montblanc-project.eu | @MontBlanc_EU

The Mont-Blanc project
Updates from the Barcelona Supercomputing Center

Filippo Mantovani

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

The “legacy” Mont-Blanc vision

Vision: to leverage the fast growing market of mobile technology for scientific computation, HPC and data centers.

2012 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018

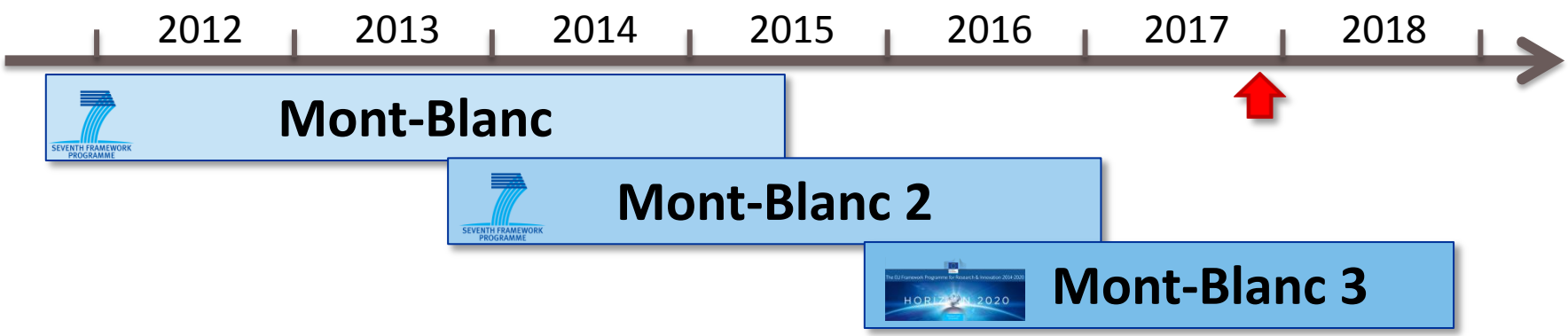
Mont-Blanc

Mont-Blanc 2

Mont-Blanc 3

The “legacy” Mont-Blanc vision

Vision: to leverage the fast growing market of mobile technology for scientific computation, HPC and data centers.



Phases share a common structure

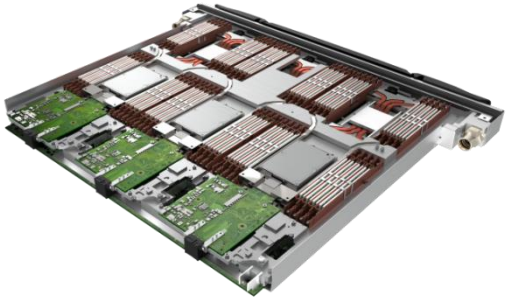
- **Experiment with real hardware**
  - Android dev-kits, mini-clusters, prototypes, production ready systems
- **Push software development**
  - System software, HPC benchmarks/mini-apps/production codes
- **Study next generation architectures**
  - Learn from hardware deployment and evaluation for planning new systems

Hardware platforms

We started here

We ended up here

Bull
atos technologies

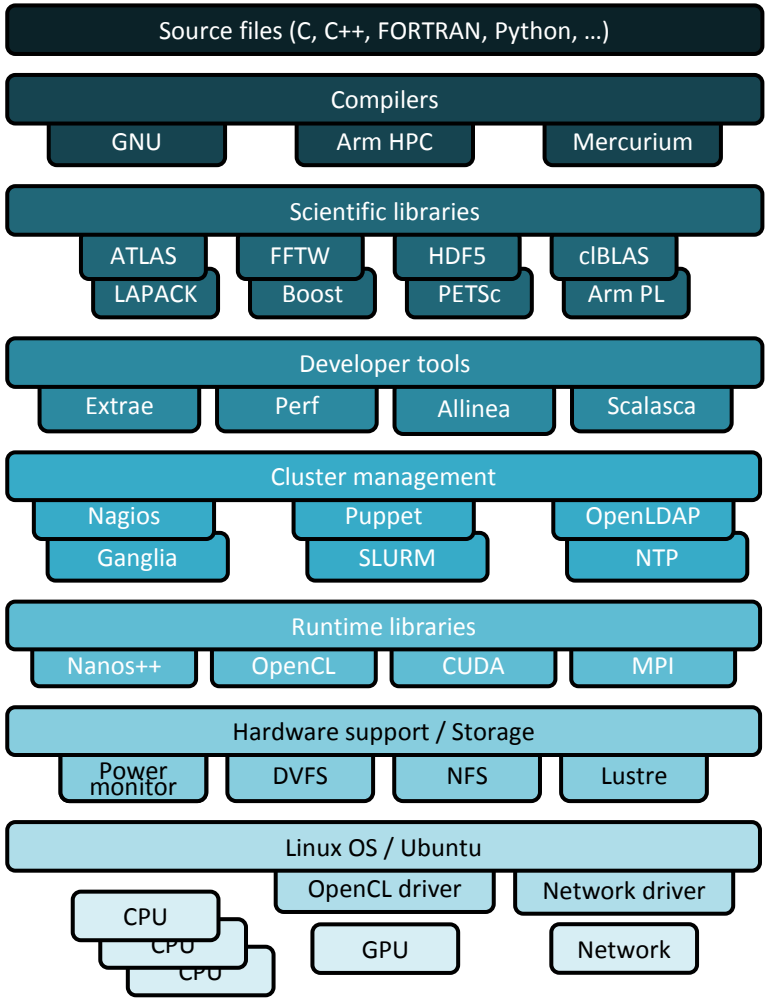


N. Rajovic et al., "The Mont-Blanc Prototype: An Alternative Approach for HPC Systems," in Proceedings of SC'16, p. 38:1–38:12.

