COMPRESSIBLE LATTICE BOLTZMANN SOLVER
CPU BENCHMARK

MuCoSim WS 2023/24

PHASE: 2

Sudesh Rathnayake

Introduction: Summary

- Lattice Boltzmann Method(LBM) commonly adopt a structured grid in practice.

“Represented by a regular grid; points on grid are conceptually updated together. It has high spatial locality”’[1]

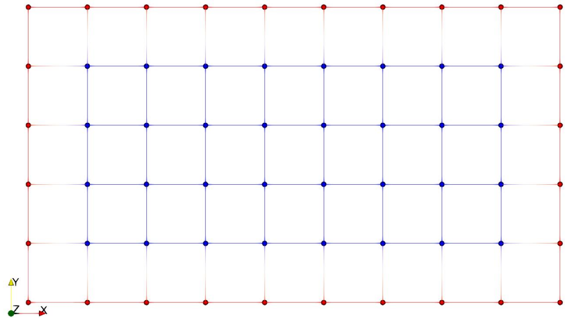


Figure 1: Exemplary structured grid

- Naturally adapted to parallel processes computing. [3].
- The “Collide and Stream” algorithms.

$$\boldsymbol{f}_i^*(\boldsymbol{x},\boldsymbol{t}) = \boldsymbol{f}_i(\boldsymbol{x},\boldsymbol{t}) + \Omega_i(\boldsymbol{x},\boldsymbol{t}) \qquad \boldsymbol{f}_i(\boldsymbol{x} + \boldsymbol{c}_i\Delta \boldsymbol{t},\ \boldsymbol{t} + \Delta \boldsymbol{t}) = \boldsymbol{f}_i^*(\boldsymbol{x},\boldsymbol{t})$$

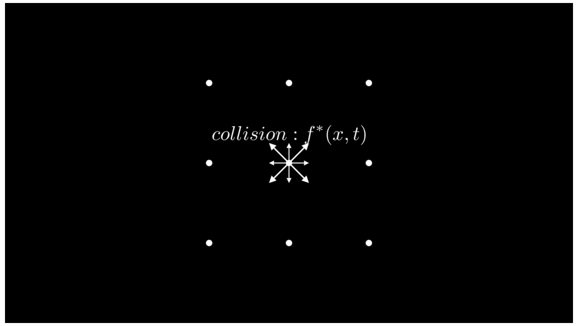


Figure 02: Collide and Stream

- Compressible LBM → $\boldsymbol{f}_i^{eq}$ using discreate entropy function by formulating a minimization problem.

- 2 populations to integrate energy equation apart from the mass, and momentum equations.
- The present solver based on Entropic Lattice Boltzma… Method (ELBM)

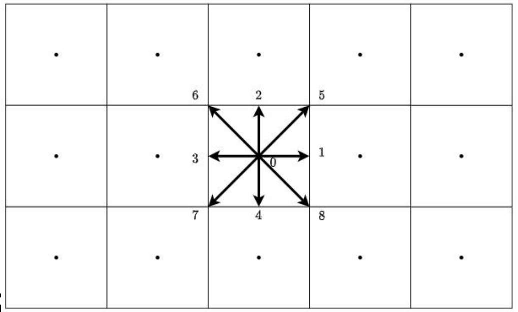


Figure 04:Lattice with D2Q9 stencil

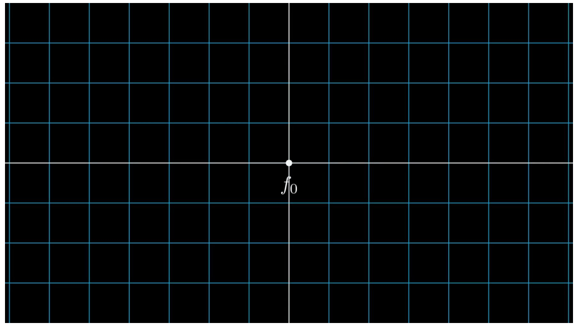


Figure 03: f and g populations in a lattice node

git@gitlab.cs.fau.de:yh54ojyn/mucosim.git

Test system & CLBM solver

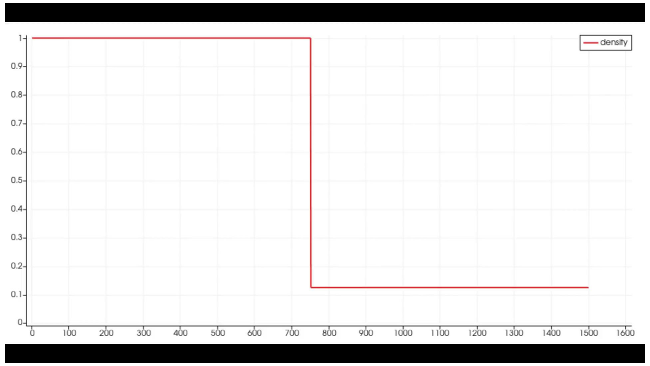


Figure 05: Results for density variation of air inside a 2D shock tube using CLBM solver

Algorithm

- Solver is based on C++ with OpenMP and build using Cmake 3.23.1.
- Input parameter file defines the simulation properties.
- For benchmarking, a 2D shock tube simulation is considered.
- Stencil configuration : D2Q9 (D2Q9 X 2 in one LB node)

Tools

- LIKWID version : 5.3.0
- LIKWID flags : likwid-perfctr -g MEM/ENERGY/MEM_DP
  : likwid-topology

Compiler

- GNU g++
- compiler flags : -O3 –march=native –mavx –ftree-vectorize

Test system

Name	Fritz
Processor	Intel Xeon Platinum 8360Y
Micro architecture	Icelake
Frequency [GHz]	2.4
Cores	72
Sockets	2 (No SMT)
NUMA domains	4
Main memory [GB]	256
Thermal design power [W]	250

Issues: Phase01

- According to single core performance doamin size of 500x500 fits to L3 cache.
- Bad scaling behaviour after the second socket.
- Total memory bandwidth and performance has a saturating trend in 1st NUMA domain.
- expf4; calculation of exponential function using math.h library is critical inside the newtonRaphson kernel.

Grouping: Function / Call Stack

Function / Call Stack	CPU Time ▼	Module	Function (I
▼ expf64	1687.306s	libm.so.6	expf64
▶ ↖ j23 ← newtonRaphson._omp_fn.17	261.509s	CLBM	j23(node&)
▶ ↖ j13 ← newtonRaphson._omp_fn.17	255.705s	CLBM	j13(node&)
▶ ↖ j33 ← newtonRaphson._omp_fn.17	131.396s	CLBM	j33(node&)
▶ ↖ func2 ← newtonRaphson._omp_fn.17	128.771s	CLBM	func2(node&, double)
▶ ↖ j22 ← newtonRaphson._omp_fn.17	128.538s	CLBM	j22(node&)
▶ ↖ j21 ← newtonRaphson._omp_fn.17	127.912s	CLBM	j21(node&)
▶ ↖ j11 ← newtonRaphson._omp_fn.17	127.344s	CLBM	j11(node&)
▶ ↖ func3 ← newtonRaphson._omp_fn.17	126.925s	CLBM	func3(node const&, double)
▶ ↖ j12 ← newtonRaphson._omp_fn.17	126.888s	CLBM	j12(node&)
▶ ↖ func1 ← newtonRaphson._omp_fn.17	121.991s	CLBM	func1(node&, double)
▶ calcGeq._omp_fn.5	95.709s	CLBM	calcGeq._omp_fn.5
▶ [No call stack information]	54.618s		
▶ stream._omp_fn.15	679.169s	CLBM	stream._omp_fn.15
▶ collision._omp_fn.9	610.130s	CLBM	collision._omp_fn.9
▶ calcQuasiEqG._omp_fn.8	505.590s	CLBM	calcQuasiEqG._omp_fn.8
▶ resetDDFshitS._omp_fn.11	475.925s	CLBM	resetDDFshitS._omp_fn.11
▶ pTensor._omp_fn.6	425.356s	CLBM	pTensor._omp_fn.6
▶ swap._omp_fn.16	422.428s	CLBM	swap._omp_fn.16
▶ pEqTensor._omp_fn.7	410.548s	CLBM	pEqTensor._omp_fn.7
▶ calcGeq._omp_fn.5	323.667s	CLBM	calcGeq._omp_fn.5
▶ calcFeq._omp_fn.4	256.802s	CLBM	calcFeq._omp_fn.4
▶ func@0x770e4	248.515s	libm.so.6	func@0x770e4
▶ func@0x1dfd4	227.869s	libgomp.so.1	func@0x1dfd4
▶ setWeights._omp_fn.3	214.170s	CLBM	setWeights._omp_fn.3

Figure 6: Vtune hotspot results..

