MONT-BLANC

http://montblanc-project.eu @MontBlanc_EU

# Applications: evolution & behaviour on new Arm HPC platforms

Mont-Blanc workshop @ Arm Research Summit

*Cambridge – 2018, Sep 18th*

Filippo Mantovani

filippo.mantovani@bsc.es

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement n. 671697

# Introduction: Why?

**Objective #1, from the DoW of Mont-Blanc 1:**

[...] We will also deploy a complete system software stack to enable development, analysis, and efficient execution of HPC applications targeting Exascale scientific challenges.

Dibona, a playground for...

1. **Micro-benchmarking**: for measuring micro-architectural implementation and integration
2. **Performance measurements**: "simple" execution of benchmarks and applications studying execution time and scalability
3. **Survival course**: since 2011 we tested 12 clusters! We need ways to survive hardware refactoring with decent performance

# Introduction: Who and What?


Alya, OpenIFS, HPCG, Linpack, Lulesh, NTchem


HPGMG, Tangaroa


Solvers, CARP, Parallelization in time

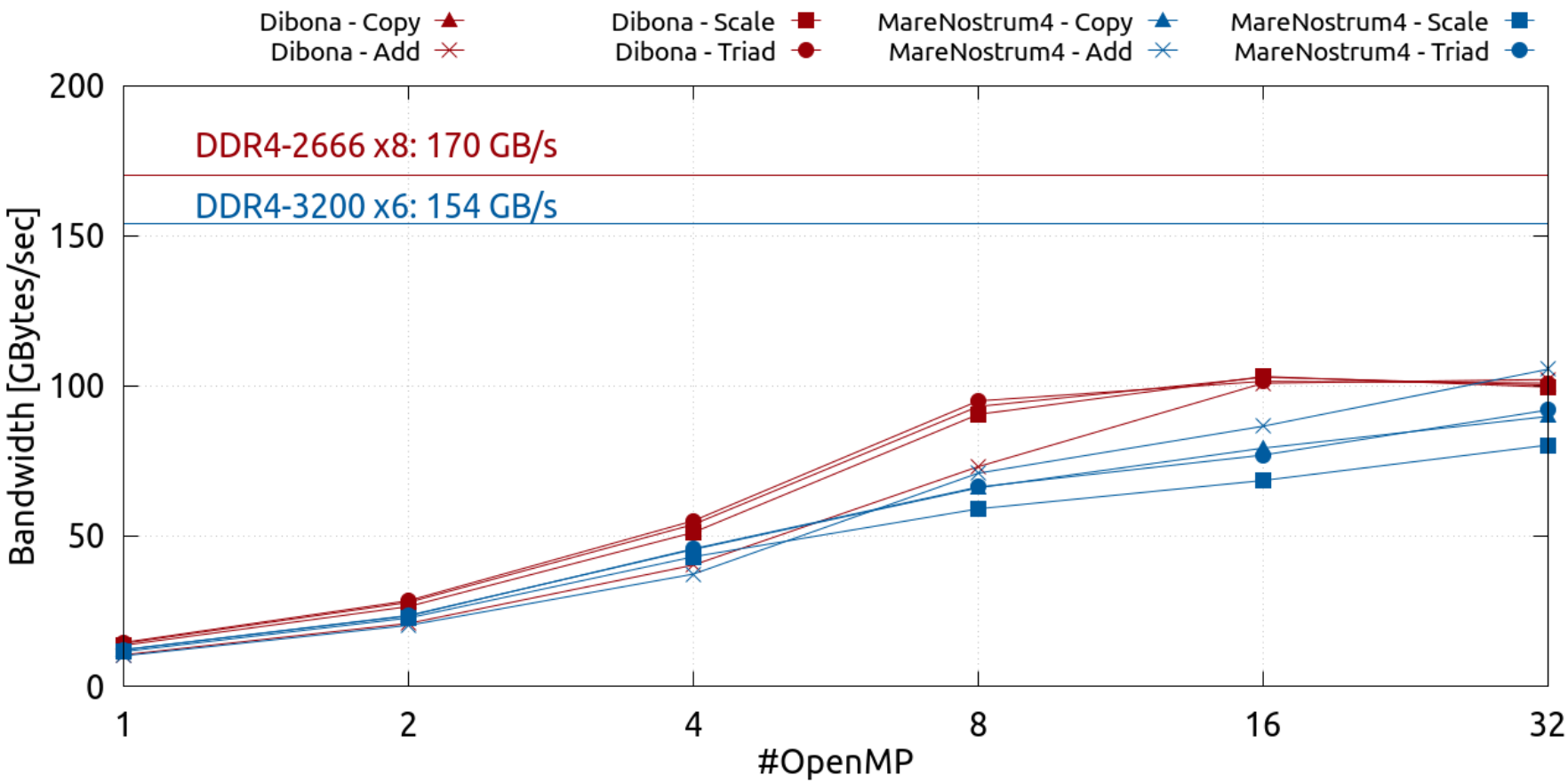

Industrial applications (engine CFD)


Extraction of kernels for architectural simulations


Platform provider (Codename “Dibona”)