MONT-BLANC

System Software and Use Cases

We started here



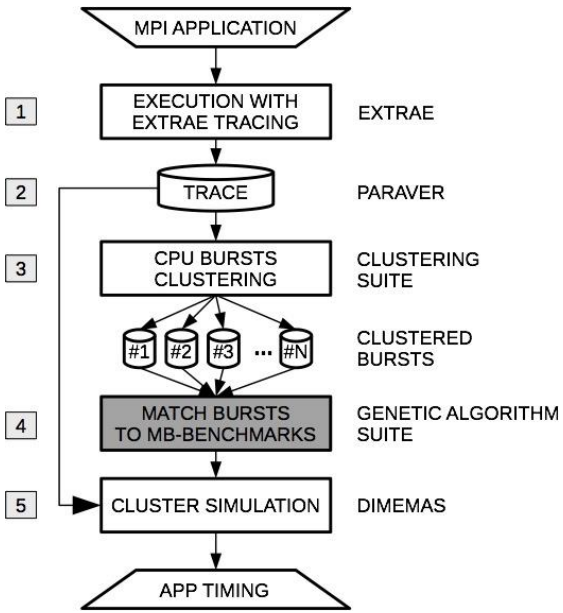
We ended up here

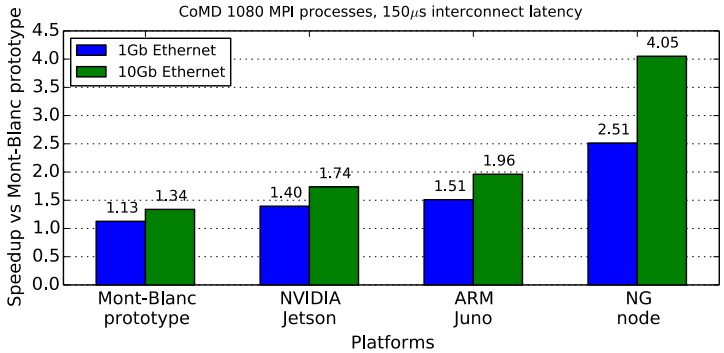
- Different OS flavors
- Arm HPC Compiler
- Arm Performance Libraries
- Allinea tools
- …
- All well packed and distributed through OpenHPC

- Several complex HPC production codes have run on Mont-Blanc
  - Alya
  - AVL codes
  - WRF
  - FEniCS

Study of Next-Generation Architectures

We started here





Credits: N. Rajovic

We ended up here

- **A Multi-level Simulation Approach (MUSA) allows us:**
  - To gather performance traces on any current HPC architecture
  - To replay them using almost any architecture configuration
  - To study scalability and performance figures at scale, changing the number of MPI processes simulated

Credits: MUSA team @ BSC

Where BSC is contributing today?

- **Evaluation of solutions**
  - Hardware solutions
    - Mini-clusters deployed liaising with SoC providers and system integrators
  - Software solutions
    - Arm Performance Libraries, Arm HPC Compiler

F. Banchelli et al., "Is Arm software ecosystem ready for HPC?", poster at SC17.

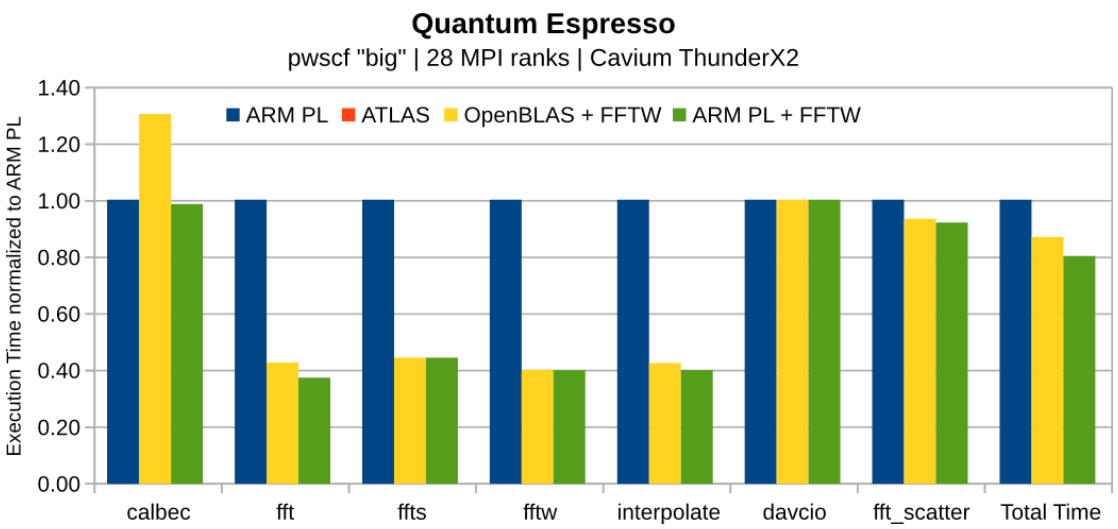
- **Use cases**
  - Alya: finite element code where we experiment atomics-avoiding techniques
    - GOAL: test new runtime features to be pushed into OpenMP
  - HPCG: benchmark where we started looking at vectorization
    - GOAL: explore techniques for exploitation of the Arm Scalable Vector Extension

- **Simulation of next generation large clusters**
  - MUSA: Combining detailed trace driven simulation with sampling strategies for exploring how architectural parameters affects the performance at scale.

T. Grass et al., "MUSA: A Multi-level Simulation Approach for Next-Generation HPC Machines," in SC16 proceedings, pp. 526–537.