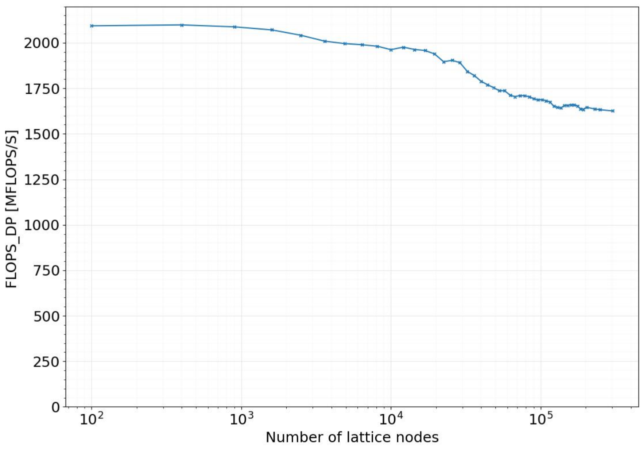


Figure 7: single core performance for fixed 2.4GHz.

BAD IMPLEMENTATION?
NEED CODE OPTIMIZATIONS?
OpenMP sheduling?

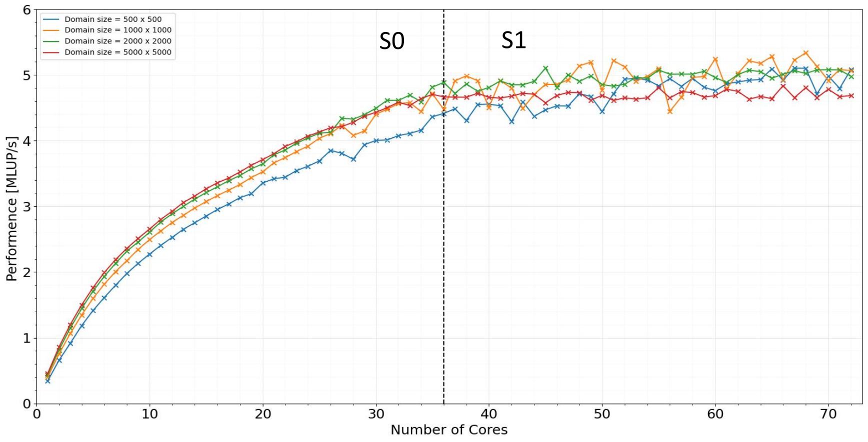


Figure 6: Performance saturation in one node. (phase:01)

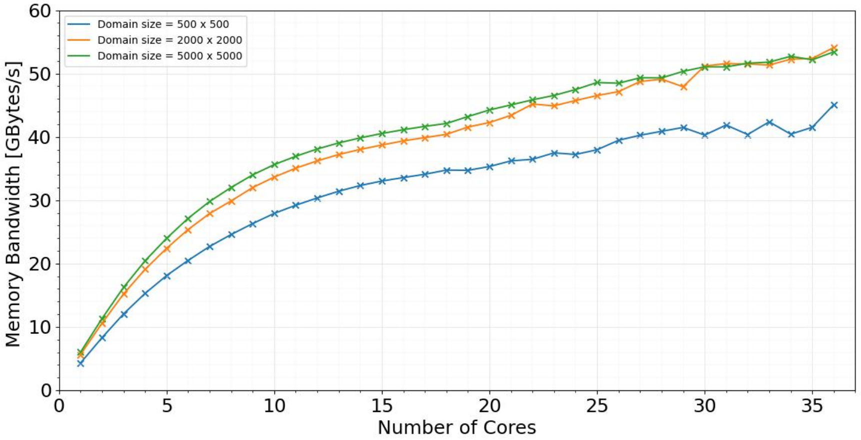


Figure 8: Memory bandwidth saturation in one node. (phase:01)

Issues: Phase01

- According to the initial profiling results; investigated possible issues in newtonRaphson kernel.
- Repeated calls for expf4 unnecessarily.
  - ✓ Since it's the hotpost, investigated the possible code optimizations.
  - ✓ Therefore, modifiedthe newtonRapson subroutines.
- Further, experimented OpenMP sheduling chunk size.
  - ✓ High OpenMP overhead for small domain sizes.
  - ✓ Tried performance vs chunk size investigations
  - ✓ Use the static sheduling.
  - ✓ Found out that peformance is better in default settings compared to settings with chunk size.
- Ready for phase to tests.

Strong scaling.

- Better strong scaling results compared to phase 01
- Performance saturation in 1st NUMA domain.
- Maximum performance of ~26 MLUP/S with using full node.

