Nemo analyses
POP_AR_078

Jesus Labarta, Marta Garcia, Joan Vinyals (BSC)

EU H2020 Center of Excellence (CoE)

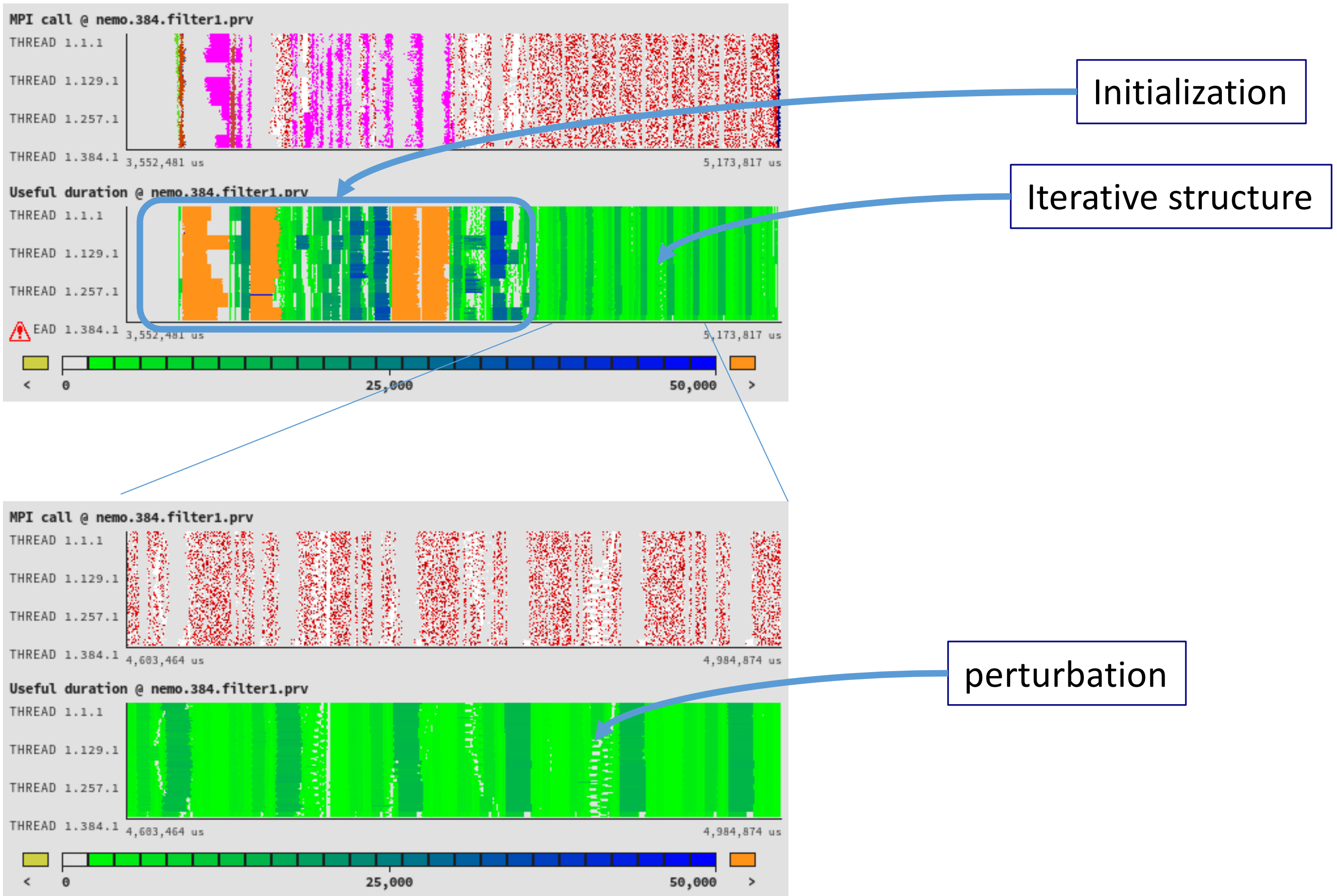
November 4th 2020

The code

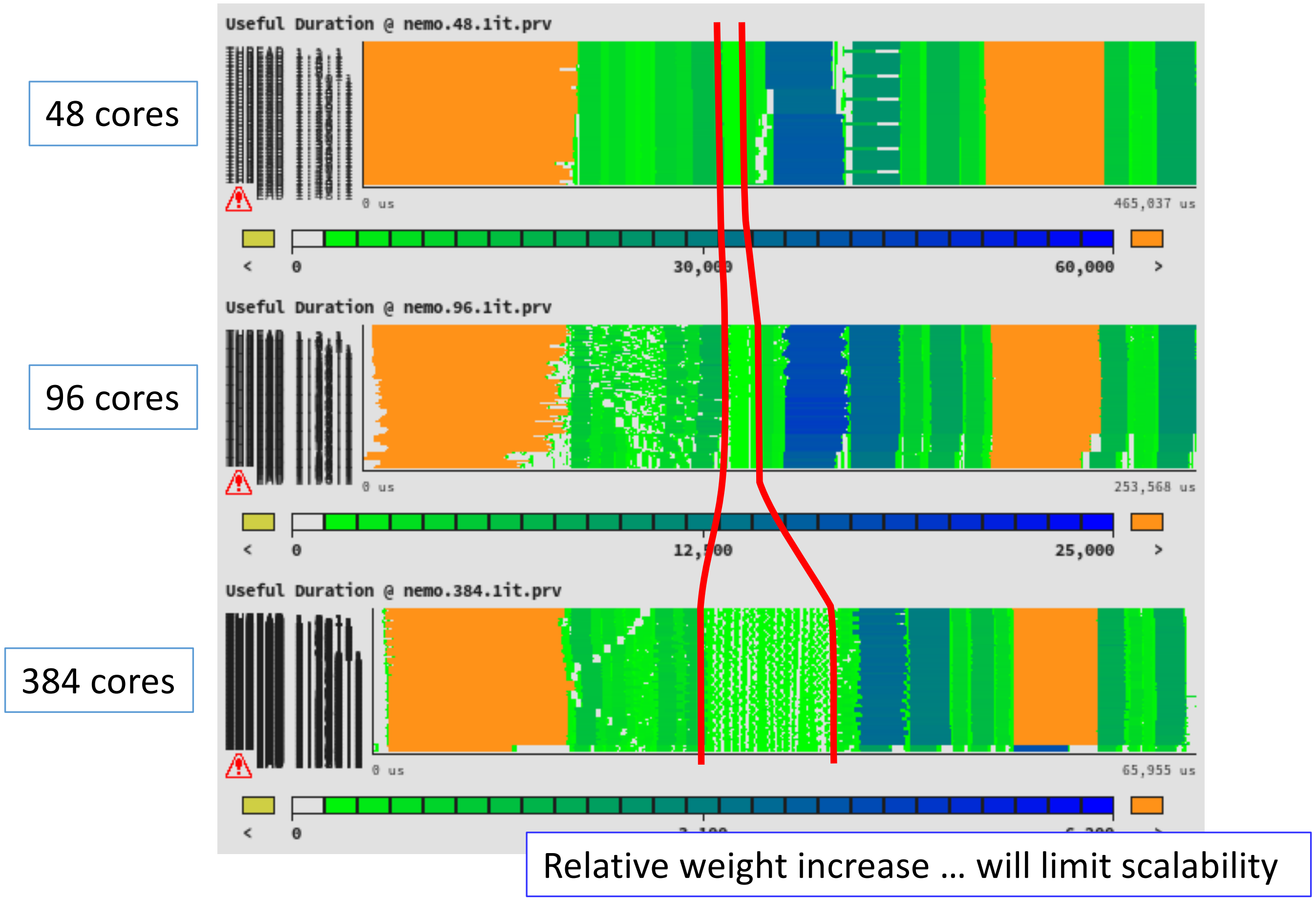
- Code
  - NEMO version 4.0.2
  - MPI
- Problem:
  - Bench ORCA 1 like
- Traces:
  - Run @ MareNostrum, filling node (48 processes/core)
  - 48, 96 and 384 processes

Structure

384 cores

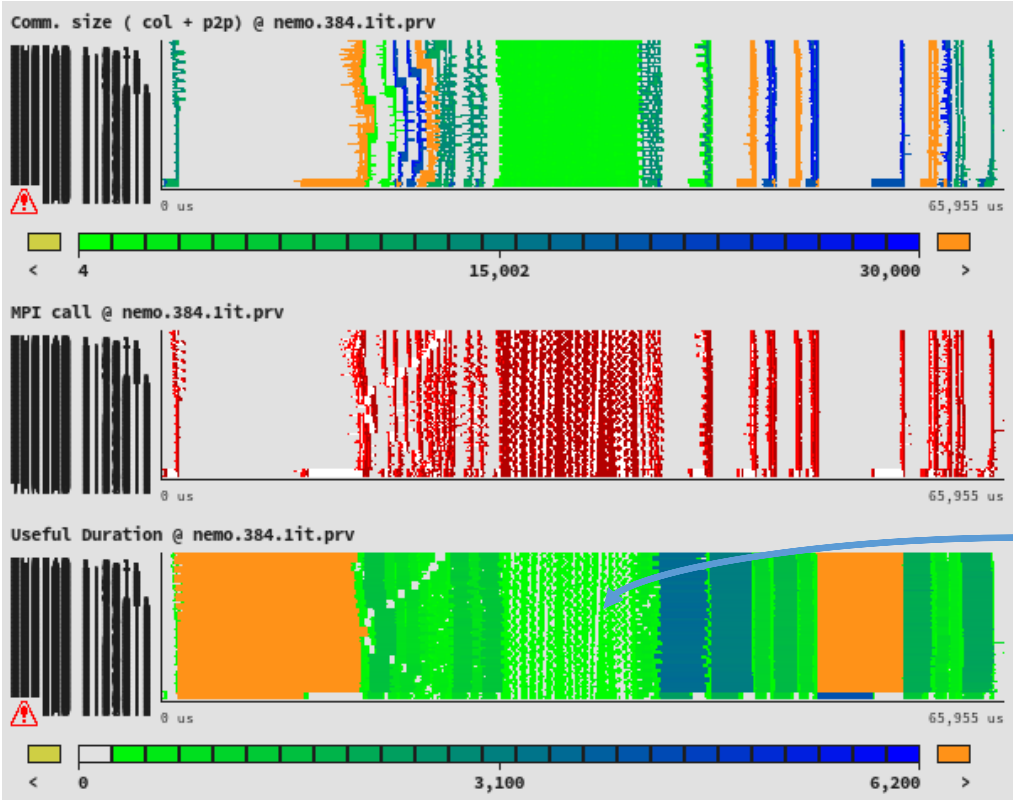


Structure


48 cores
96 cores
384 cores
Useful Duration @ nemo.48.1it.prv
0 us
465,037 us
0
30,000
60,000
Useful Duration @ nemo.96.1it.prv
0 us
253,568 us
0
12,500
25,000
Useful Duration @ nemo.384.1it.prv
0 us
65,955 us
0
Relative weight increase … will limit scalability

Structure

384 cores



Too fine grain !! → hybrid

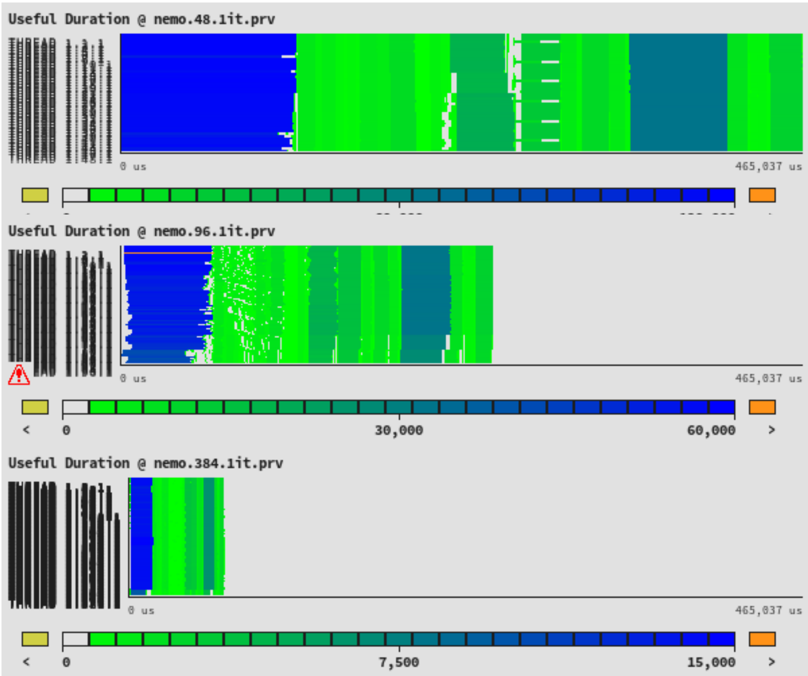
Scaling

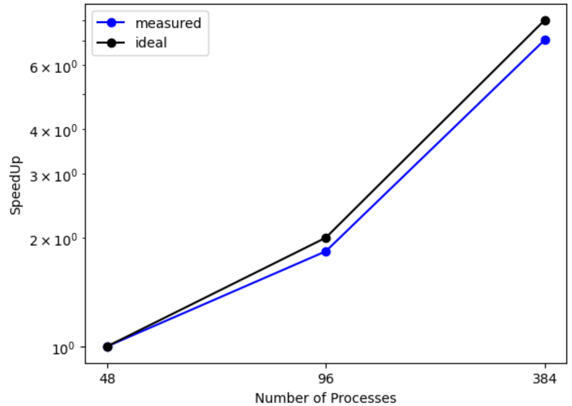
T(48) =465 ms

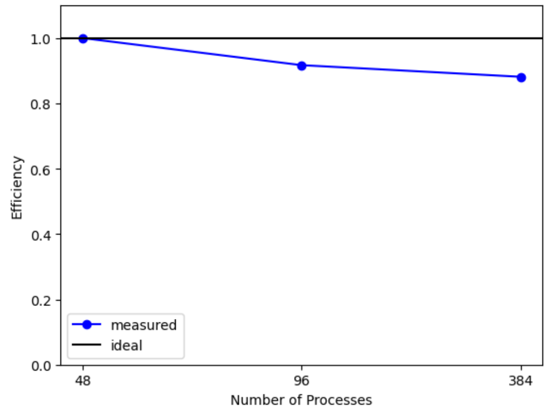
48

96

384

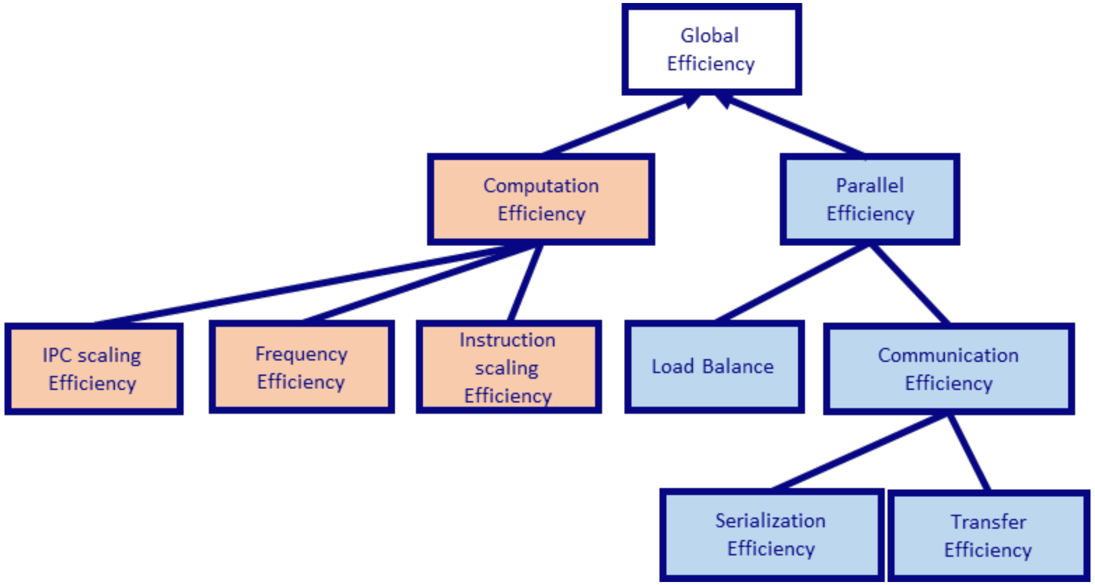






Hierarchical Performance Model

Efficiencies: ~ (0,1]
Multiplicative model



$$\boxed{CompEff = Ieff * IPCeff * Feff}$$

$$\eta_{\parallel} = LB * \overbrace{Ser * Trf}^{CommEff}$$

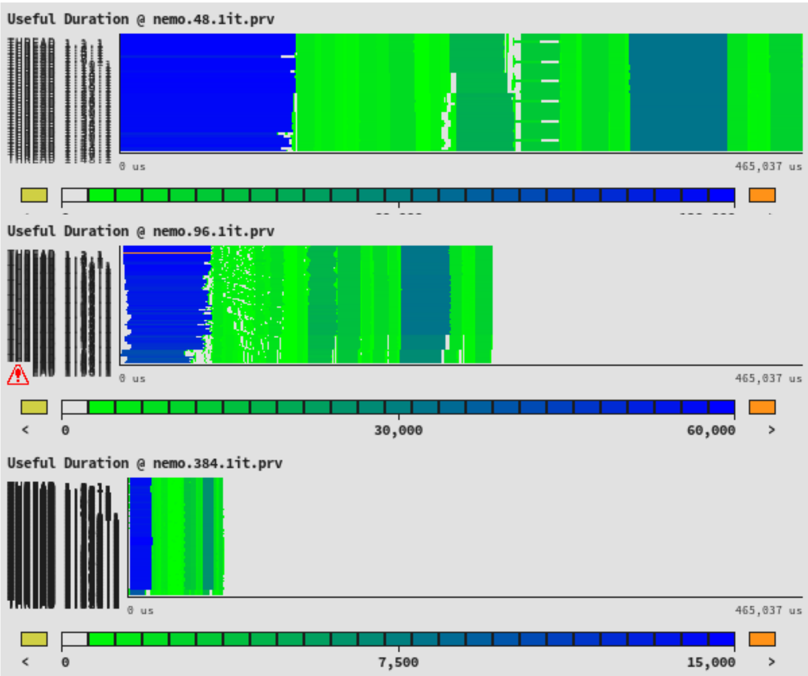
M. Casas et al, "Automatic analysis of speedup of MPI applications". ICS 2008.