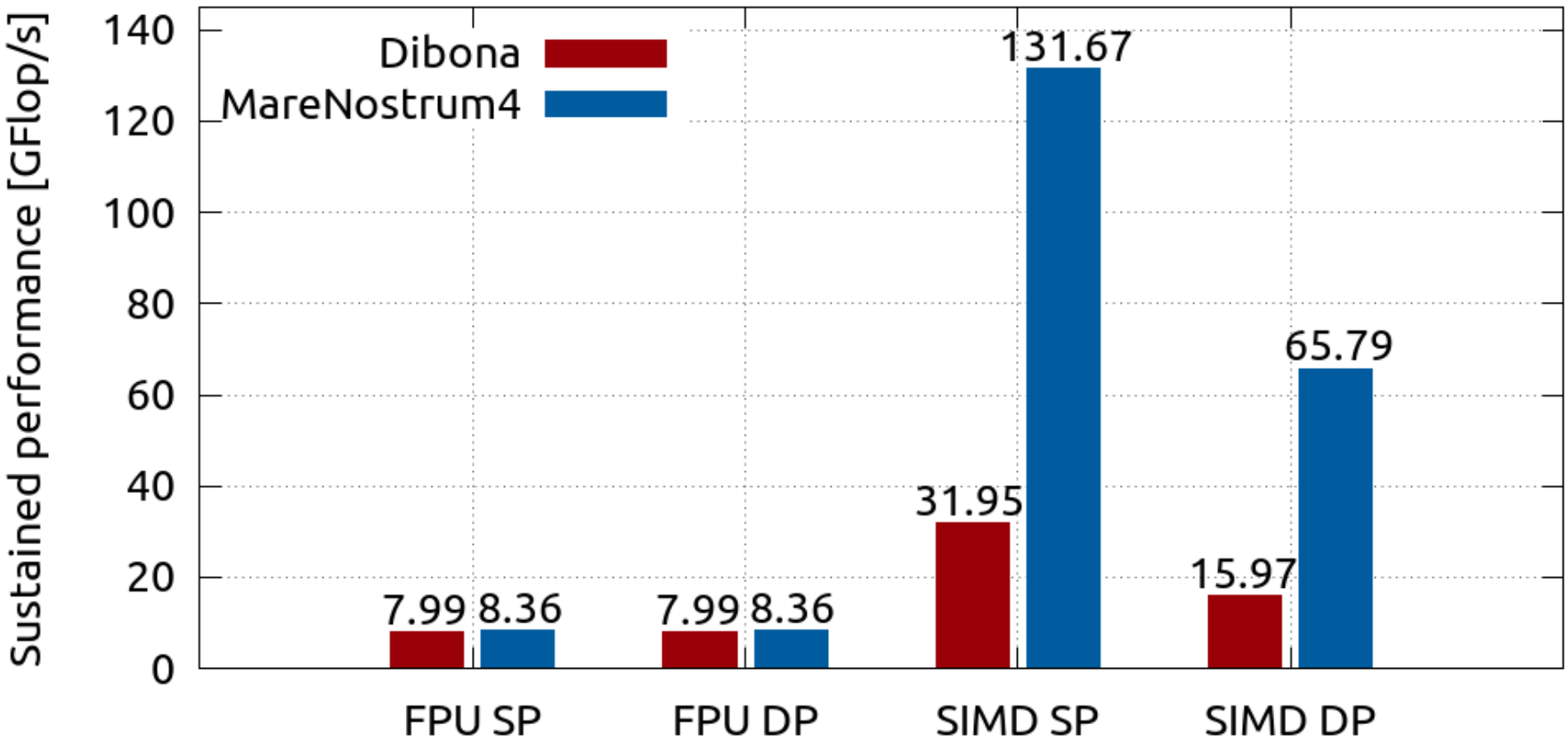
# Micro-benchmarks

# Micro-benchmarks: Memory / STREAM



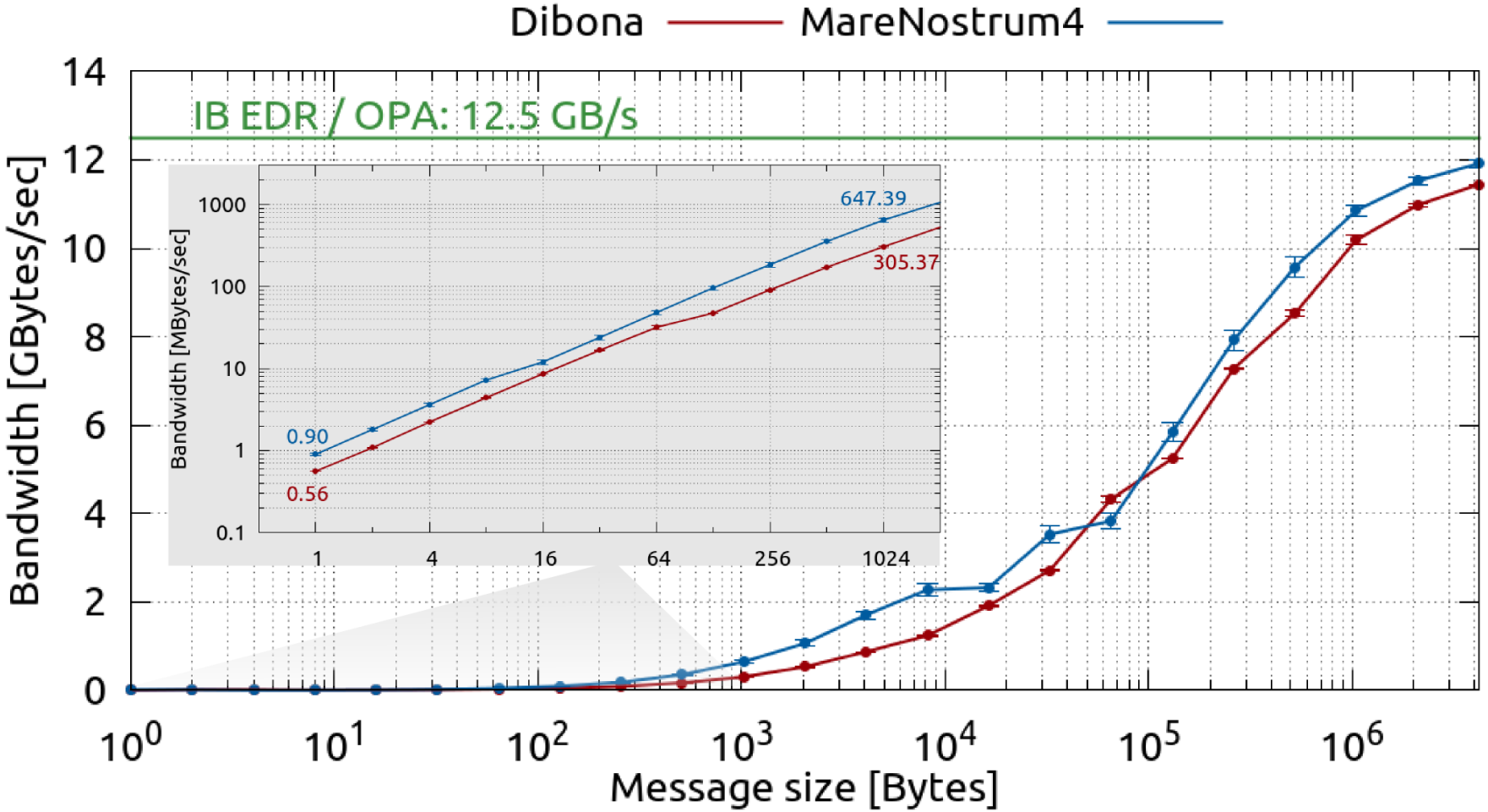

Single socket data

# Micro-benchmarks: Floating point throughput



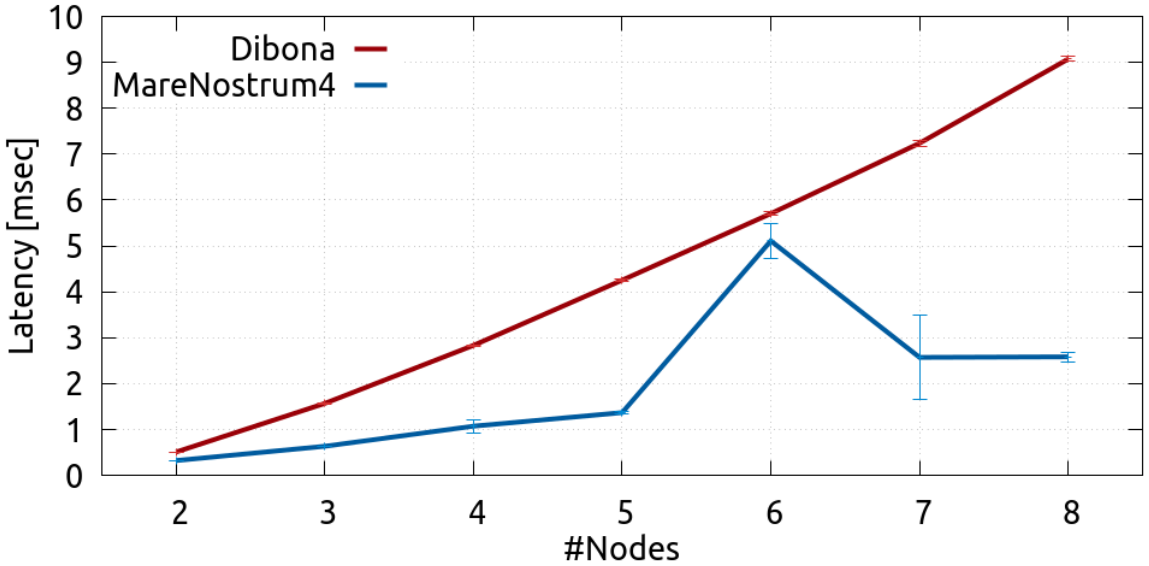

- Evaluation of the throughput of FPU (inline assembly)
