POP2_AR_092

POP CoE

Piernik Assessment (POP2)

Federico Panichi (federico.panichi@nag.co.uk), NAG - September 2020

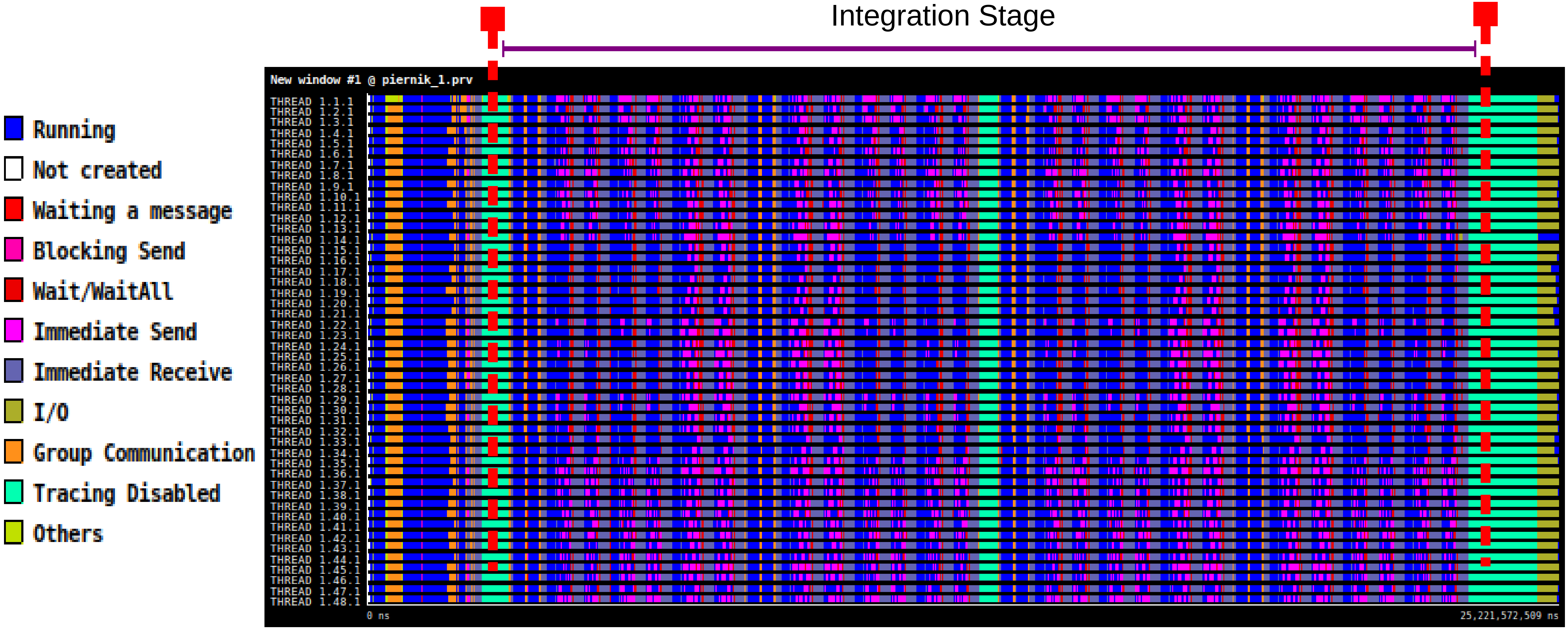
EU H2020 Centre of Excellence (CoE) **1 December 2018 – 30 November 2021**

Grant Agreement No 824080

Background

- Applicant: Michał Hanasz
- Name of code: Piernik (August 2020)
- Scientific/technical area: CFD
- Programming: Fortran with MPI parallelism
- Platform: MareNostrum4
  - 1 node with 2x 24 core Intel Xeon Platinum 8160 24C at 2.1 GHz
- Scale: 1 to 3 nodes
- POP collected the performance data, code has been compiled with INTEL compiler 2017.4
- Tools used: Extrae 3.8 & Paraver 4.8
- Input test case is named: NAG_galdisk (only `fluid_update()`)
- Input cases: $b_z = 16^3$ and 32^3;