Efficiency model

48

96

384



	48	96	384
Global efficiency	94.16	86.35	82.99
-- Parallel efficiency	94.16	84.11	74.80
-- Load balance	98.67	95.34	95.77
-- Communication efficiency	95.43	88.22	78.10
-- Serialization efficiency	97.86	93.05	89.61
-- Transfer efficiency	97.52	94.81	87.15
-- Computation scalability	100.00	102.66	110.95
-- IPC scalability	100.00	115.98	182.39
-- Instruction scalability	100.00	91.37	62.62
-- Frequency scalability	100.00	96.87	97.13

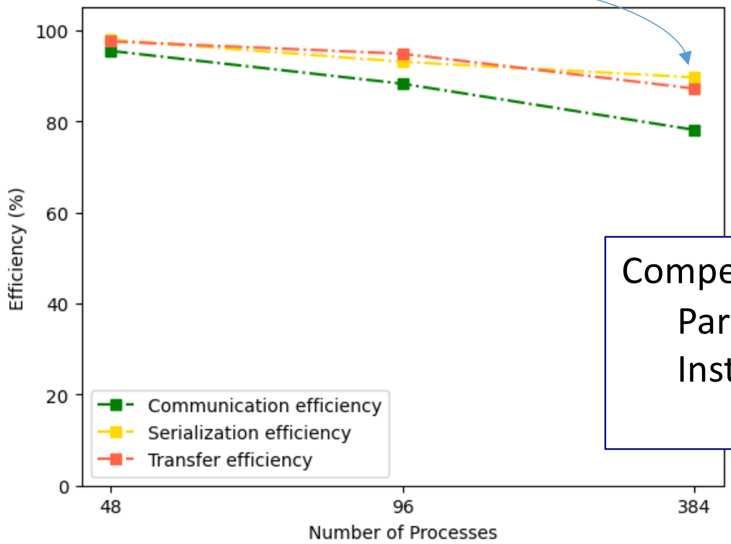
Percentage(%): 0 – 20 – 40 – 60 – 80 – 100

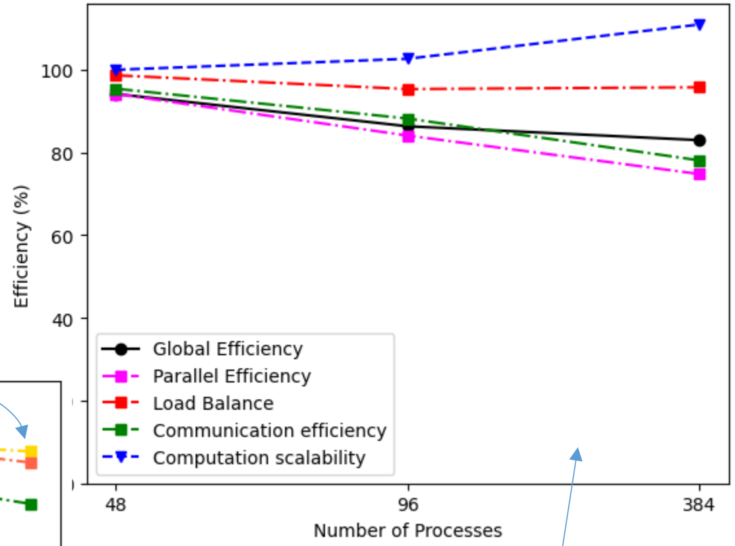
Avg Useful IPC(48) =0.67

Avg Useful Frequency(48) =2.061 GHz

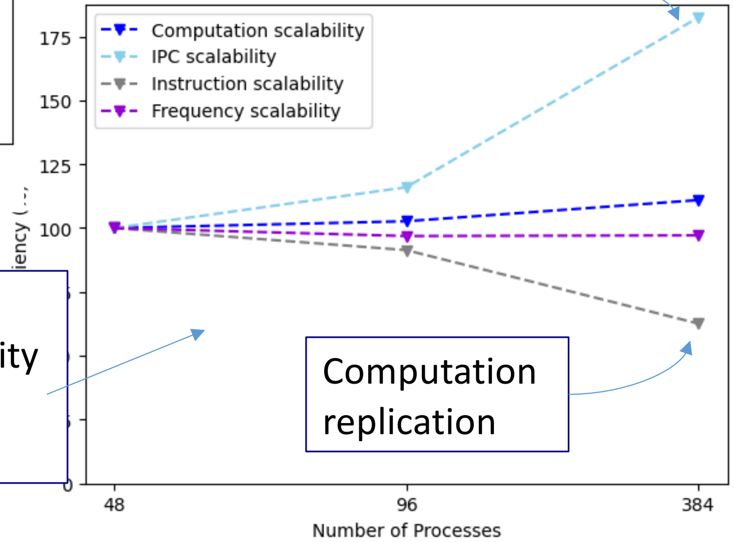
Efficiency model

Transfer and Serialization
Transfer scaling worse





Very good IPC scaling
→ probably very bad starting value
→ will not go on forever



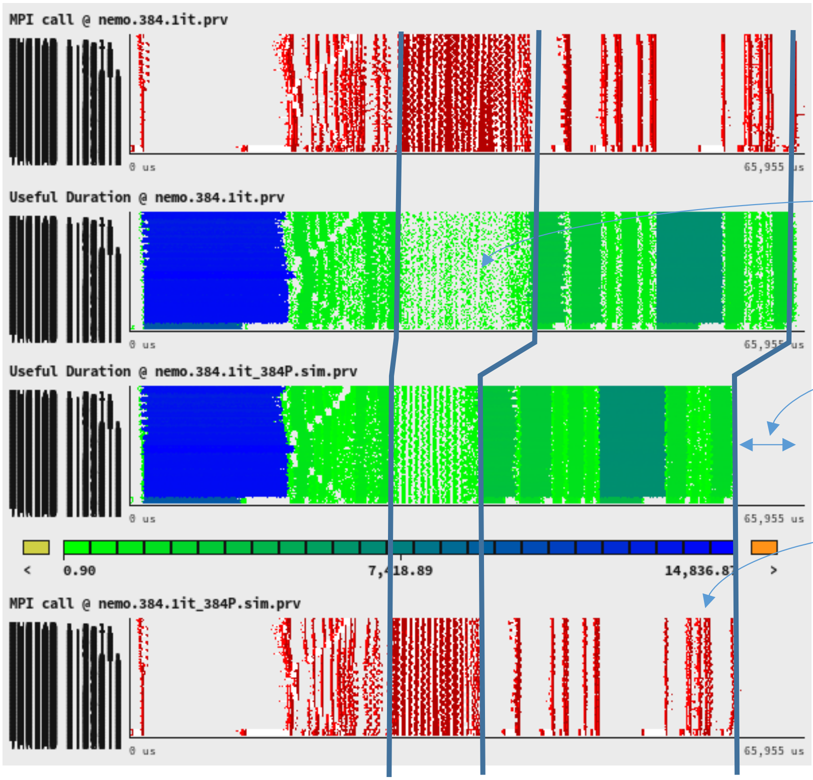
Compensation effects
Parallel efficiency ↔ Computation scalability
Instruction scalability (replication) ↔ IPC

Computation replication

Communication analysis

Actual run

Ideal Network



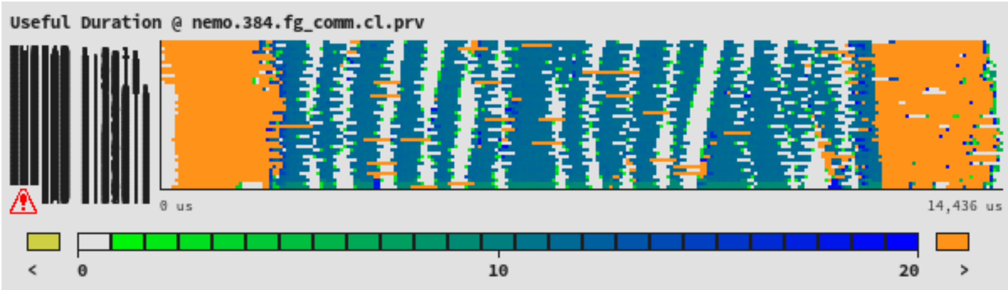
Fine granularity region sensitive to Interconnect latency and bandwidth

Transfer

MPI does not disappear …
Elapsed time for the region does not shrink

Communication analysis

- Focus on Inner fine grain communication phase

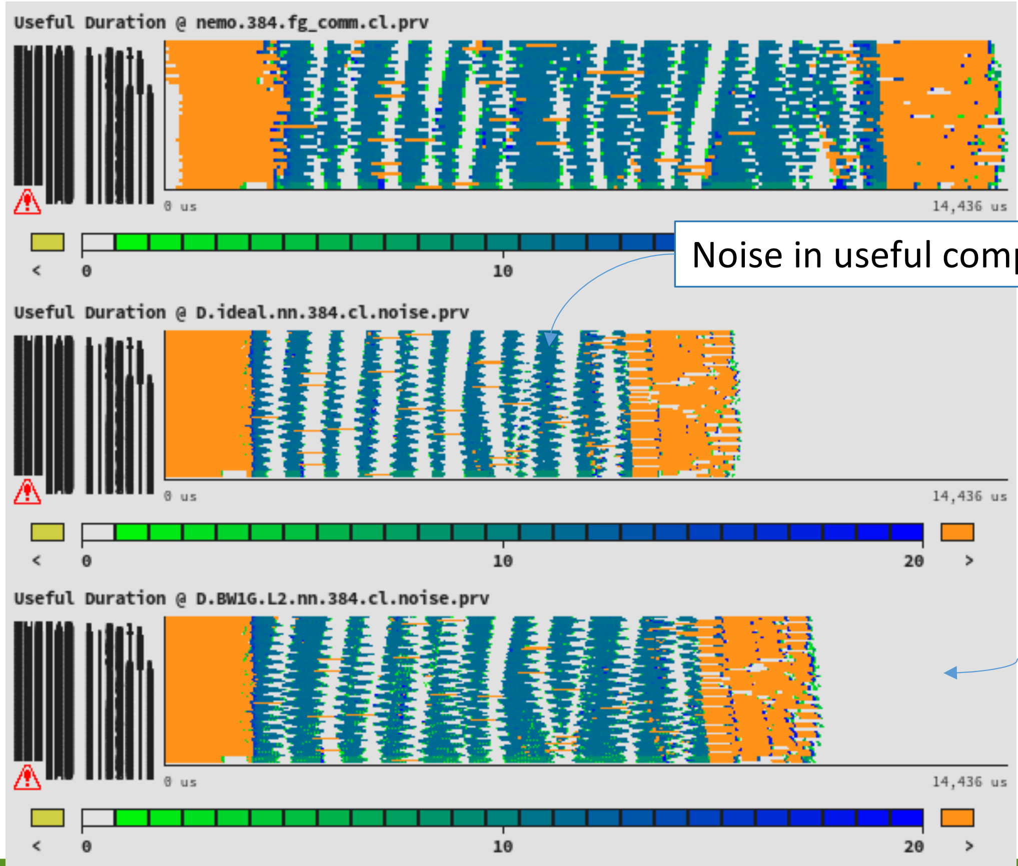


Communication analysis

Actual run

Ideal Network

“Nominal” Network



Noise in useful computation in fine grain phase

Difference → there was noise in the internals of MPI implementation

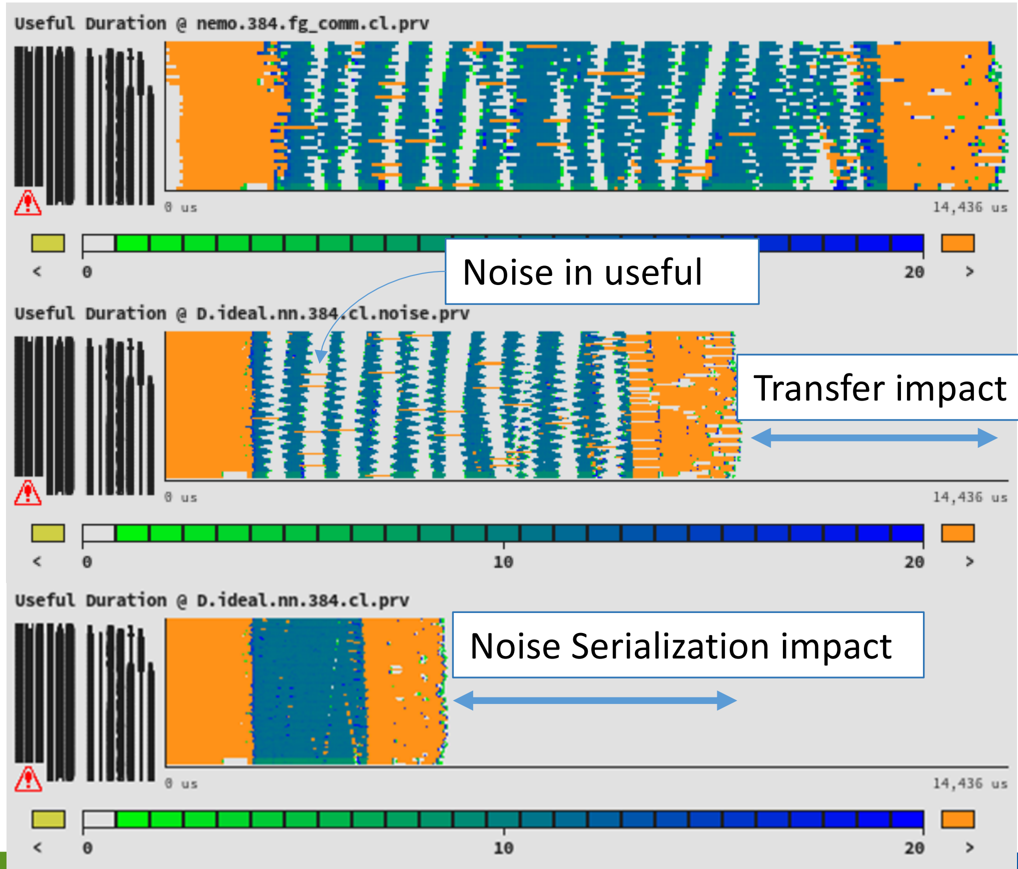
Communication analysis

- Serialization caused by OS Noise

Actual run

Ideal Network

Ideal Network “Eliminating” OS noise

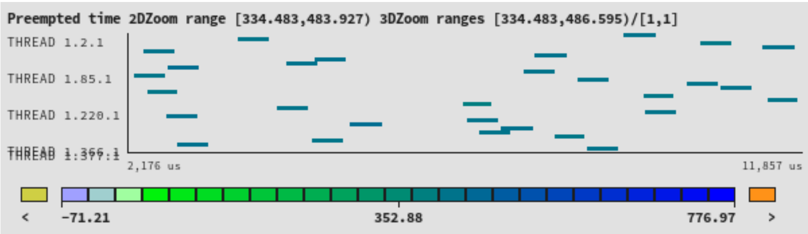


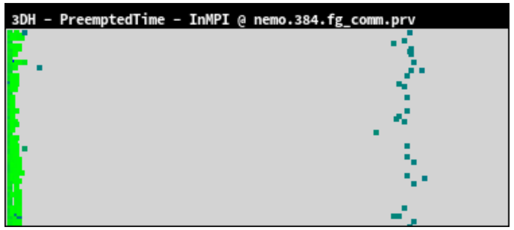
Close look at noise

"Preemptions"