- Similar scalar FP performance, both SP and DP ($< 5\%$)
- Vector length of NEON is 128b, AVX512 is 512b $\rightarrow 4\times$

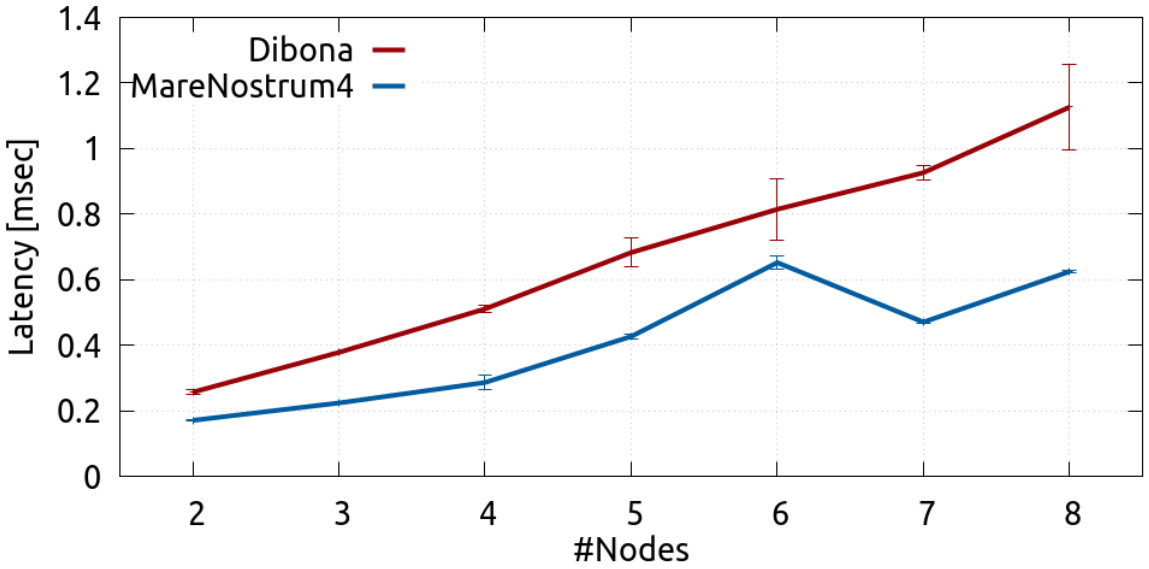
# Micro-benchmarks: Network ping-pong




- Similar bandwidth at large message size ($< 5\%$)
- IB EDR looks $\sim 2\times$ slower than OPA transferring small packages