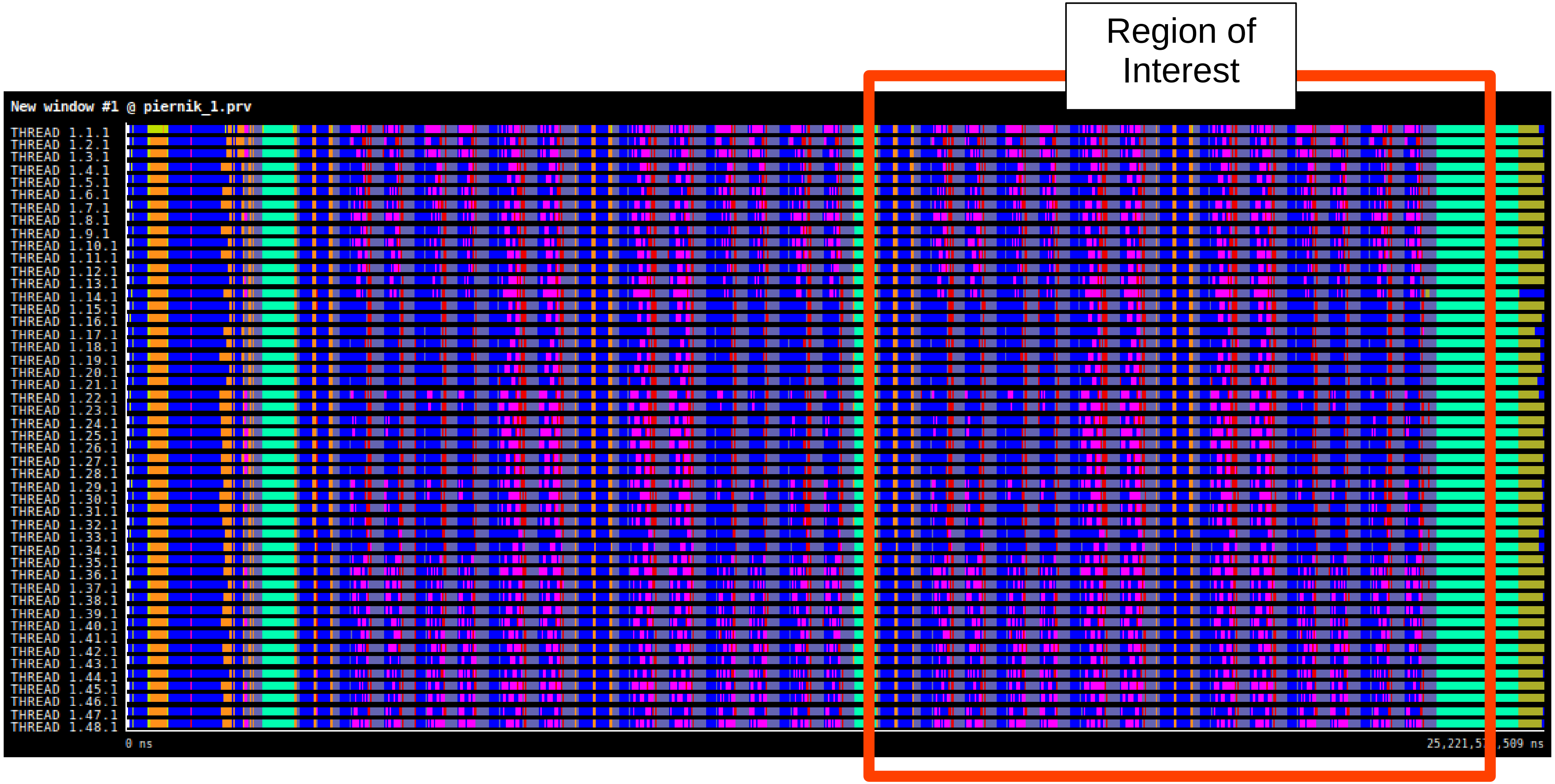
Application Structure



Application Structure

- the Region of Interest (RoI) is defined as the **second** iteration step of the Integration region;

- First time step is longer due to additional initialization and may skew the analysis, the second and next steps have all constant length;