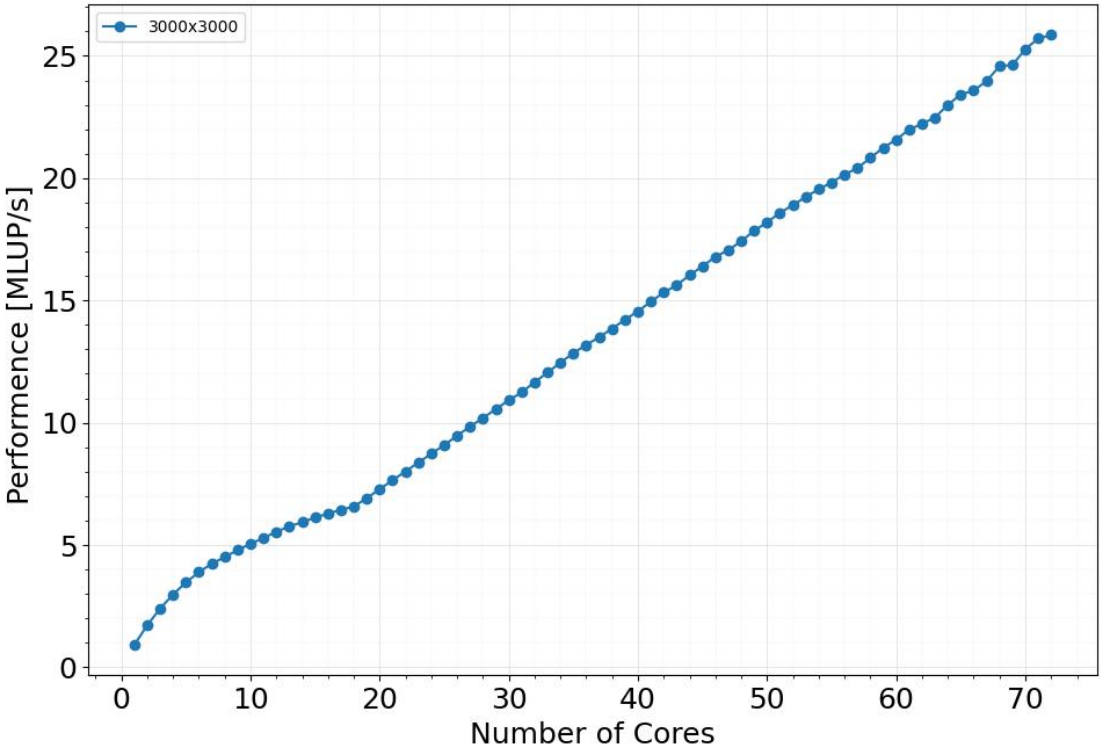


Figure 7: Performance measurements in one node at 2.4 GHz

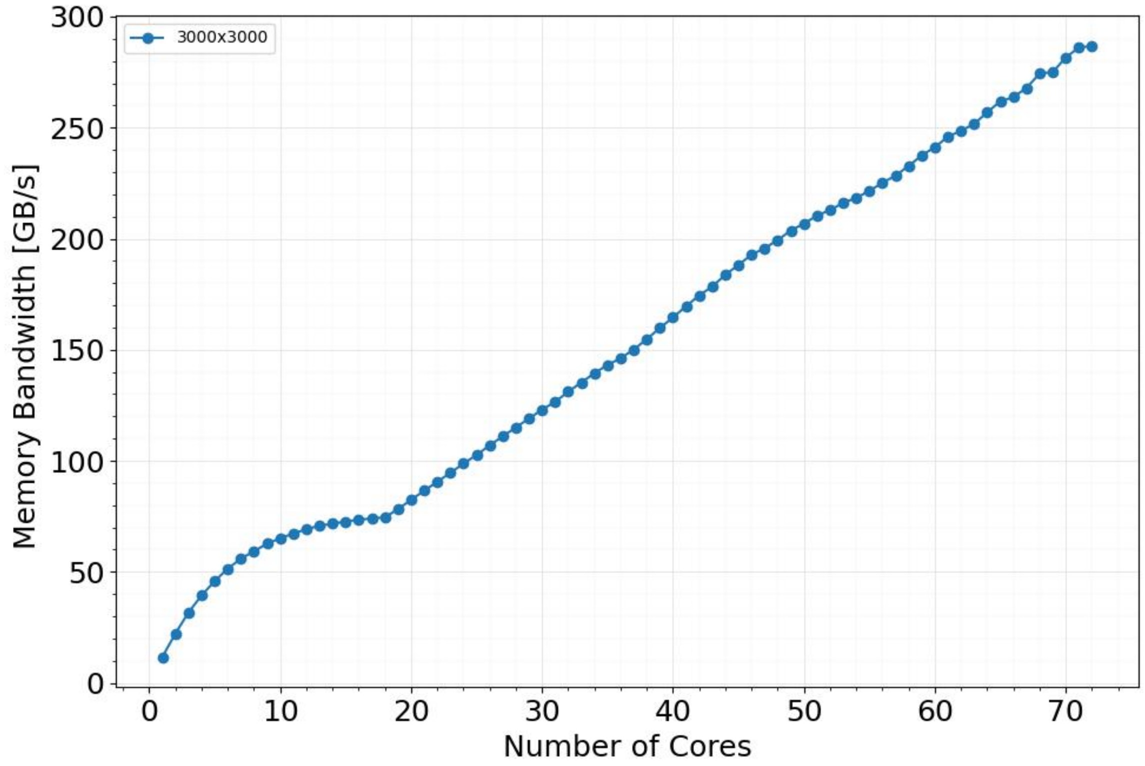


Figure 8: Memory bandwidth measurements in one node at 2.4 GHz

- Memory Bandwidth also saturates in 1st NUMA domain. (74.5 GB/s).
- single core performance is ~2x compared to phase 01.

Profiling

- Focused kernels considering energy measurements
  - ✓ Stream
  - ✓ Collision
  - ✓ newtonRaphson

Elapsed Time: 114.938s

CPU Time:	7763.840s
Total Thread Count:	72
Paused Time:	0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time	% of CPU Time
stream._omp_fn.18	CLBM	769.654s	9.9%
collision._omp_fn.12	CLBM	685.792s	8.8%
resetDDFshitS._omp_fn.14	CLBM	631.249s	8.1%
swap._omp_fn.23	CLBM	591.541s	7.6%
calcQuasiEqG._omp_fn.10	CLBM	576.830s	7.4%
[Others]	N/A*	4508.774s	58.1%

*N/A is applied to non-summable metrics.

Figure 9: Vtune Hotspot results.

Welcome | hotspots_sudesh_72.f0454.nhr.fau.de

Hotspots

Analysis Configuration | Collection Log | Summary | Bottom-up | Caller/Callee | Top-down Tree | Flame Graph | Platform

Grouping: Function / Call Stack

Function / Call Stack	CPU Time	Module	Function (Full)	Source File	Start Address
stream._omp_fn.18	769.654s	CLBM	stream._omp_fn.18		0x404ce0
collision._omp_fn.12	685.792s	CLBM	collision._omp_fn.12		0x4042d0
resetDDFshitS._omp_fn.14	631.249s	CLBM	resetDDFshitS._omp_fn.14		0x404570
swap._omp_fn.23	591.541s	CLBM	swap._omp_fn.23		0x406e60
calcQuasiEqG._omp_fn.10	576.830s	CLBM	calcQuasiEqG._omp_fn.10		0x403e40
expf64	574.681s	libm.so.6	expf64		0xfb40
calc_parameters_from_PDF._omp_fn.5	513.967s	CLBM	calc_parameters_from_PDF._omp_fn.5		0x406fe0
pTensor._omp_fn.8	472.088s	CLBM	pTensor._omp_fn.8		0x403900
pEqTensor._omp_fn.9	459.827s	CLBM	pEqTensor._omp_fn.9		0x403ba0
func@0x1dfd4	456.211s	libgomp.so.1	func@0x1dfd4		0x1dfd4
calcFeq._omp_fn.6	312.084s	CLBM	calcFeq._omp_fn.6		0x403380
calcGeq._omp_fn.7	307.521s	CLBM	calcGeq._omp_fn.7		0x403520
allfuncs	305.787s	CLBM	allfuncs(node const&, double, double*)		0x409d50
newtonRaphson._omp_fn.24	297.915s	CLBM	newtonRaphson._omp_fn.24		0x409fb0
setWeights._omp_fn.4	269.078s	CLBM	setWeights._omp_fn.4		0x403280
calcRelaxationFactors._omp_fn.11	222.741s	CLBM	calcRelaxationFactors._omp_fn.11		0x4041b0
pow	171.325s	libm.so.6	pow		0xf3f0
func@0x1de24	52.950s	libgomp.so.1	func@0x1de24		0x1de24
func@0x401bb0	25.340s	CLBM	func@0x401bb0		0x401bb0
func@0x770e4	21.810s	libm.so.6	func@0x770e4		0x770e4
createFlags._omp_fn.0	8.784s	CLBM	createFlags._omp_fn.0		0x402d20
func@0x1ca74	6.797s	libgomp.so.1	func@0x1ca74		0x1ca74

Figure 10: Vtune call stack

- Now the stream and collision kernels plays a critical role as in standard LBM schemes.
- According to literature there was a strong focus for the root finding scheme considering compressible LBM performance [4].

