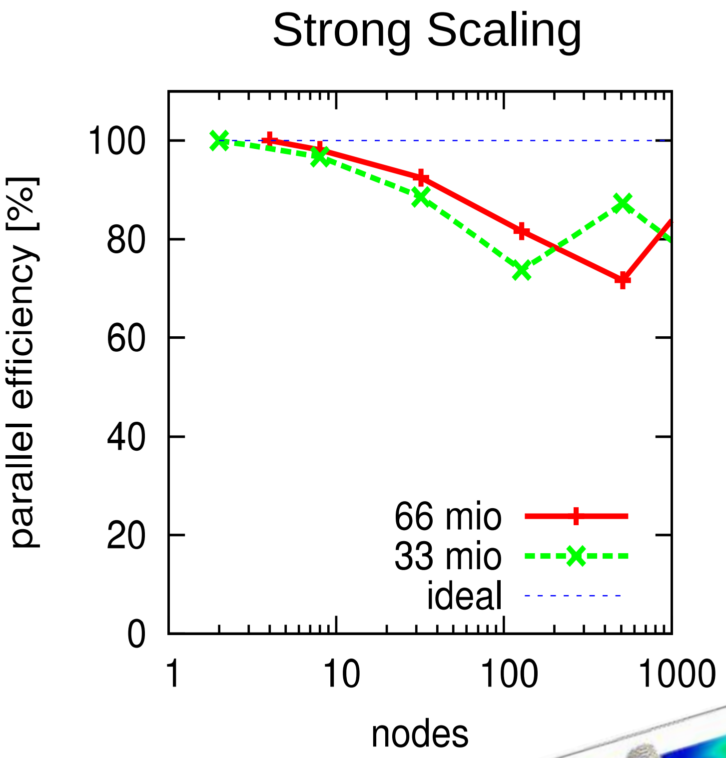


Performance Map

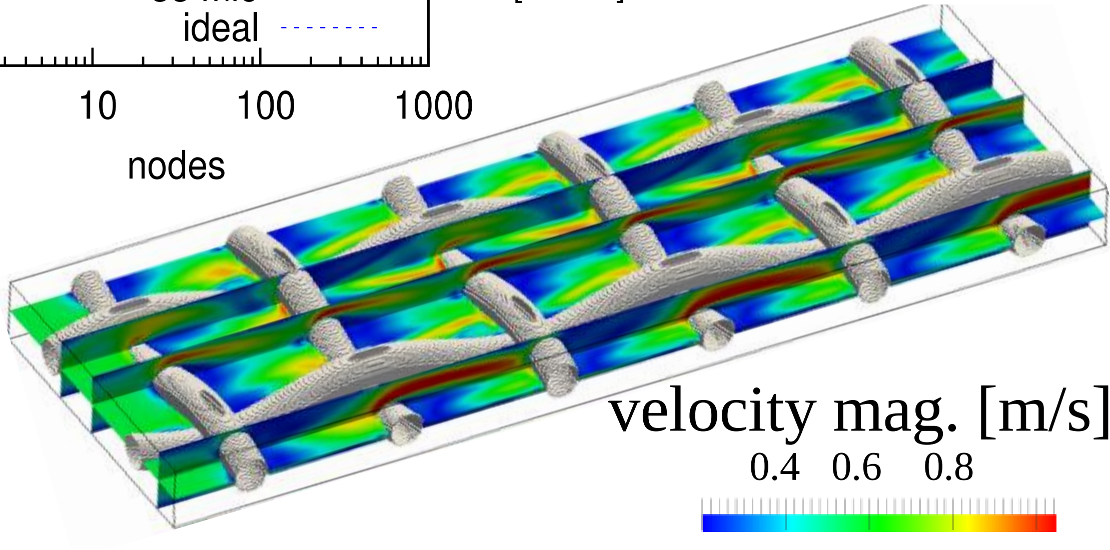


Musubi with Uniform Mesh on Hermit





[Hasert]



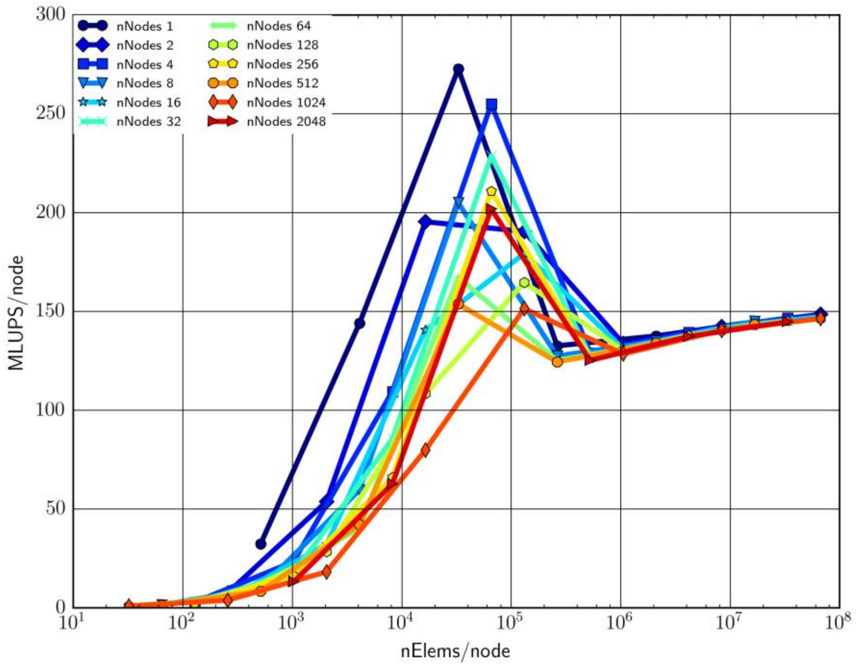
Spacer geometry for the uniform grid test case, the corresponding performance map and extracted parallel efficiency for strong scaling. The map shows the performance per node over the problem size per node when scaling beyond a node. The closer the lines to single node, the better. Measured on *Hermit*.

Haswell Systems: Different Interconnects

HLRS Hornet

(Intel Haswell, Xeon E5-2680v3, 12 Cores)

