Performance Optimization and Productivity

EU H2020 Center of Excellence (CoE)

1 October 2015 – 31 March 2018 (30 months)

POP CoE

- **A Center of Excellence**
  - **On Performance Optimization and Productivity**
  - **Promoting best practices in performance analysis and parallel programming**
- **Providing Services**
  - Precise understanding of application and system behavior
  - Suggestion/support on how to refactor code in the most productive way
- **Horizontal**
  - Transversal across application areas, platforms, scales
- **For academic AND industrial codes and users**

Partners

- **Who?**
  - BSC (coordinator), ES
  - HLRS, DE
  - JSC, DE
  - NAG, UK
  - RWTH Aachen, IT Center, DE
  - TERATEC, FR

A team with

- Excellence in performance tools and tuning
- Excellence in programming models and practices
- Research and development background AND
  proven commitment in application to real academic and industrial use cases

Motivation

Why?

- Complexity of machines and codes
  → Frequent lack of quantified understanding of actual behavior
  → Not clear most productive direction of code refactoring
- Important to maximize efficiency (performance, power) of compute intensive applications and the productivity of the development efforts

Target

- Parallel programs , mainly MPI /OpenMP … although can also look at CUDA, OpenCL, Python, …

3 levels of services

? Application Performance Audit

- Primary service
- Identify performance issues of customer code (at customer site)
- Small Effort (< 1 month)

! Application Performance Plan

- Follow-up on the service
- Identifies the root causes of the issues found and qualifies and quantifies approaches to address the issues
- Longer effort (1-3 months)

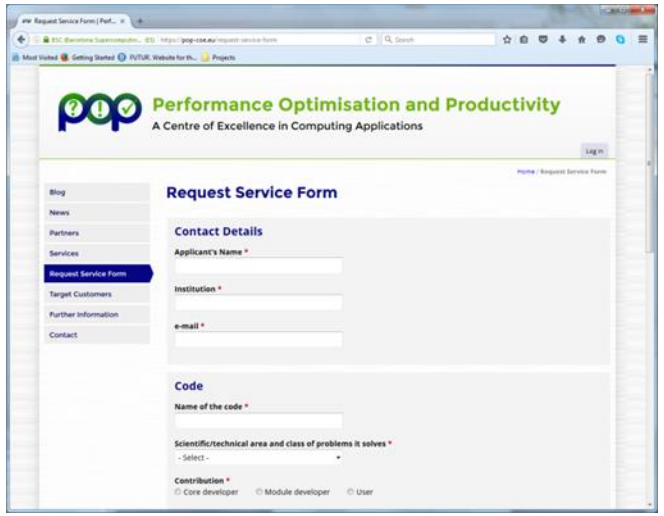
✓ Proof-of-Concept

- Experiments and mock-up tests for customer codes
- Kernel extraction, parallelization, mini-apps experiments to show effect of proposed optimizations
- 6 months effort

Reports

Software demonstrator

Apply @
http://www.pop-coe.eu



Target customers

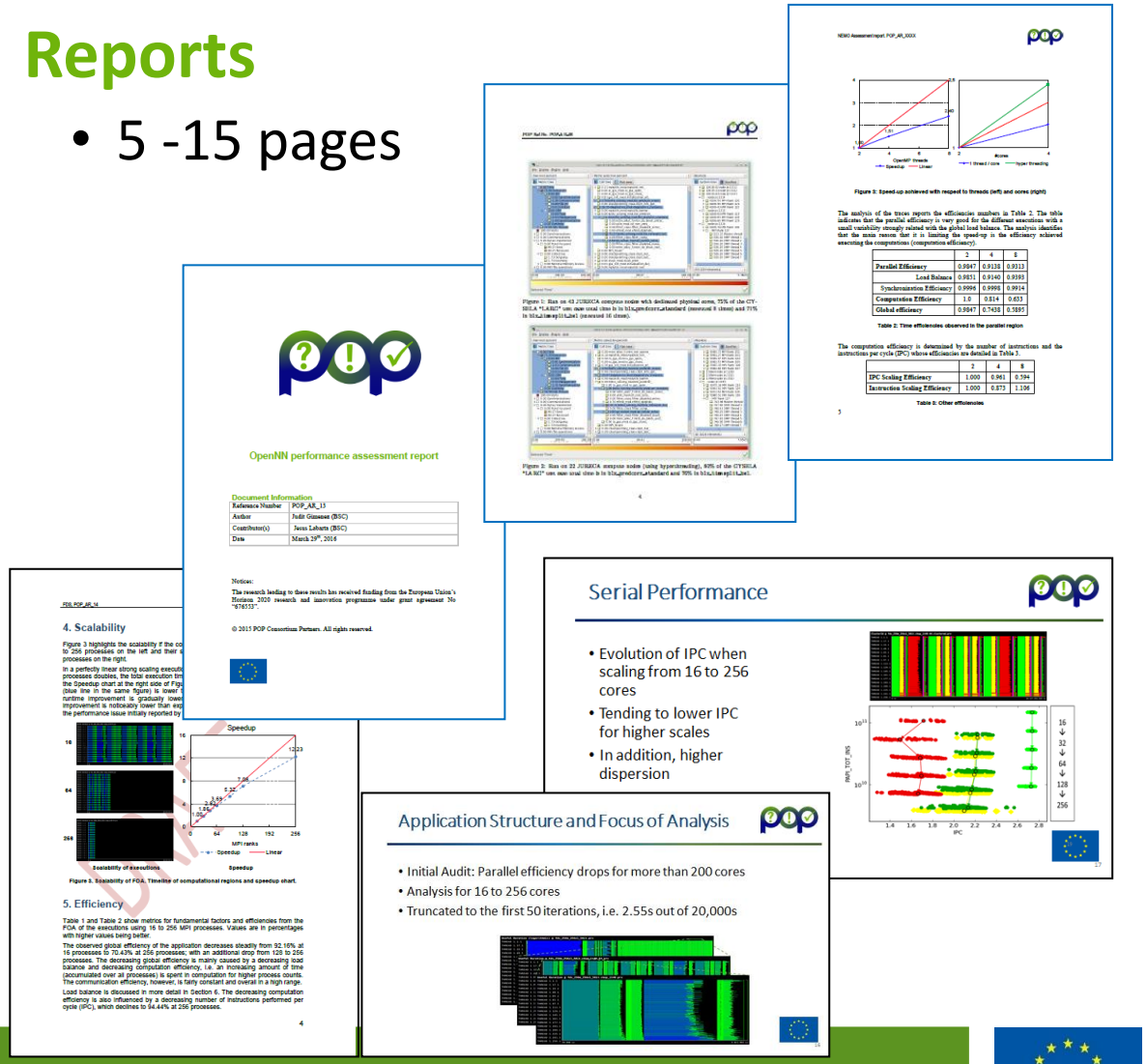
- **Code developers**
  - Assessment of detailed actual behavior
  - Suggestion of more productive directions to refactor code
- **Users**
  - Assessment of achieved performance on specific production conditions
  - Possible improvements modifying environment setup
  - Evidences to interact with code provider
- **Infrastructure operators**
  - Assessment of achieved performance in production conditions
  - Possible improvements modifying environment setup
  - Information for allocation processes
  - Training of support staff
- **Vendors**
  - Benchmarking
  - Customer support
  - System dimensioning/design