Energy and power measurements; main

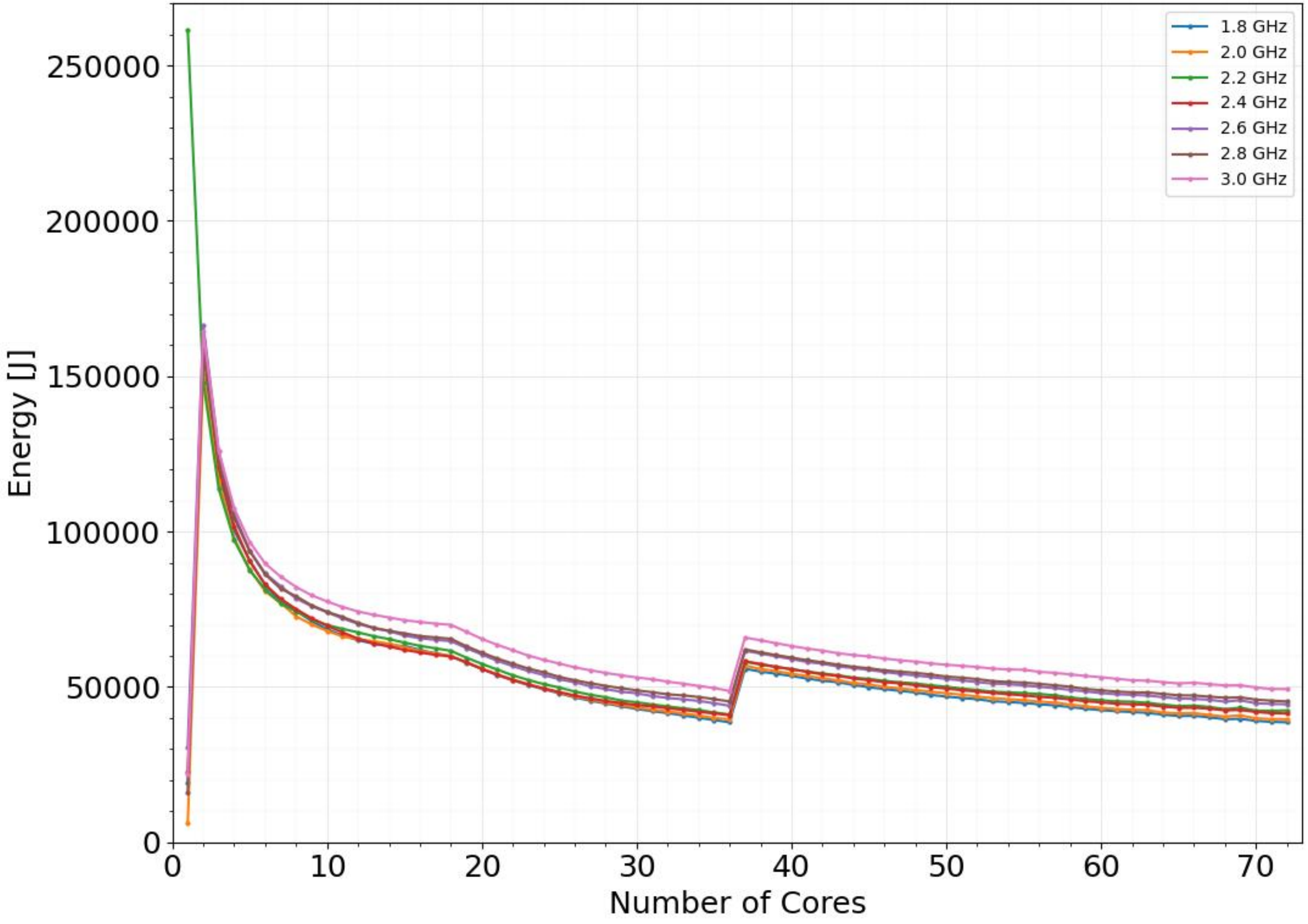


Figure 10: Energy measurements in one node

- An unusual behaviour can be observed at single core power measurment.
  - ✓ This behaviour persists for serveral data samples
  - ✓ However, for 2.2 GHz the intended energy measurements can be obtained.
- The energy values have a saturating optimum point at 1st NUMA domain before it reaches second optimum point at 1st socket.
- Sudden energy jump when moving to second socket before decrase in a quite linear fashion in second socket.
- The frequencies (GHz) 1.8, 2.0, 2.2 and 2.4 shows the lowest CPU energy measurements in first socket.

Energy and power measurements; main

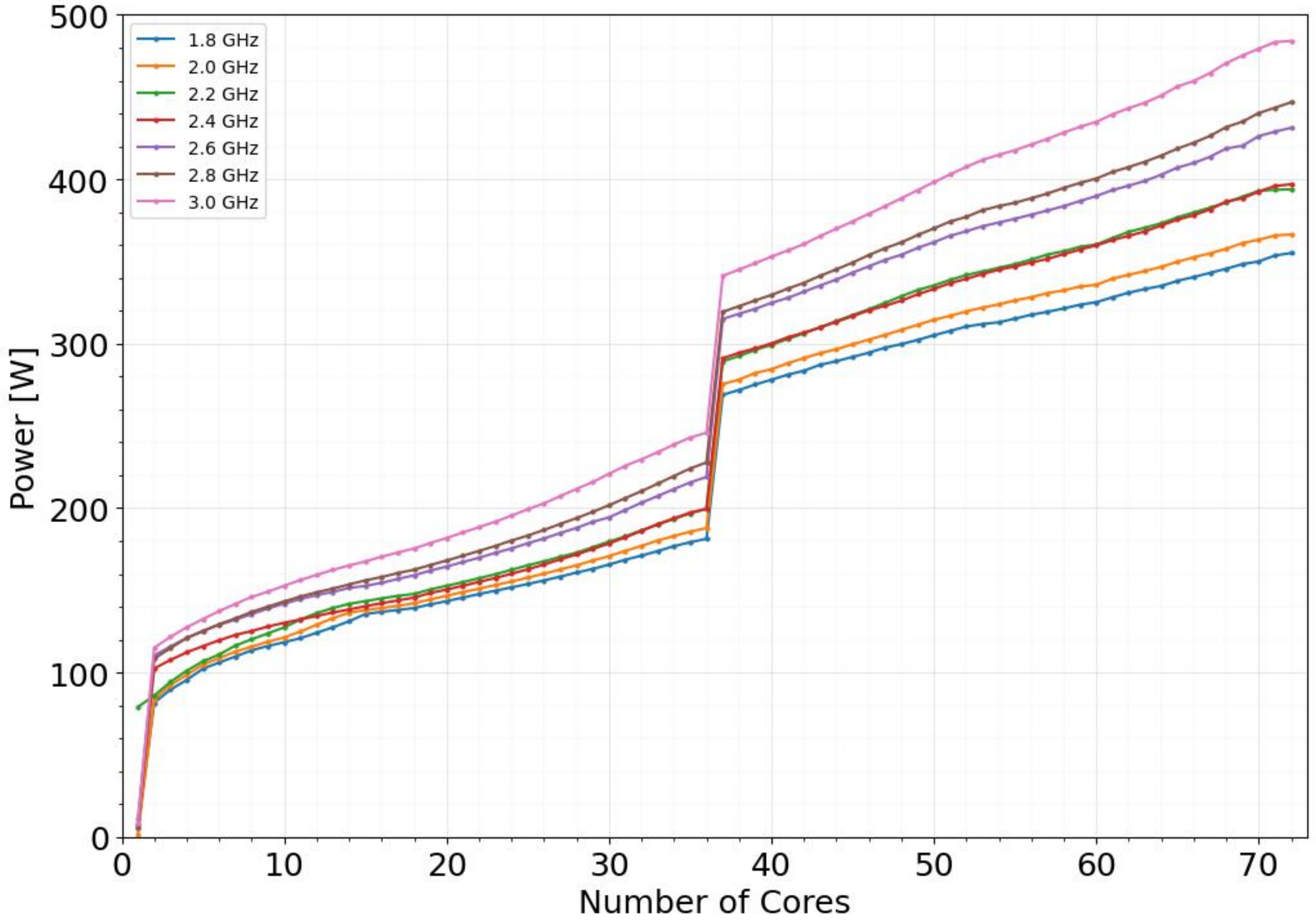


Figure 10: Power measurements in one node

- An unusual behaviour also can be observed at first core power measurment.
- LIKWID-POWERMETER also produced the same results.
- Sudden power jump when moving to second socket.