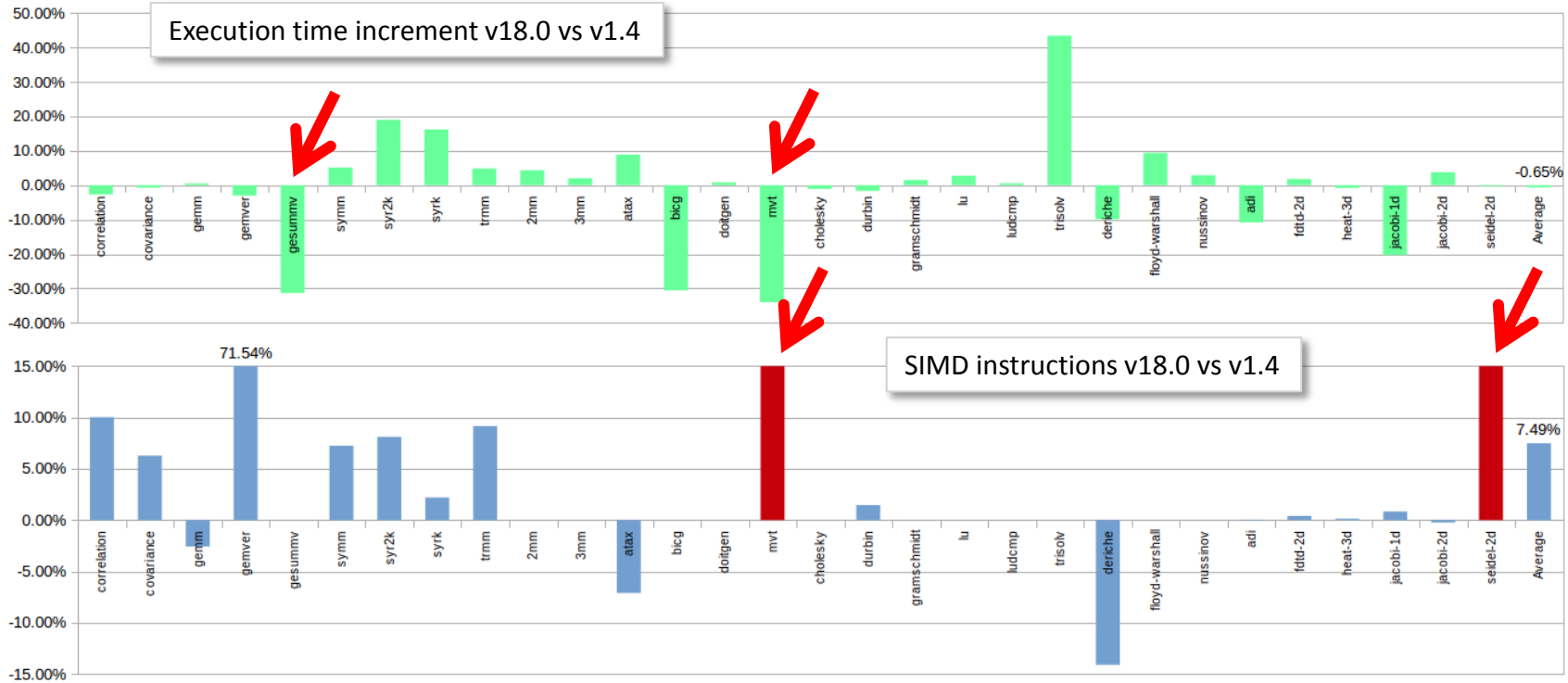
Evaluation of Arm Performance Libraries

- **Goal**
  - Test an HPC code making use of arithmetic and FFT libraries
- **Method**
  - Quantum Espresso pwscf input
  - Compiled with GCC 7.1.0
- **Platform configuration #1 (poster SC17)**
  - AMD Seattle
  - Arm PL 2.2
  - ATLAS 3.11.39
  - OpenBLAS 0.2.20
  - FFTW 3.3.6
- **Platform configuration #2**
  - Cavium ThunderX2
  - Arm PL v18.0
  - OpenBLAS 0.2.20
  - FFTW 3.3.7



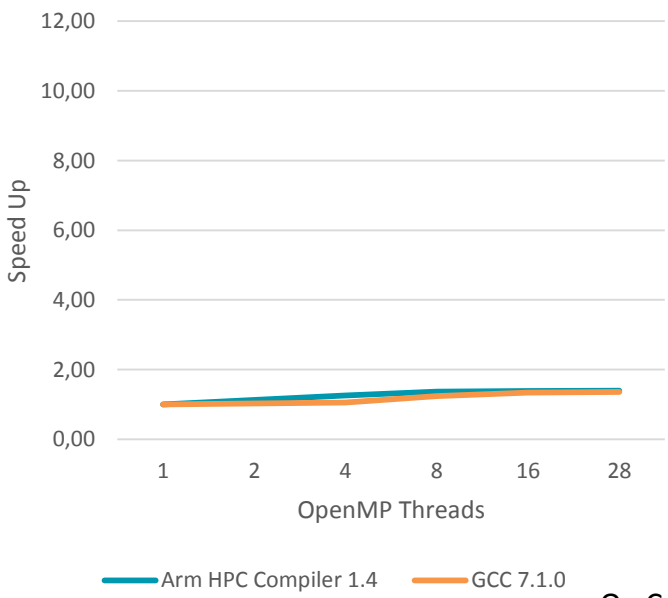
Evaluation of the Arm HPC Compiler

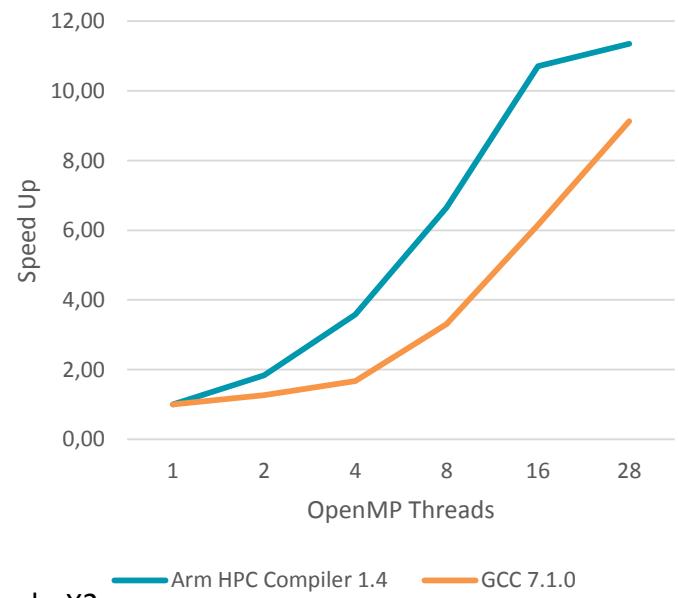
- **Goal**
  - Evaluate the Arm HPC Compilers v18.0 vs v1.4
- **Method**
  - Run Polybench benchmark suite
  - Including 30 benchmarks by Ohio State University
  - Run on Cavium ThunderX2