POP Metrics Terminology - 1

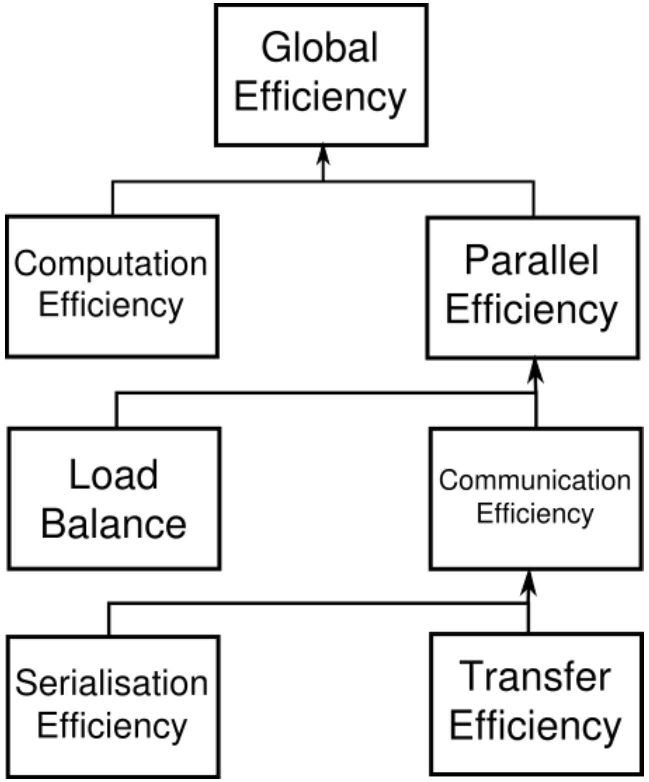
1. **Global efficiency** = parallel efficiency * computational scaling

2. **Parallel efficiency** = average (useful computation) / runtime
   - *If the code is perfectly parallelised → runtime = average (useful computation)*
   - *Useful computation excludes time within MPI*

3. **Load Balance** = average (useful computation) / maximum (useful computation)

4. **Communication efficiency** = maximum (useful computation) / runtime
   - *If time overhead of MPI = 0 → runtime = maximum (useful computation)*

5. **Transfer efficiency** = ideal runtime / runtime
   - *Assuming infinite bandwidth and zero latency network*



POP Metrics Terminology - 2

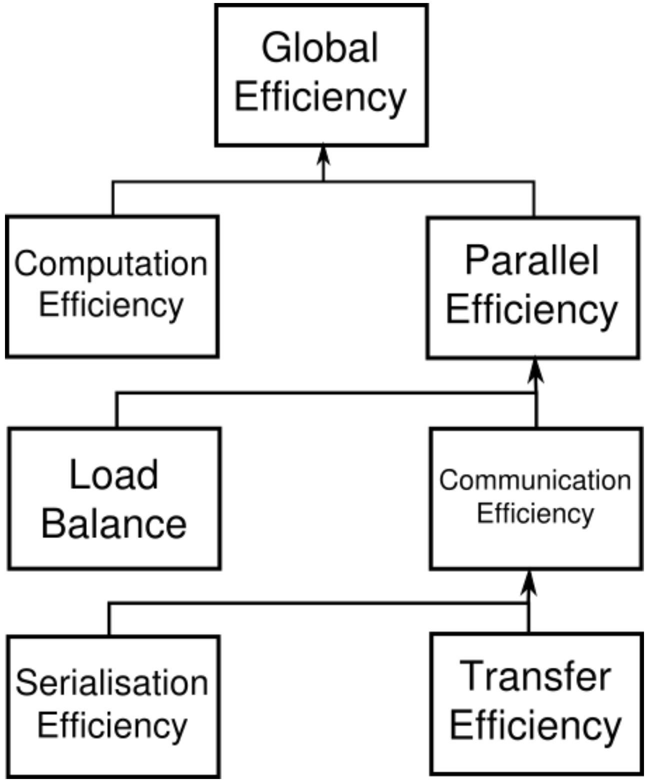
6. **Serialisation efficiency** = remaining cost of MPI
   - *Time spent waiting idle for other MPI ranks to end/start communicate*

7. **Computational scaling** = Instruction scaling * IPC scaling * Frequency scaling.