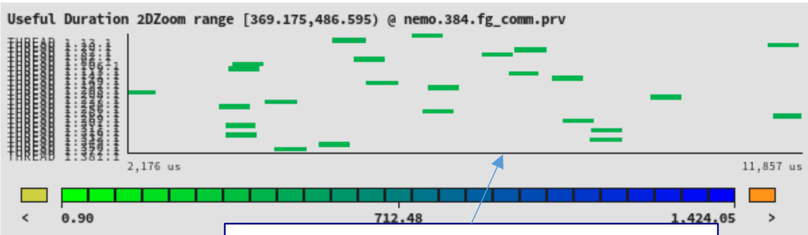
"Preempted" time

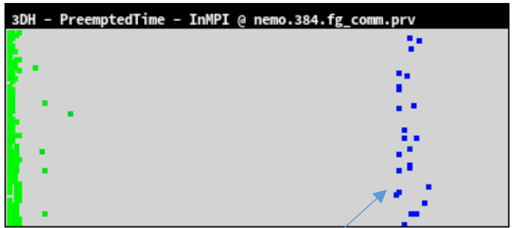
In MPI

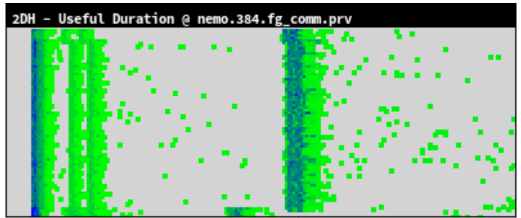




In Useful







Scattered in space and time
Both in MPI and useful

Mode ~400 us

Compared to ~11 us
(and less) computations

Hybrid MPI+OpenMP with relatively dynamic scheduling would be a way to reduce the impact of noise

"Noise" cause ?
Cant fight noise, learn to live with it

Close look at noise

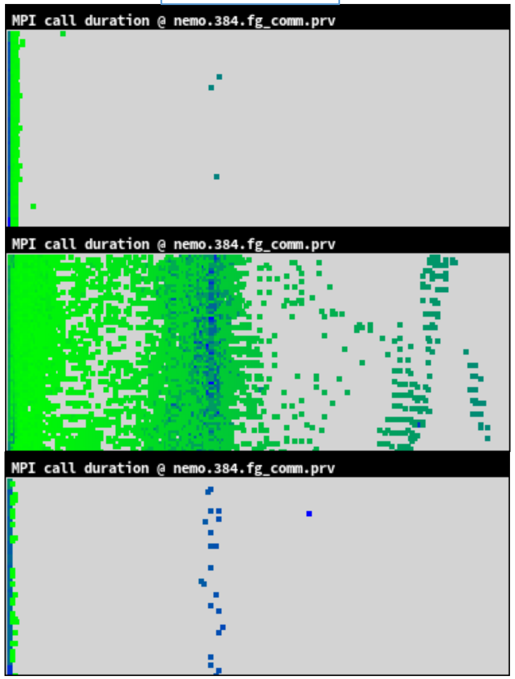
- MPI call durations affected & impacted by noise

Duration

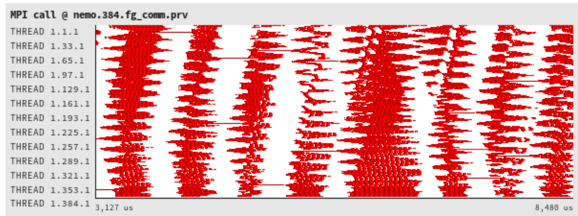
Isend

Recv

Wait



Mode ~400 us

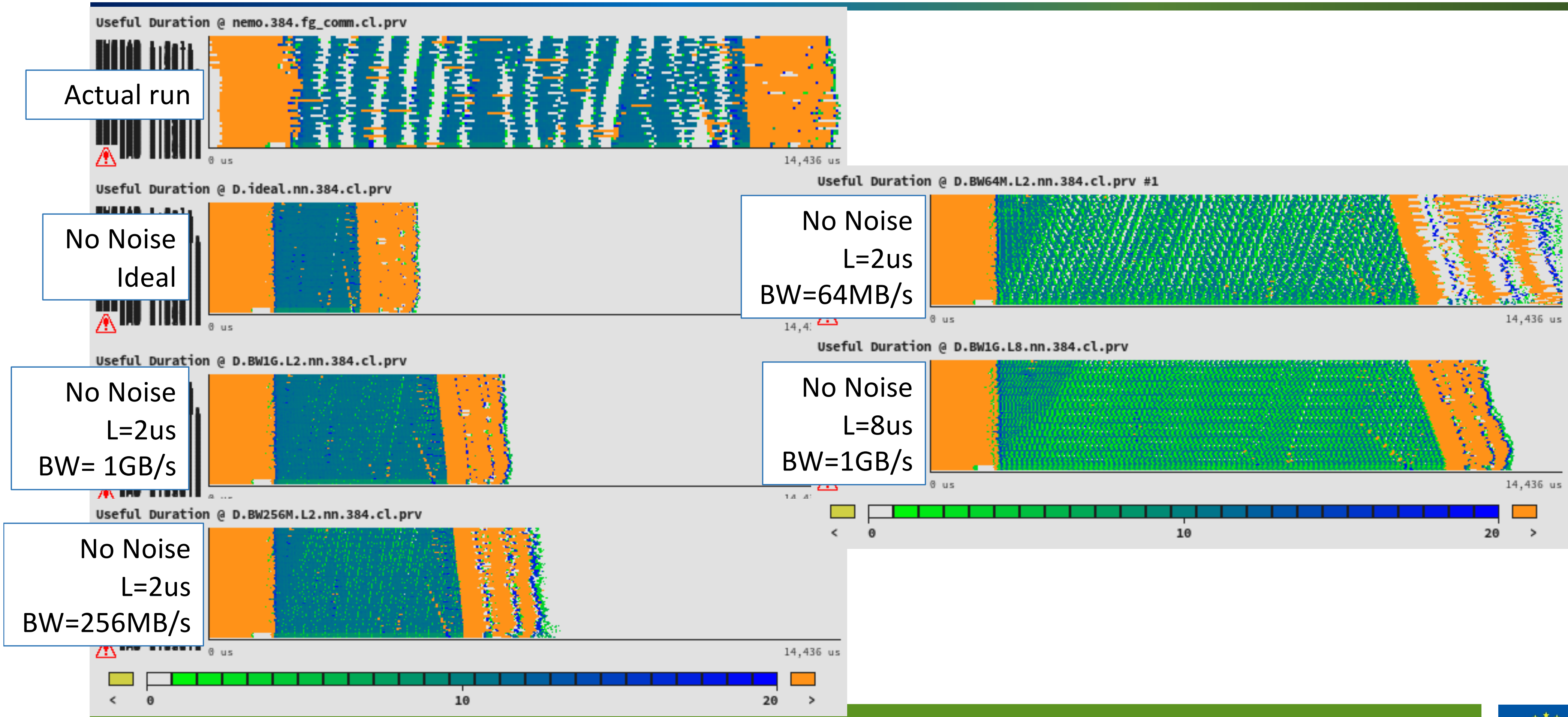


"Noise" cause ?

Noise within MPI

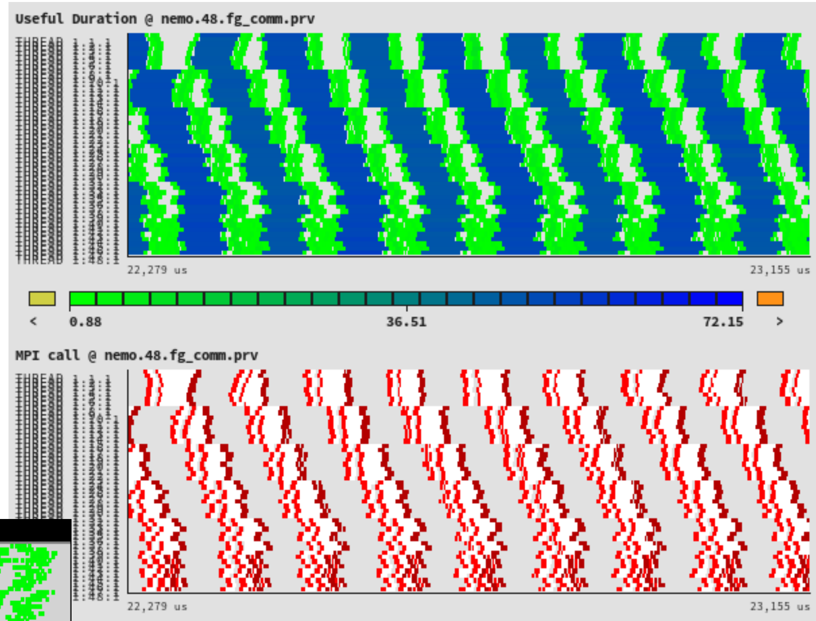
Requirement to vendor to implement noise toleration mechanisms within the node in their MPI implementation

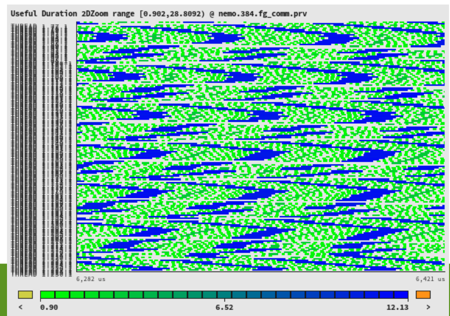
Transfer analysis

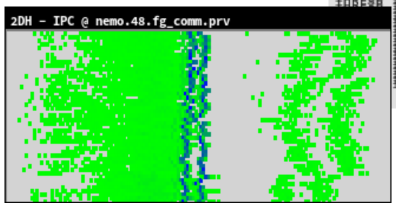


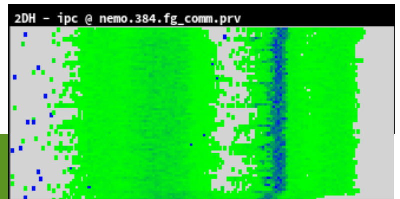
The skew

- Pattern in fine grain communication phase
- Efficiency loss on 48 processes (0.66)
- Significant impact at 384
  - Parallel efficiency ~0.5
  - Less overlap of computations & strong scaling → better IPC
  - Compensating effects