High Performance Conjugate Gradient

- **Problem**
  - Scalability of HPCG is very limited
  - OpenMP parallelization of the reference HPCG version is poor
- **Goals**
  1. Improve OpenMP parallelization of HPCG
  2. Study current auto-vectorization for leveraging SVE
  3. Analyze other performance limitations (e.g. cache effects)

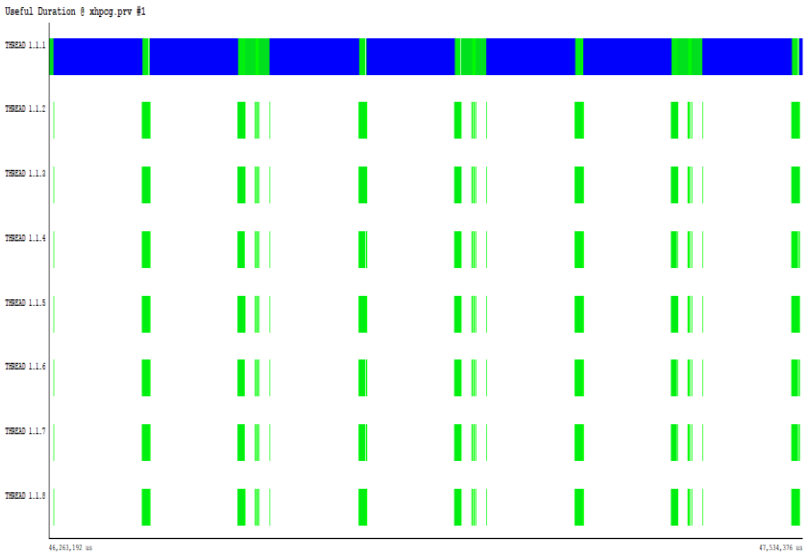


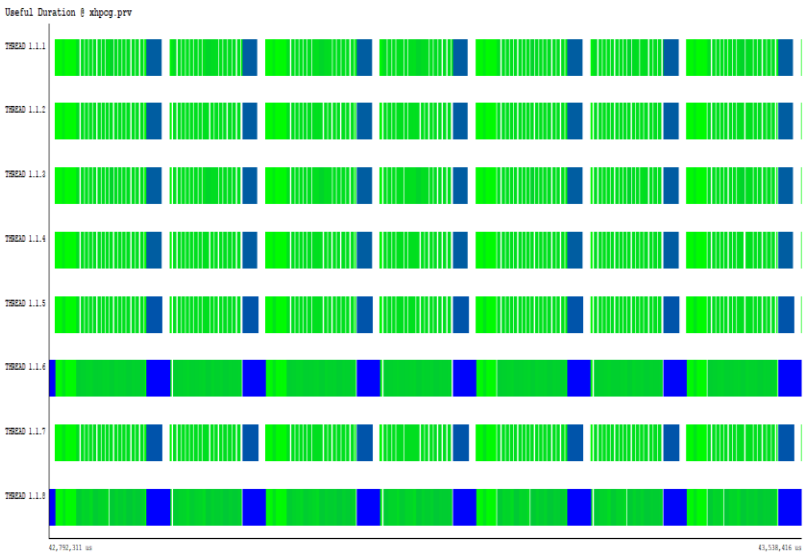


On Cavium ThunderX2

High Performance Conjugate Gradient

- **Problem**
  - Scalability of HPCG is very limited
  - OpenMP parallelization of the reference HPCG version is poor
- **Goals**
  1. Improve OpenMP parallelization of HPCG
  2. Study current auto-vectorization for leveraging SVE
  3. Analyze other performance limitations (e.g. cache effects)