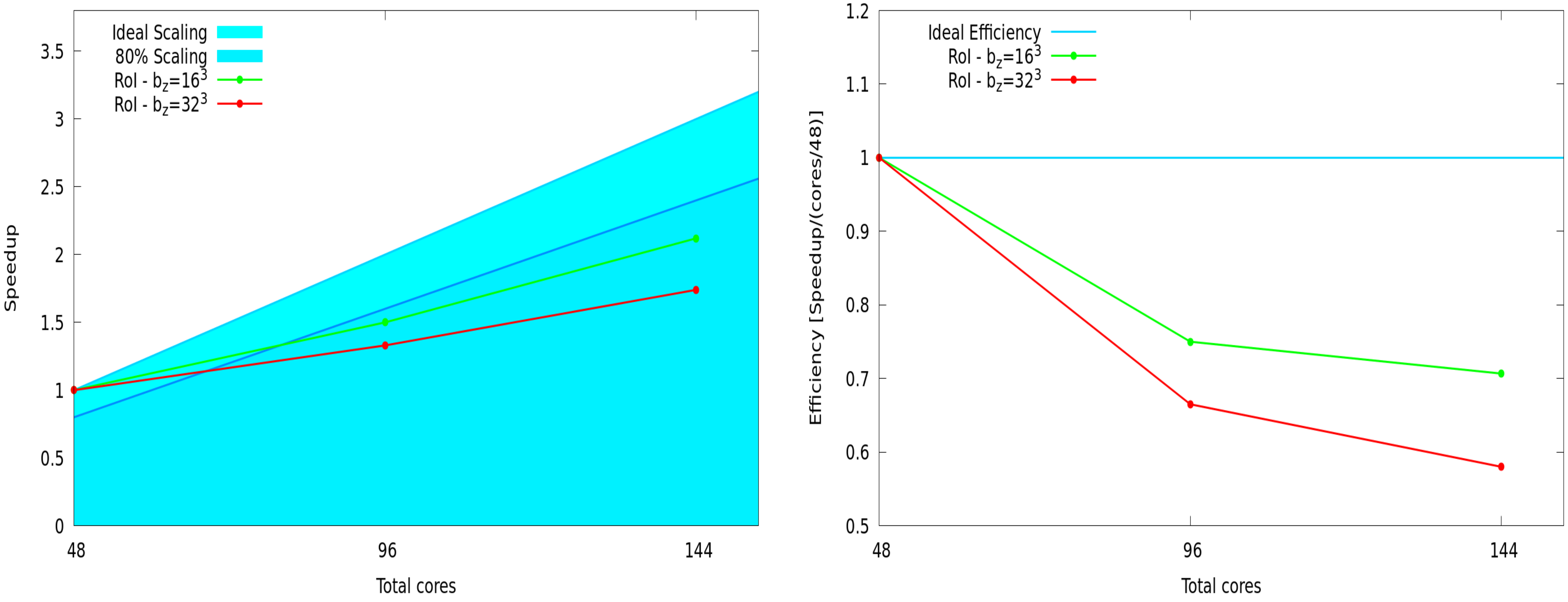
   IPC scaling = it compares IPC to the reference;

   Instruction scaling = ratio of total number of useful instructions for a reference case (e.g., 1 processor) compared to values when Increasing the numbers of processes.

   Frequency Efficiency = ratio of processor frequencies for a reference case compared to values when increasing the number of Processes.



Scaling and Efficiency Plot



POP Metrics: b_z = 16, 16, 16

Number of nodes	1	2	3
Global efficency	78.02	58.71	55.04
Parallel efficency	78.02	65.72	60.08
Load balance	89.92	81.32	78.00
MPI Communication efficiency	86.76	80.82	77.02
Serialization efficency	92.92	92.08	90.72
Transfer efficency	93.38	87.77	84.90
Computational Scaling	100.00	89.33	91.61
IPC scalability	100.00	90.28	92.91
Instruction scalability	100.00	99.05	98.78
Frequency scalability	100.00	99.89	99.82