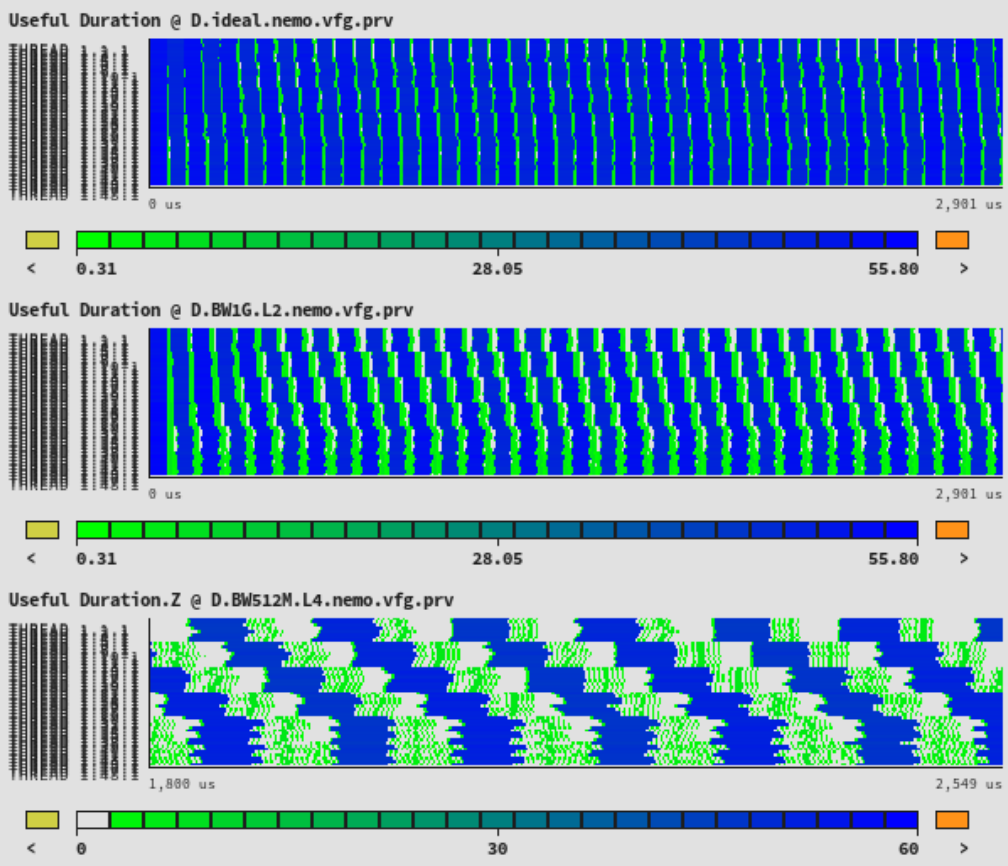


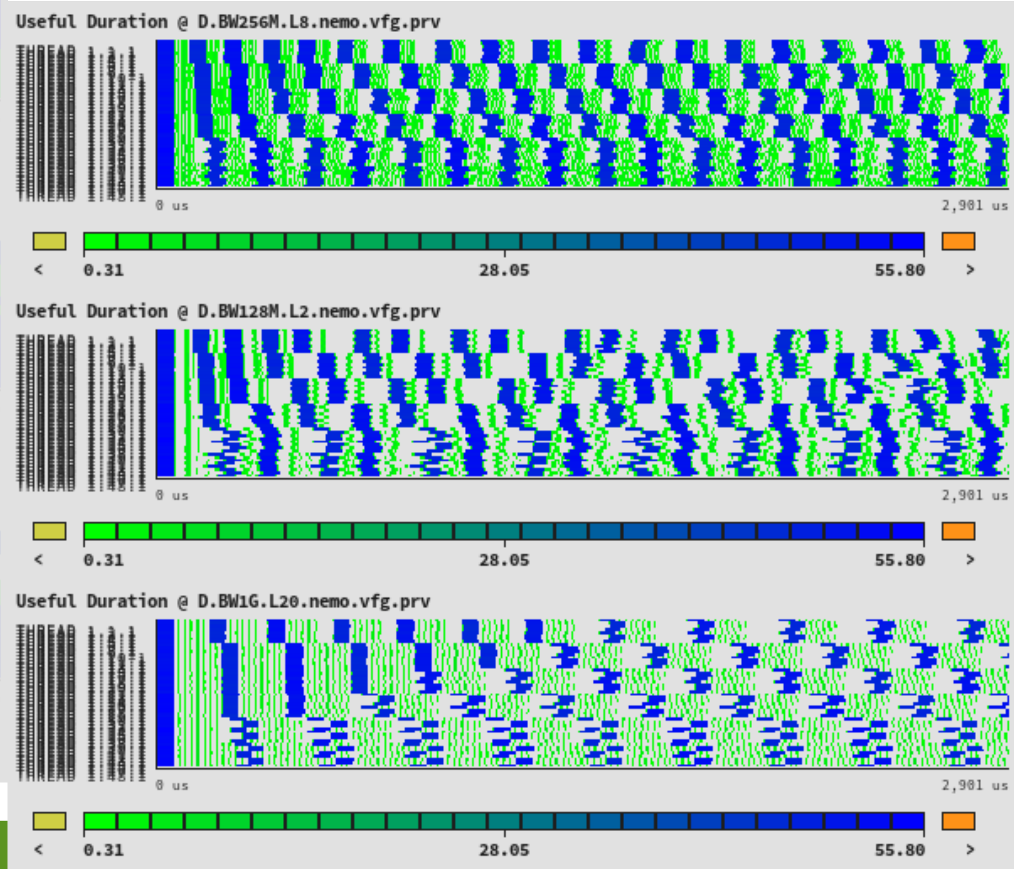




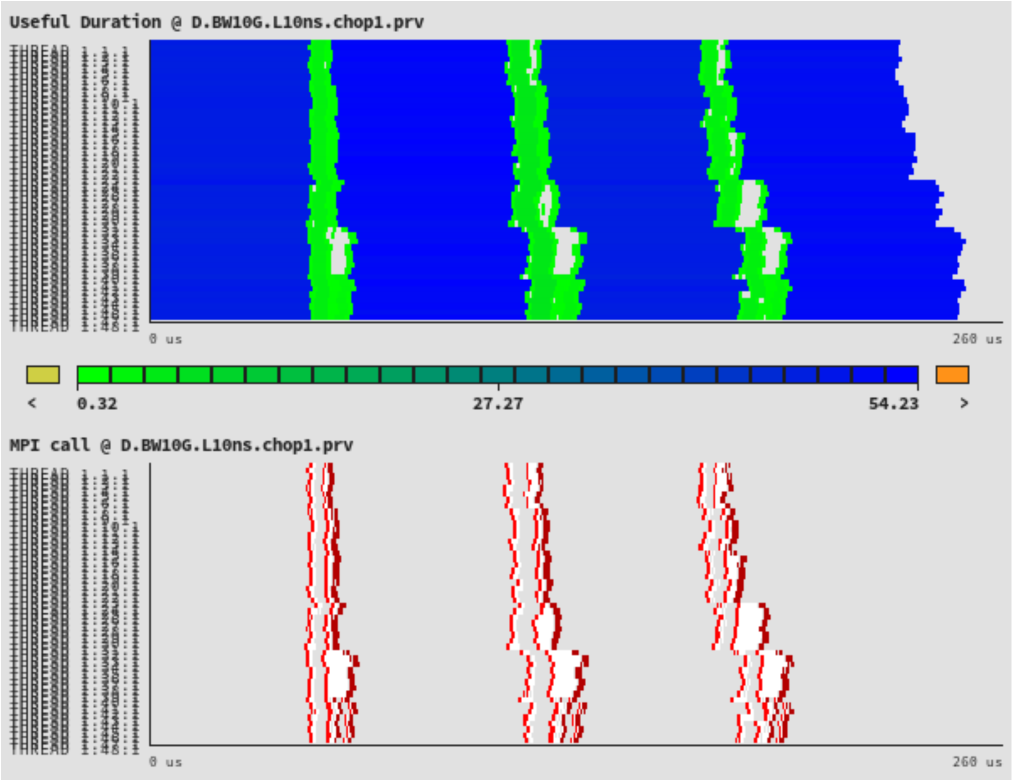
IPC: 0.5 -- 3.5

Impact of latency and BW on the skew

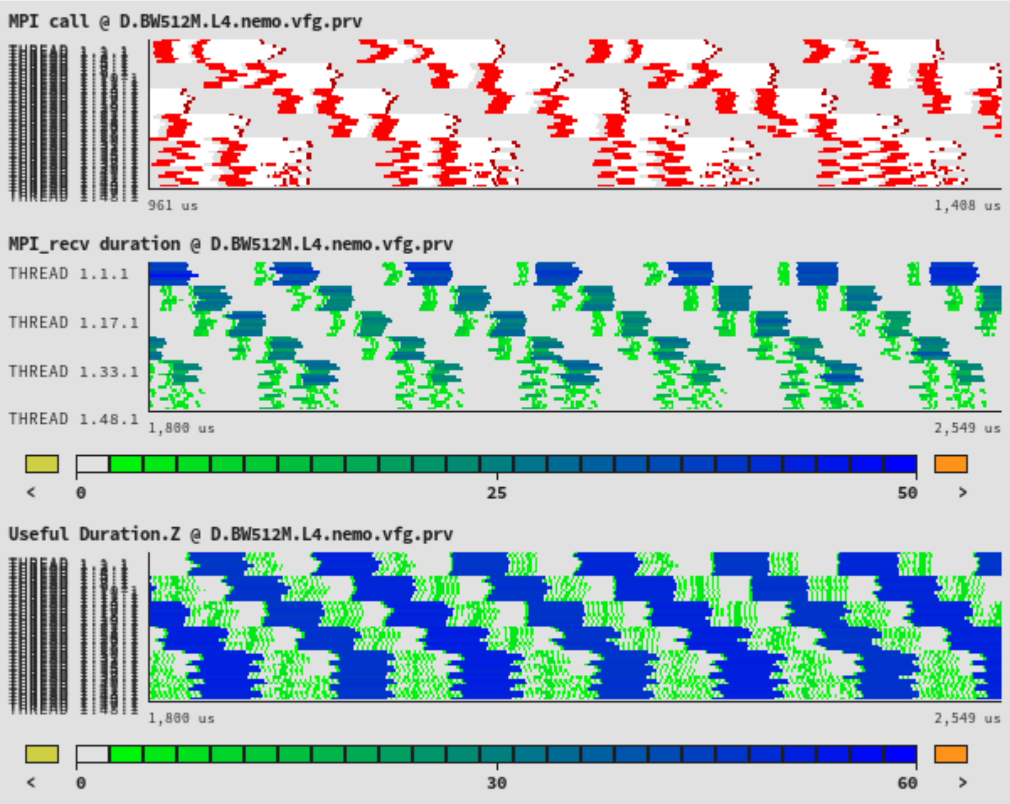


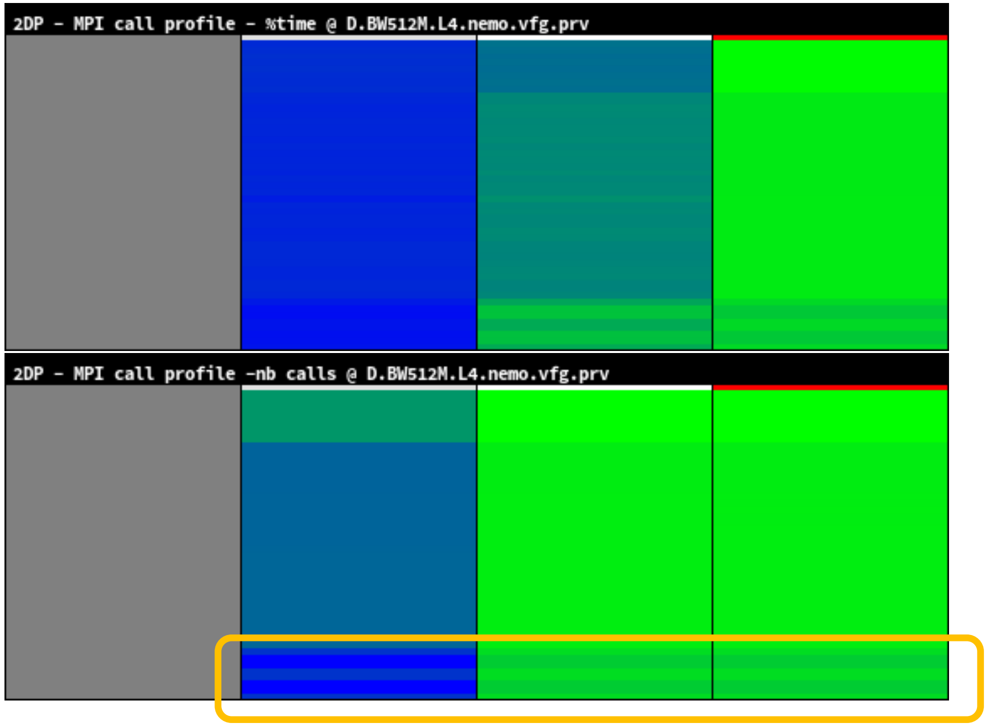


Genesis of the skew



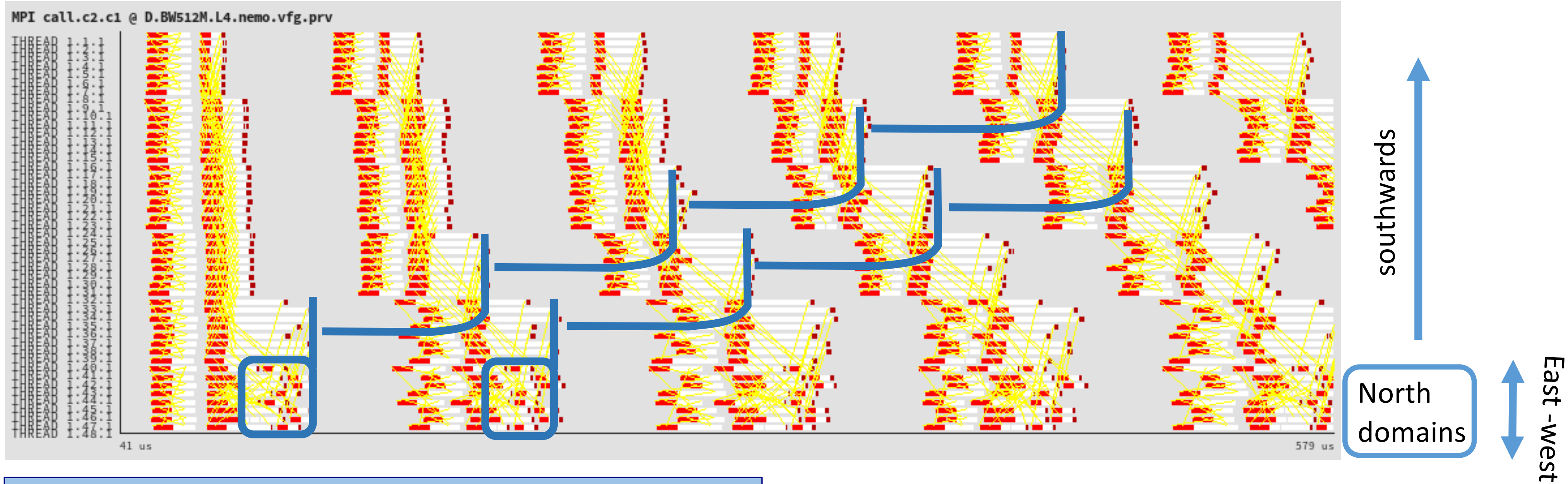
Genesis of the skew





Larger number of communication of processes in the north domain

Genesis of the skew



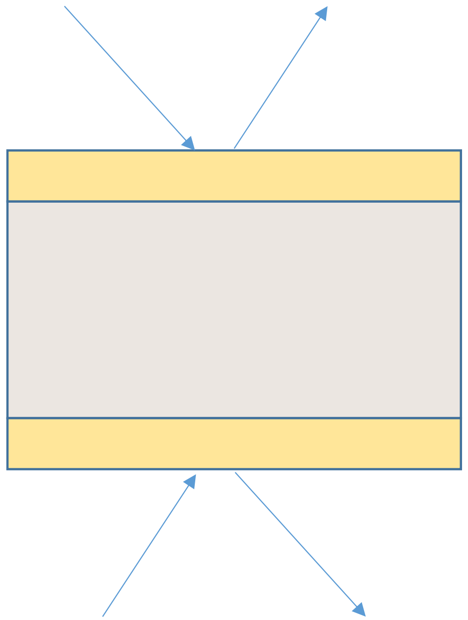
Compensate skew → Assign less load to last processes ??

Reorder communication from below and computation may “alleviate” propagation ??

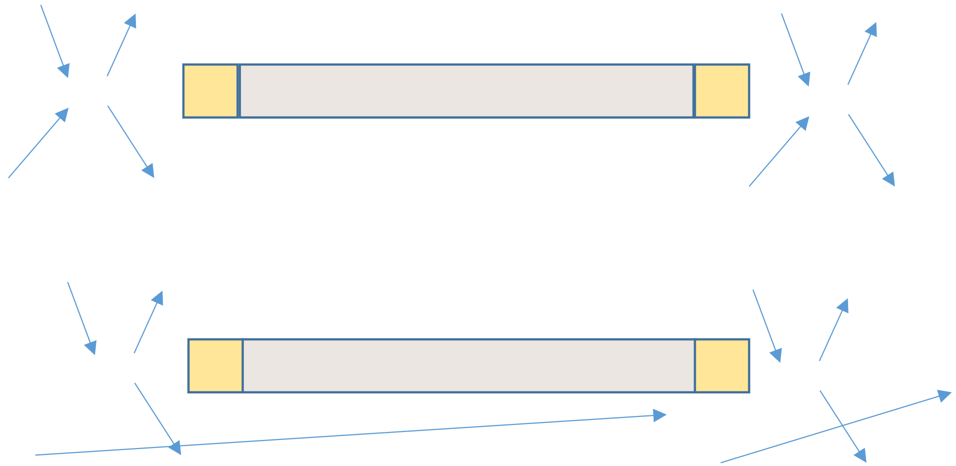
all isends before any receive ? Just on the pole?

Overlap communication - computation?

"Physical/algorithmic" structure



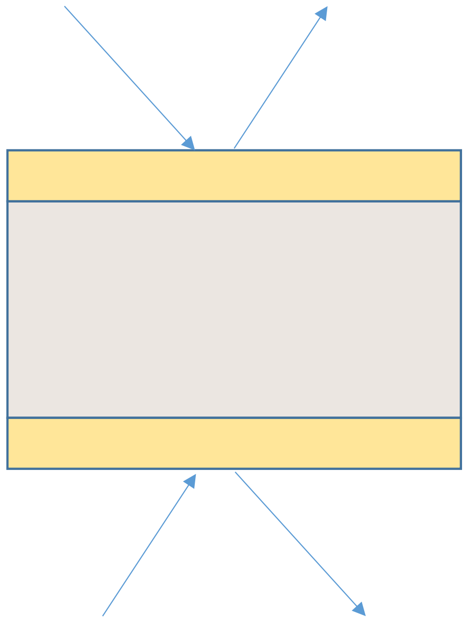
"computational" structure



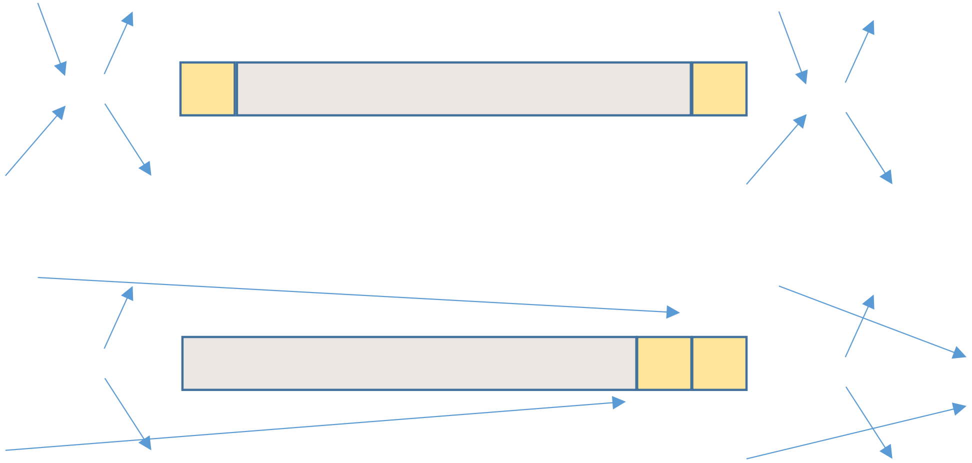
Can be reordered?
Source code reordering (irecv, wait)
tasks and dependences
(more dynamic, adaptive, overhead?)

Reorder computation ?