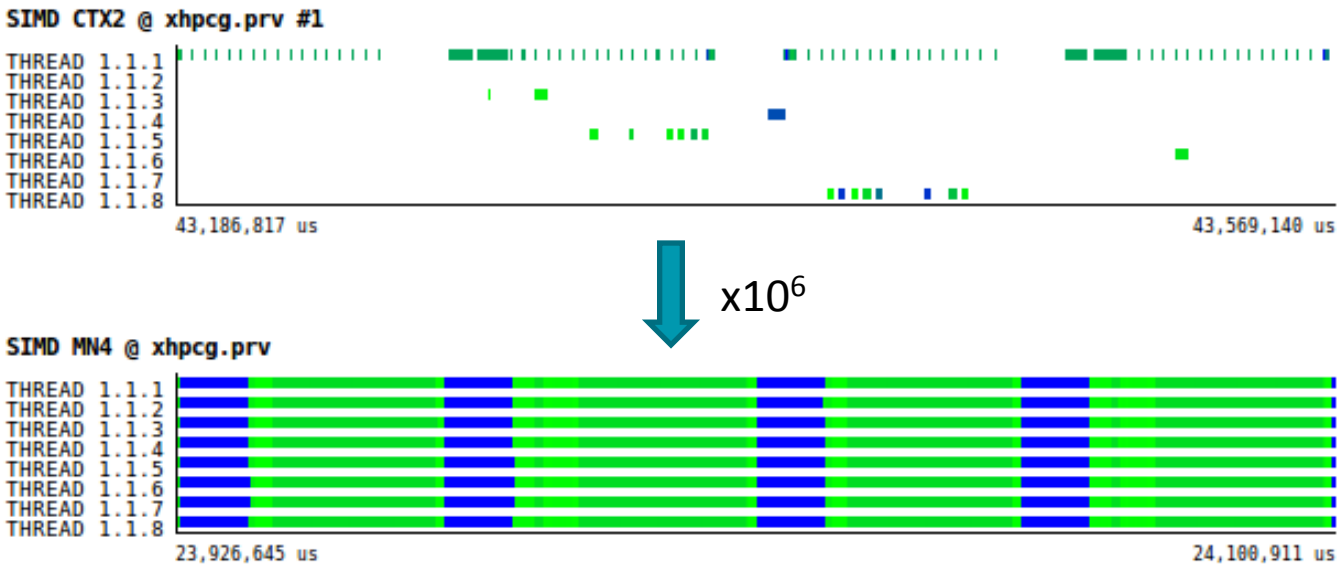


On Cavium ThunderX2

HPCG - SIMD parallelization

- **First approach**
  - Check auto-vectorization in current platforms

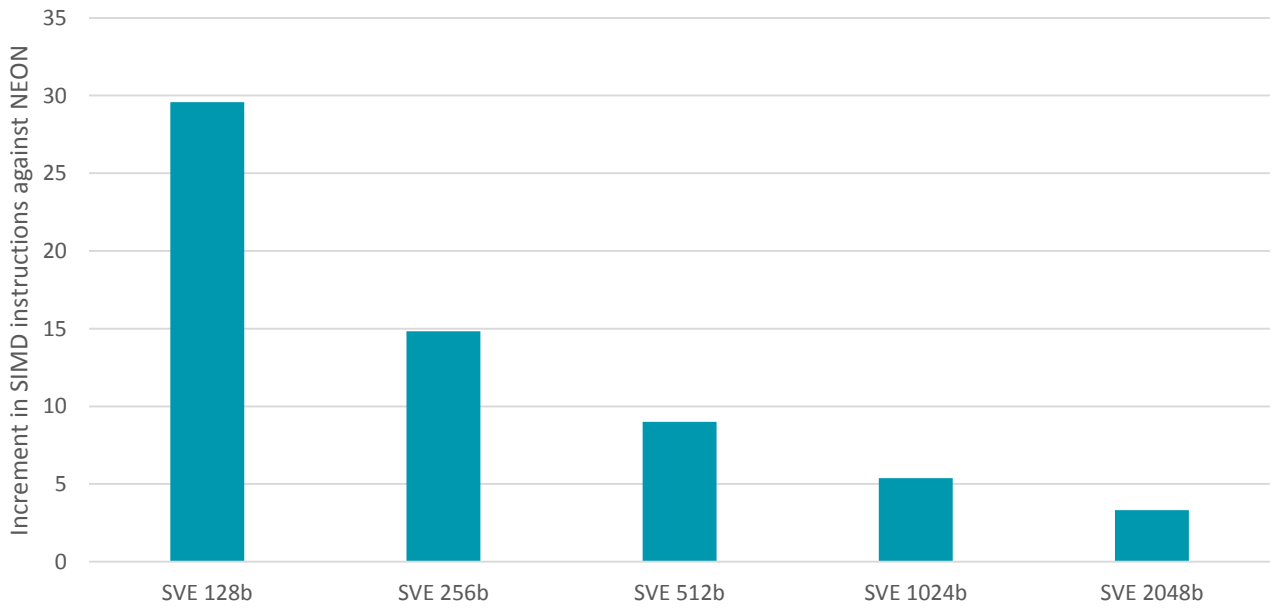
- **Method**
  - Count SIMD instructions in the “ComputeSYMGS” region
  - On Cavium ThunderX2 using Arm HPC Compiler v18.0
  - On Intel Xeon Platinum 8160 (Skylake) using ICC supporting AVX512



HPCG - SVE emulation

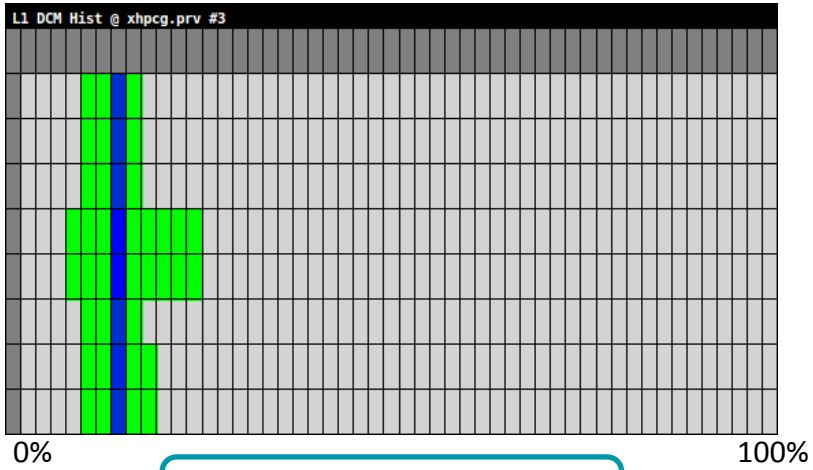
- **First approach**
  - Check auto-vectorization when SVE is enabled

- **Method**
  - Evaluate auto-vectorization in a whole execution of HPCG (one iteration)
  - Generate binary using Arm HPC Compiler v1.4 enabling SVE
  - Emulate SVE instruction using Arm Instruction Emulator in Cavium ThunderX2

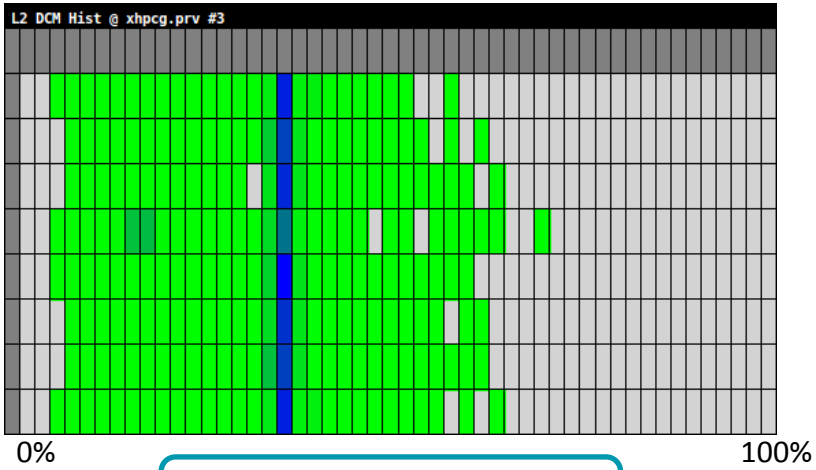


HPGC - Memory access evaluation

- **Cache hit ratio degraded when using multi-coloring approaches**
  - Data related to ComputeSYMGS
  - Gathered on Cavium ThunderX2
  - Compiled with GCC