Activities (June 2017)

- **Services**
  - Completed/reporting: 80
  - Codes being analyzed: 21
  - Waiting user / New: 22
  - Cancelled: 10
- **By type**
  - Audits: 95
  - Plan: 15
  - Proof of concept: 13

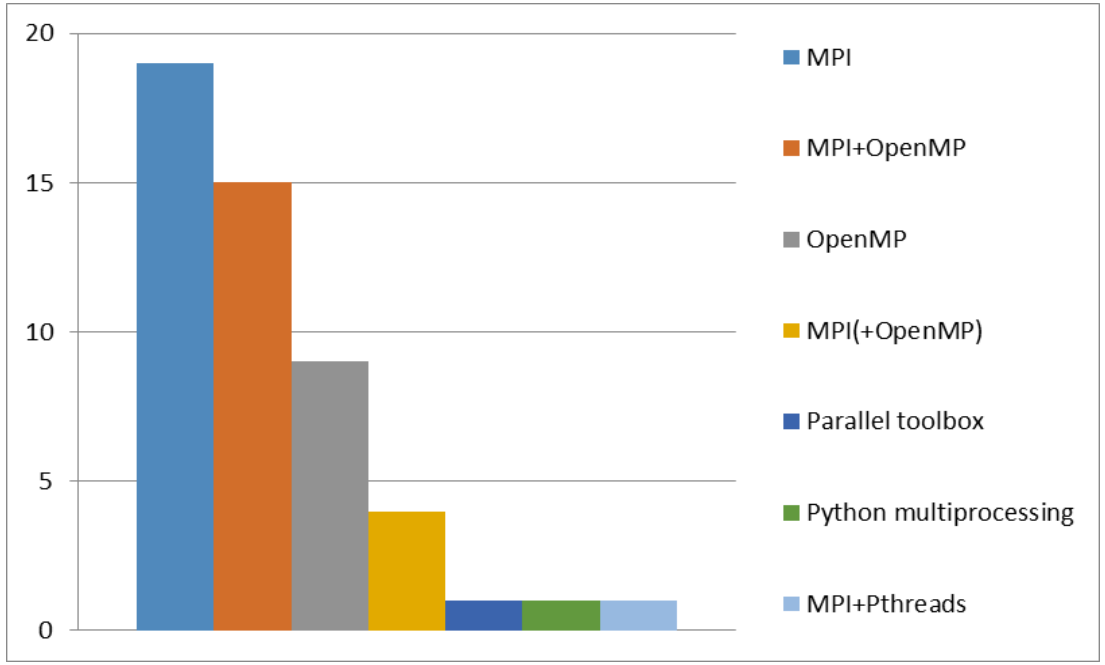
+ 5 training workshops

- **Reports**
  - 5 -15 pages



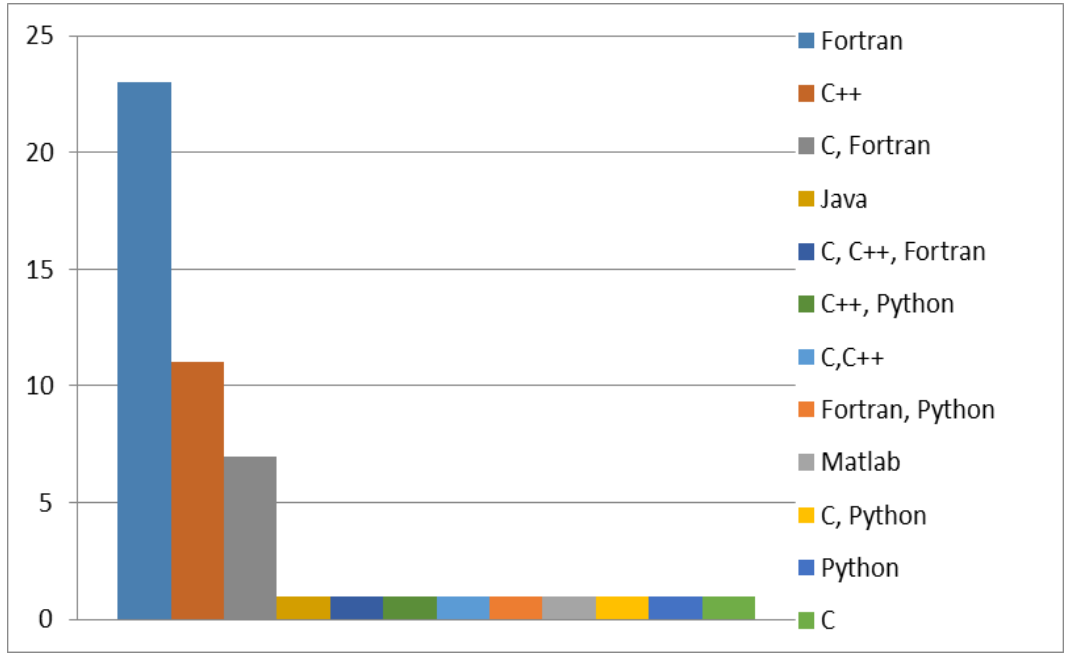
WP4 – Audit characterization

Code



- **Parallel programming model**
  - 77% MPI or MPI+X
  - 17% pure OpenMP