“Physical/algorithmic” structure



“computational” structure



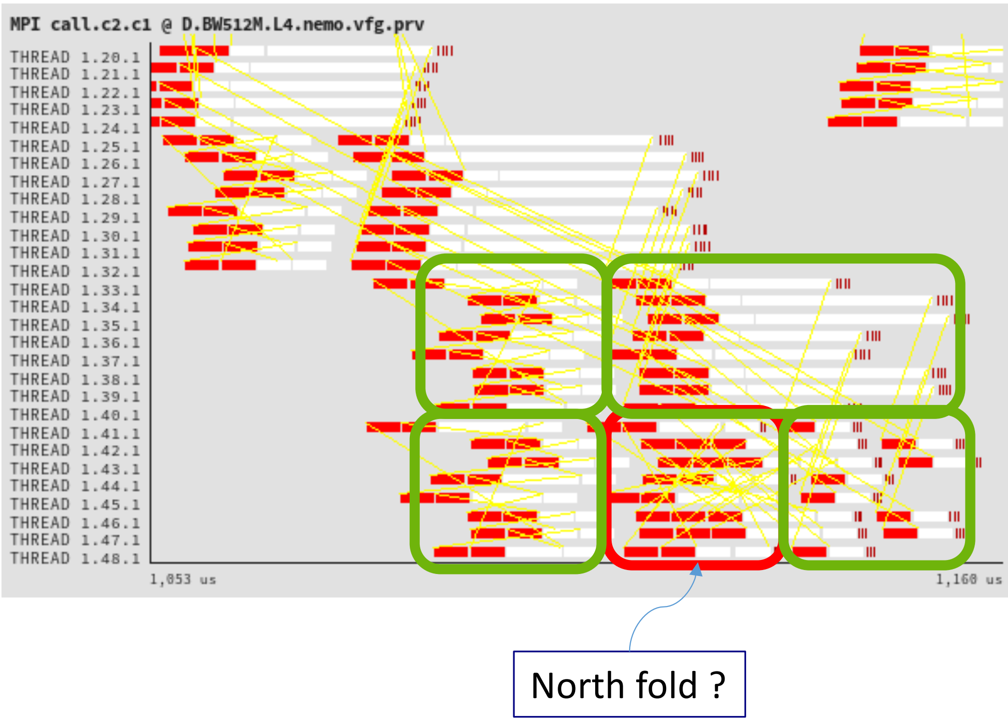
Can be reordered? Dependences?

Source code reordering

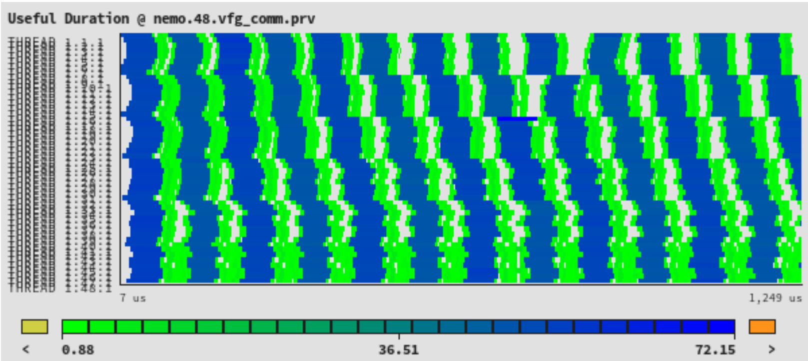
tasks and dependences

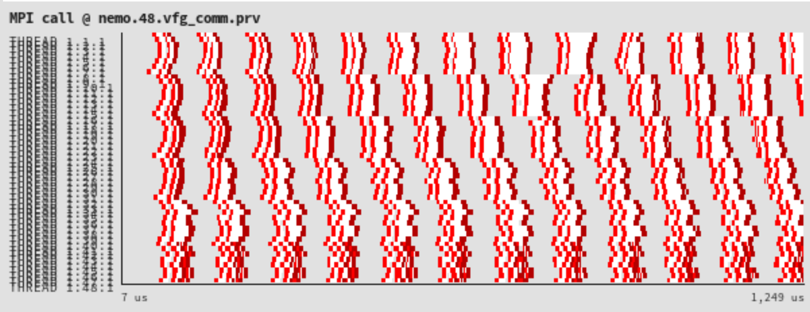
(more dynamic, adaptive, overhead?)

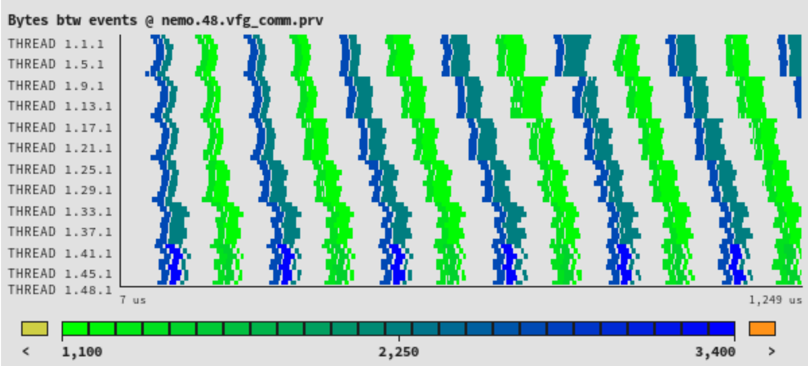
Genesis of the skew

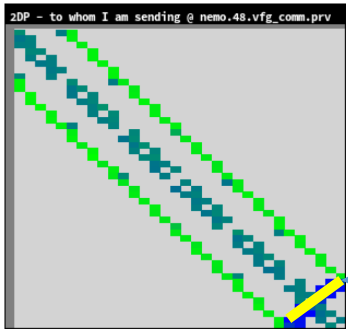


North fold







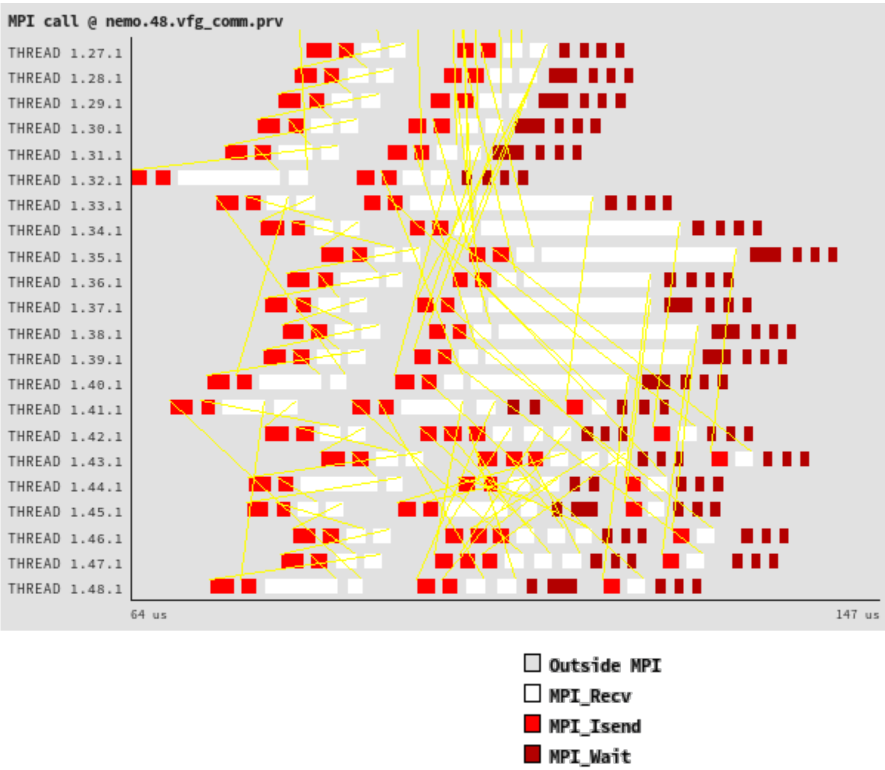


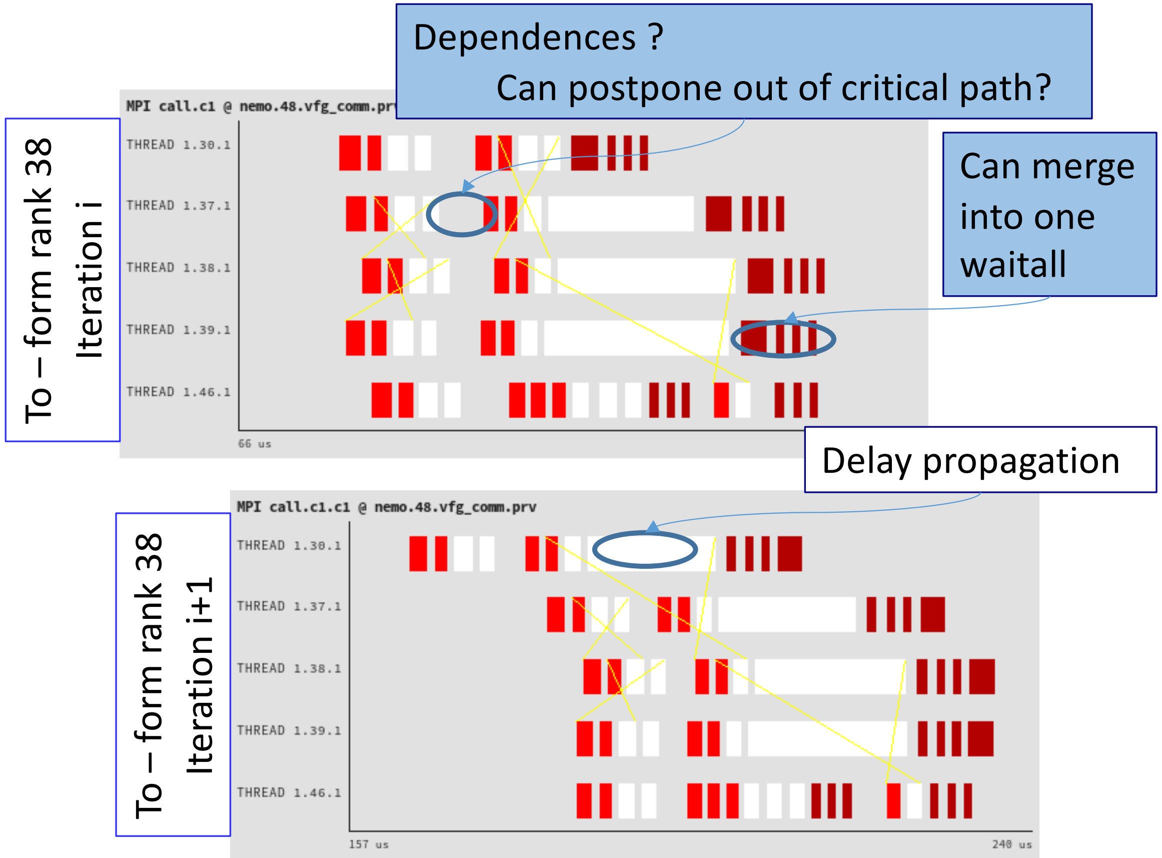
North fold local exchange

Substructure:

are the two phases dependent (potential to overlap)?

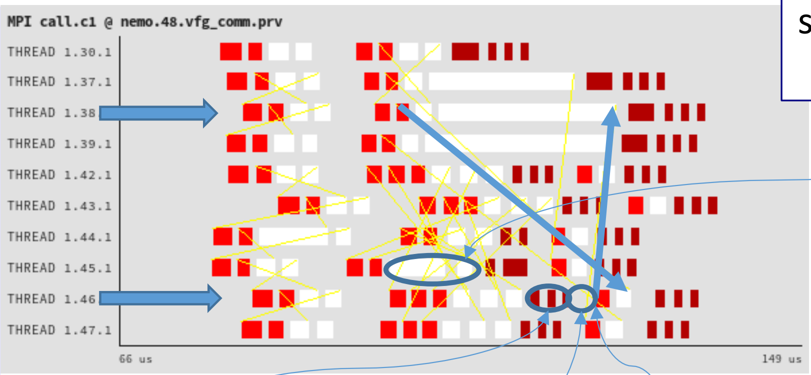
Detailed MPI call sequence





Detailed MPI call sequence

To – form rank 38 & 46
Iteration i



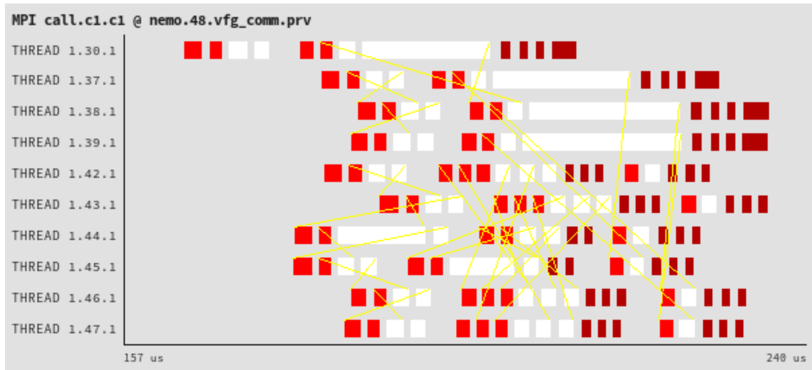
Could reorder receptions to avoid some long receives delaying others
Irecv + waitall

Reorder MPI calls?
Decouple isends from recvs?

Can be postponed ?

Can be advanced ?
before other isends??

Actual compute ?



Potential of imbalance?

Skew appear and propagates

Ideal Network

Skew “disappears”

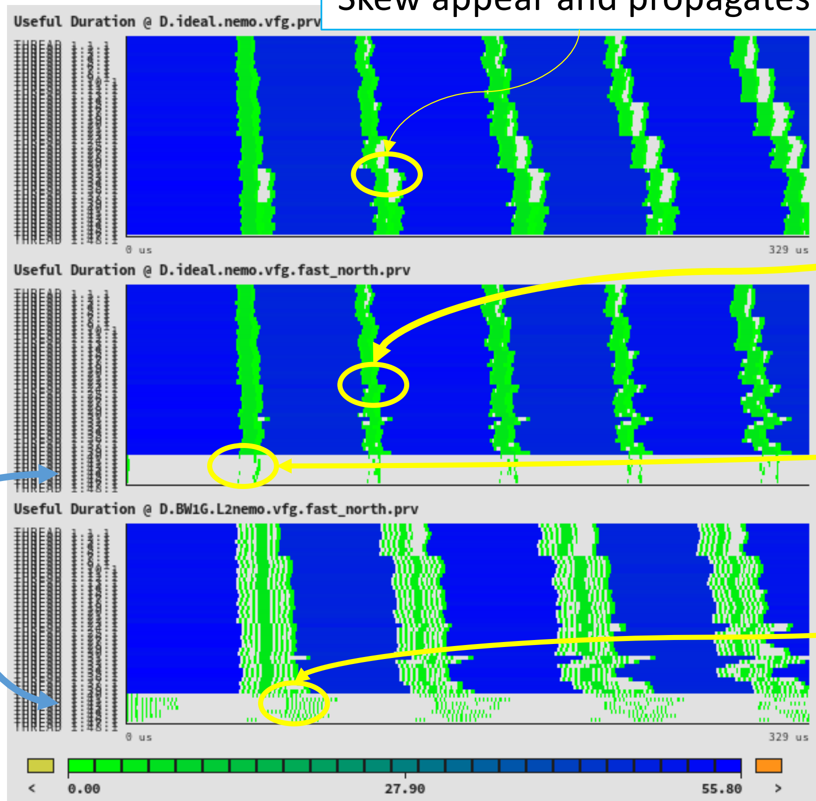
Ideal Network

Local communication not advanced !
→ Dependence

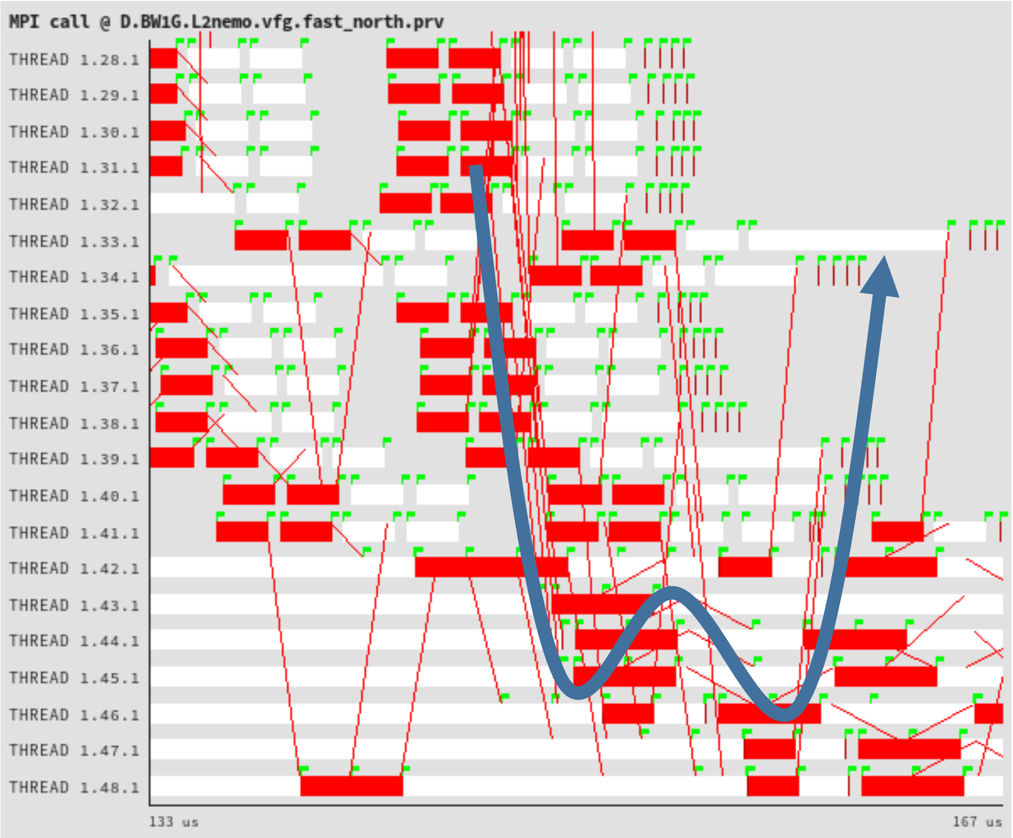
Simulation of much faster north row of processors

Latency still in the critical path …
… may be overlapped ?