~13% L1D miss ratio



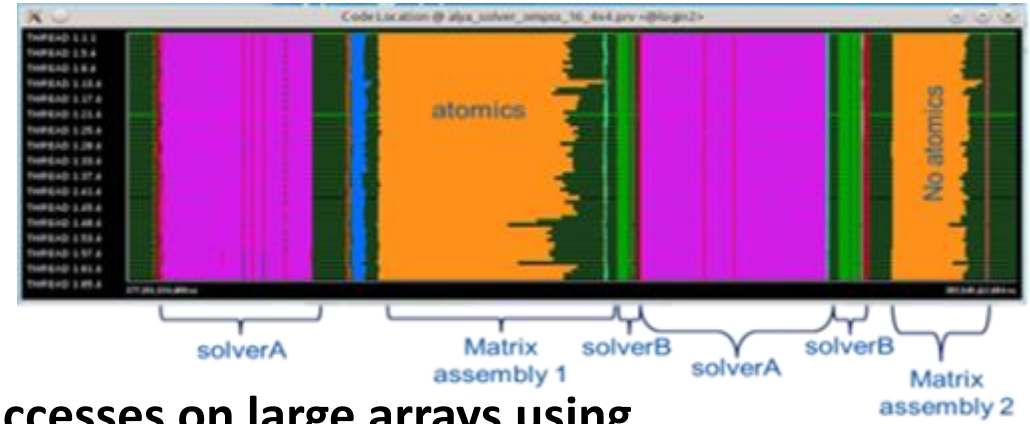
~35% L2D miss ratio

- **Next steps**
  - Optimize data access patterns in memory
  - Simulate “SVE gather load” instructions in order to quantify the benefits

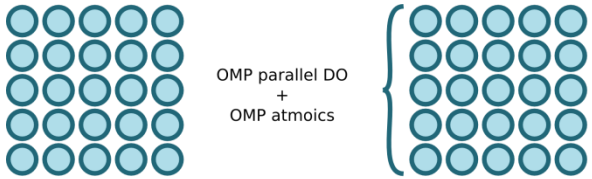
Alya: BSC code for multi-physics problems

Parallelization of finite elements code

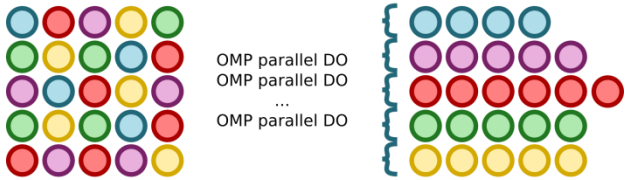
- **Analysis with Paraver:**



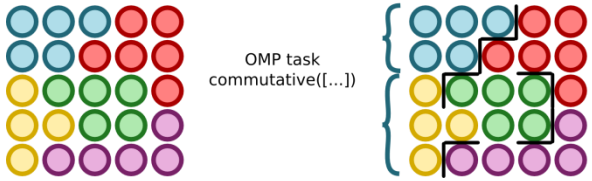
- **Reductions with indirect accesses on large arrays using**
  - No coloring
    Use of atomics operations harms performance



  - Coloring
    Use of coloring harms locality



  - Commutative Multidependences
    - (OmpSs feature to be hopefully included in OpenMP)



Credits: M. Garcia, J. Labarta

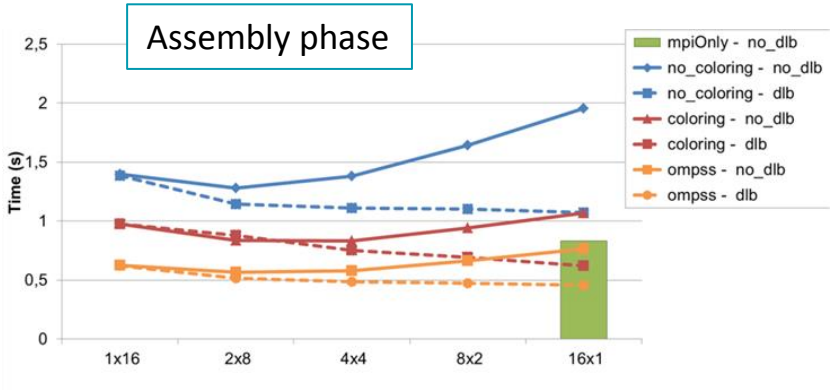
Alya: taskification and dynamic load balancing

Goal

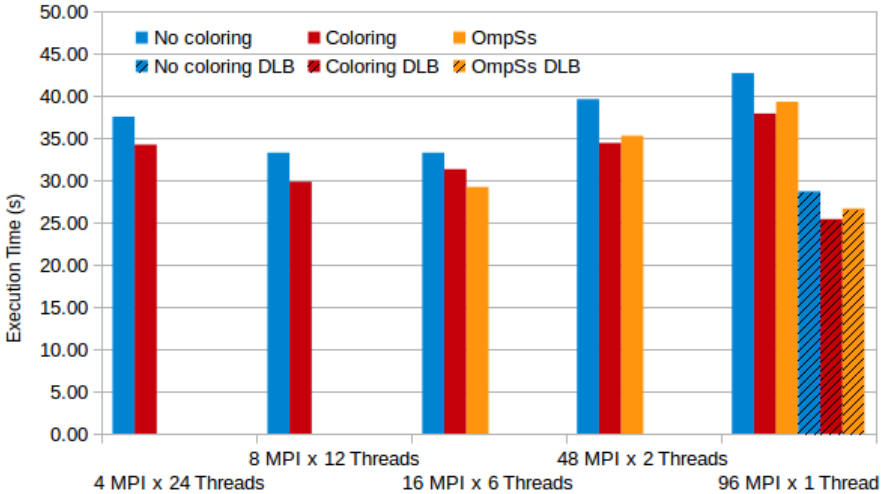
- Quantify the effect of commutative dependences and DLB on an HPC code