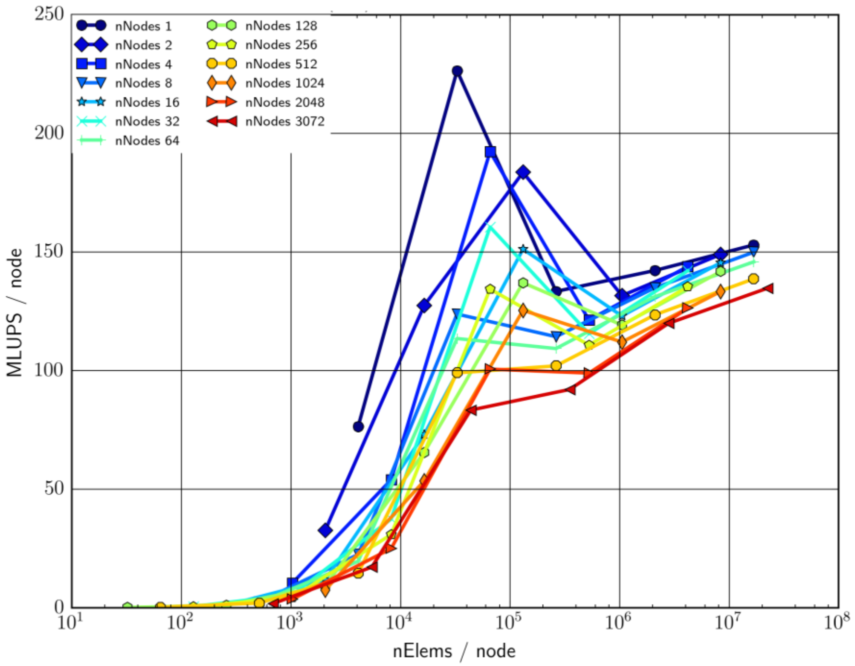
Aries Interconnect



LRZ SuperMUC2

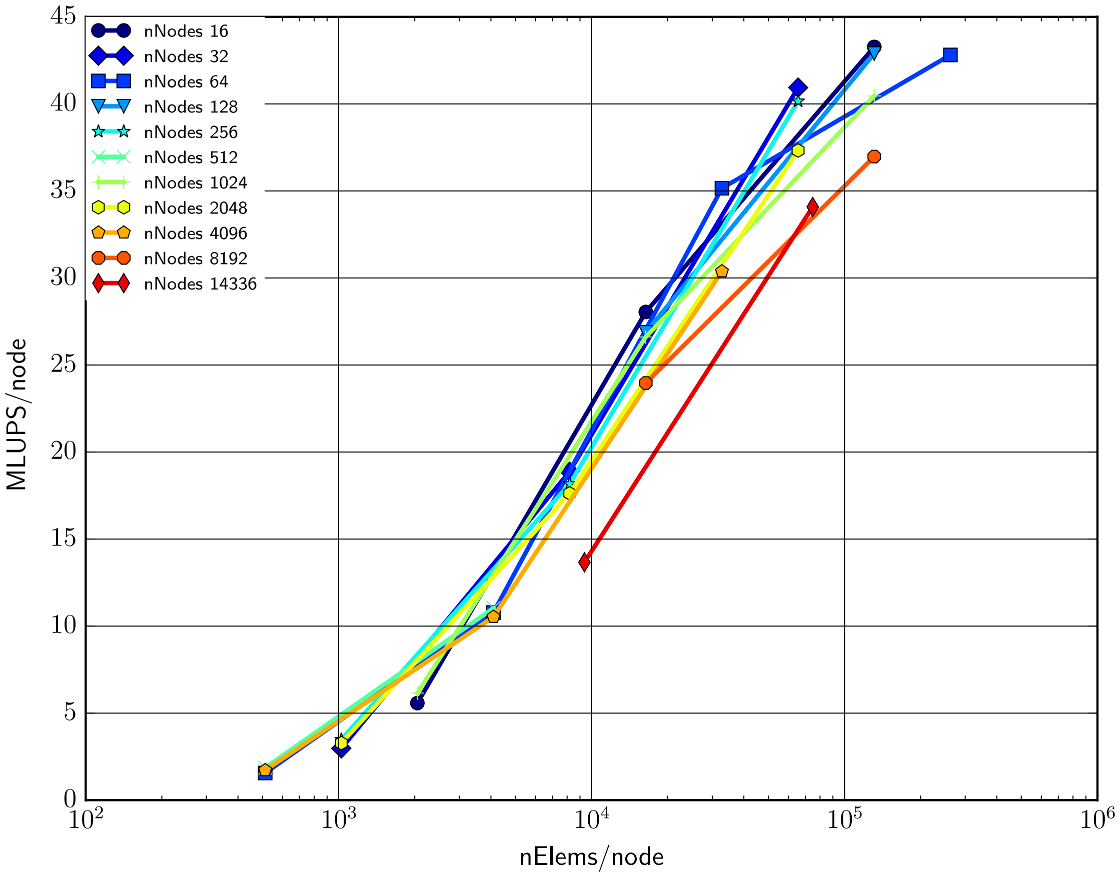
(Intel Haswell, Xeon E5-2697v3, 14 Cores)

Infiniband FDR14 Tree



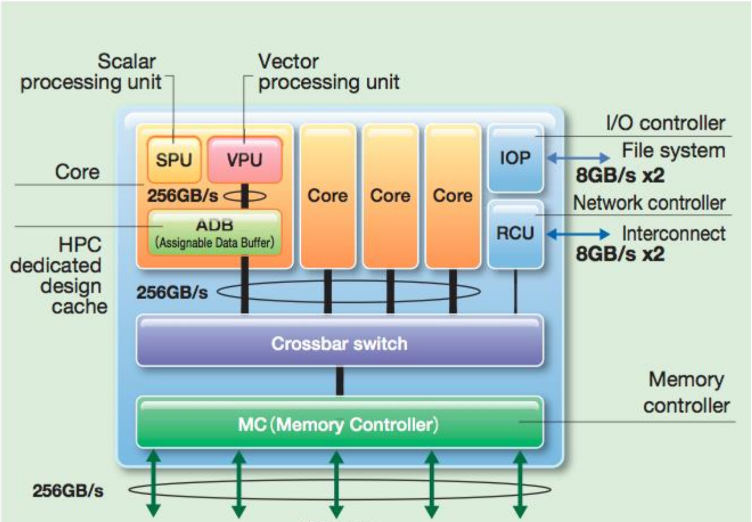
Juqueen: Limited Memory per Core

- IBM BlueGene/Q: PowerPC A2
- Per node: 16 cores / 16 GB
- Exhibited slow initialization