  - Few from new paradigms

- **Programming language**
  - 64% Fortran (+X) as expected
  - 9.4% Python (+X) not really expected

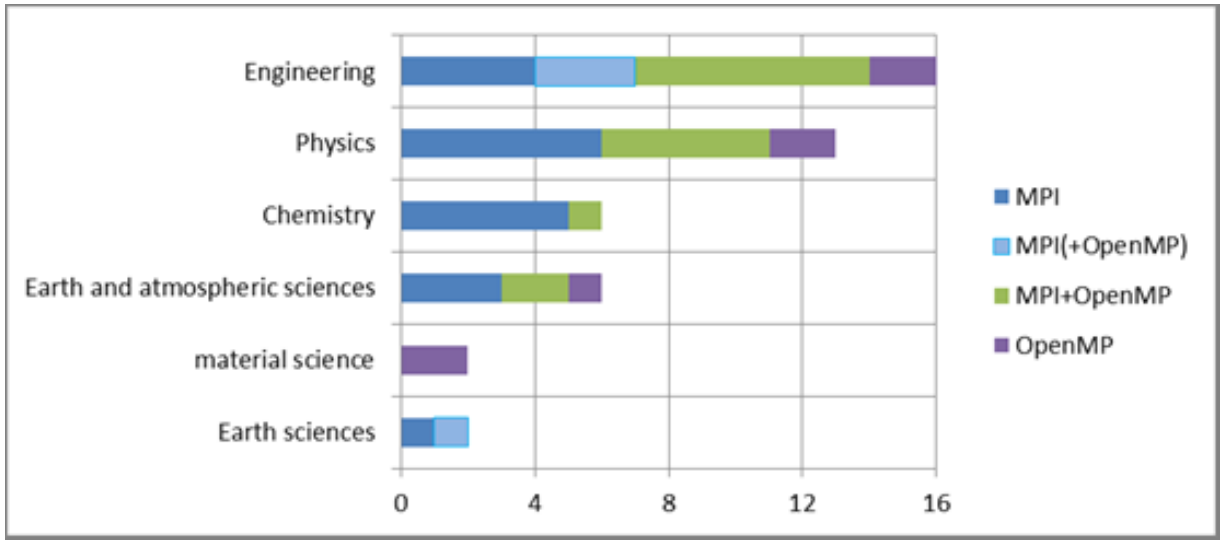


WP4 – Audit characterization

Code

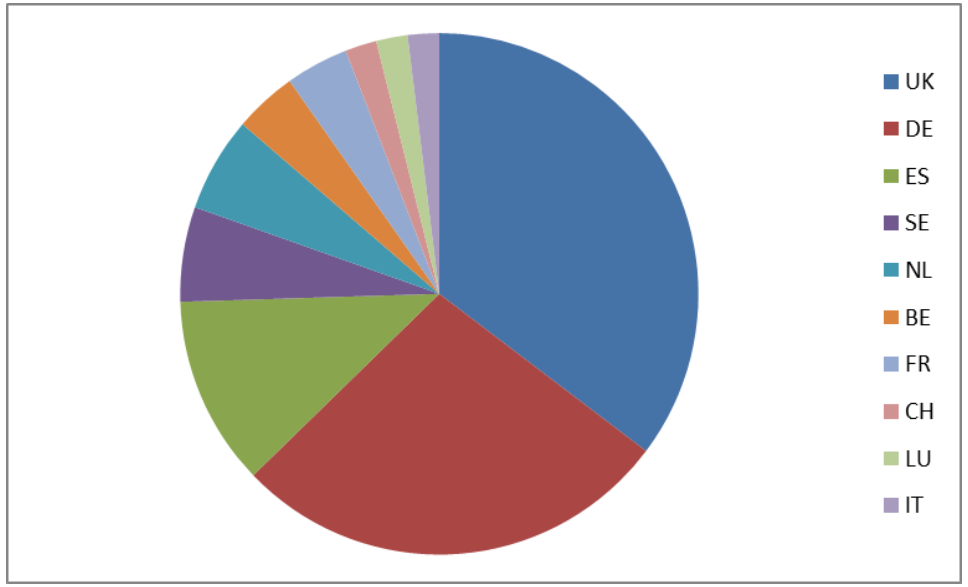
- **Scientific/technical area**
  - Dominated by Engineering and Physics
  - 90.5% of the requests from traditional HPC sectors
  - But also some requests on Data analytics, Deep learning, Medical, Media film, Text processing