Method

- Run the "Assembly phase" of Alya (containing atomics)
- On MareNostrum 3, 2x Intel Xeon SandyBridge-EP E5-2670
- On Cavium ThunderX, 2x CN8890



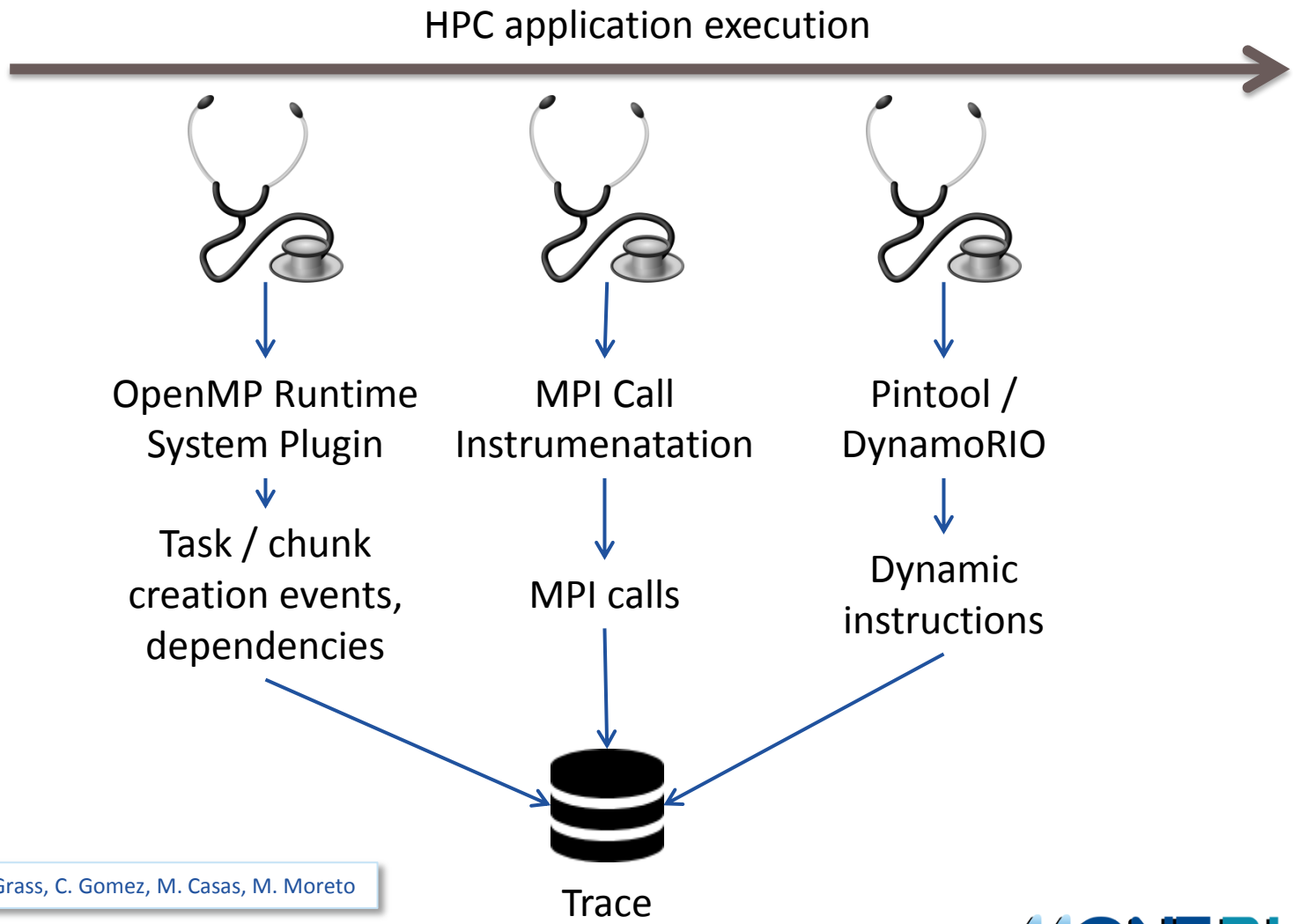
16 nodes x P processes/node x T threads/process