Energy and power measurements; main

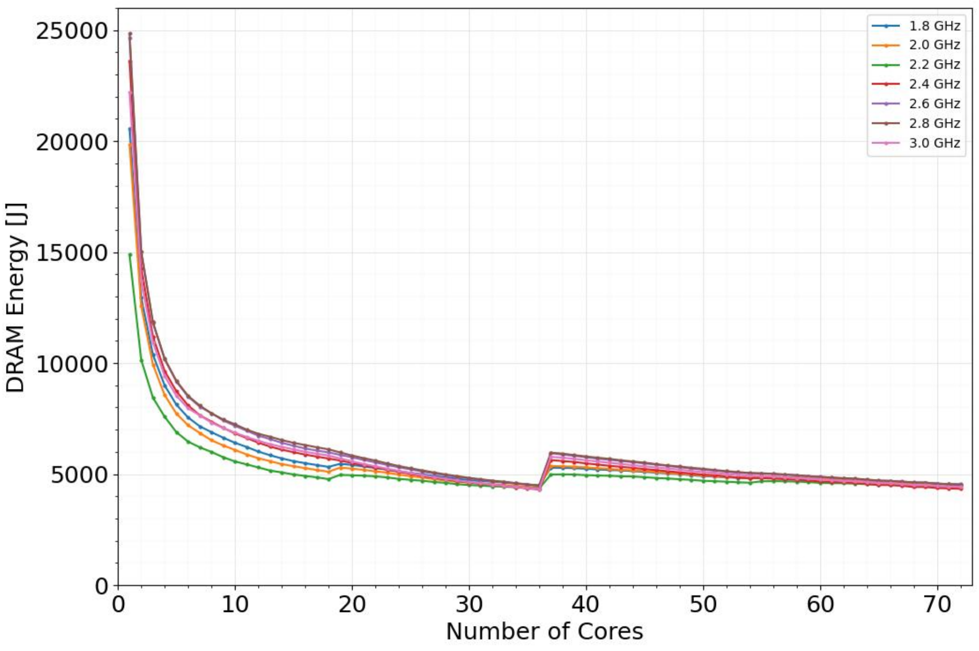


Figure 11: DRAM Energy measurements in one node

- DRAM energy measurements do not show an unexpected behaviour at single core measurements all the time.
- 2.2 GHz seems to be the lowset energy curve for a wider domain.
- The DRAM energy contribution to the total energy is in the range of 5%-11%.

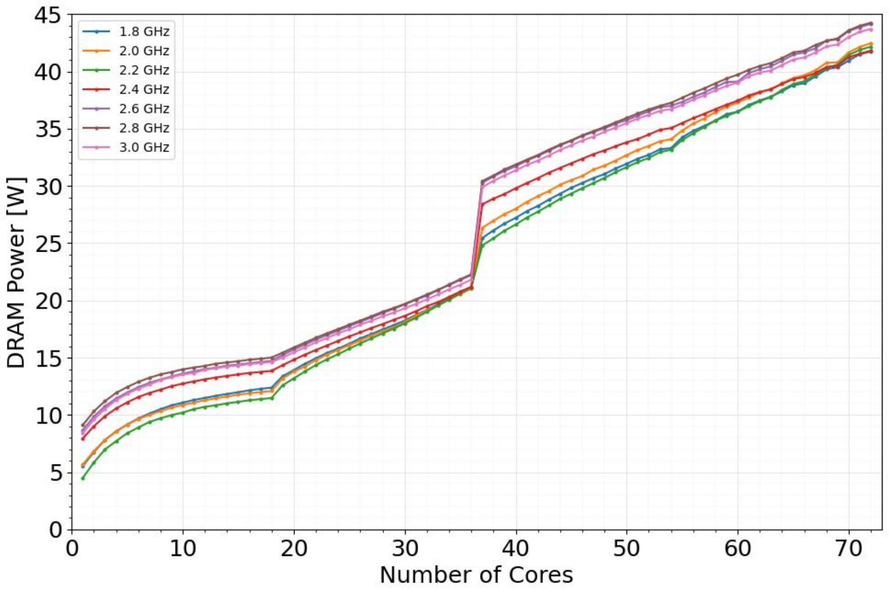


Figure 11: DRAM Power measurements in one node

- First NUMA domain there is a saturating trend considering DRAM power.

Energy Delay Product(EDP)

- Z-plot considers dissipated energy versus any suitable performance metric for a given program[5][6].
  - ✓ EDP is the gradient of a line passing through in z-plot.
- First socket measurements is taken for the Z-plot.

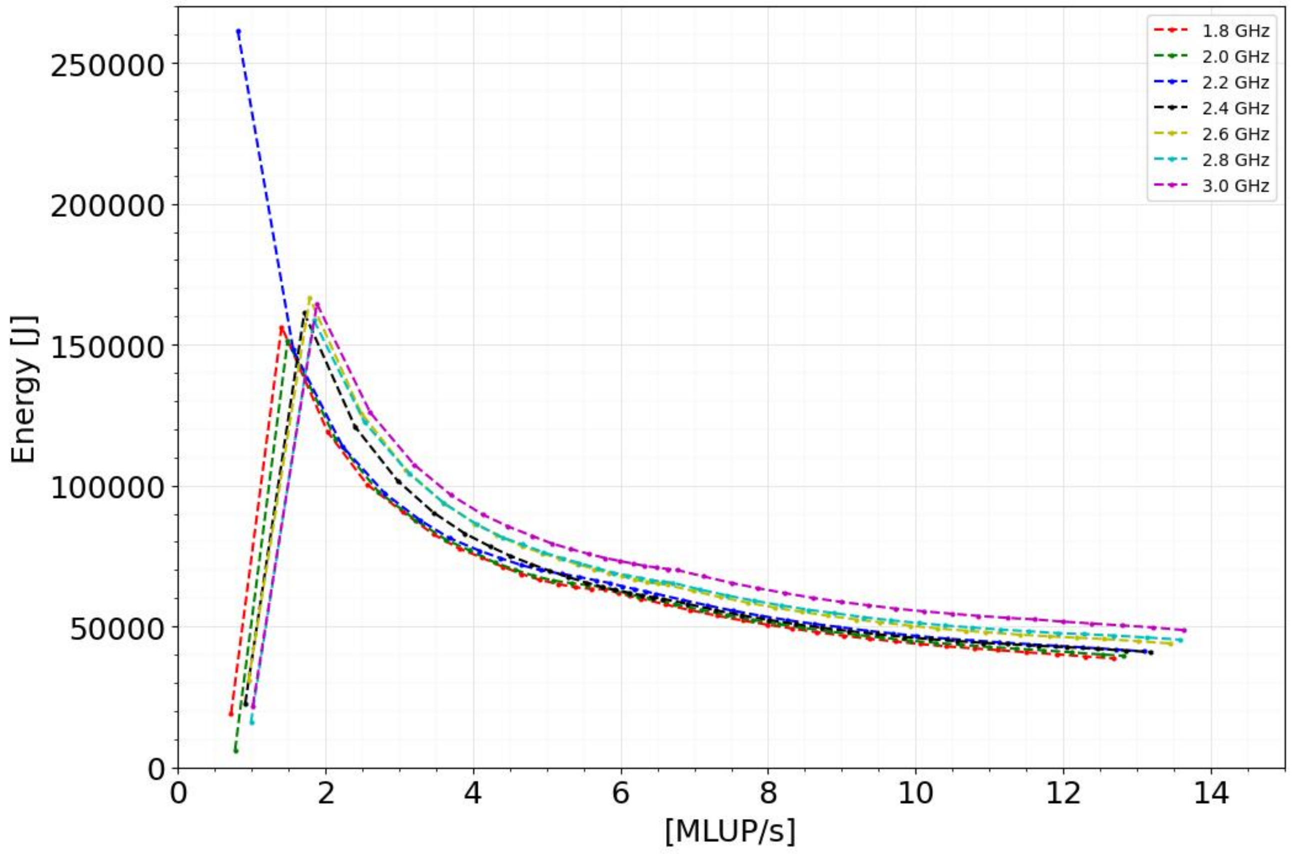


Figure 11: Z-plot considering one socket.