Area versus parallel programing model



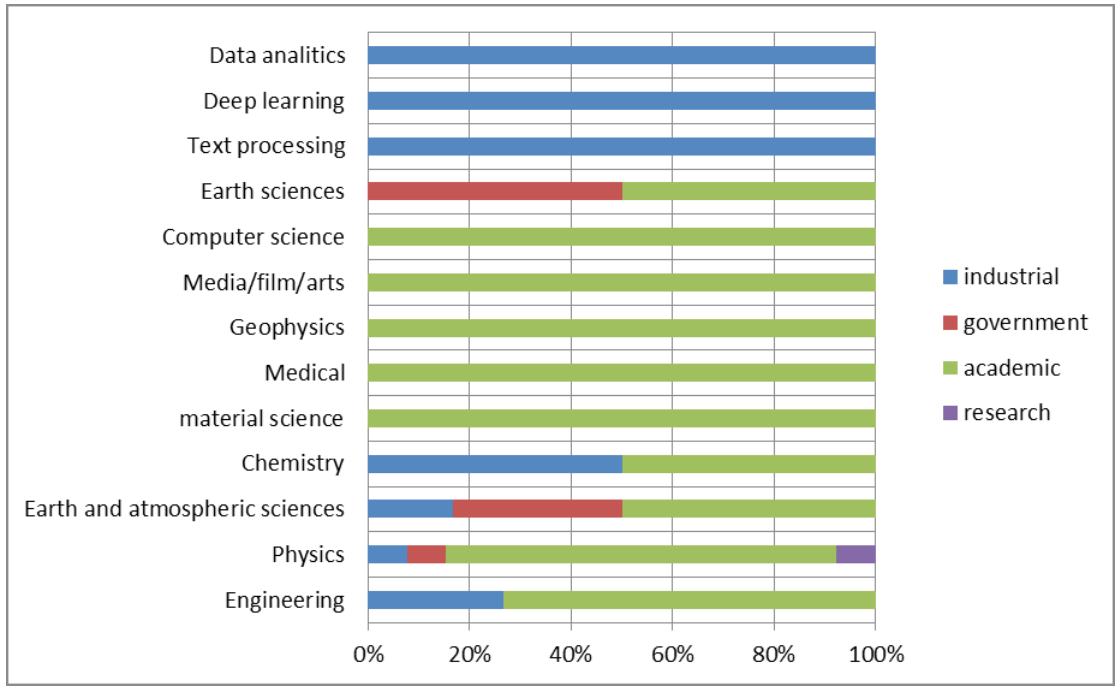
WP4 – Audit characterization

User profile



- **Country**
  - 23% requests from countries outside the consortium
  - 33.9% UK, 26.3% DE, 13.2% ES, 3.6% FR

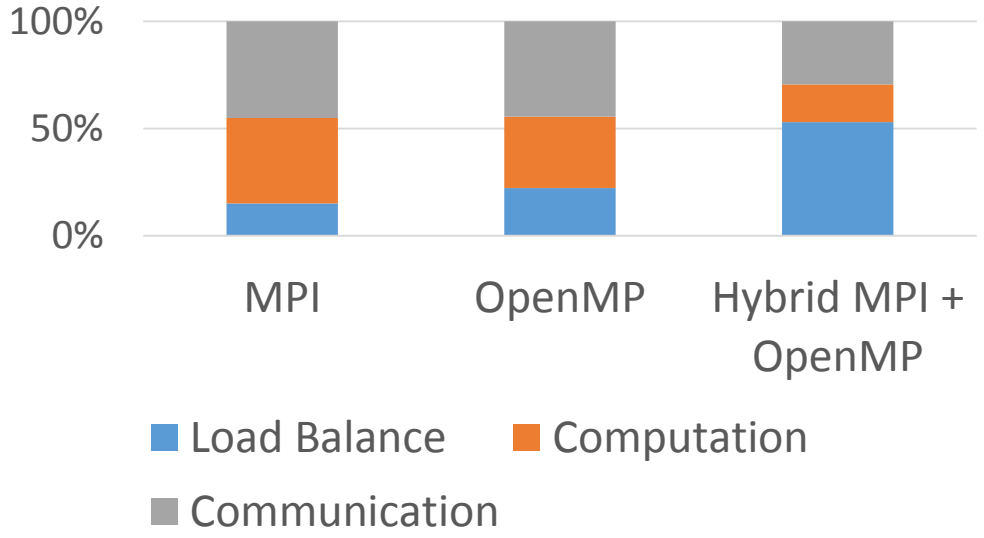
- **User institution versus code area**
  - Industrial companies provide all cases from new HPC sectors