Number of nodes	1	2	3
Average IPC	2.04	1.84	1.89
Frequency [Ghz]	2.09	2.09	2.09

POP Metrics: b_z = 32, 32, 32

Number of nodes	1	2	3
Global efficency	80.92	53.62	46.90
Parallel efficency	80.92	61.95	53.95
Load balance	90.16	71.10	65.99
MPI Communication efficiency	89.75	87.13	81.76
Serialization efficency	94.51	91.94	88.90
Transfer efficency	94.96	94.78	91.98
Computational Scaling	100.00	86.56	86.93
IPC scalability	100.00	86.91	87.35
Instruction scalability	100.00	99.59	99.53
Frequency scalability	100.00	100.00	99.98

Number of nodes	1	2	3
Average IPC	2.04	1.83	1.83
Frequency [Ghz]	2.09	2.09	2.09

POP metrics: Discussion

1. Global efficiency for the RoI drops to less than 60% on 3 nodes for the 16^3 case and less than 50% on 3 nodes for the 32^3 case;

2. Parallel efficiency drops for two and three nodes:

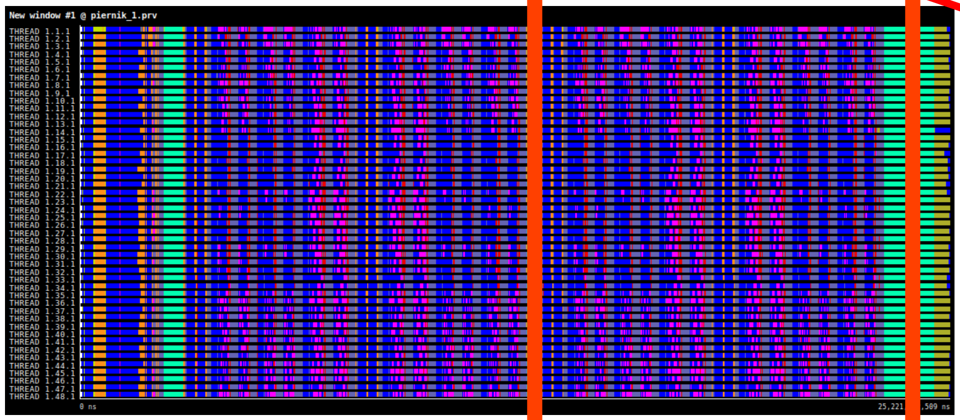
- From 78% on one node down to 60% on three nodes, for the $b_z = 16^3$ case;
- From 80% on one node down to 53% on three nodes, for the $b_z = 32^3$ case.
- *Transfer efficiency is high (>90%);*
- *For this test-case, poor Load balance is the main factor that limits scalability of the application;*

3. Computational Scaling in the RoI is high for all the nodes (e.g.: >80% on 3 nodes):

- *Good IPC scaling (and very high average IPC): good management of vectorization;*
- *Low IPC for specific functions (e.g.: compute_mr_recv) that has low impact on the overall calculation and are related to MPI calls;*

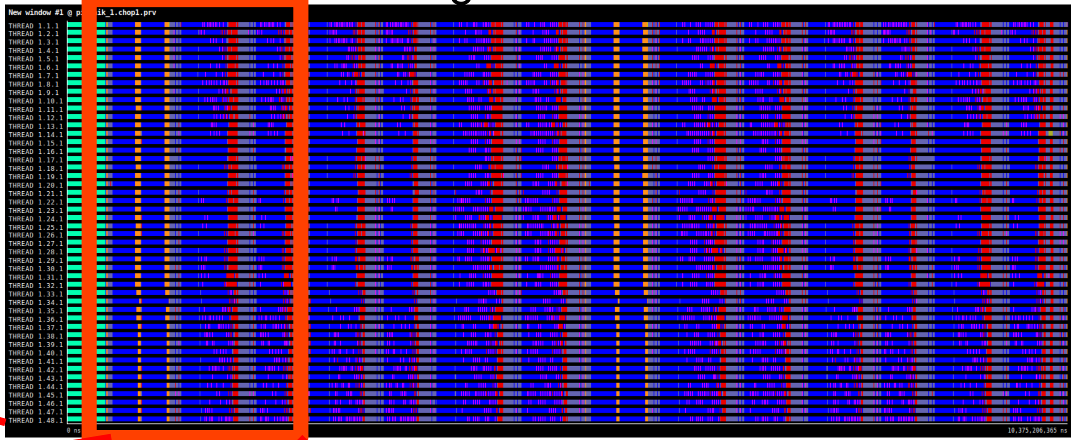
4. Fixing the Load Ballance to 100% boosts Global efficiency up to 70% on three nodes for both the $b_z = 16^3$ and $b_z = 32^3$ cases.