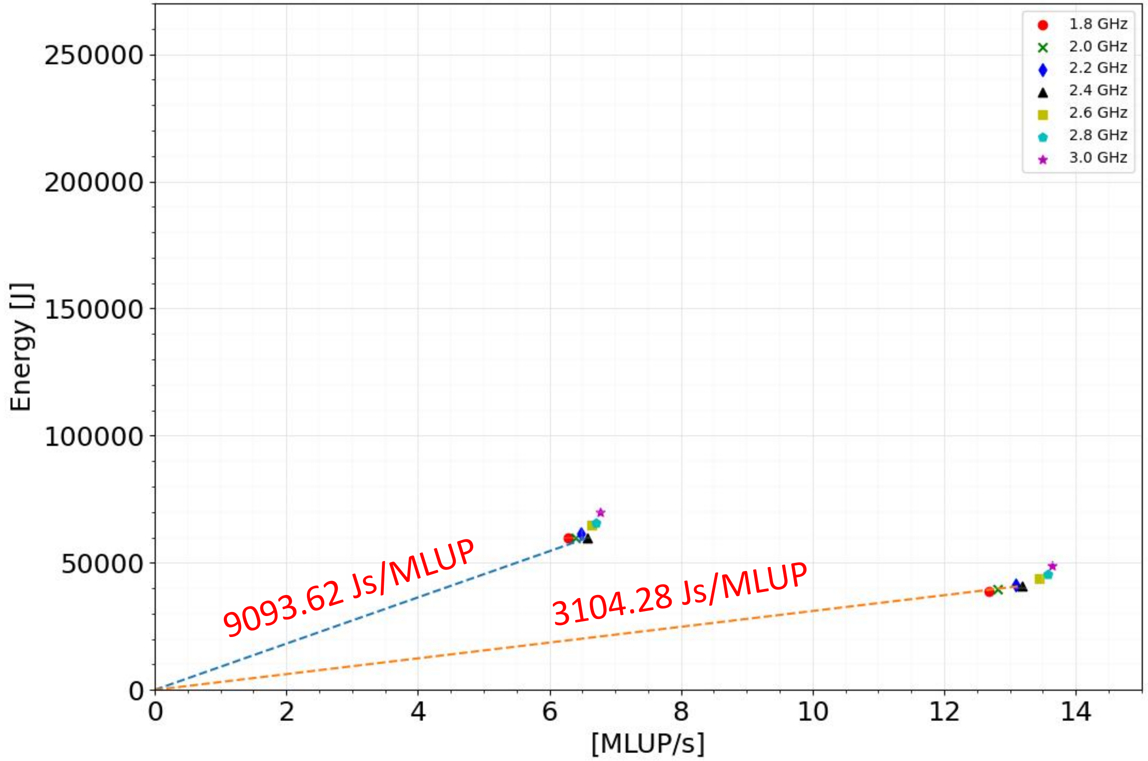


Figure 12: EDP related to 18 and 36 cores at 2.4 GHz

- This verifies the earlier point relating energy with first four frequencies.

Energy Delay Product(EDP)…

- With EDP the trade of between energy and performance can be discribed.

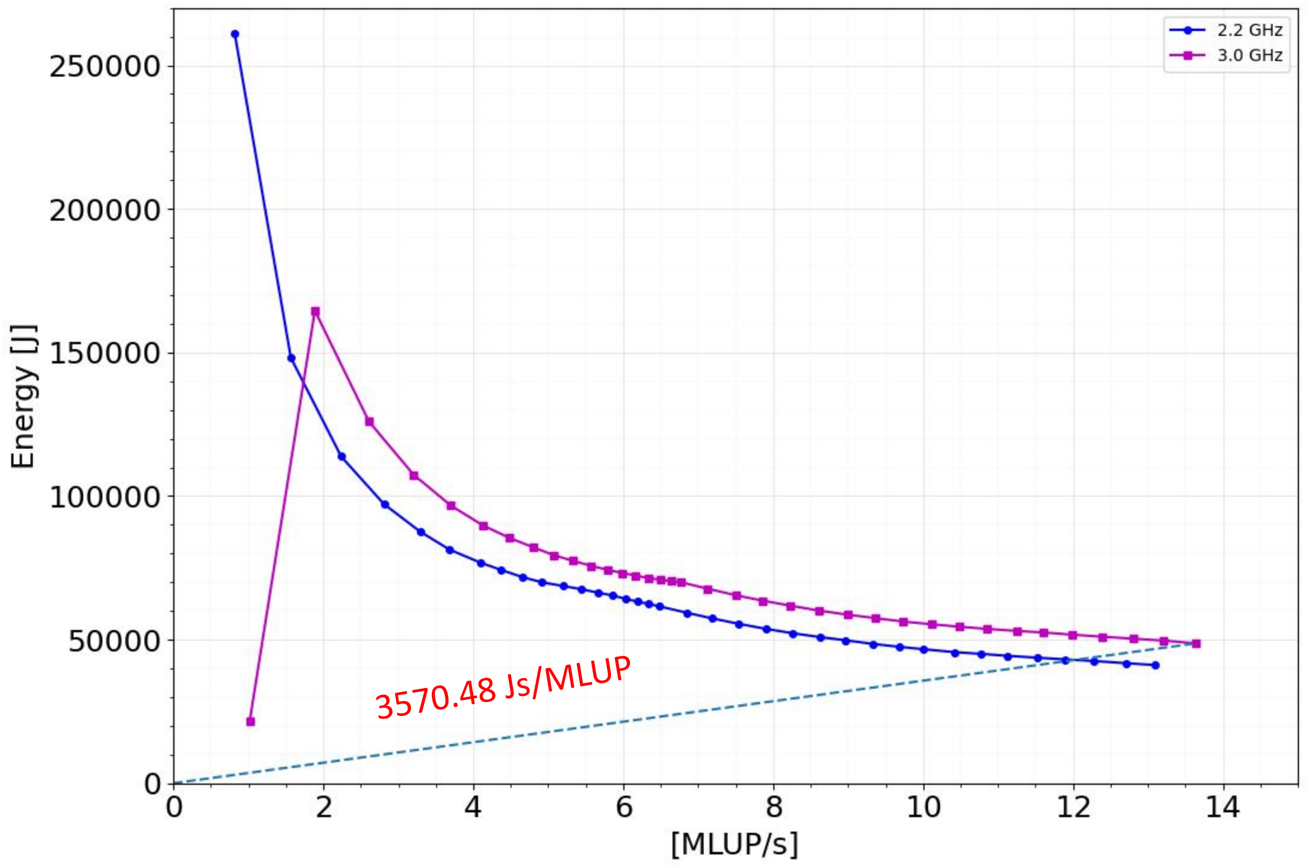


Figure 11: Energy and performance trae-off

- Considering EDP it is vissible that
  - ✓ 33 cores @ 2.2 GHz have the same energy efficiency as the 36 cores @ 3.0 GHz.
  - ✓ But, performance wise it loose nearly 10%.

Hotspot measurements

- Considered the stream, coliision and newtonRaphson kernels.
- Observed the performance, memory bandwidth saturations and energy contributions.