WP4 – Audit characterization

Performance Audit results

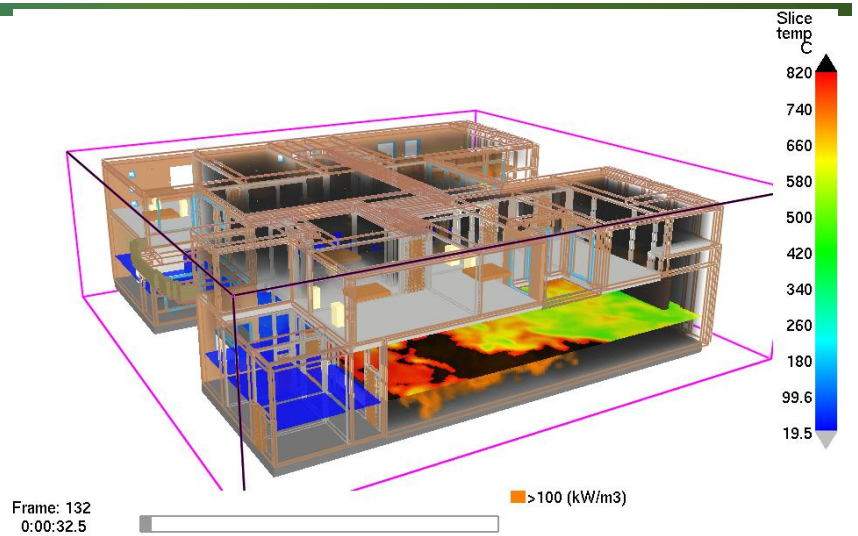
- **Parallel efficiency**
  - At least 67% would benefit / require optimizations (acceptable + bad)
  - Most frequent reason for acceptable efficiency is data transfer and for bad efficiency is load balance (+ data transfer)

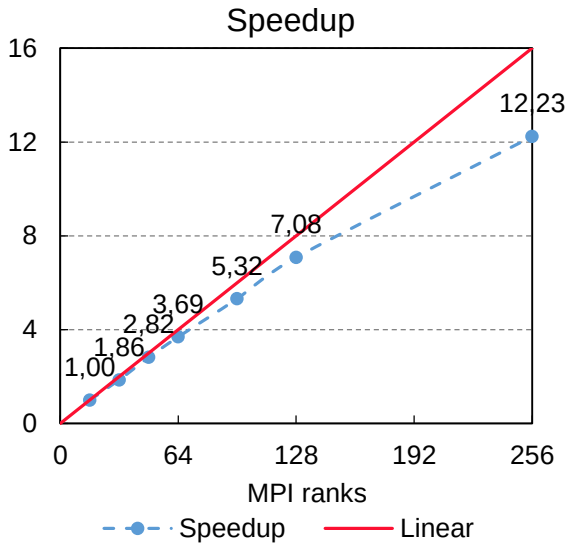


- **Serial performance (IPC)**
  - 44% have IPC >1 for all regions
  - Others may benefit from a serial performance improvement
    - 24% general IPC < 1

Case study: FDS Audit

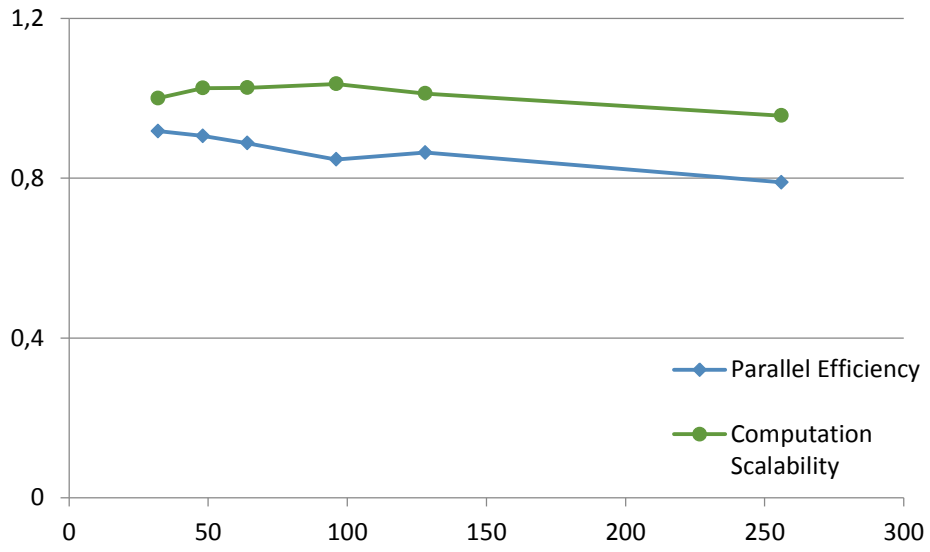
- User: Spanish SME
- Code: FDS (Fire dynamics simulation)
  - Simulates fire and smoke development in structures
- Code Area: Engineering
- Performance Audit:
  - Parallel efficiency drops for more than 200 cores
  - Evaluate efficiency running @ MareNostrum