“nominal” Network



Dependence chain



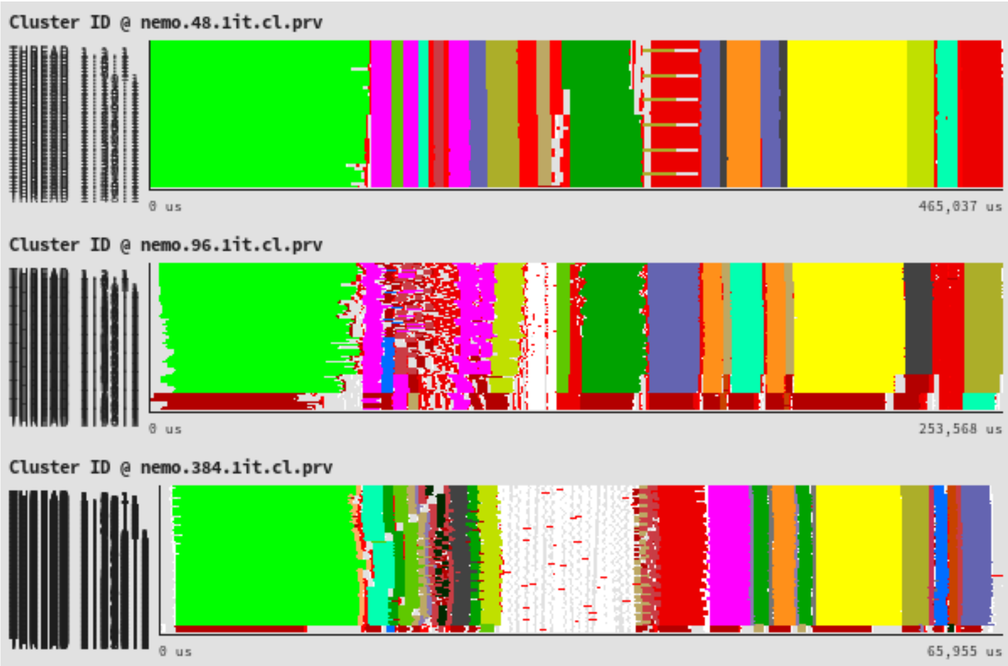
Dependence chain?
Mostly communications?
What computation?

Can be embedded within less processes ?

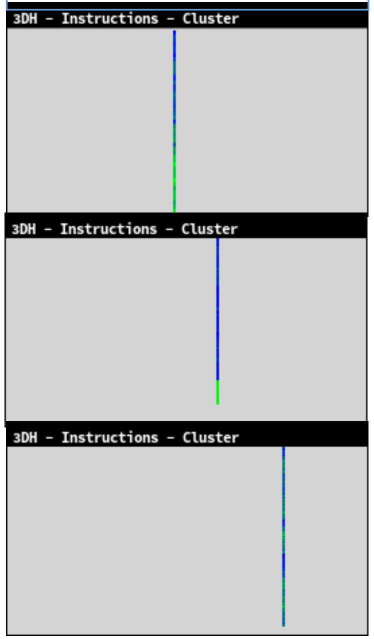
Can reorder north computations/communications ?

Can reduce #north processes? Only one?

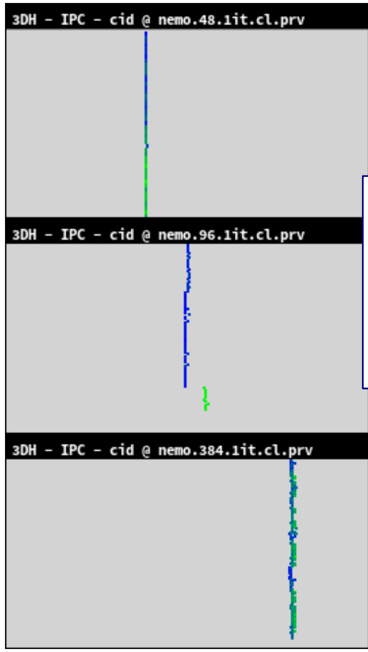
Computation scaling



Yellow Instructions “strong scaled”



Yellow IPC



Caused by locality at L1, L2, L3 level !

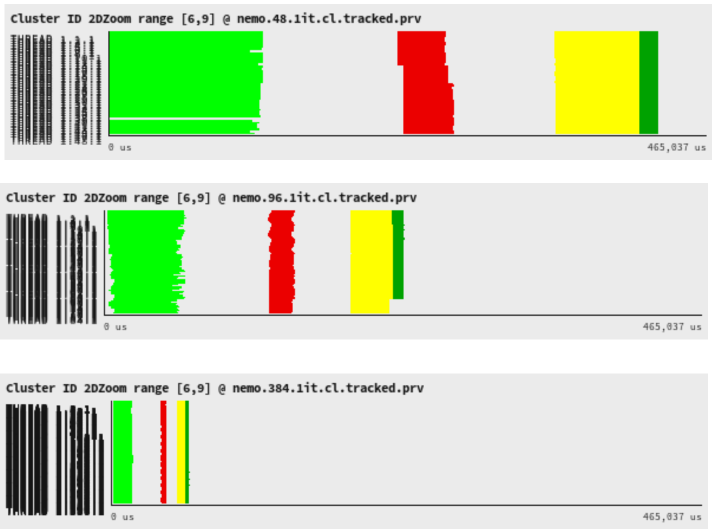
Compensating effects.
Interesting to understand individually why !

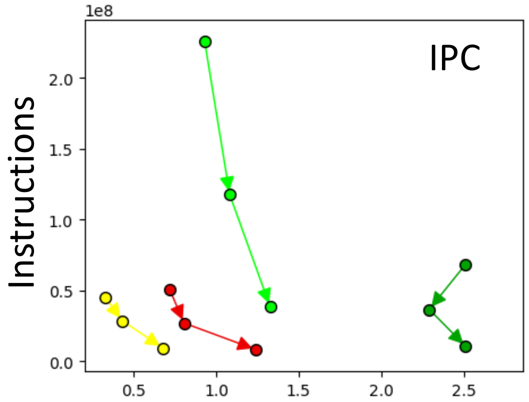
Computation Scaling

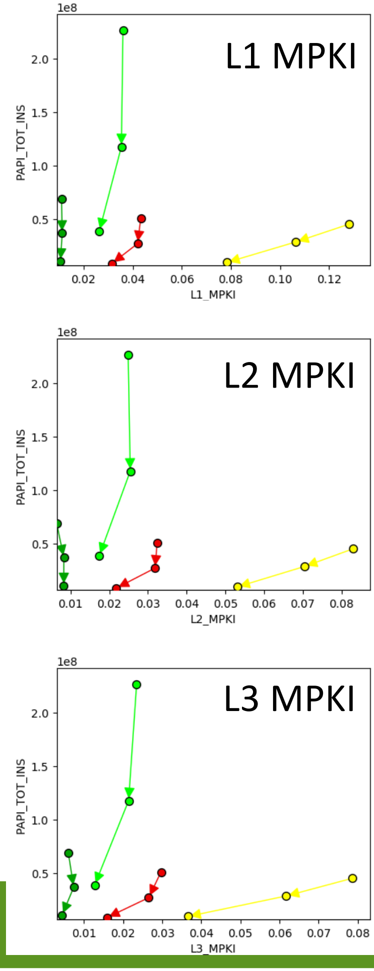
48

96

384

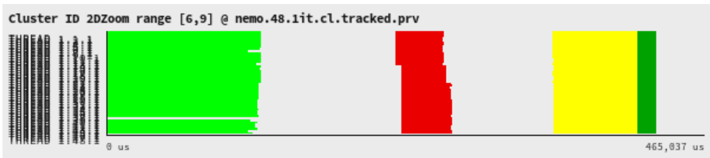




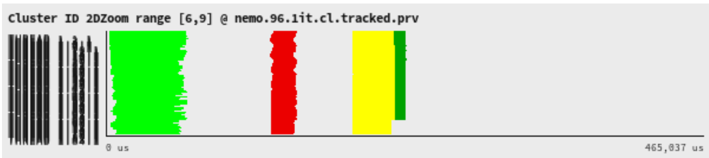


Computation Scaling

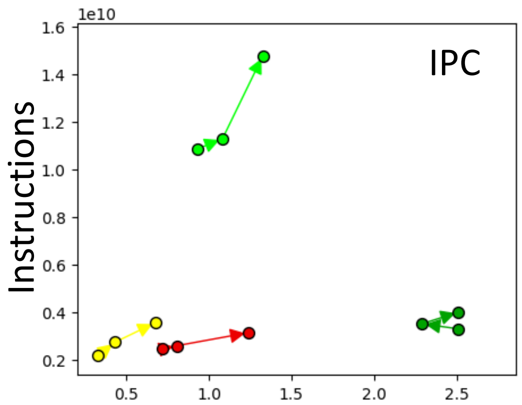
48

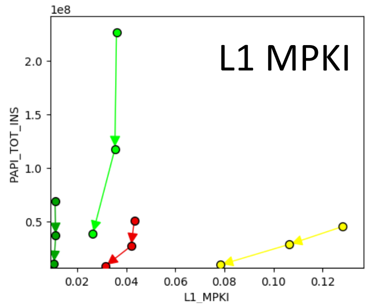


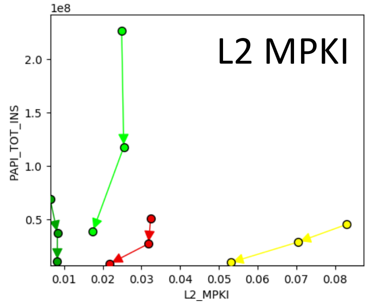
96

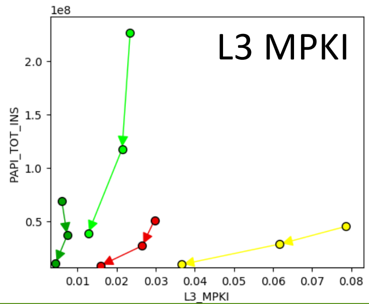


384

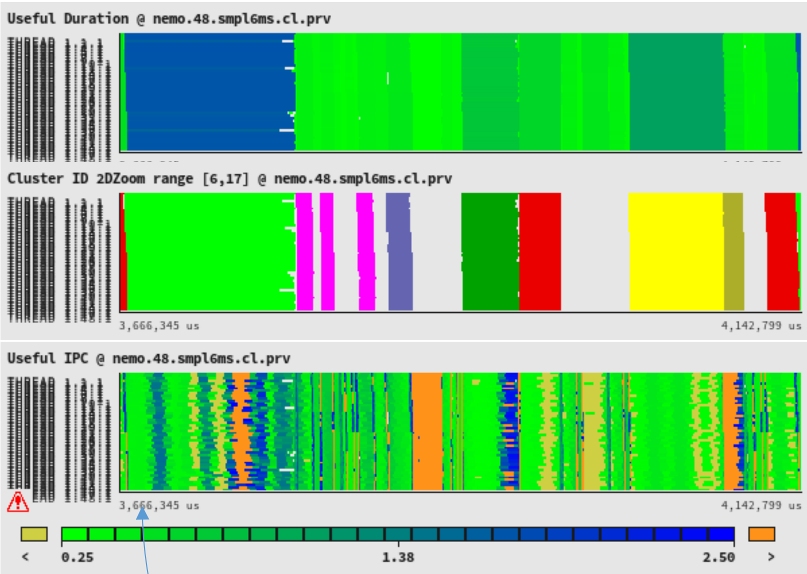


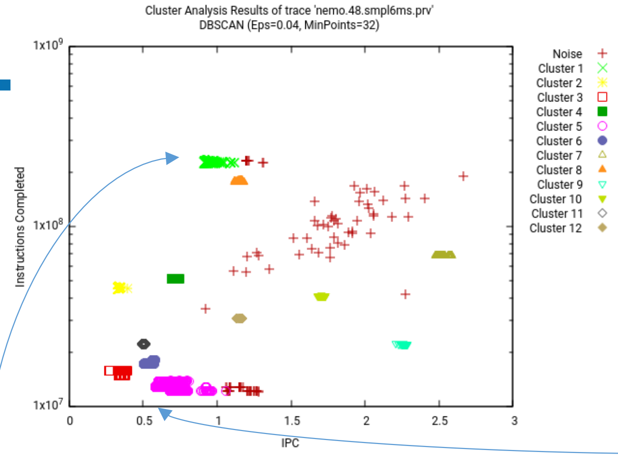






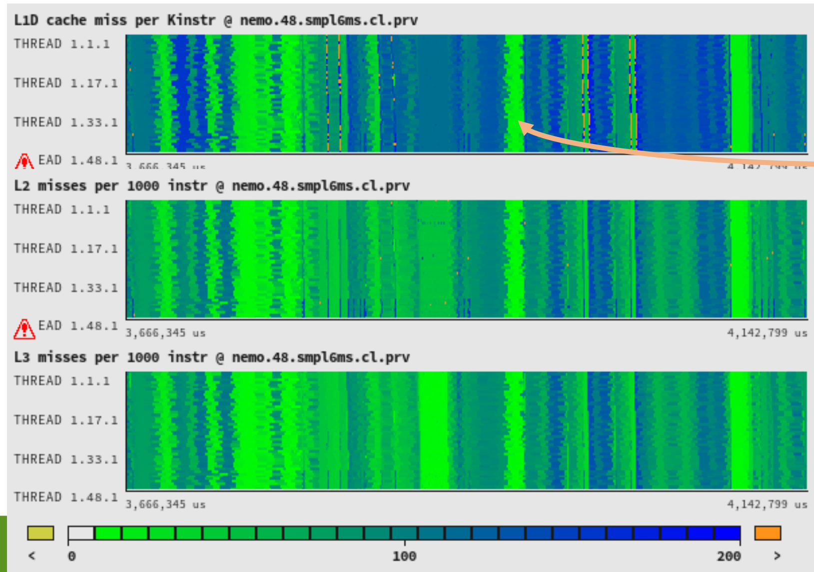
Sampled traces





Regions with poor IPC

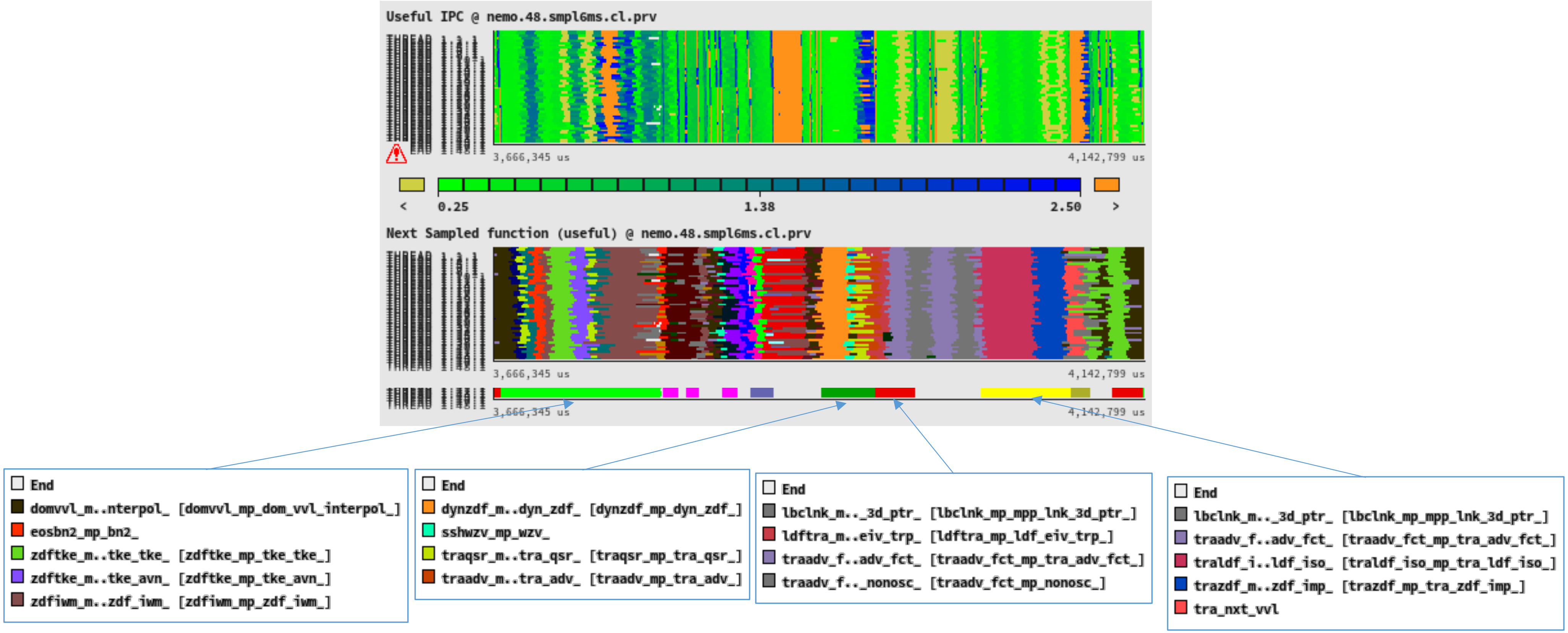
Very poor IPC sub regions within region of moderate average IPC