# Micro-benchmarks: Network collectives

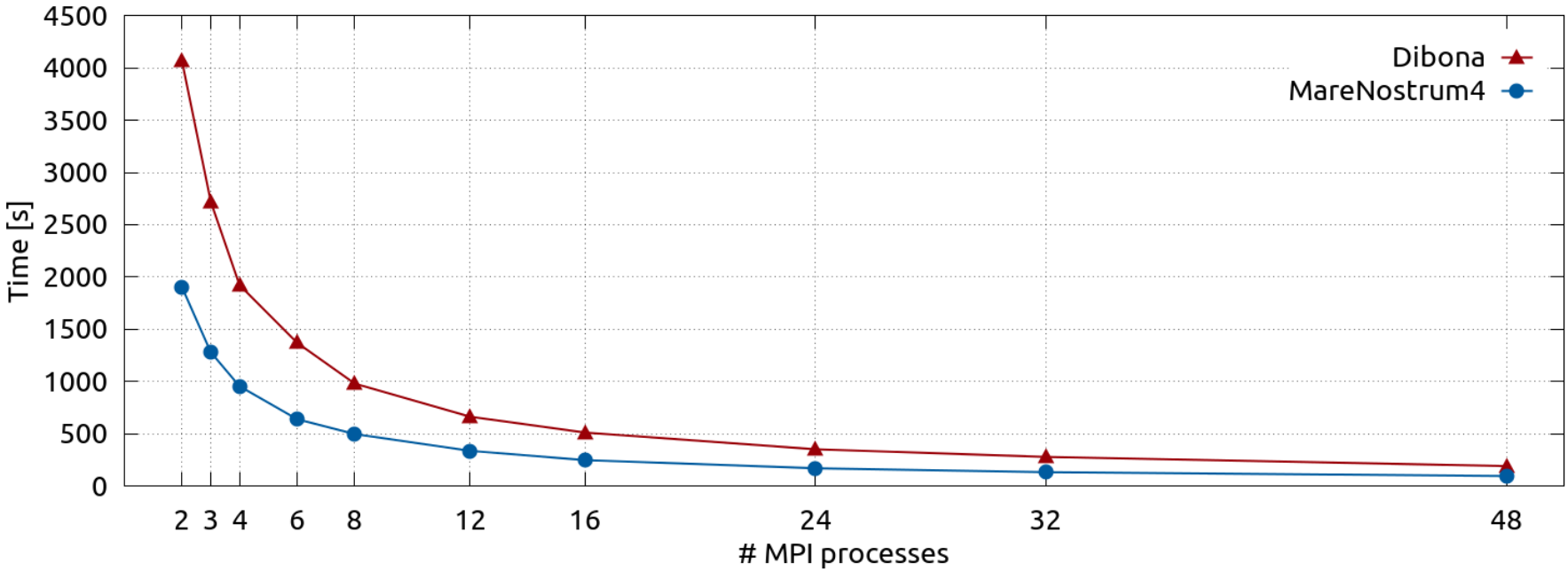


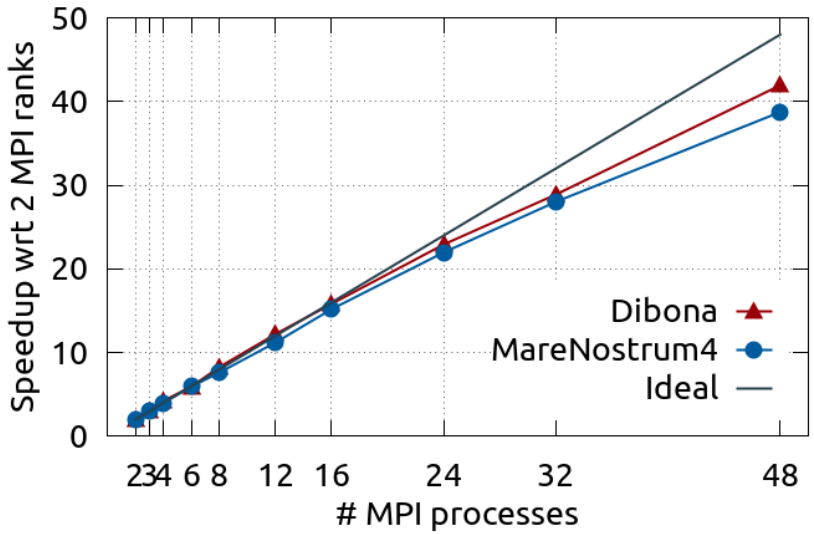



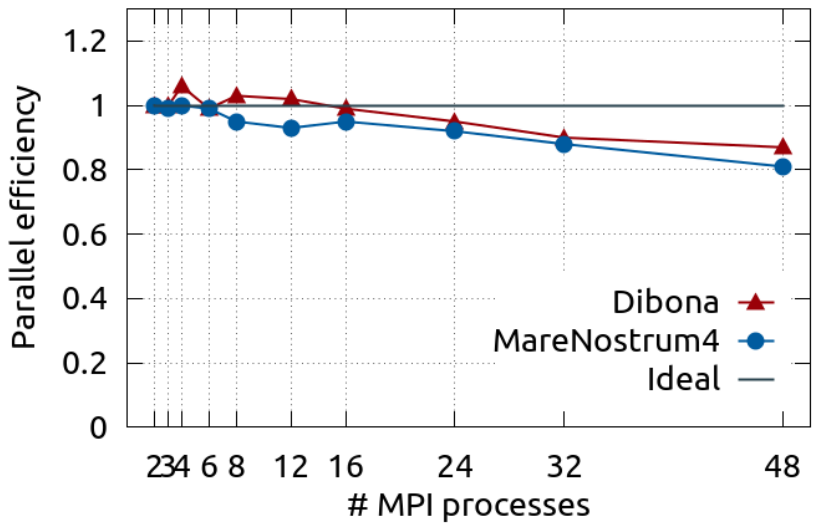

- Overall consistent trend in Dibona
- Benchmarking in production clusters $\rightarrow$ High variability especially in MN4

# Applications – basic evaluation

# OpenIFS (preliminary)



# OpenIFS (Configurations)

## MareNostrum4

Compiler: Intel 18.0.2

Fortran Flags

```
$OIFS_FC{?}      = mpif90
$OIFS_FFLAGS{?} = -qopenmp -pg \
  -m64 -O1 -xHost -fp-model precise \
  -convert big_endian
$OIFS_FFIXED{?} = -r8
$OIFS_FCDEFS{?} = BLAS LITTLE LINUX \
  INTEGER_IS_INT
$OIFS_LFLAGS{?} = -qopenmp -pg
```

C Flags

```
$OIFS_CC{?}      = mpicc
$OIFS_CFLAGS{?} = -pg -O -qopenmp
$OIFS_CCDEFS{?} = BLAS LITTLE LINUX \
  INTEGER_IS_INT _ABI64
```

## Dibona

Compiler: GNU 7.1.0

Fortran Flags

```
$OIFS_FC{?}      = mpif90
$OIFS_FFLAGS{?} = -O3 -march=native \
  -mcpu=native -fstack-arrays \
  -fconvert=big-endian
$OIFS_FFIXED{?} = -fdefault-real-8 \
  -fdefault-double-8 \
  -ffixed-line-length-132
$OIFS_FCDEFS{?} = BLAS LITTLE LINUX \
  INTEGER_IS_INT F90 PARAL NONCRAYF
$OIFS_LFLAGS{?} =
```

C Flags

```
$OIFS_CC{?}      = mpicc
$OIFS_CFLAGS{?} = -O3 -march=native \
  -mcpu=native
$OIFS_CCDEFS{?} = BLAS LITTLE LINUX \
  INTEGER_IS_INT _ABI64
```