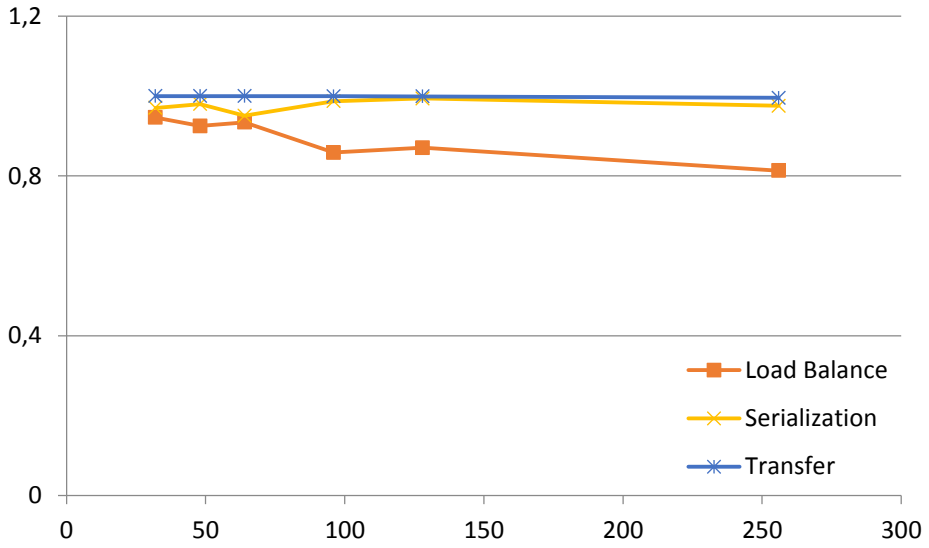




FDS Efficiency Analysis

- Analysis of MPI version with 32 – 256 ranks @ MN3



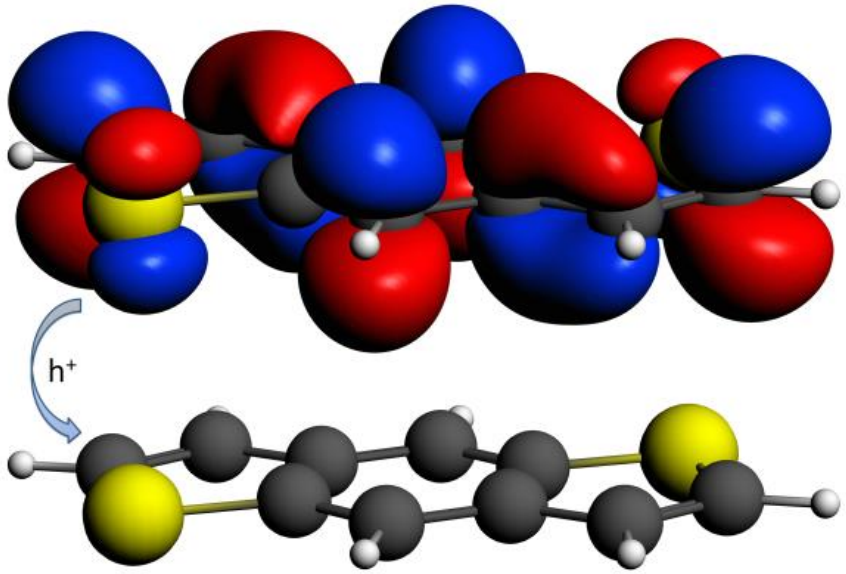


- Efficiencies still good at that scale
- Main lose of efficiency: unbalanced amount of work
- In MN3 a XYZ decomposition would improve balance and improve 20%

Case study: ADF Audit

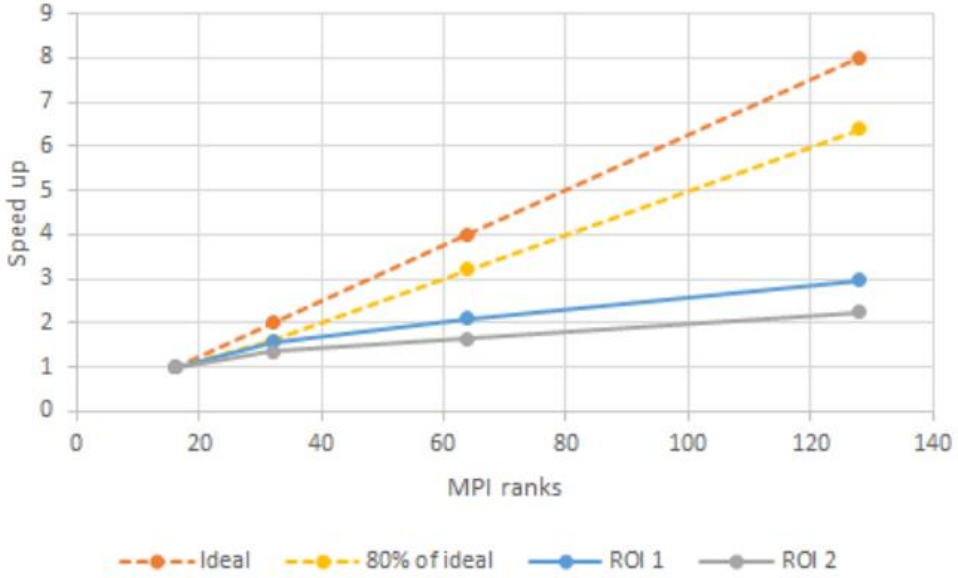
- User: Amsterdam-based SW company
- Code: ADF(Amsterdam Density Functional)
  - Understanding and predicting structure, reactivity and spectra of molecules
- Code Area: Computational chemistry
- Performance Audit:
  - Check application scalability and potential optimizations