Credits: M. Josep, M. Garcia, J. Labarta

Multi-Level Simulation Approach

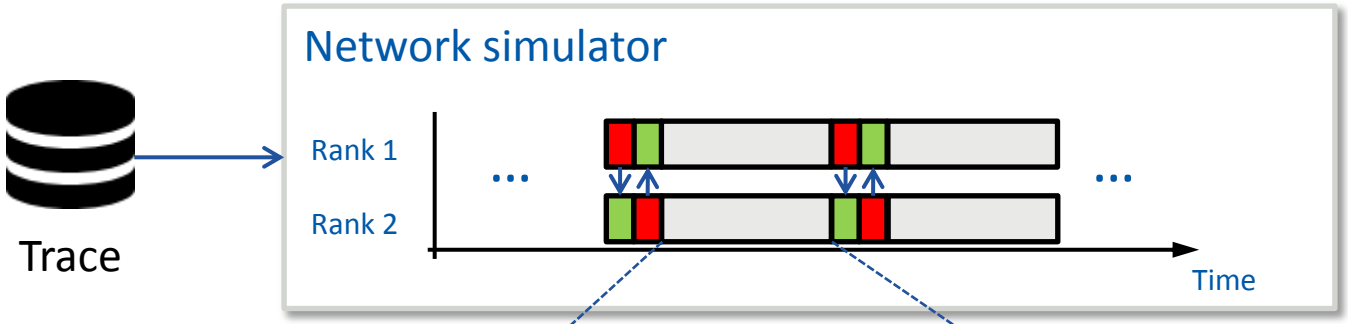
- **Level 1: Trace generation**



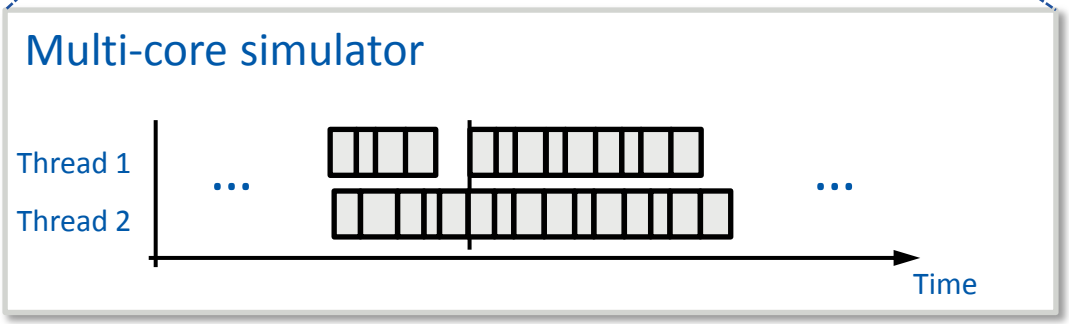
Credits: T. Grass, C. Gomez, M. Casas, M. Moreto

Multi-Level Simulation Approach

- **Level 2: Network simulation (Dimemas)**



- **Level 3: Multi-core simulation (TaskSim + Ramulator + McPAT)**



Credits: T. Grass, C. Gomez, M. Casas, M. Moreto

Multi-Level Parameters

- **Architectural**
  - CPU architecture
  - Number of cores
  - Core frequency
  - Threads per core
  - Reorder buffer size
  - SIMD width

- **Micro-architectural**
  - L1/2/3 Cache size/latency

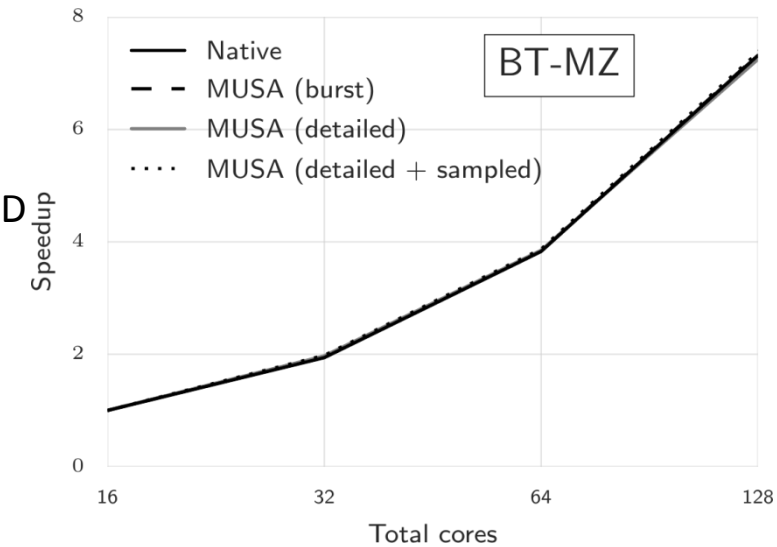
- **Main memory**
  - Memory technology
  - Capacity
  - Bandwidth
  - Latency

Problem:
Simulation time diverges
Solution:
We supported different modes (Burst, Detailed, Sampling) trading accuracy for speed

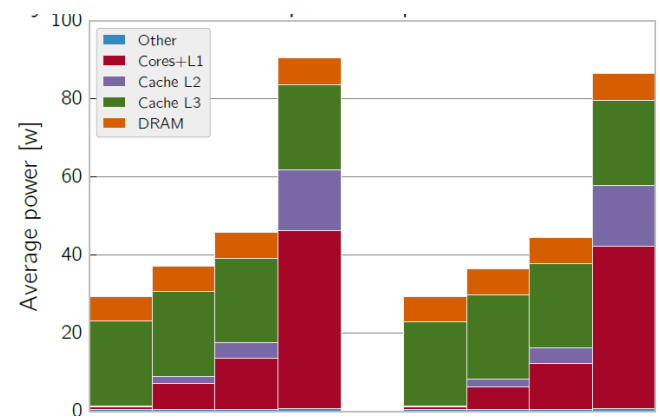
Credits: T. Grass, C. Gomez, M. Casas, M. Moreto

MUSA: status

- **SC'16 paper**
  - Validation of the methodology with 5 applications
    - BT-MZ, SP-MZ, LU-MZ, HYDRO, SPECFEM3D
  - Proven performance figures at scale up to 16 kMPI ranks



- **Status update**
  - Added parameter sets for state-of-the art architectures
  - Support for power consumption modeling
    - Including CPU, NoC and memory hierarchy
  - Incremented set of applications
  - Expanded trace database
    - Including traces gathered on MareNostrum4 (Intel Skylake + OmniPath)
  - Included support for DynamoRIO



Credits: T. Grass, C. Gomez, M. Casas, M. Moreto

Student Cluster Competition

- **Rules**
  - 12 teams of 6 undergraduate students
  - 1 cluster operating within 3 kW power budget
  - 3 HPC applications + 2 benchmarks

- **One team from University Politècnica de Catalunya (UPC-Spain)**
  - Participating with Mont-Blanc technology

- **3 awards to win**
  - Best HPL
  - 1st, 2nd, 3rd overall places
  - Fan favorite

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
E4 COMPUTER ENGINEERING
CAVIUM
arm
MONT-BLANC

Interested in any of the topics presented?

Visit our booths @ SC17!

booth #1694

booth #1925

booth #1975

Follow us!

montblanc-project.eu

@MontBlanc_EU

filippo.mantovani@bsc.es