Zoom on the RoI and repetitive pattern



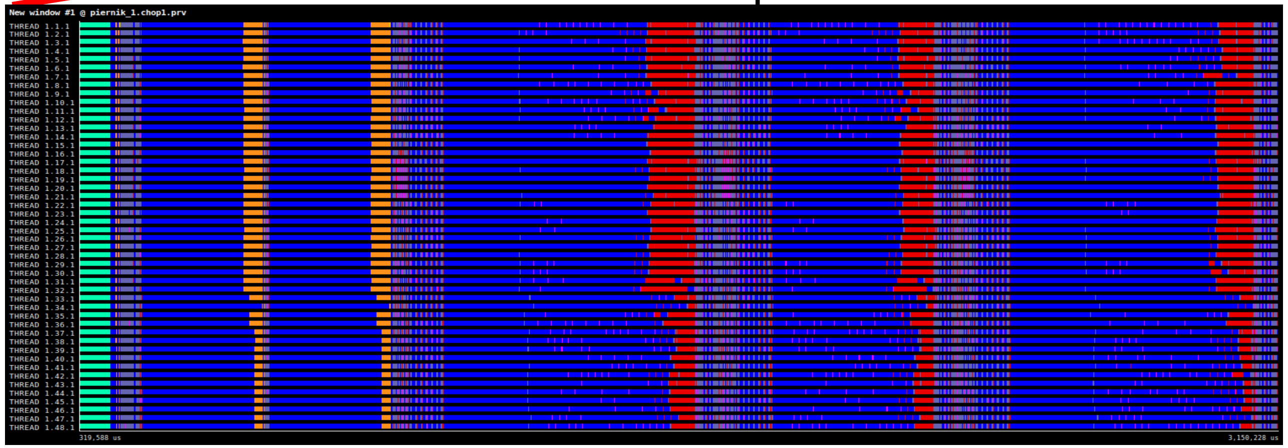
Full timeline

Region of Interest



- Running
- Not created
- Waiting a message
- Blocking Send
- Wait/WaitAll
- Immediate Send
- Immediate Receive
- I/O
- Group Communication
- Tracing Disabled
- Others