SCM
Software for Chemistry & Materials
www.scm.com



ADF Audit analysis

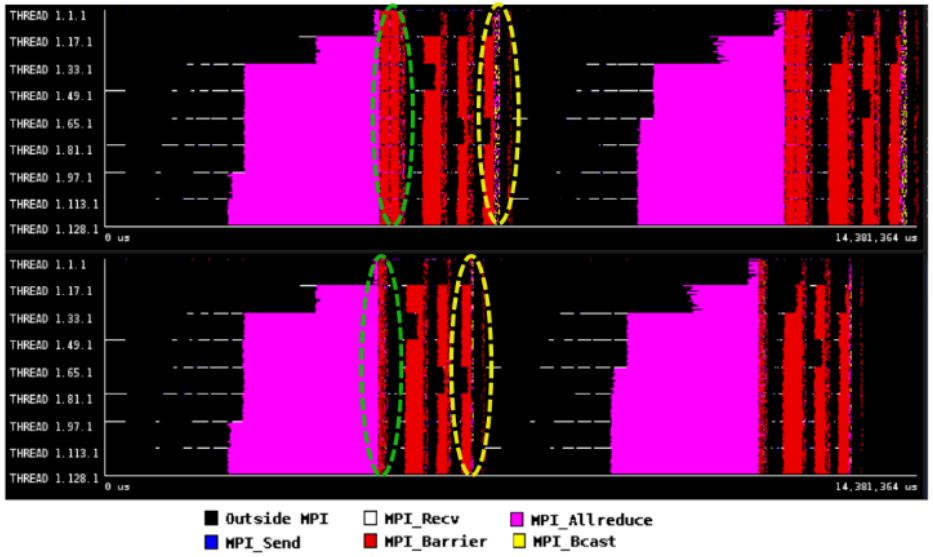
- Fortran with MPI and low-level shared arrays
- Very poor parallel efficiency caused by both load unbalance and communications



- Suggested a performance plan for a more detailed analysis

ADF Performance Plan results

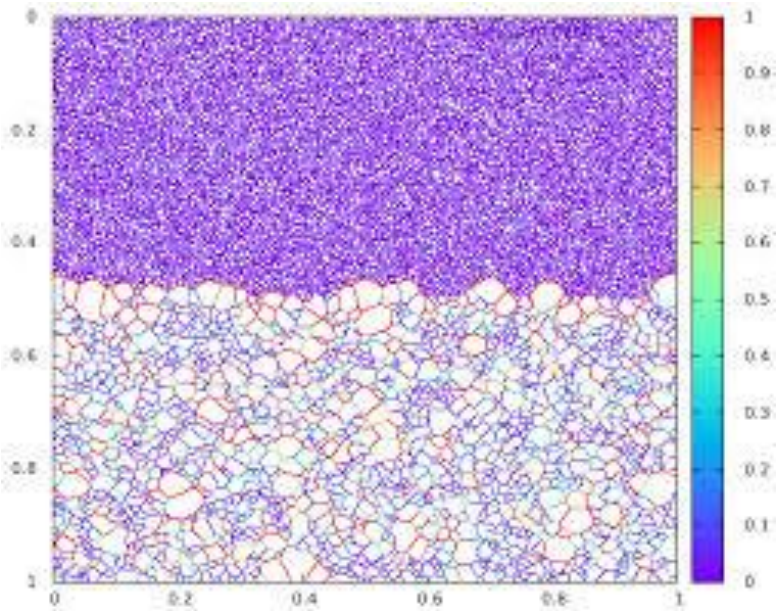
- Key Plan results:
  - Located unequal division of work
  - Work sharing amongst ranks was not frequent enough -> time spent waiting
  - Potential for up to a factor of two performance improvement



- Code changes implemented by the developers and released in their most recent update