# Vectorization

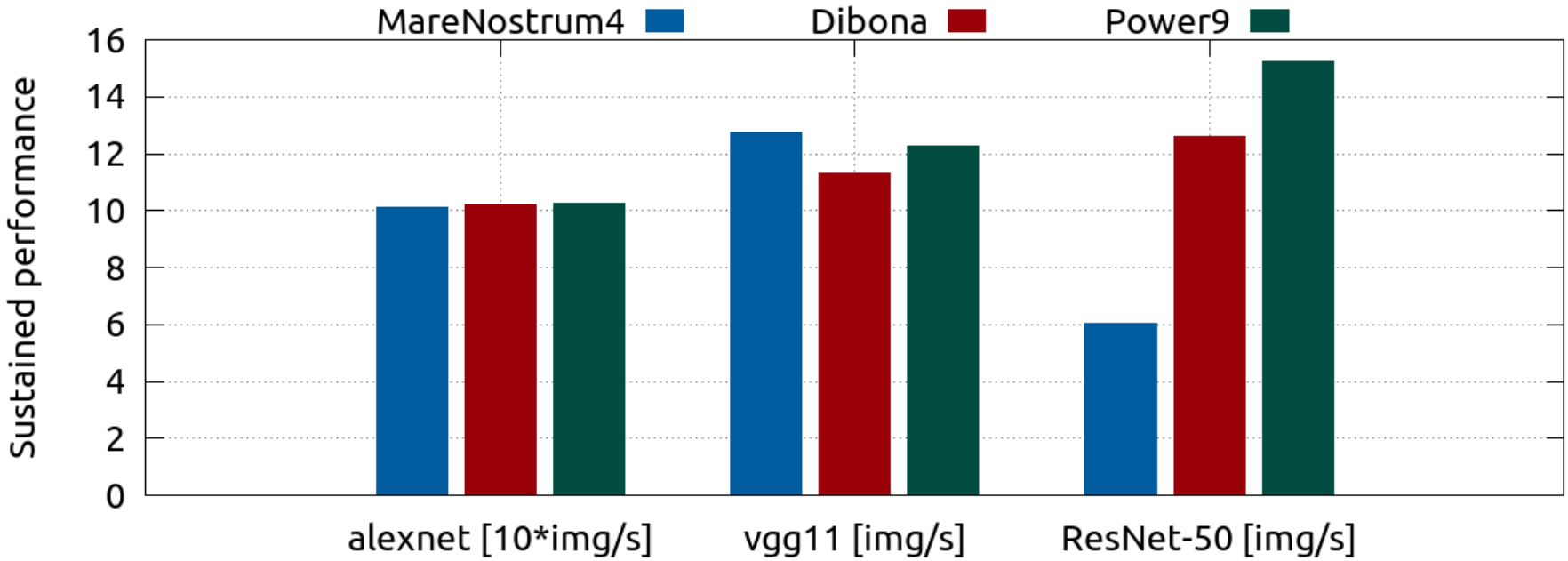
**Goal:** Study the state-of-the-art of auto-vectorization

**Method:** Compare auto-vectorization of applications
Count numbers of "vector instructions per 1000 instructions"

| Application | GCC7 | GCC8 | Arm Comp. |
|---|---|---|---|
| HPCG | 1.41 | 1.52 | 0.03 |
| OpenIFS | 8.08 | – | 13.37 |

# TensorFlow

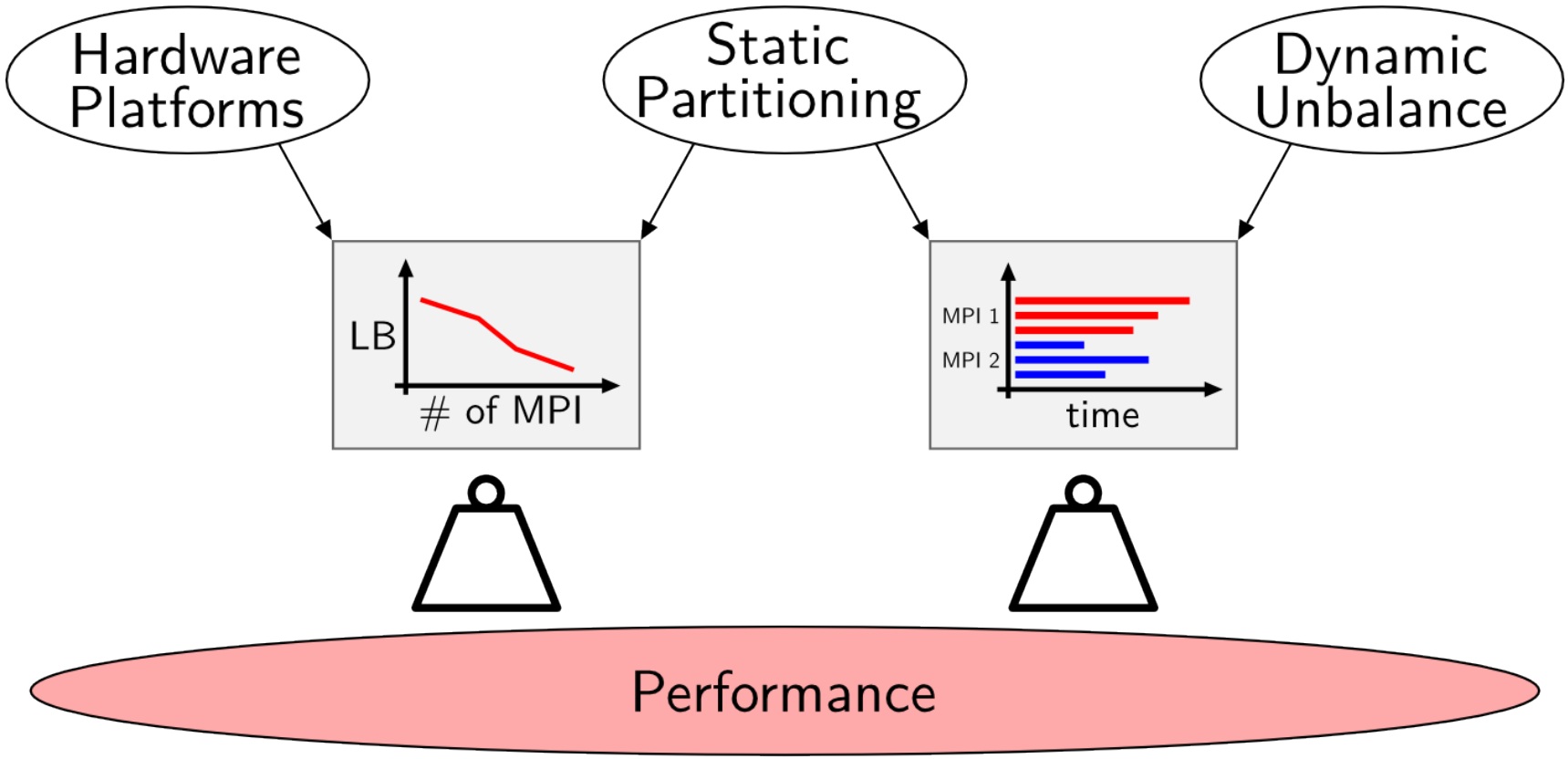
Focusing on "Training", not "Inference"
**NOTE:** Training is mostly a GPU task nowadays