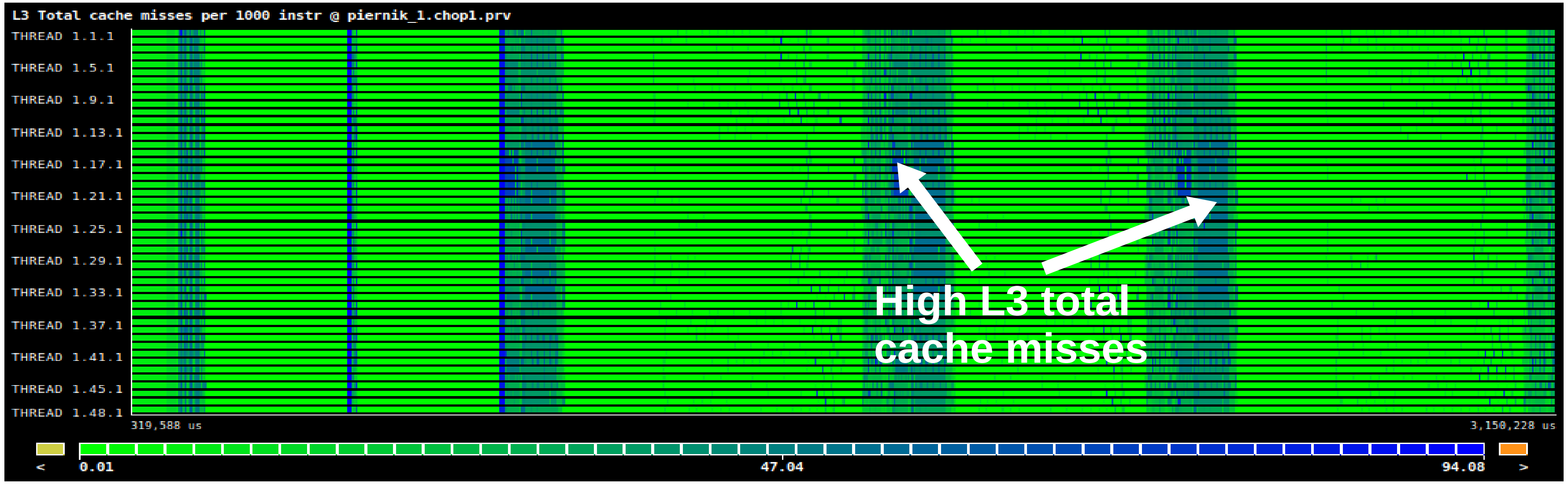
Zoom on the part of the RoI

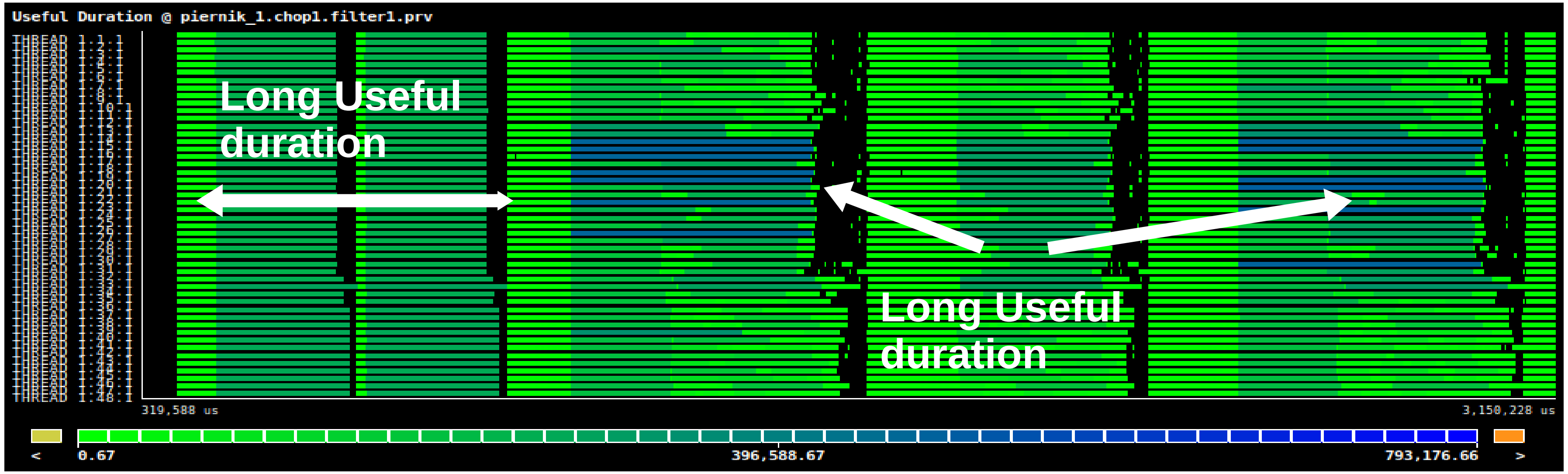


Two series of zooms in the timeline:
- First zoom: RoI (red rectangle on first image);
- Second zoom: beginning of the RoI.

Repetitive pattern in the RoI, thus we focus on a sub-region of it

L3 cache misses and Useful duration (b_z = 16)





- No relation between L3 cache missing per 1000 instructions and useful Duration:

No clear relation w.r.t. IPC;

Summary and recommendations

SUMMARY:

1. L3 cache misses doesn’t seems to relate with Useful duration;

2. Instruction scalability is good with both block sizes (16 and 32);

3. For this test-case, the scalability of the application was limited mainly due to poor load balance (consequence of the test-case set-up);

RECOMMENDATIONS:

1. Using a different block size (e.g. from 16 <--> 32) might improve MPI communication efficiency;

2. Investigate the reason for L3 cache missing: addressing the problem might slightly improve the Global efficiency;

Performance Optimisation and Productivity

A Centre of Excellence in HPC

Contact:
https://www.pop-coe.eu

Mail to:
pop@bsc.es

Twitter:
@POP_HPC

This project has received funding from the European Union‘s Horizon 2020 research and innovation programme under grant agreement No 676553 and 824080.