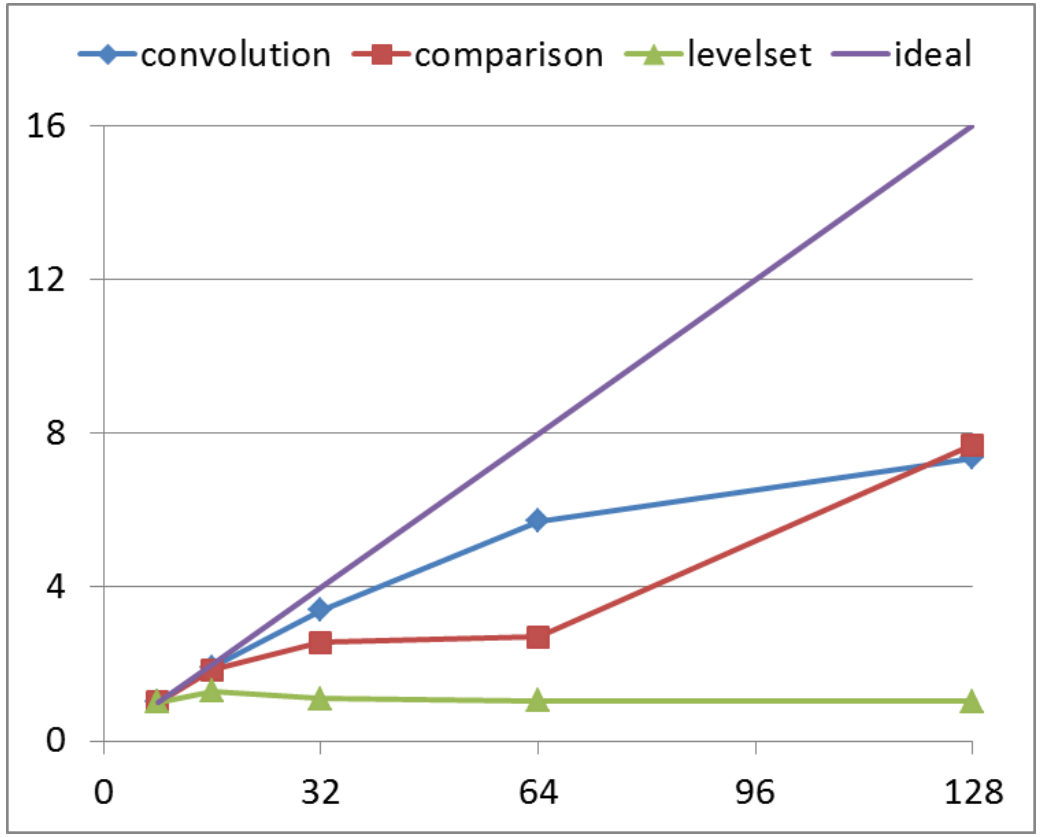
Case study: GraGLeS2D Audit

- User: German University
- Code: GraGLeS2D
  - Simulates the grain growth in polycrystalline materials
- Code Area: Material Science
- Performance Audit:
  - Poor scaling on a NUMA machine with 128 cores



GraGLeS2D Audit Analysis

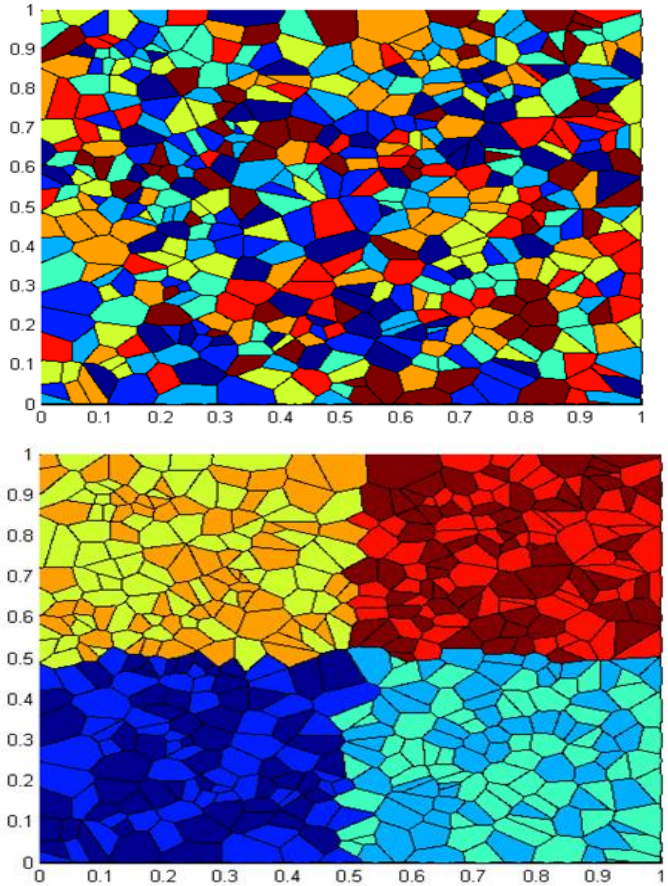
- Analysis of OpenMP with 8 – 128 cores
  - 4 boards x 4 sockets x 8 cores
- Observations from Audit
  - Work balance good except for the first iteration
  - Data sharing causing remote memory access reduces scalability
  - Detected consuming loops that can be vectorised
- PoC proposed and implemented



GraGLeS2D Proof of Concept

- PoC Plan
  - improve data-locality by thread pinning and load-distribution
  - improve vectorisation and serial performance

- Results on test input
  - parallel regions: speedup 6.4
  - overall application: speedup 2.2



Codes analyzed

- DPM
- Quantum Espresso
- DROPS
- Ateles
- SHP-Fluids
- GraGLeS2D
- NEMO
- VAMPIRE
- psOpen
- GYSELA
- AIMS
- OpenNN
- FDS
- Baleen
- Mdynamix
- ParFlow
- GITM
- BPMF
- FIRST
- SHEMAT
- GS2
- ADF
- DFTB
- ICON
- dwarf2-ellipticsolver
- EPW
- Code Saturne
- ONETEP
- Ms2
- SIESTA
- Oasys GSA
- SOWFA
- BAND
- NGA
- Fidimag
- LAMMPS
- ScalFMM
- CHAPSIM K.W.
- ArgoDSM
- CIAO
- FFEA
- k-Wave
- DSHplus
- RICH
- COOLFluiD
- Ondes3D
- ATK
- Molcas
- GBMol_DD
- Kratos
- cf-python

+ few under NDAs

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.