Cache miss ratios “explaining” IPC

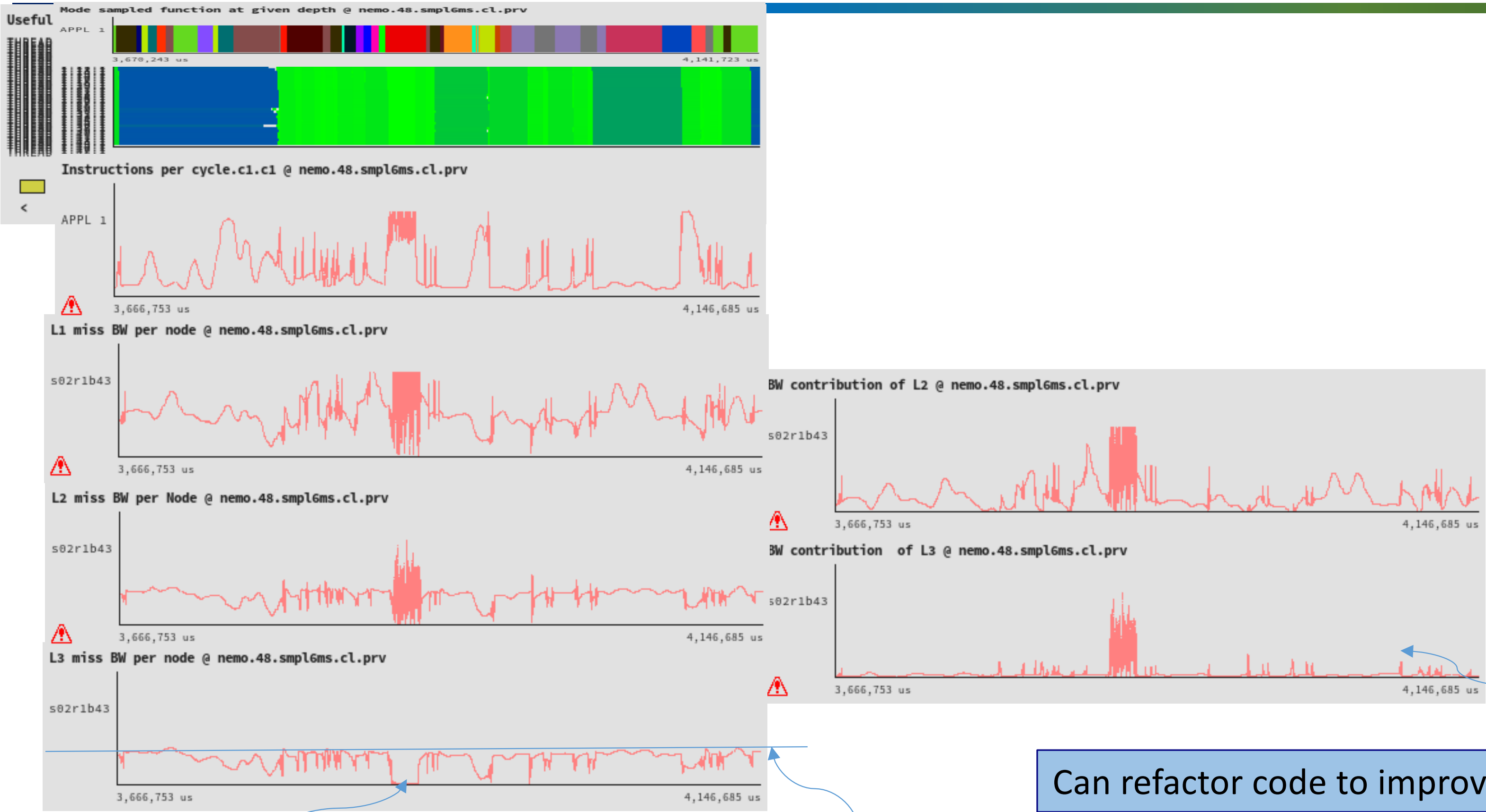
Limited benefit of L3

Sampled traces



Computation behavior

48 processes



L3 not useful elsewhere ?

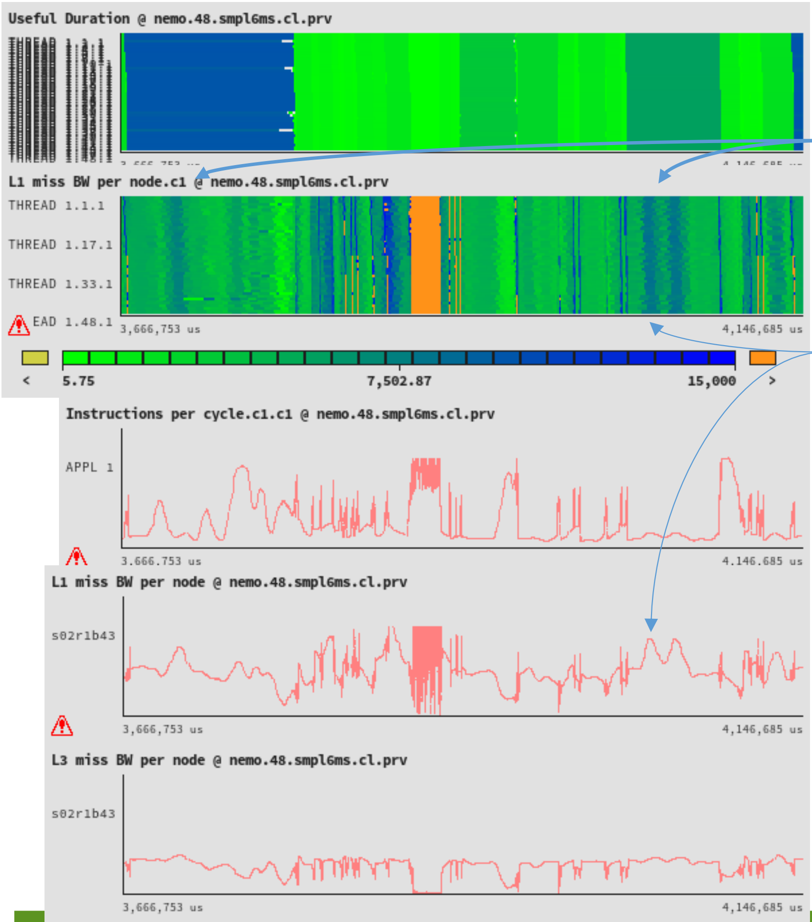
Can refactor code to improve L3 use? Blocking?

Working out of L3 in fine grain phase

Node BW limit

Co-design: no need of L3 ?

Computation behavior



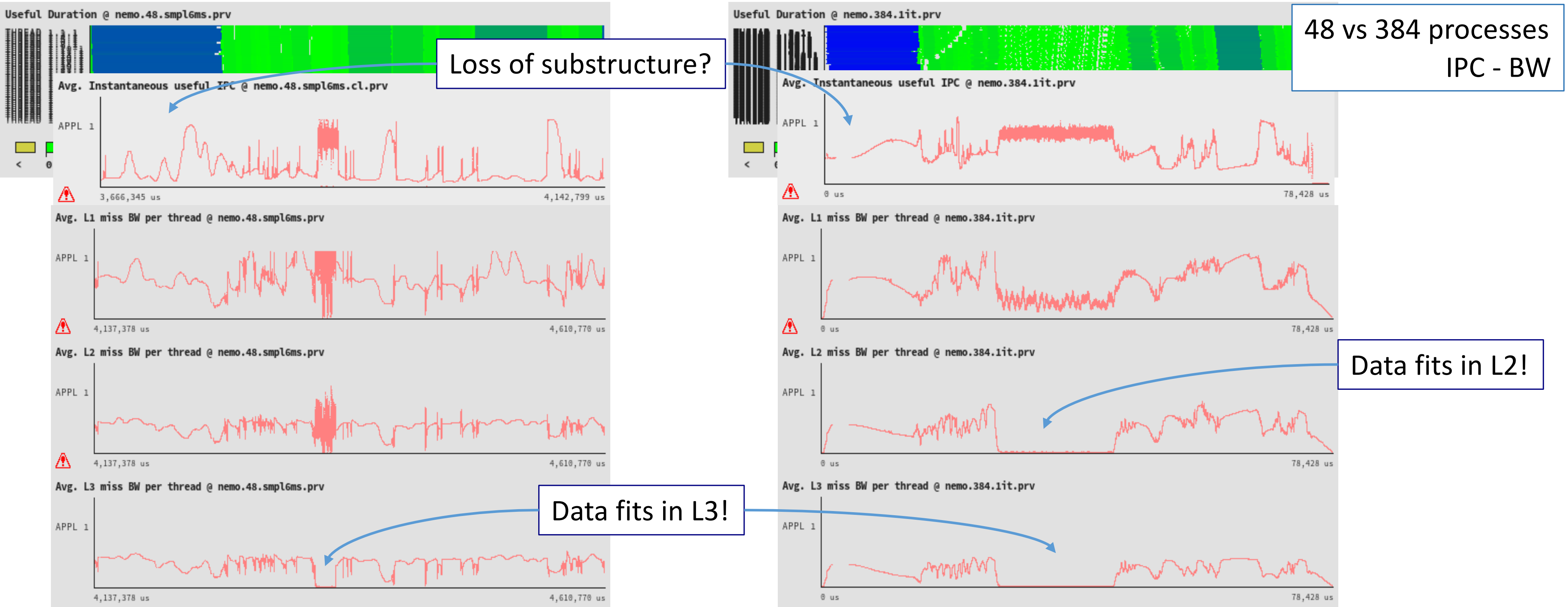
48 processes

Sub substructure

Synchronized BW demands.

Possibility to unsynchronize?