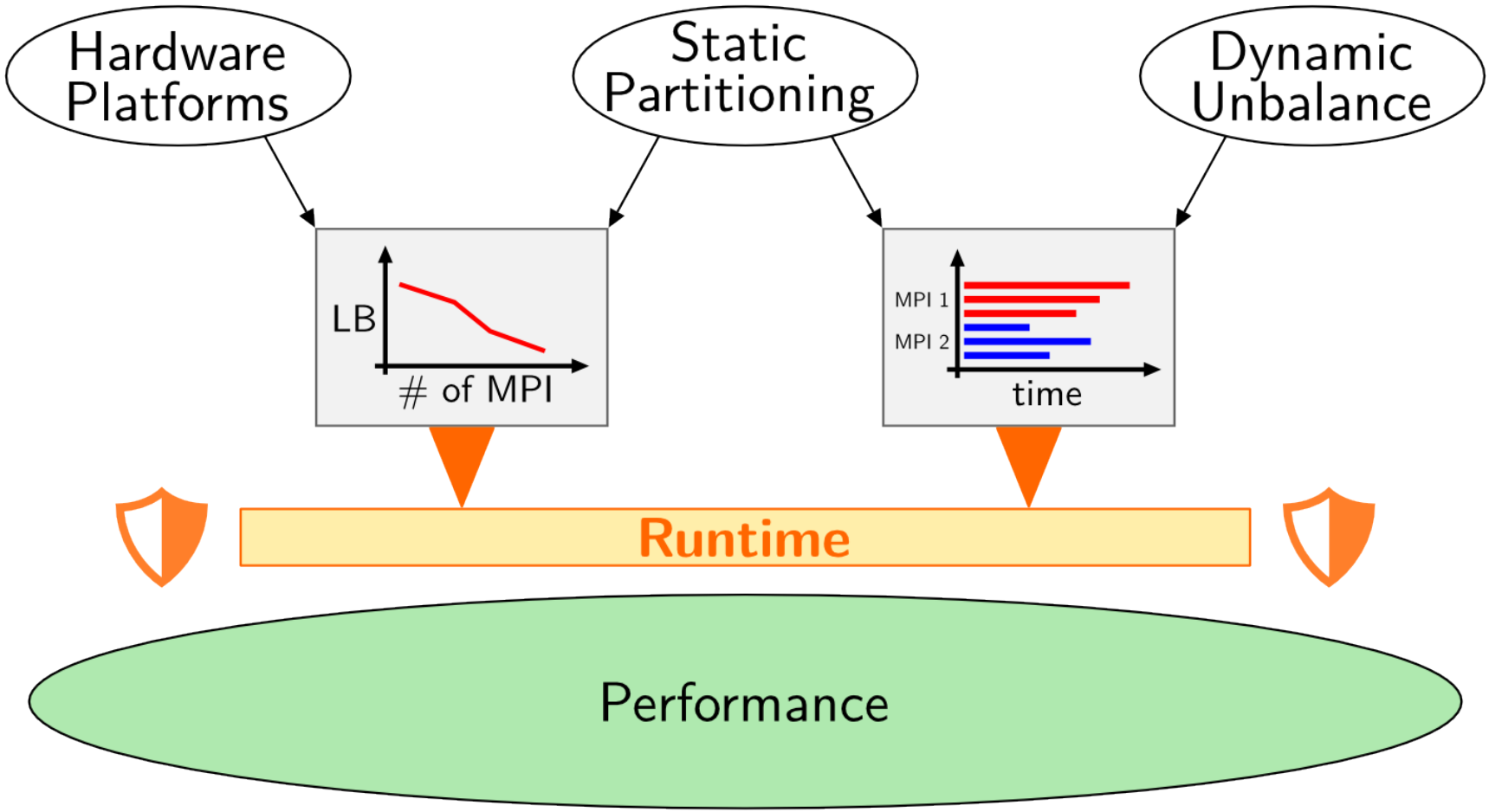

- Preliminary single node evaluation
- On Dibona TF has been compiled with Arm HPC Compiler 18.3
- On Power9 TF has been compiled with gcc8.2

# Applications – advanced

# Motivation


Hardware Platforms
Static Partitioning
Dynamic Unbalance
LB
# of MPI
MPI 1
MPI 2
time
Performance

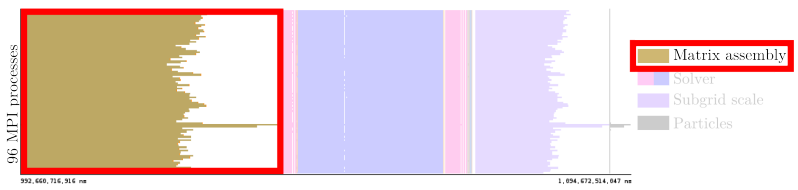
# Motivation

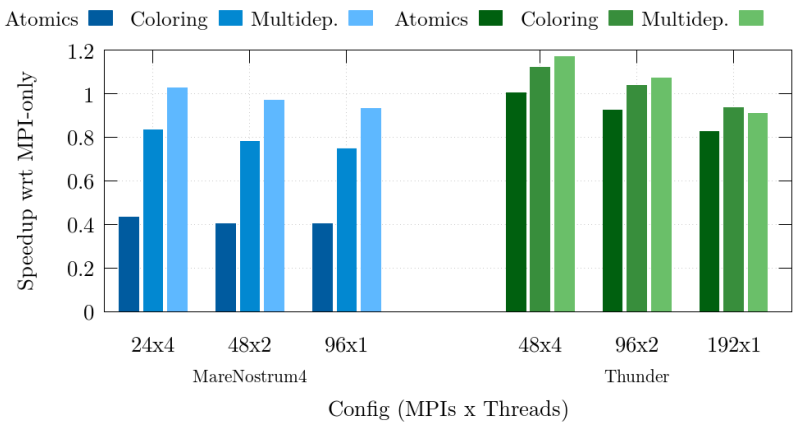

Hardware Platforms
Static Partitioning
Dynamic Unbalance
LB
# of MPI
MPI 1
MPI 2
time
Runtime
Performance