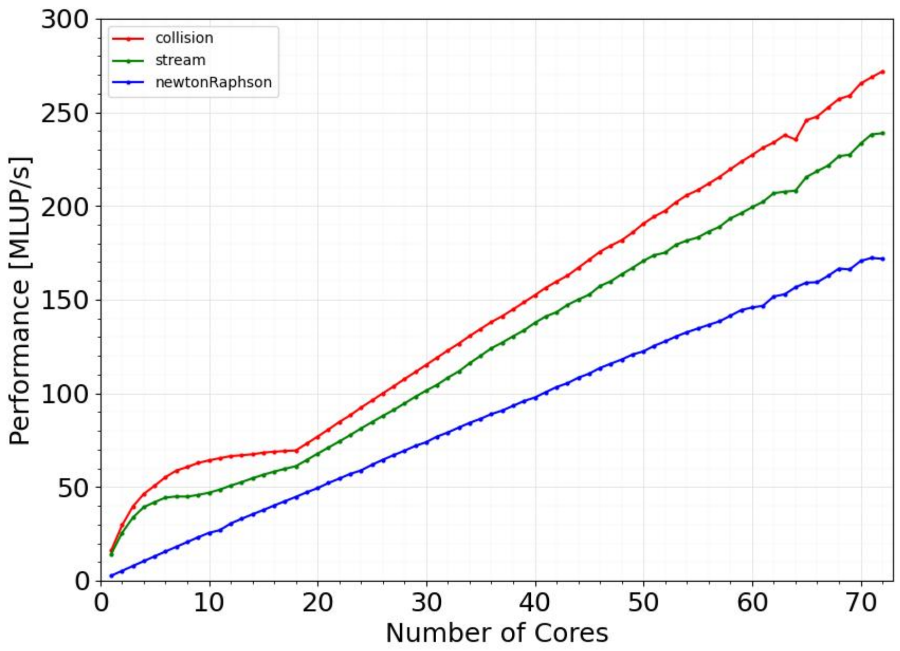


Figure 11: Performance vs. core count

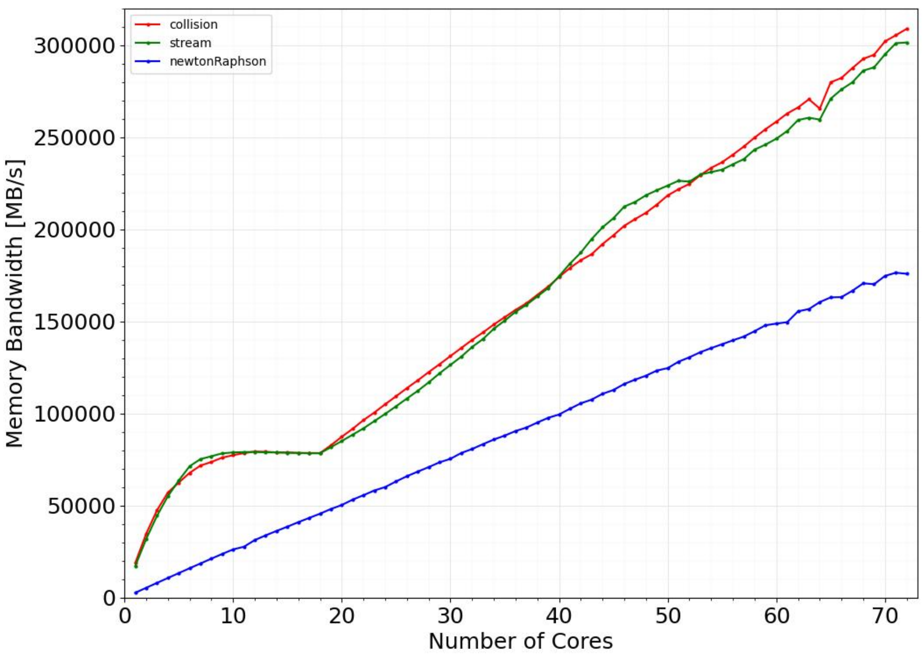
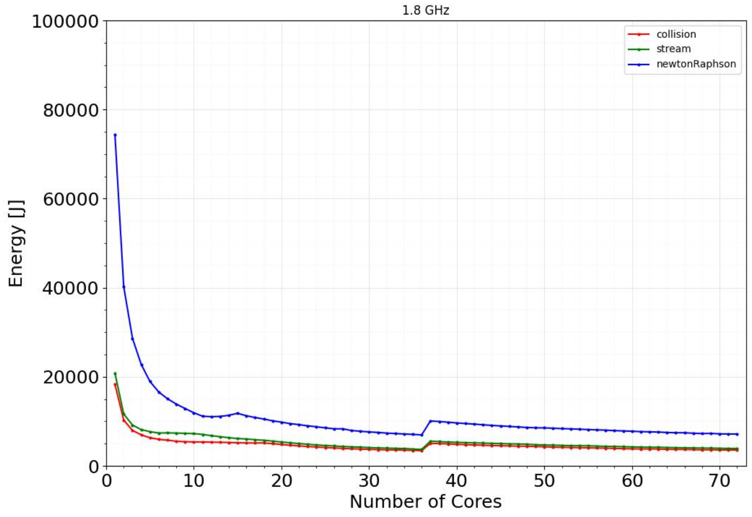


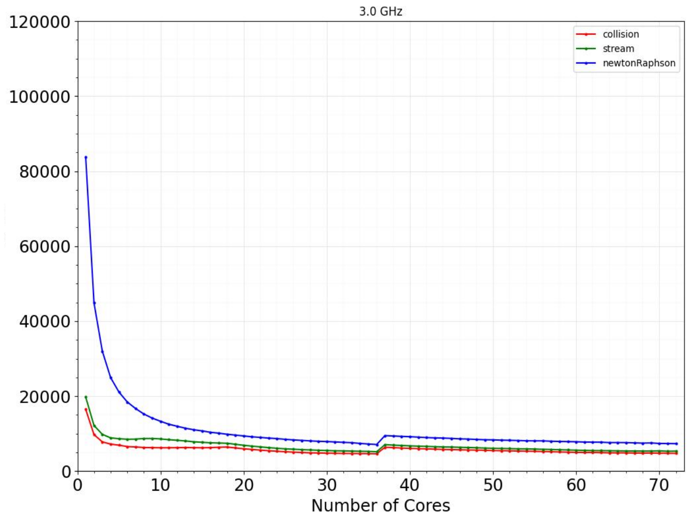
Figure 11: Memory bandwidth vs. core count

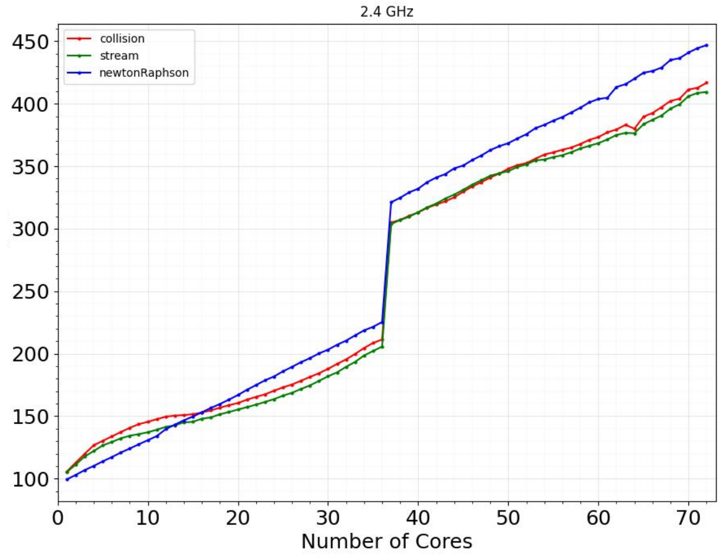
- newtonRaphson kernel does not show any saturation in full node.
- collision and streaming shows the sathrating trend inside the first NUMA doamin.

Hotspot measurements…

- The previously observed, unexpected measurement point at single cores does not appear in energy plots .