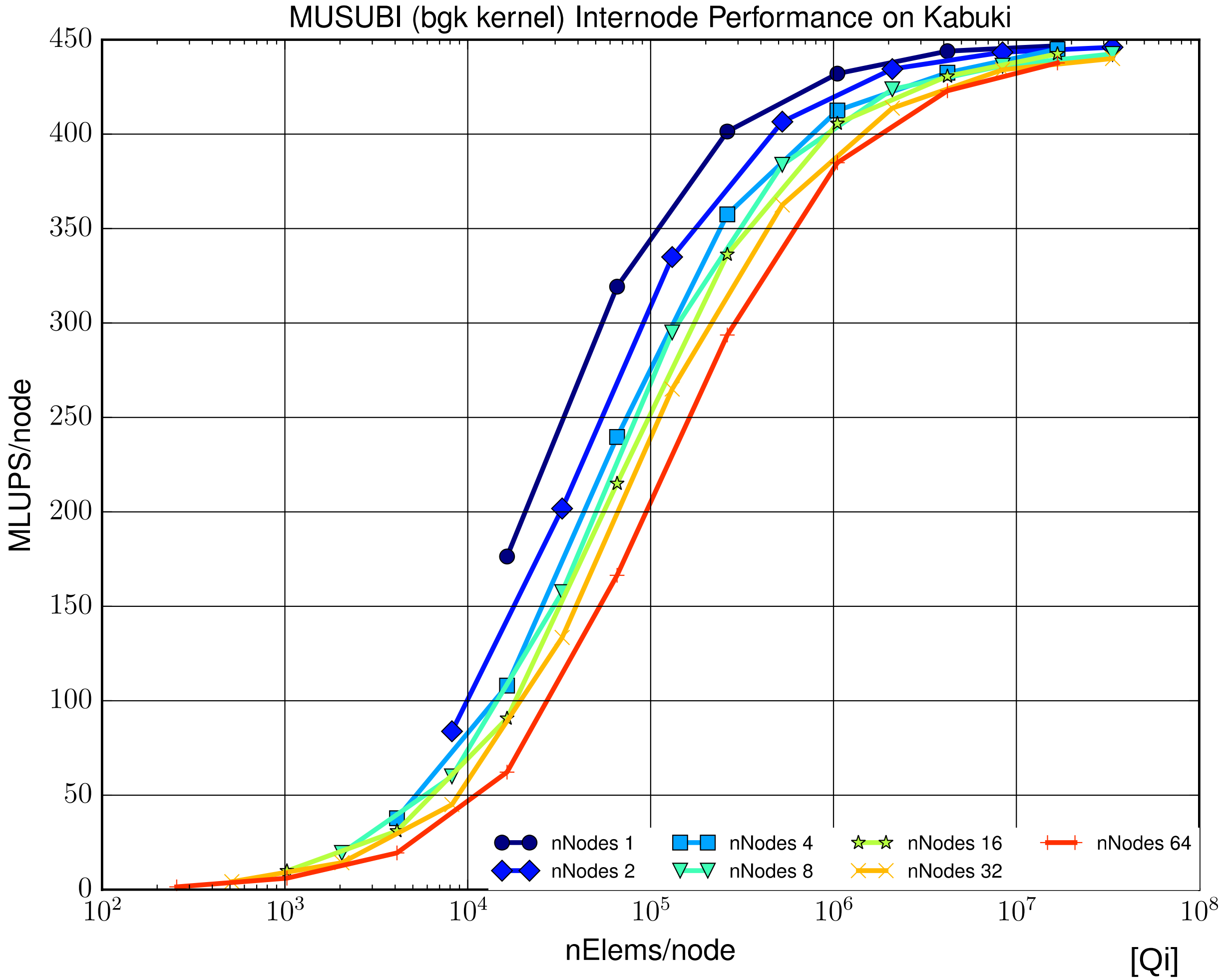


SX-ACE (HLRS Kabuki)

- Requirement of long vectors
- 4 cores / 64 GB
- High Memory Bandwidth
  - Larger relative communication fraction



[Courtesy NEC]



[Qi]

Note on Vectorization

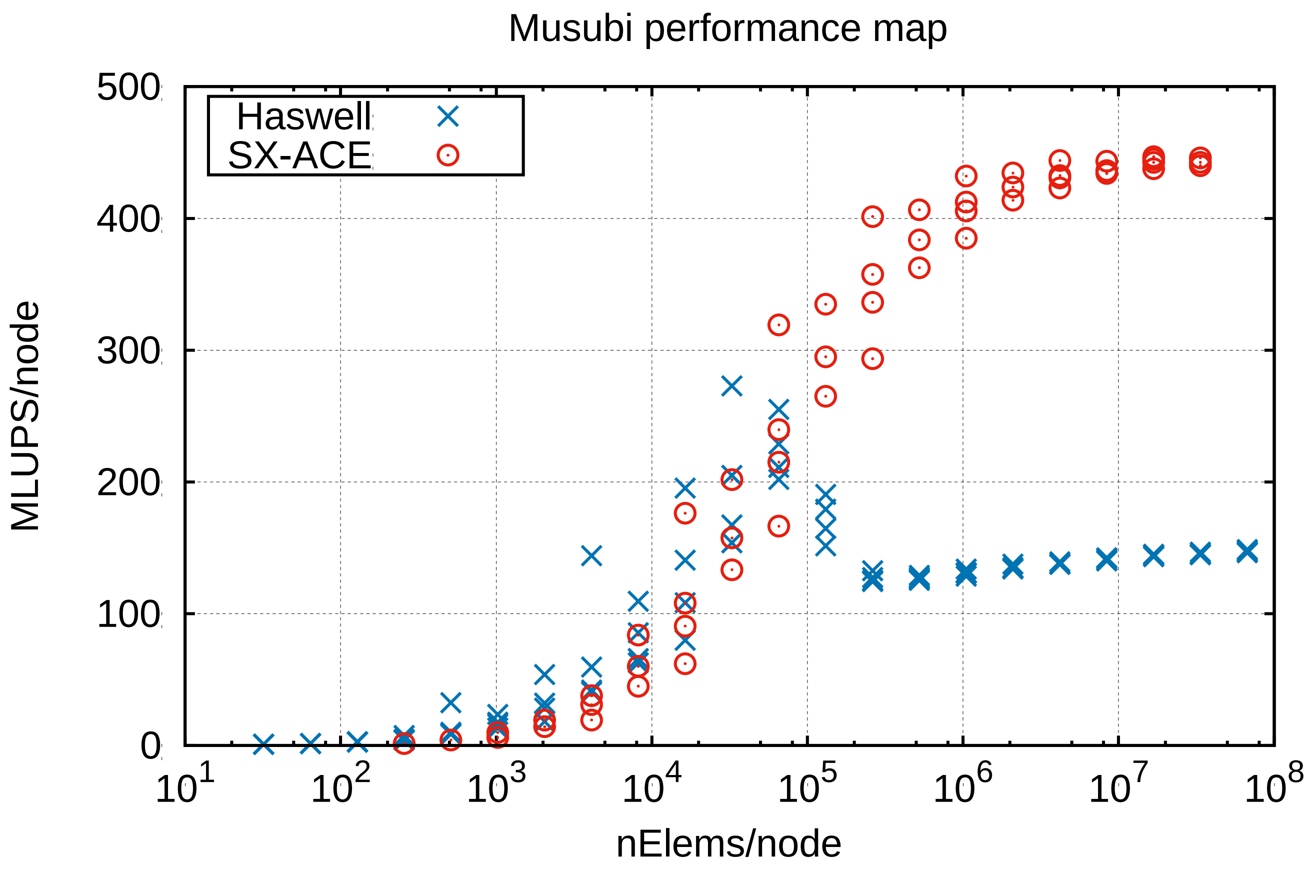
- Code mostly automatically vectorized
- Compiler sometimes needs some help

```
Do iVal = 1, nVals
        var(1) = a(iVal)
        var(2) = b(iVal)
        …
        …
        result = var(1)+var(2)+…
End do
```

```
!CDIR NODEP
Do iVal = 1, nVals
        var1 = a(iVal)
        var2 = b(iVal)
        …
        …
        result = var1+var2+…
End do
```

- Vectorization often also helps performance on scalar systems
- Blocking to limit memory, and possibly exploit caches / registers

Haswell – SX-ACE Comparison



Max Performance (MLUPS/node)	
Haswell	SX-ACE
144.8	437.6

960 GFLOPS	256 GFLOPS

Investigation

- European Project Performance Optimisation and Productivity
- *Analysis by the kind colleagues at HLRS*