# Alya

## Multidependences

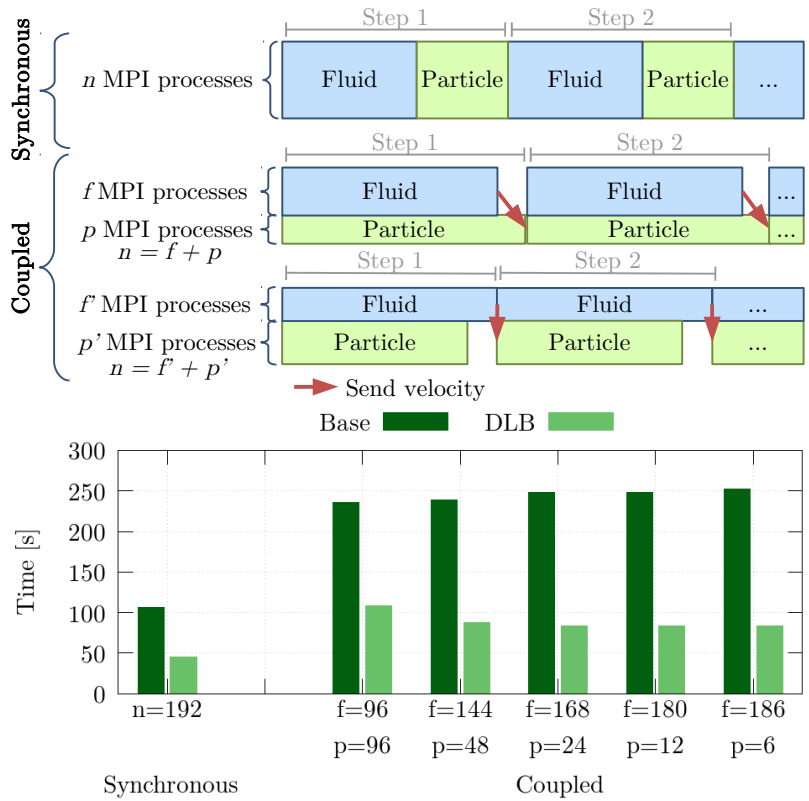
**Goal:** To avoid the race condition between two threads updating elements of the grid sharing variables.



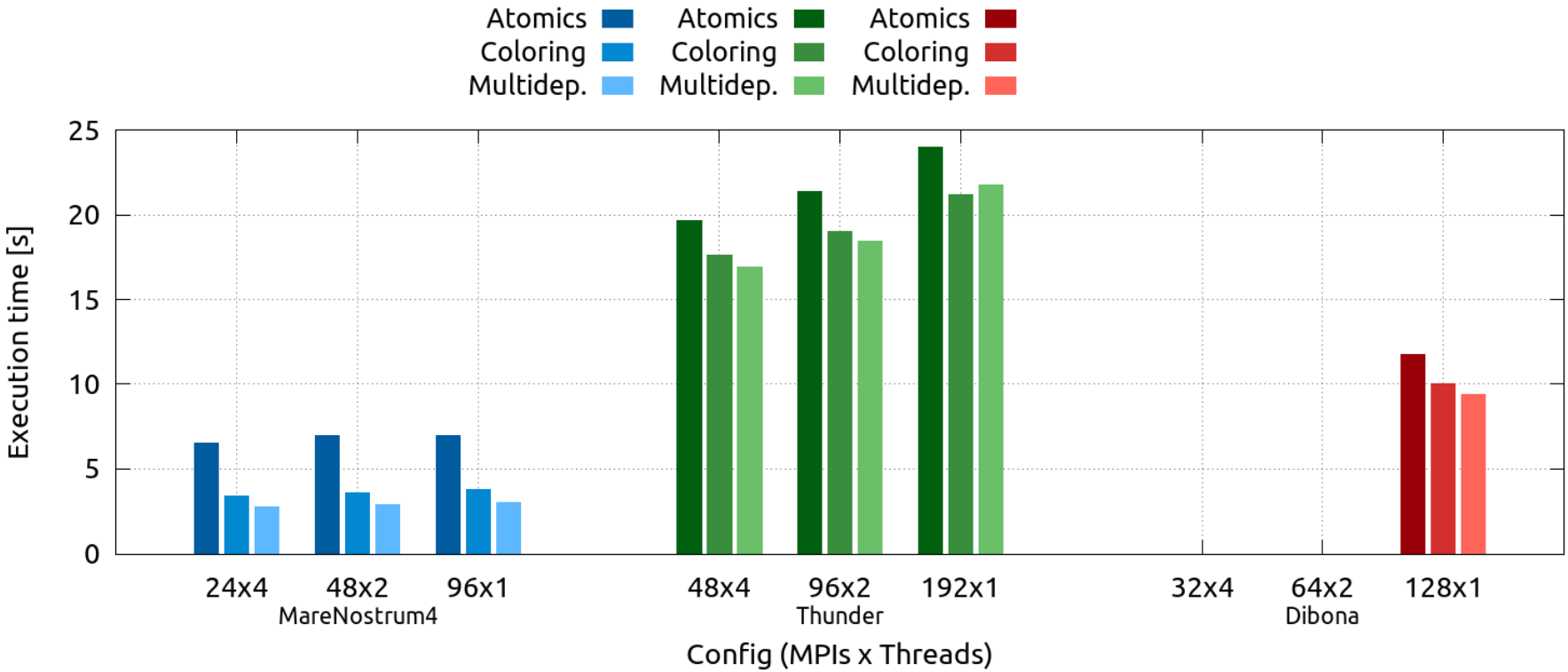

## Dynamic Load Balancing

**Goal:** Survive with load unbalance.




Paper presented at BioHPC Workshop / ICPP18 conference
https://dl.acm.org/citation.cfm?id=3229736
Contribution to OpenMP5.0 standard (TBA Nov 2018)