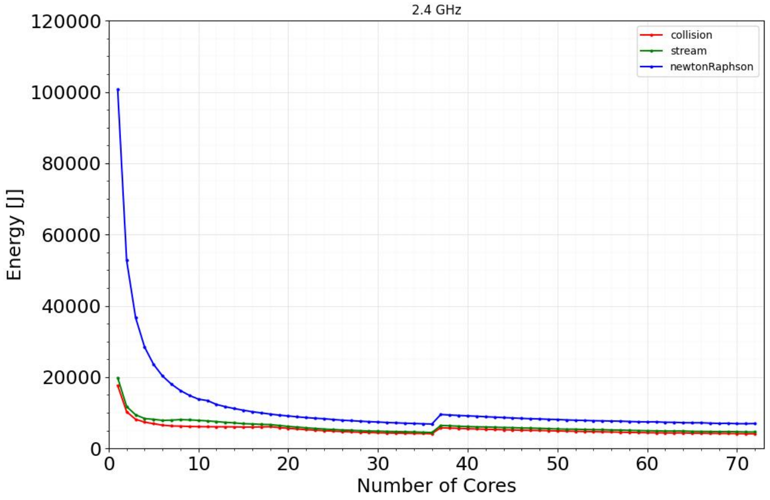


Figure 11: Energy vs. core count

Figure 11:Power vs. core count

- All three kernels contributes 9% - 16% to the main energy measurements.
- newtonRaphson has the highest contribution while showing a scalable performance within the node.

- Bothe the highest hotspot ranks seems to be have similiar variation in terms of power and energy.

Roofline modelling

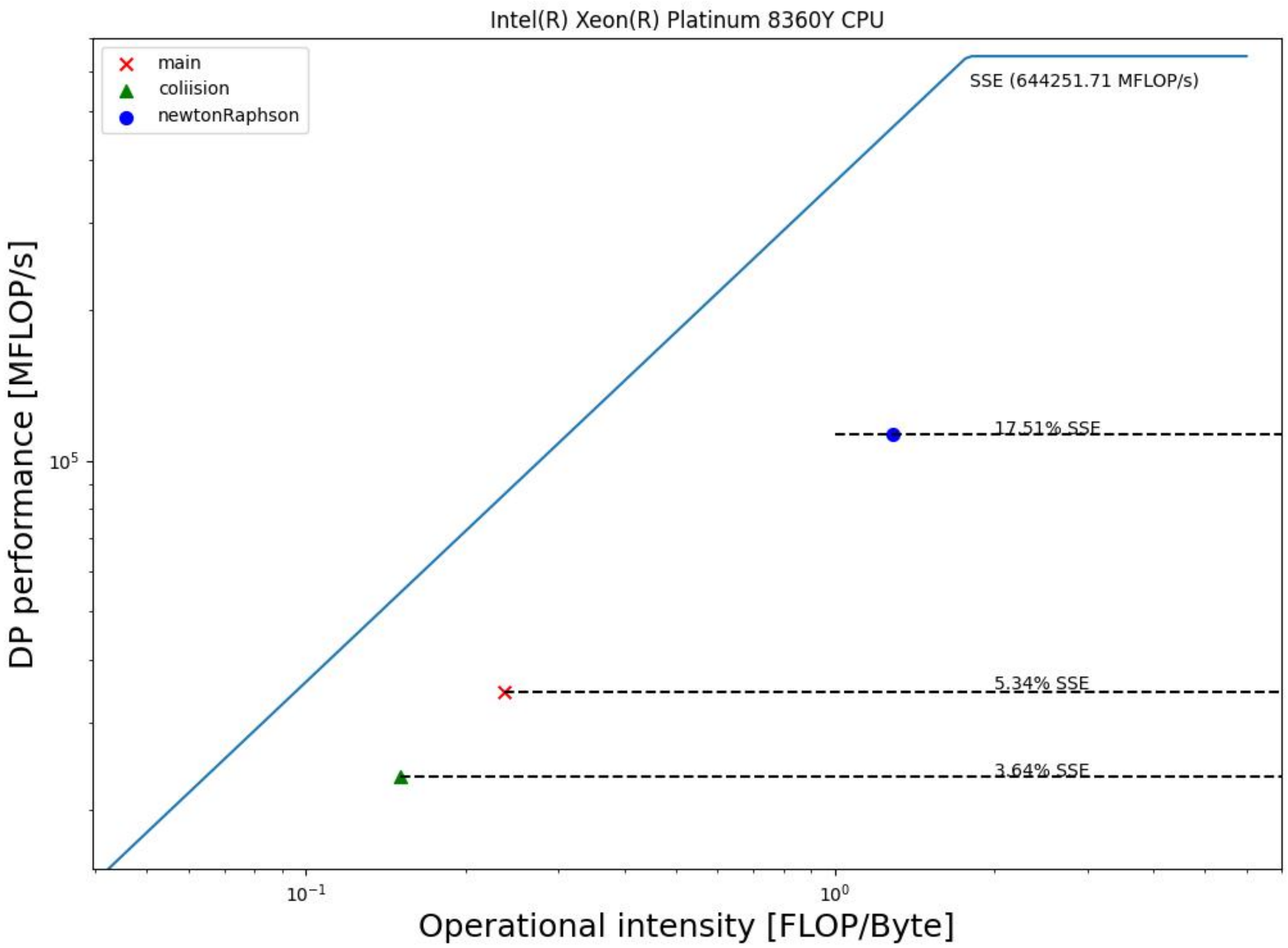


Figure 11: Roofline diagram.

- LIKWID-bench tool is utilized to construct the
- empherical roofline diagram [7].
- Maximum performance is related to the SSE FLOPS.
- Maximum data throughput is related to the likwid-bench load_sse
- Here the performance metric is double precision performance.
- Stream kernal does not have any flops.
- CLBM application is close to memory bound.

Remarks.

- This is the first kind of benchmarking effort considering LBM in compressible supersonic flow.
- CLBM 2D test application shows scalable performance.
- Solver is close to memory bound.
- Optimal energy can be observed at the first socket in fritz node considering 1.8 GHz, 2.0 GHz, 2.2 GHz and 2.4 GHz
- EDP results verified the point above.
- DRAM energy contribution to the total energy is 5%-11%.
- Investigation of optimal energy point for the present implementation.
- According to the roofline analysis the application has arbitrary horizontal roofs.
  - ✓ Thers is more room for code optimization.
- Root finding scheme has less significance interms of performance, even though root finding algorithm was the highlight in previous literature regarding , this LBM regime.

Reference

1. **The Landscape of Parallel Computing Research A View from Berkeley,** 2006
   Electrical Engineering and Computer Sciences, University of California at Berkeley, *Electrical Engineering and Computer Sciences, University of California at Berkeley,*
2. A. A. Mohamad. **Lattice Boltzmann Method Fundamentals and Engineering Applications with Computer Codes**. Springer, 2019
3. Timm Krüger, Halim Kusumaatmaja, Alexandr Kuzmin, Orest Shardt, Goncalo Silva, and Erlend Magnus Viggen. **The Lattice Boltzmann Method**. Springer International Publishing, 2017.
4. Jonas Latt, Christophe Coreixas, Joël Beny, and Andrea Parmigiani. Efficient supersonic flow simulations using lattice boltzmann methods based on numerical equilibria. Philosoph ical Transactions of the Royal Society A:Mathematical, Physical and Engineering Sciences,378(2175):20190559, jun 2020.
5. https://blogs.fau.de/hager/archives/tag/energy
6. Ayesha Afzal,Georg Hager,Gerhard Wellein,SPEChpc 2021 Benchmarks on Ice Lake and Sapphire Rapids Infiniband Clusters: A Performance and Energy Case Study, SC-W '23: Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis,November 2023, Pages 245–1254, https://doi.org/10.1145/3624062.3624197
7. https://github.com/RRZE-HPC/likwid/wiki/Tutorial:-Empirical-Roofline-Model