- José Gracia
- Christoph Niethammer
- Stephan Walter
- Anastasia Shamakina

Used System

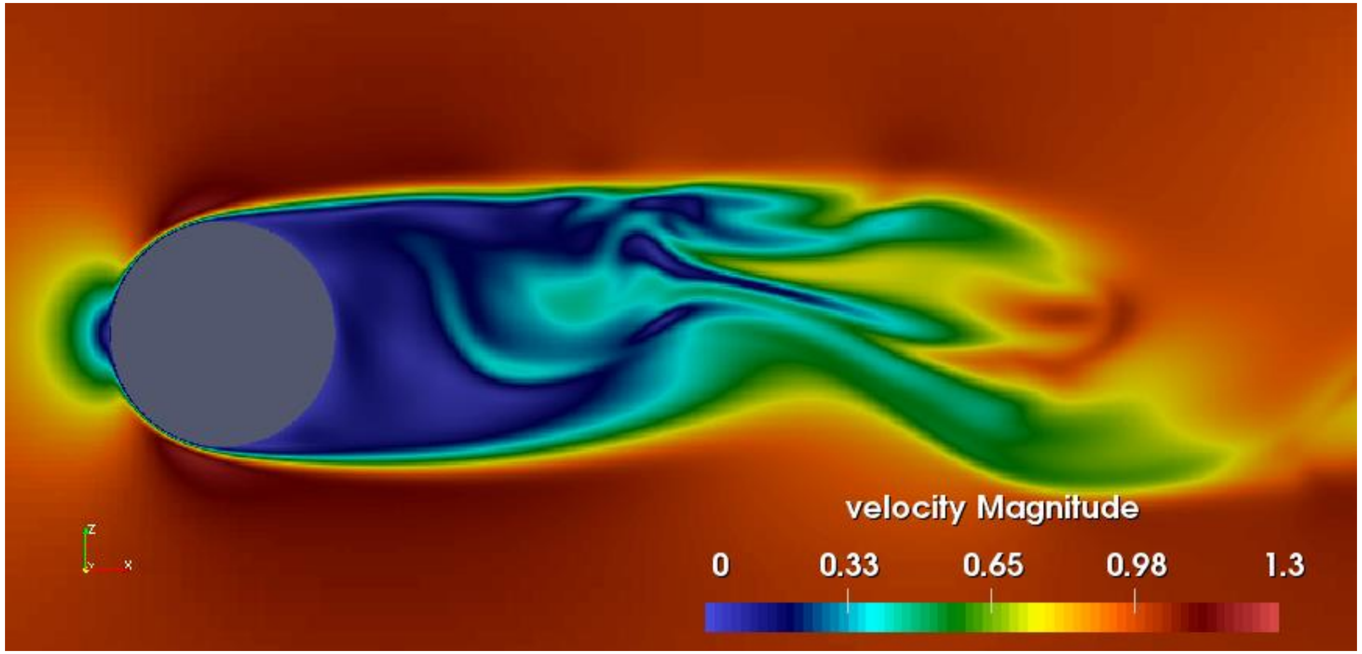
- HorUS Cluster at University Siegen

- Nodes with two Intel Xeon X5650 processors (6 cores per processor)
- Infiniband Interconnect

- Installation of analysis tools by HLRS and cluster admins
  - No action on the user side

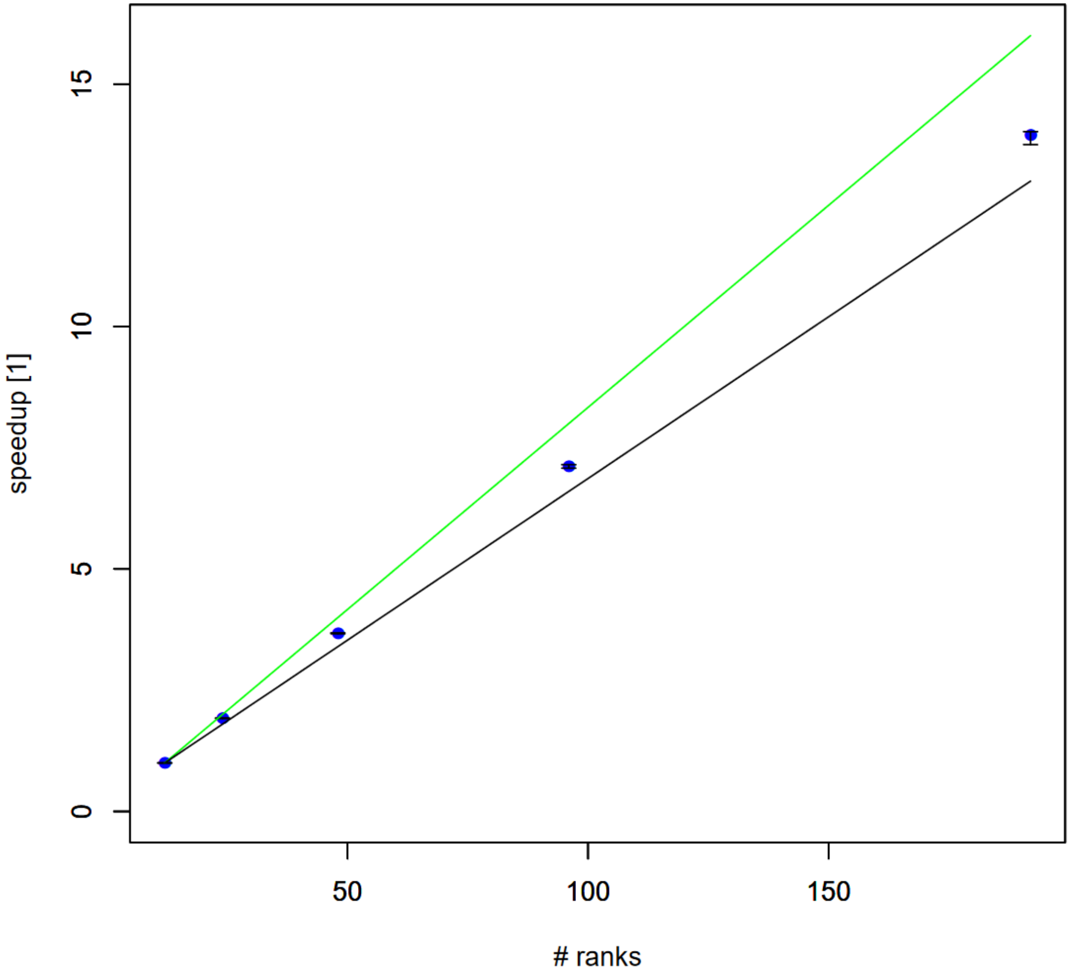
Testcase Setup

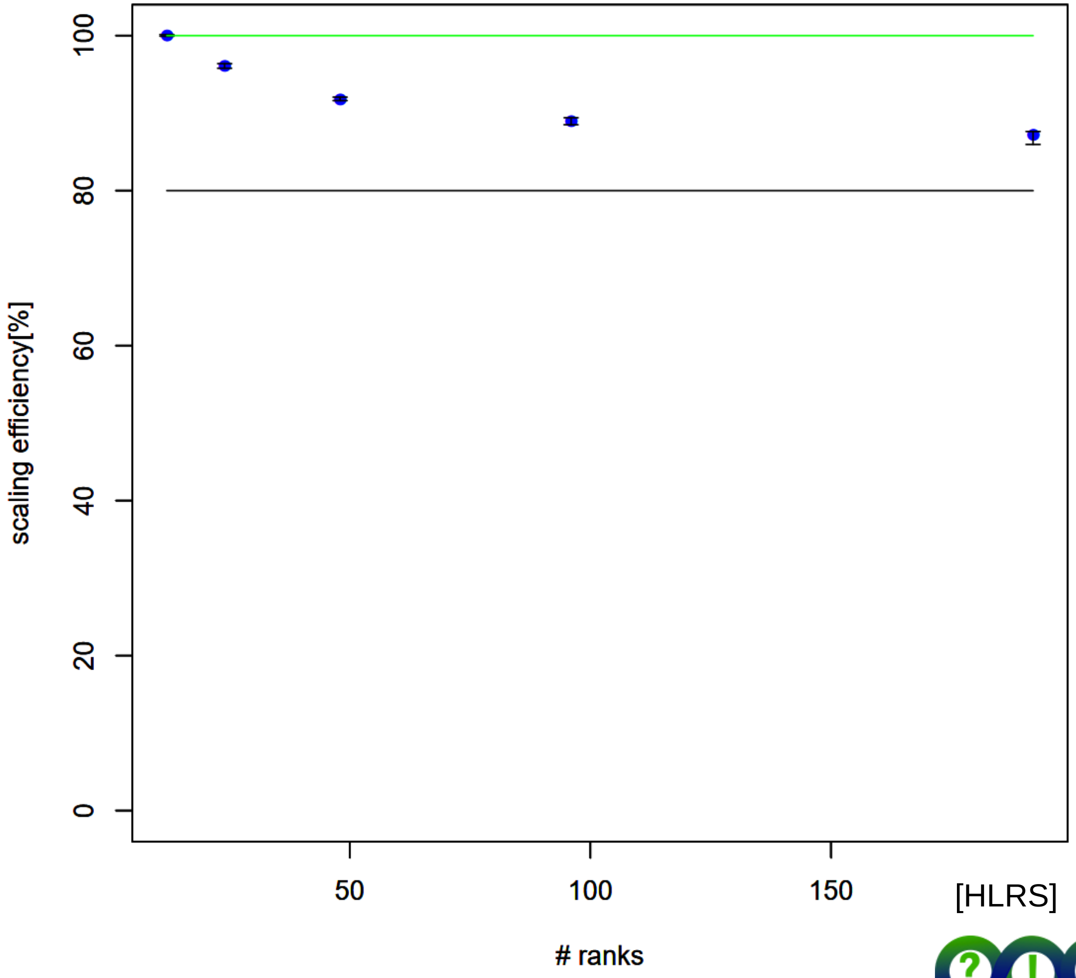
- Flow around sphere, D3Q19 stencil
  - Involves boundary conditions: q-value walls, inflow, outflow
  - Without Multilevel
  - Excluding IO, except for reading the mesh



[Qi]

Main Focus: Strong Scaling Performance

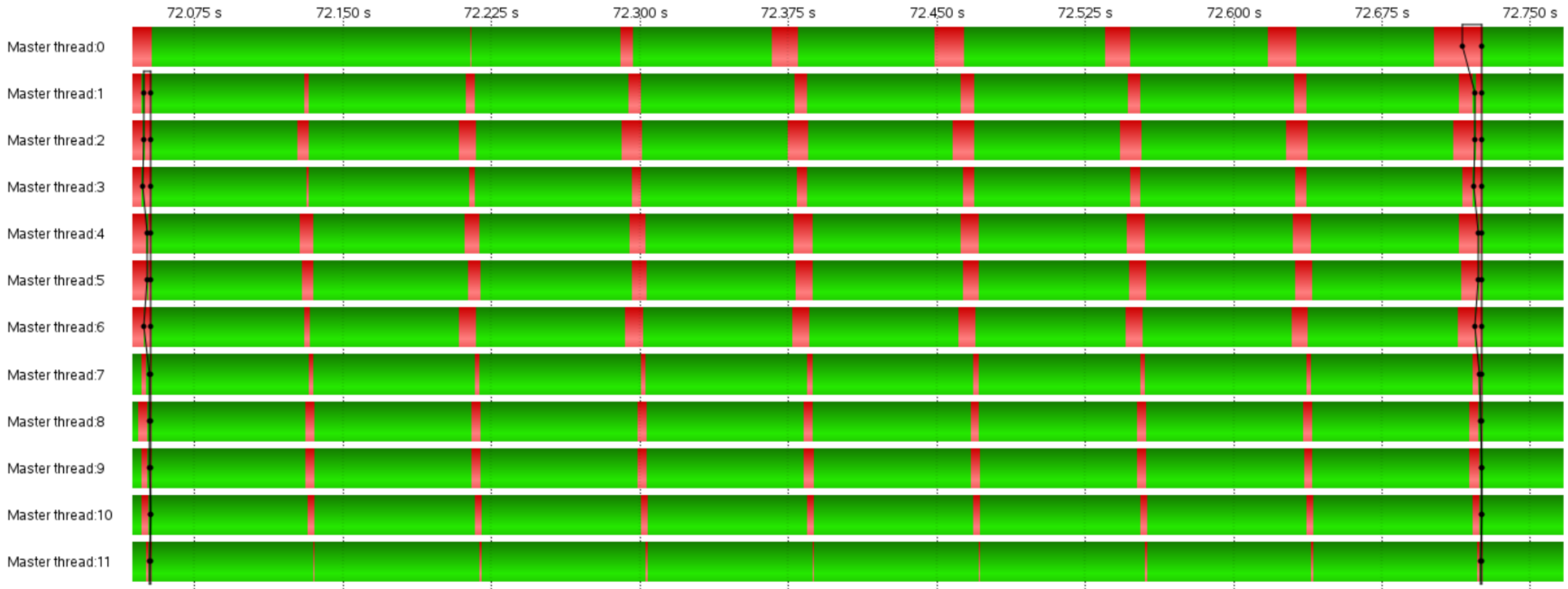




[HLRS]

Main Cause for Degradation

- Expected to see better strong scaling in this setting
- Boundary Conditions caused larger load-imbalance than expected
- Load Balance = average / maximal useful time = 86% on 192 processes

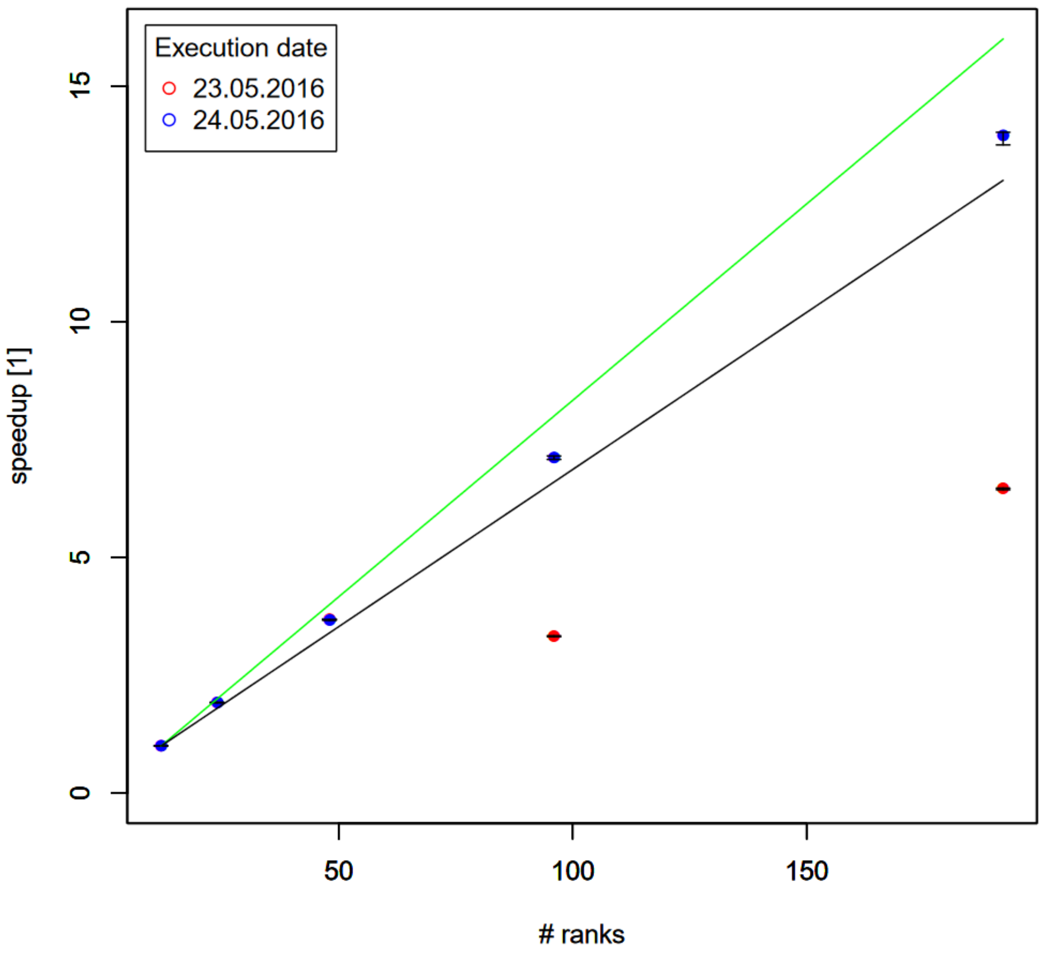


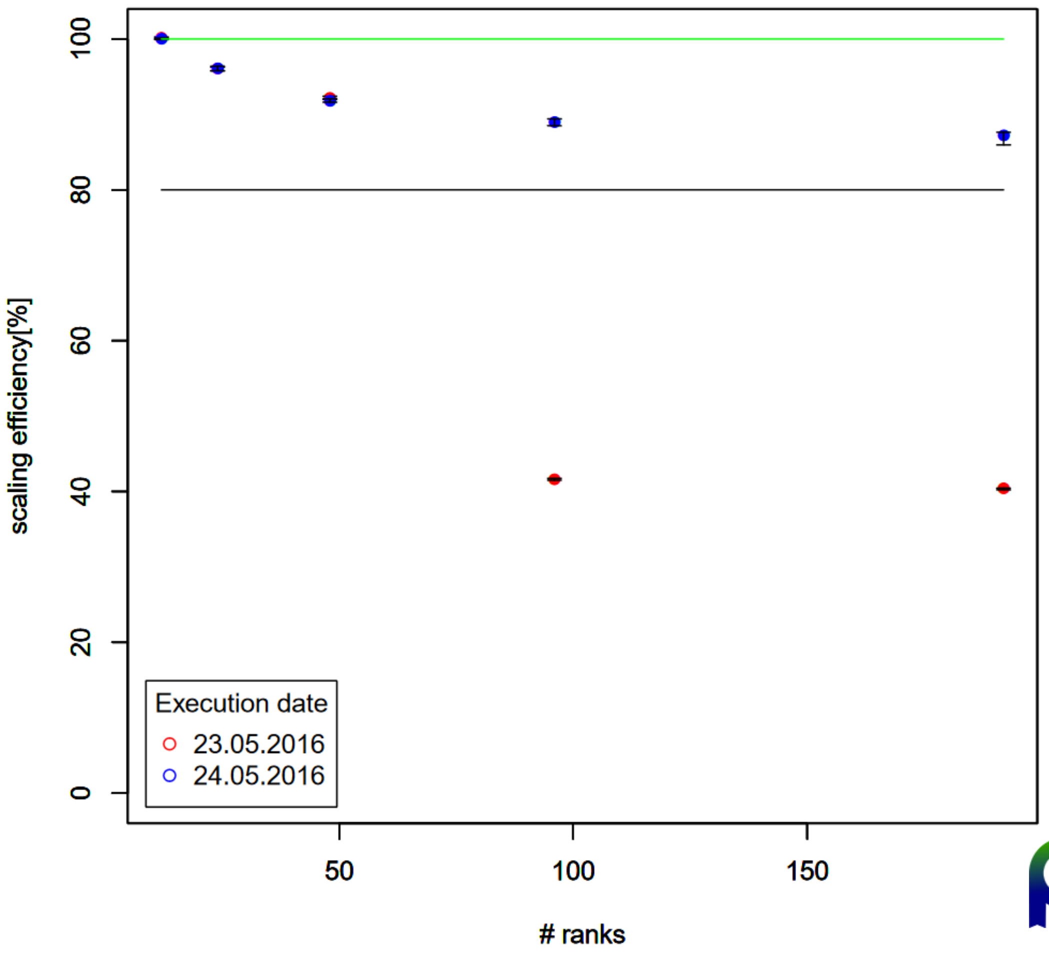
[HLRS]

Communication Pattern

- Each iteration:
  - MPI_Irecv for all neighbors on every process
  - MPI_Isend for all neighbors on every process
  - MPI_waitall: wait on all communications to complete
  - Observed up to 28 neighbors
- After M iterations (configurable):
  - MPI_Allreduce
- Available communication patterns (treelm), but all with waitall:
  - Isend_irecv
  - Isend_irecv_overlap
  - Typed_isend_ircev
  - Gathered_type

HorUS has Bad Days





[HLRS]

Recommendations for Musubi

- Take care of imbalances!
- Use non-temporal stores (SSE2 feature)
  - Implemented and improved performance by around 50%
- Overlap communication and computation
  - Can be achieved by additional indirection (not implemented)
- Avoid indirection
  - Difficult in sparse mesh

Load Balancing

- Balancing along the space-filling-curve (Chains-on-chains partitioning)

Weights

5	3	1	2	1	4	6	1	3	2	1	3	1	1	1	1	9	1	1	1	1	1	1	1	1
Σ=12					Σ=16					Σ=7					Σ=13					Σ=5				
Rank 0					Rank 1					Rank 2					Rank 3					Rank 4				

10.6 21.2 32.8 43.4

Prefix-Sum (MPI_Exscan)

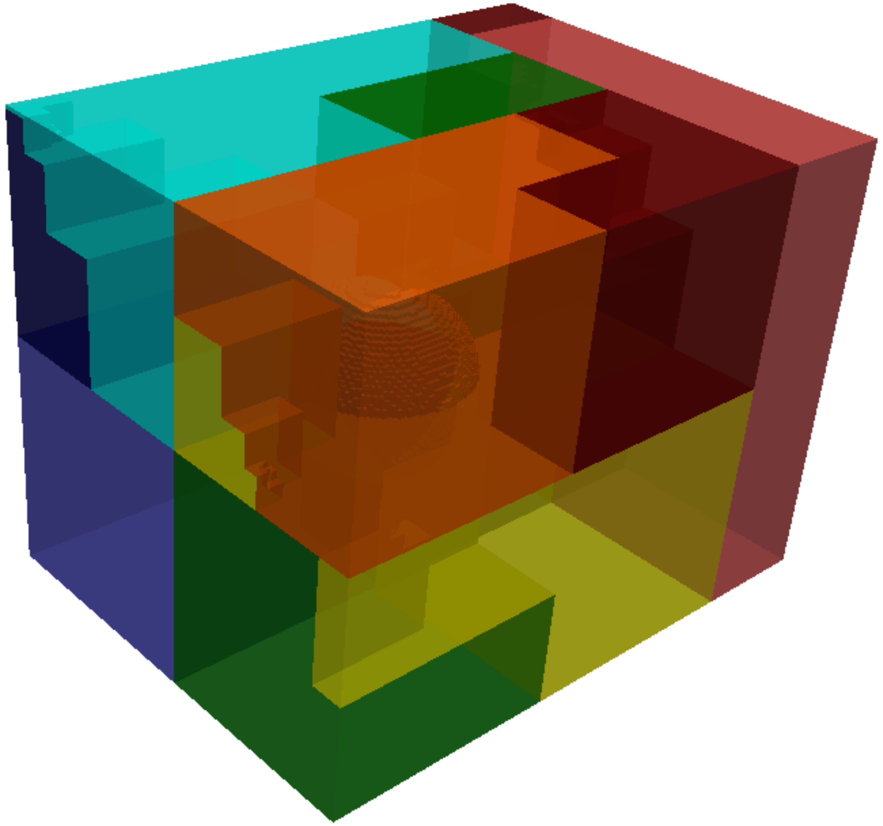
0	5	8	9	11	12	16	22	23	26	28	29	32	33	34	35	36	45	46	47	48	49	50	51	52
0	0	0	0	1	1	1	2	2	2	2	2	2	3	3	3	3	4	4	4	4	4	4	4	4
Rank 0					Rank 1					Rank 2					Rank 3					Rank 4				

New Distribution

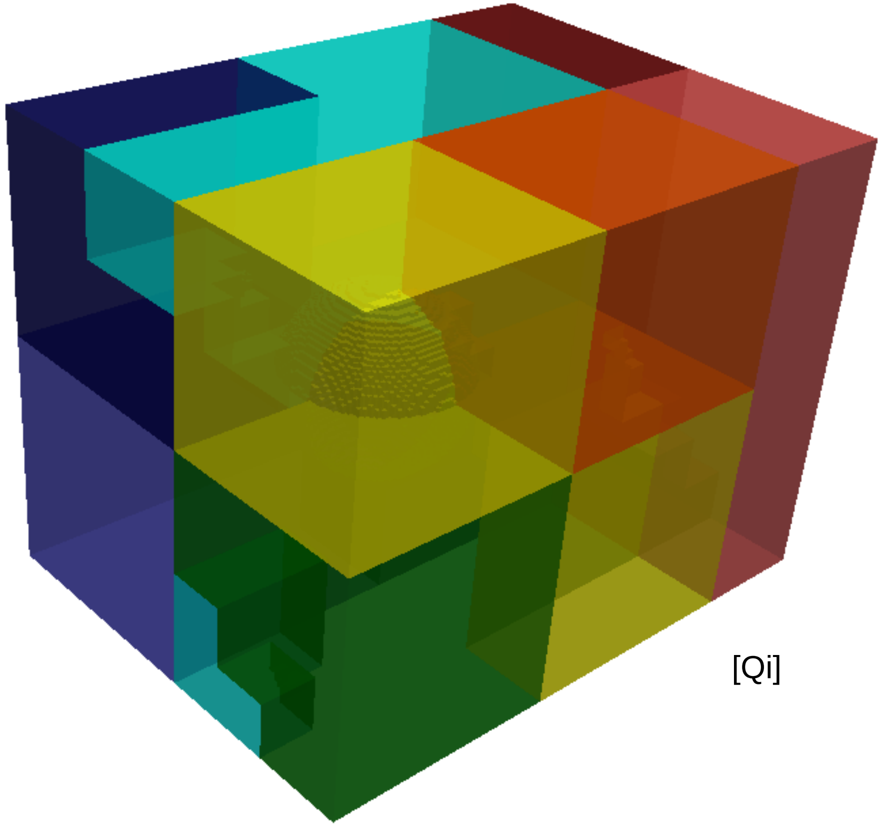
5	3	1	4	2	4	1	6	3	2	2	1	3	1	1	1	1	9	1	1	1	1	1	1	1
Rank 0				Rank1				Rank 2					Rank 3					Rank 4						

Load-Balancing Effect on Partitions

No balancing



With balancing



[Qi]

Effect on Running Time

Table 1.1: Load imbalance and simulation performance of the flow over sphere test case. Load balance is performed at the middle of the simulation.

	before balance				after balance			
procs	T_{max}	T_{mean}	imbalance	MLUPS	T_{max}	T_{mean}	imbalance	MLUPS
2	55.17	52.17	105.75%	10.21	52.07	51.48	101.15%	10.81
3	38.88	30.64	126.89%	14.43	30.60	30.33	100.89%	18.26
4	31.48	23.20	135.73%	17.81	23.95	23.24	103.09%	23.06
5	30.90	19.62	157.44%	18.19	20.54	19.71	104.21%	26.90
6	23.77	16.16	147.07%	23.57	16.91	16.55	102.15%	32.54
7	23.30	15.40	151.28%	24.02	16.33	15.68	104.11%	33.73
8	20.13	13.61	147.94%	27.71	14.33	13.93	102.86%	38.26

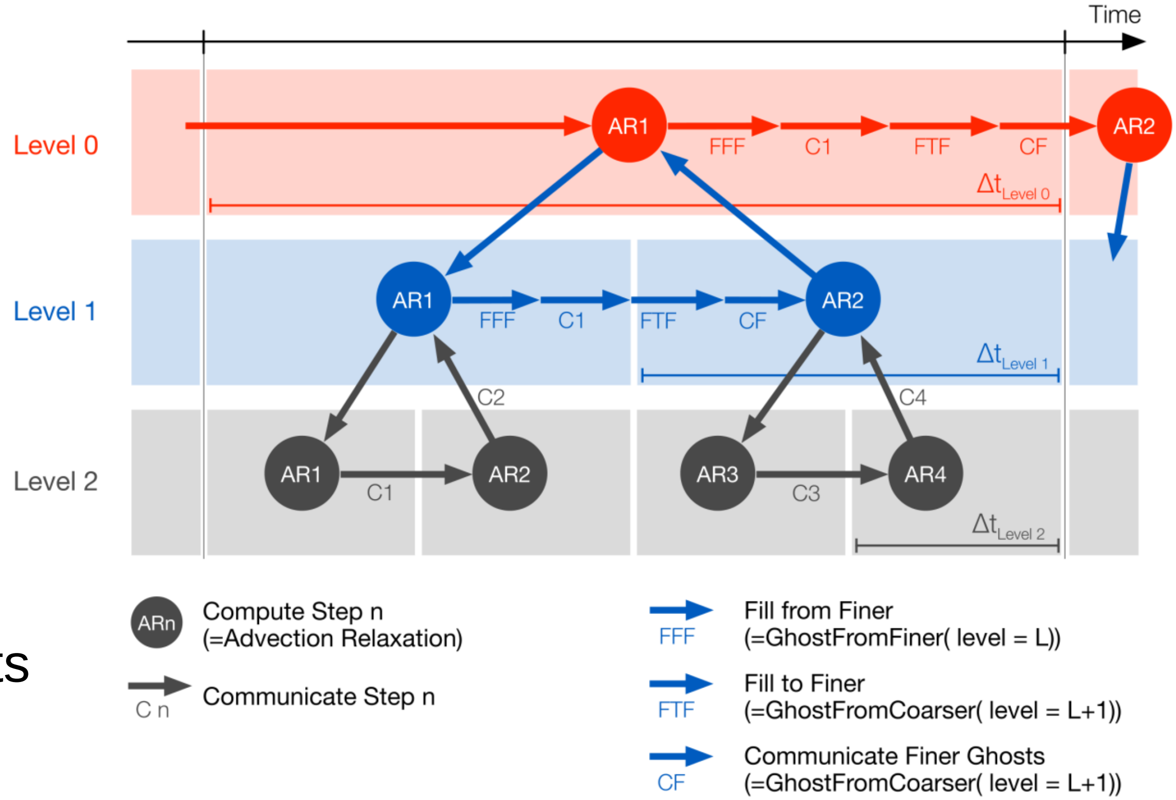
[Qi]

Issue with Balancing Multi-Level Meshes

- Balancing strategy works for **some** Multi-Level setups

- Often very limited
  - Levels serialized in LBM algorithm
  - Space-Filling-Curve „ignores“ levels

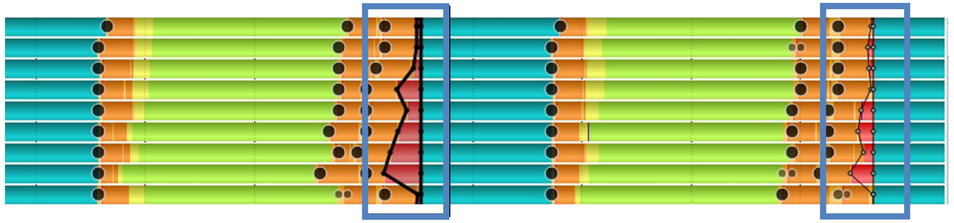
- Need to move to levelwise partitioning
  - But: breaks the concept of identifying elements without communication
  - Possibly introduce a distributed indirection



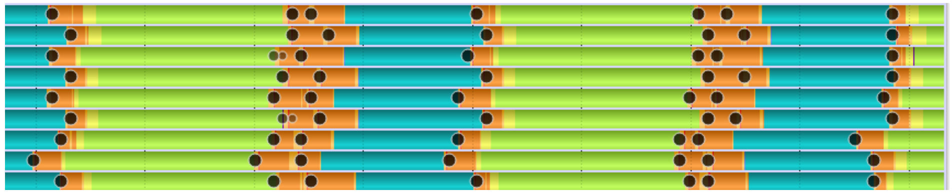
Minimizing Global Synchronization

- Health-checks and control use global reduction
  - Not critical to the actual numerical computation
  - User-defined intervals
- Introduced a scheme to use non-blocking collectives introduced in MPI-3
  - Delay the reduction completion by one check interval

Blocking `MPI_Allreduce`



Non-blocking `MPI_IAllreduce`



[Gericke]

Added Features: Reduced Computational Intensity

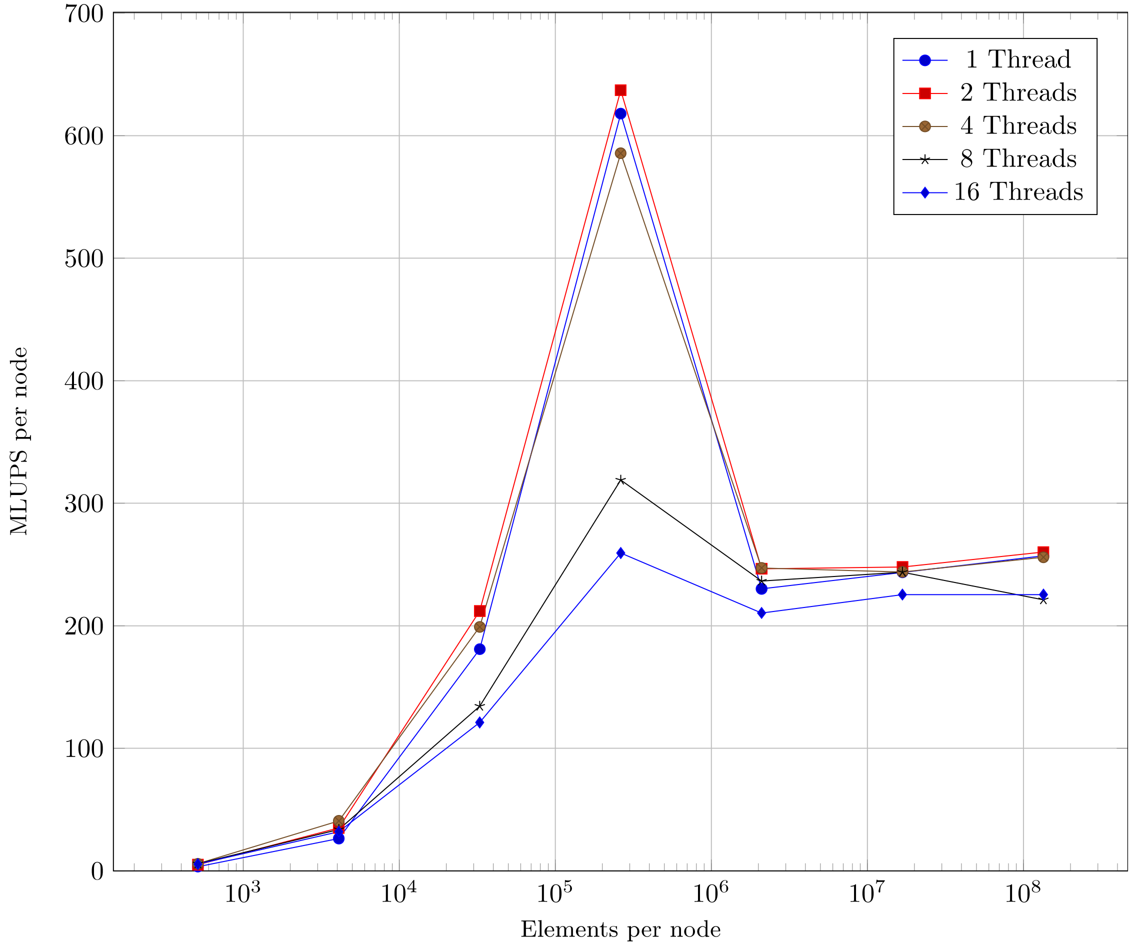
- For some features (like turbulence models) more quantities need to be kept
  - More memory accesses

- Traded performance for features deemed necessary
  - As kernels are memory-bound
  - Latest Musubi version has more features, but achieves usually lower MLUPS

Hybrid Parallelism in Musubi

- Main kernels are simple loops
  - Straight forward to add OpenMP parallel do
- Not all parts are OpenMP parallel in Musubi
- With elements ordered by space-filling-curve
  - OpenMP loop blocks behave similar to MPI partitions
- Performance measured on HAWK (2 AMD EPYC 7742 processors with 64 cores each)

BGK D3Q19 on 64 Nodes



Beyond 4 threads performance degrades.

Conclusion

- Importance to keep complete parallel workflow in mind
- Musubi as a versatile simulation tool on HPC systems
- Simple performance metrics enable categorization and comparison across implementations and machines
- POP analysis really helpful and identified regions of improvement

Impressum

Thema: **Performance Measurements for Musubi**

Datum: 2024-06-14

Autor: Harald Klimach

Institut: Institut für Softwaremethoden zur Produkt-Virtualisierung

Bildcredits: Alle Bilder „DLR (CC BY-NC-ND 3.0)“,
sofern nicht anders angegeben