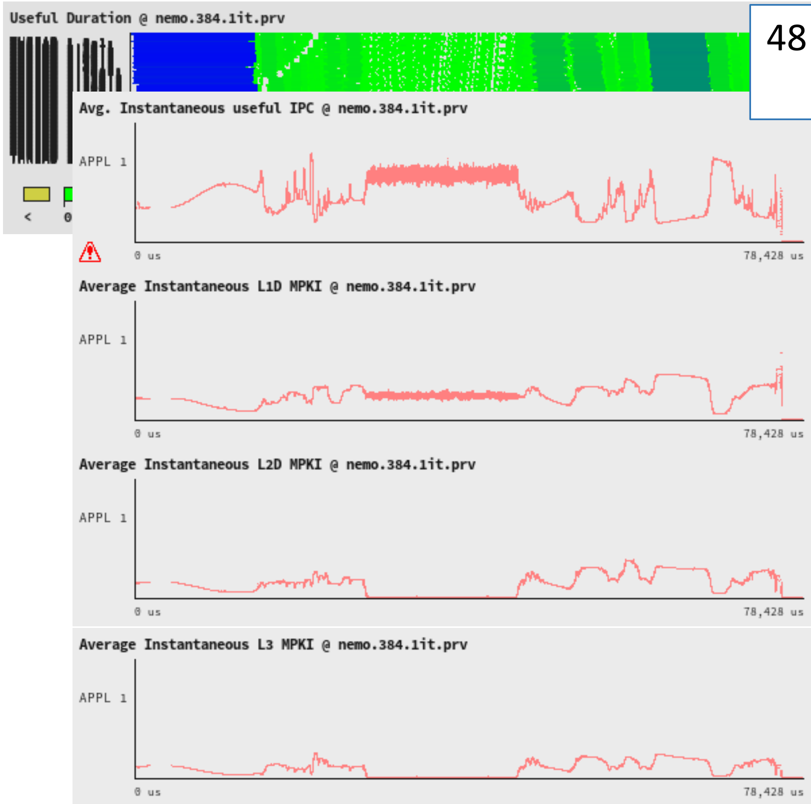
Comparative Computation behavior



Comparative Computation behavior

2.8

160

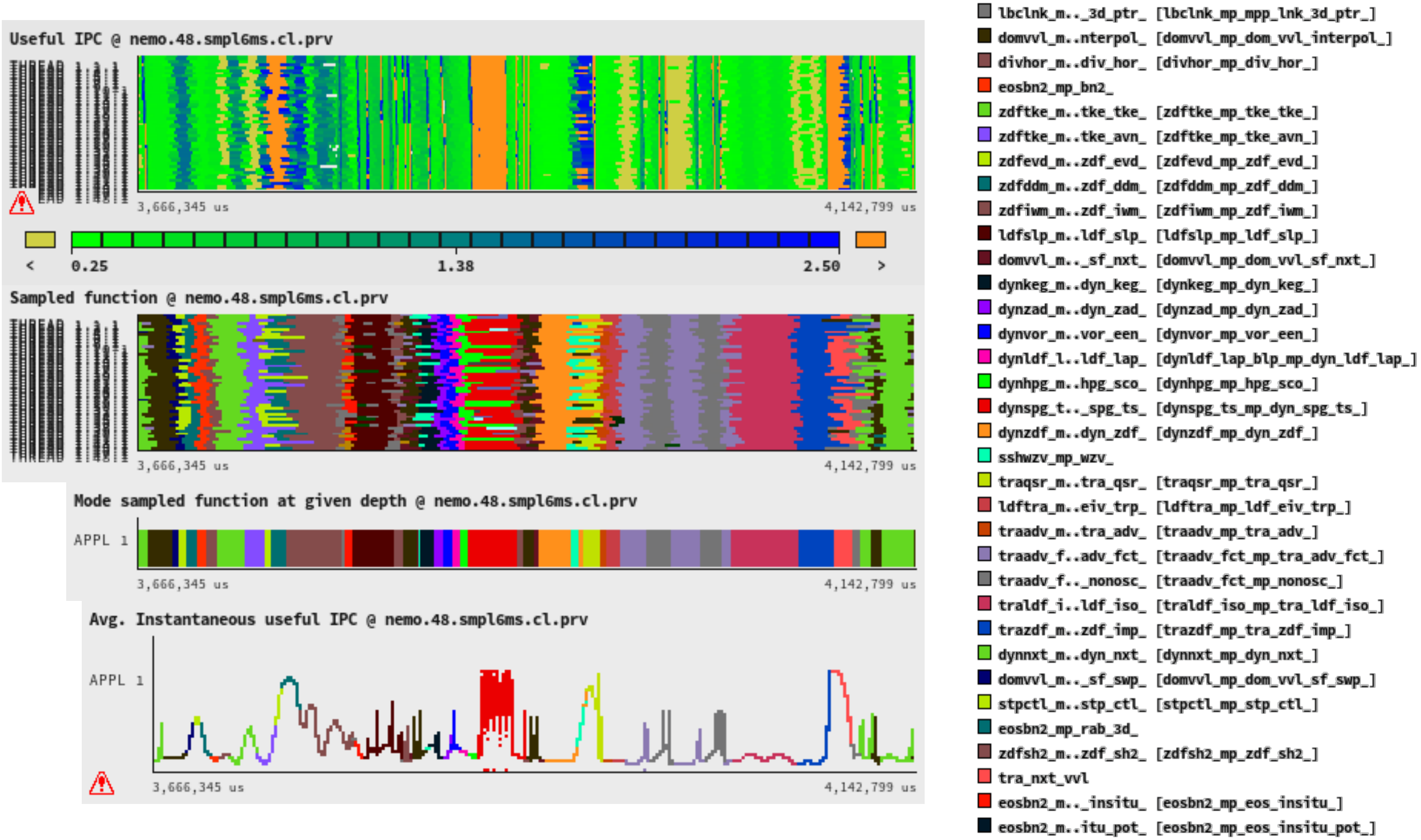


48 vs 384 processes
IPC – Miss Ratios

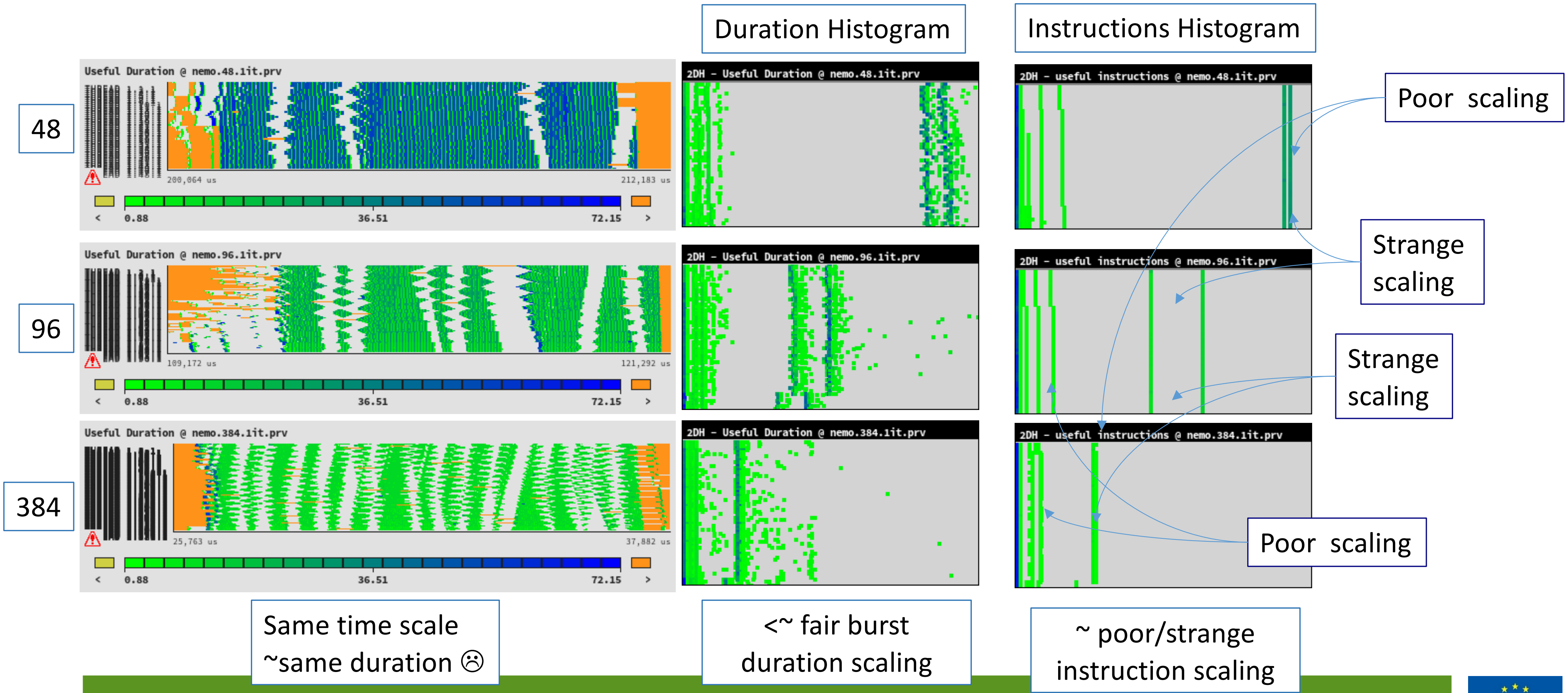
Overall improvement in Miss ratios

*Same scales for all miss ratios, same scales for all IPCs

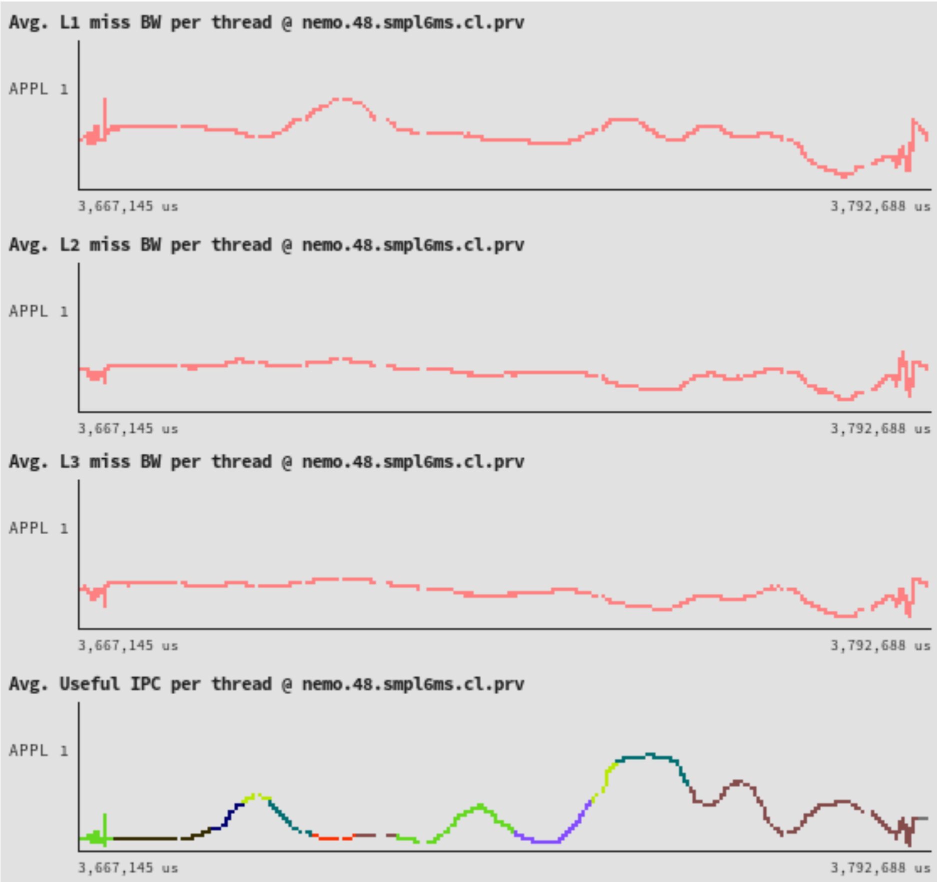
Link to Source



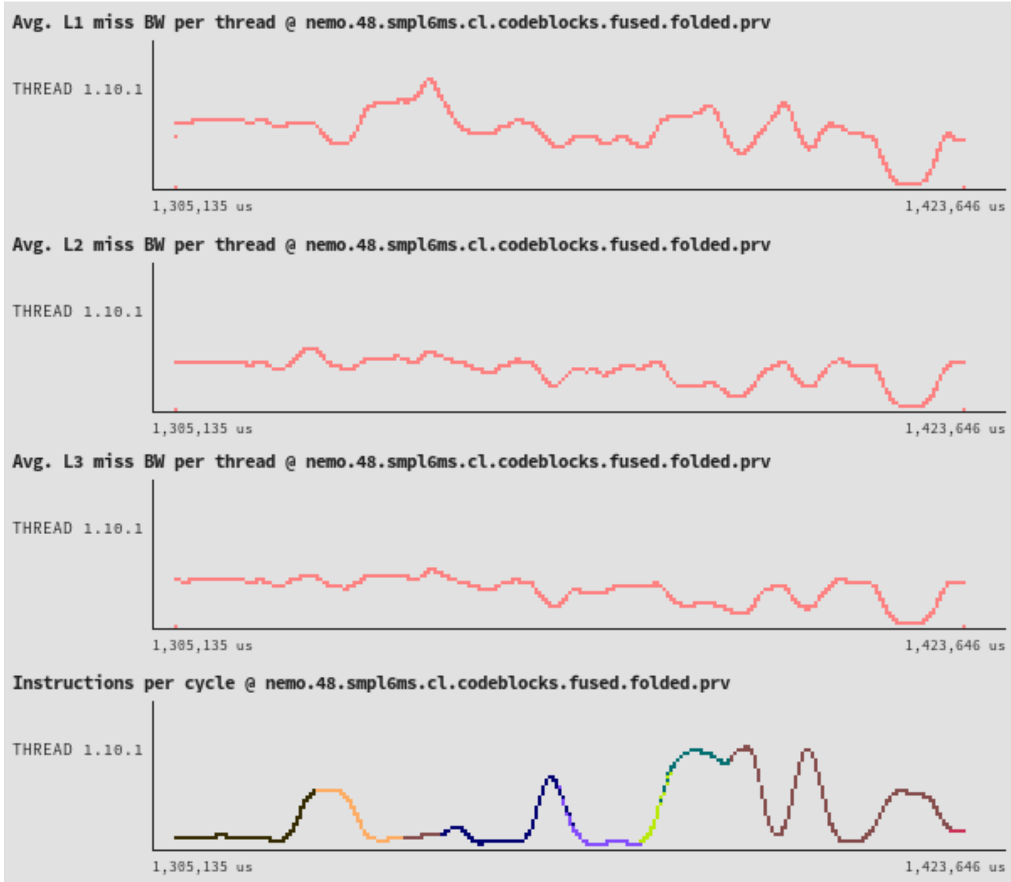
Solver Computational Scaling



Increasing accuracy ?

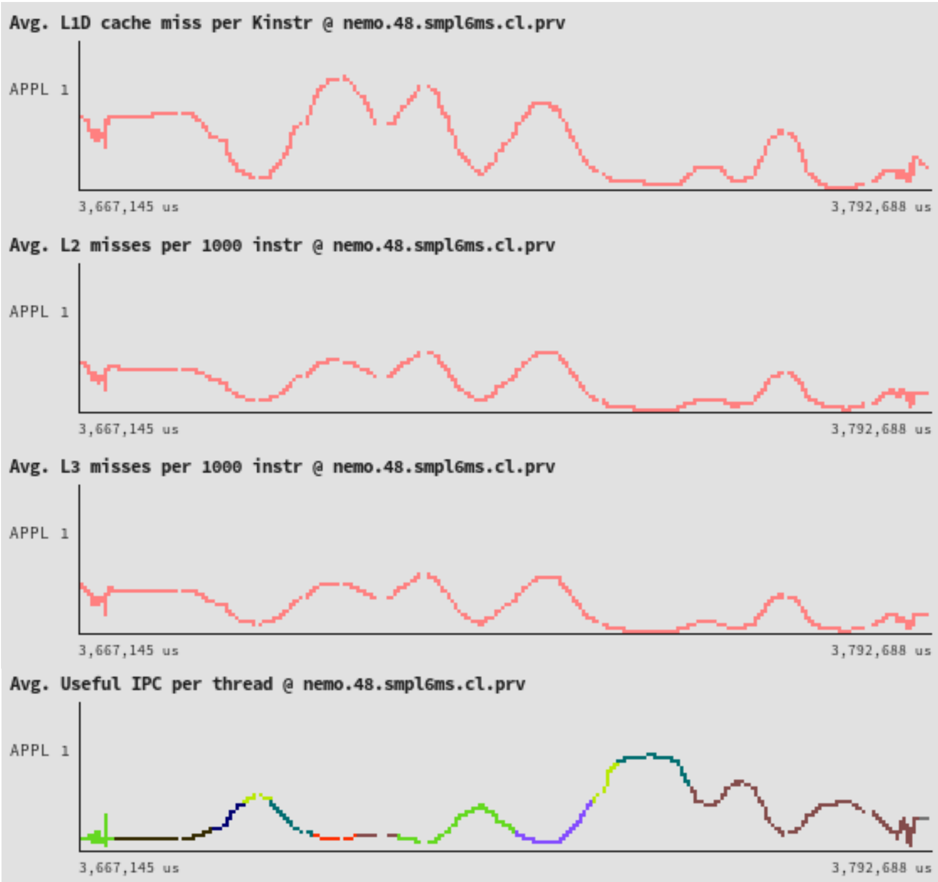


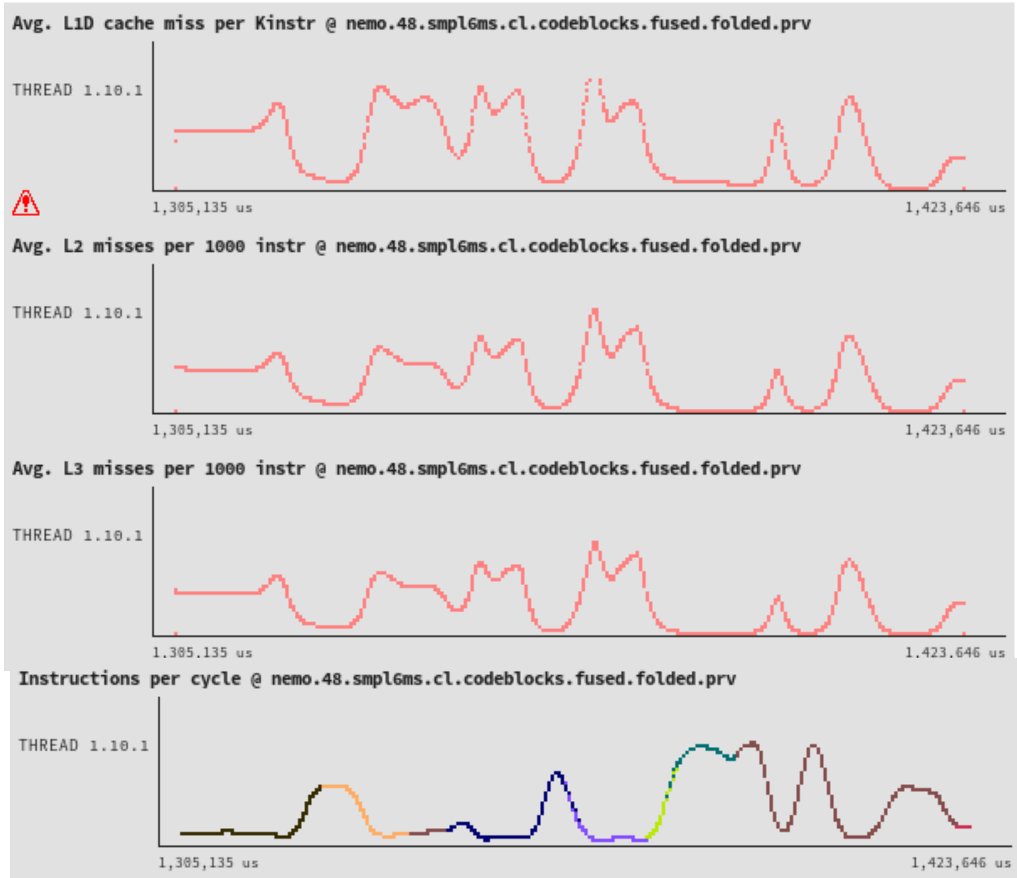
Aggregated Sampling



Folding Sampling

Increasing accuracy ?





Aggregated Sampling

Folding Sampling
Cluster 1

Recommendations

- Merge sequence of waits for Isends to waitall
- Reorder north: advance isends postpone recs   (if possible)
- Assign less load to north fold to take internal communication out of the critical path   (combined with reordering of comms)
- Reduce #north processes?
  - → only one ?
- Increase granularity in solver
  - Gather to single rank per node → Solve → Scatter
- OpenMP in solver
  - → Tasks with dependences between communications and computations for out of order execution and noise tolerance

Possibility for a Mockup POC

Recommendations

- Try to improve locality → better IPC
  - L3 usage: Blocking?
- Convergence of numerical method: Jacobi vs. Gauss-Seidel ?

Performance Optimisation and Productivity

A Centre of Excellence in Computing Applications

Contact:

https://www.pop-coe.eu

mailto:pop@bsc.es

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 676553.