# Alya: Preliminary results in Dibona

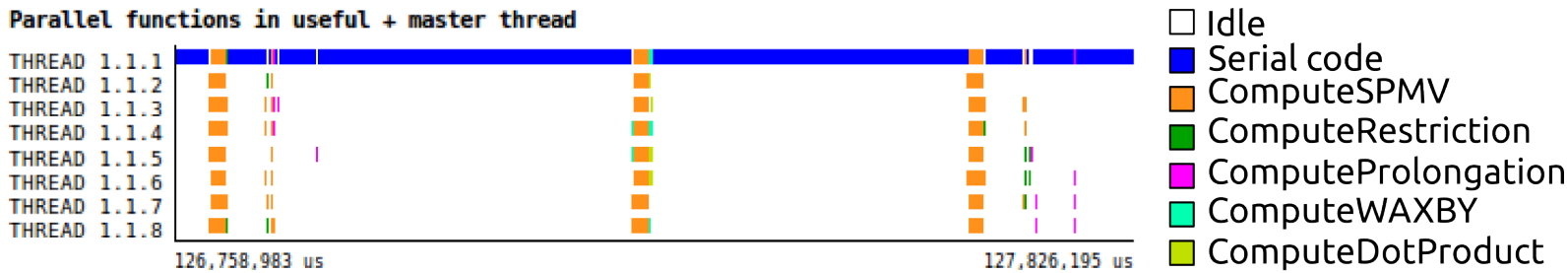



- Still the multidependences version delivers more performance
- Work in progress...

# HPCG: Explore OpenMP parallelization

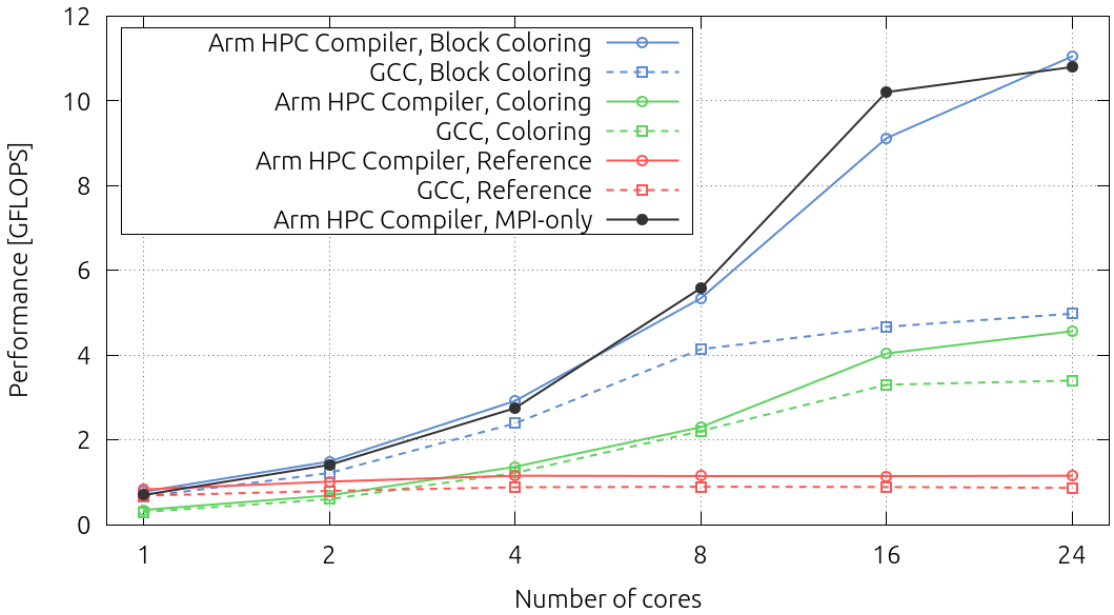
- For addressing scalability limitations mitigating the impact of MPI communications at high number of compute nodes
- To increase data reuse by threads working on the same local domain, thus improving effective bandwidth

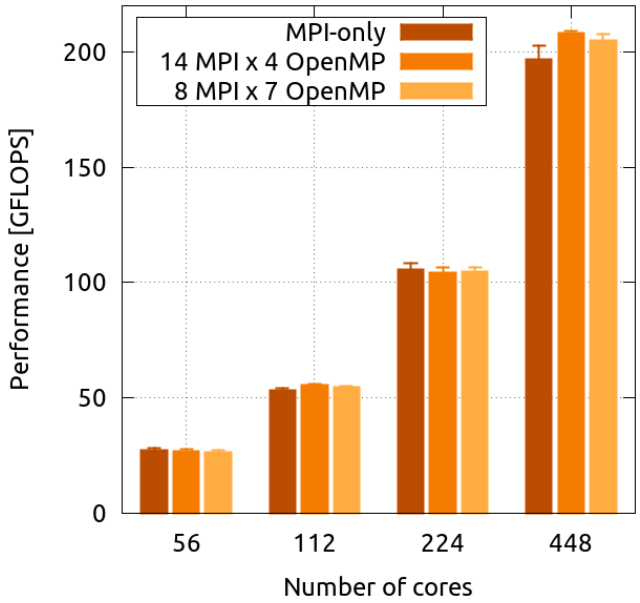
Reference OpenMP version offers very limited parallelization:




**Disclaimer** – Several scientists studied this problem before us.
Iwashita et al. https://doi.org/10.1109/IPDPS.2012.51
Iwashita et al. https://doi.org/10.1109/20.996114
Marjanovic et al. https://doi.org/10.1007/978-3-319-17248-4_9
Park et al. https://doi.org/10.1109/SC.2014.82
Zhang et al. https://doi.org/10.1007/978-3-319-11197-1_3

# HPCG: Single node evaluation




B0 silicon, OpenMPI 3.0, GCC 7.1.0, Arm HPC Compiler 18.1

Tech report: https://upcommons.upc.edu/handle/2117/116642
Contribution to the community: https://gitlab.com/arm-hpc/benchmarks/hpcg

# Conclusions

- Micro-benchmarking characterizing the micro-architecture
- Performance figures of benchmarks and applications
- Runtime techniques enabling performance portability

- Next steps enabled by this contribution:
  - Extensive performance evaluation of Dibona and Arm tools
  - Power measurements and energy study
  - Preparation of applications for architectural simulation

More details can be found in:
http://montblanc-project.euproject/deliverables#deliverables-2015
http://pm.bsc.es/dlb

**Acknowledgment:** The Mont-Blanc application team at BSC: