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)

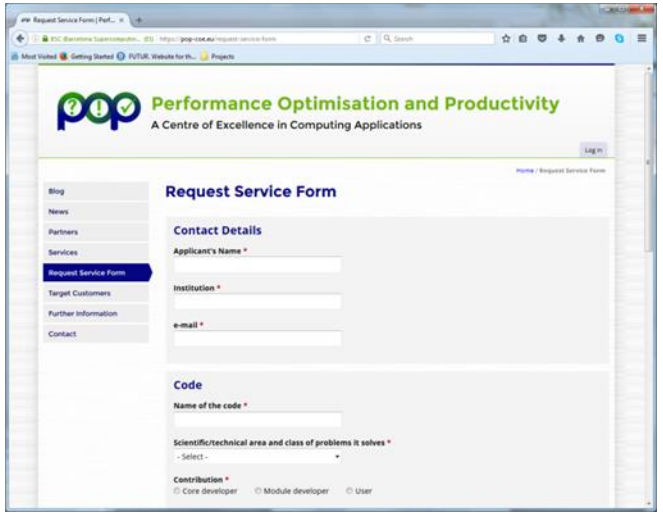
✓ Proof-of-Concept

- 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

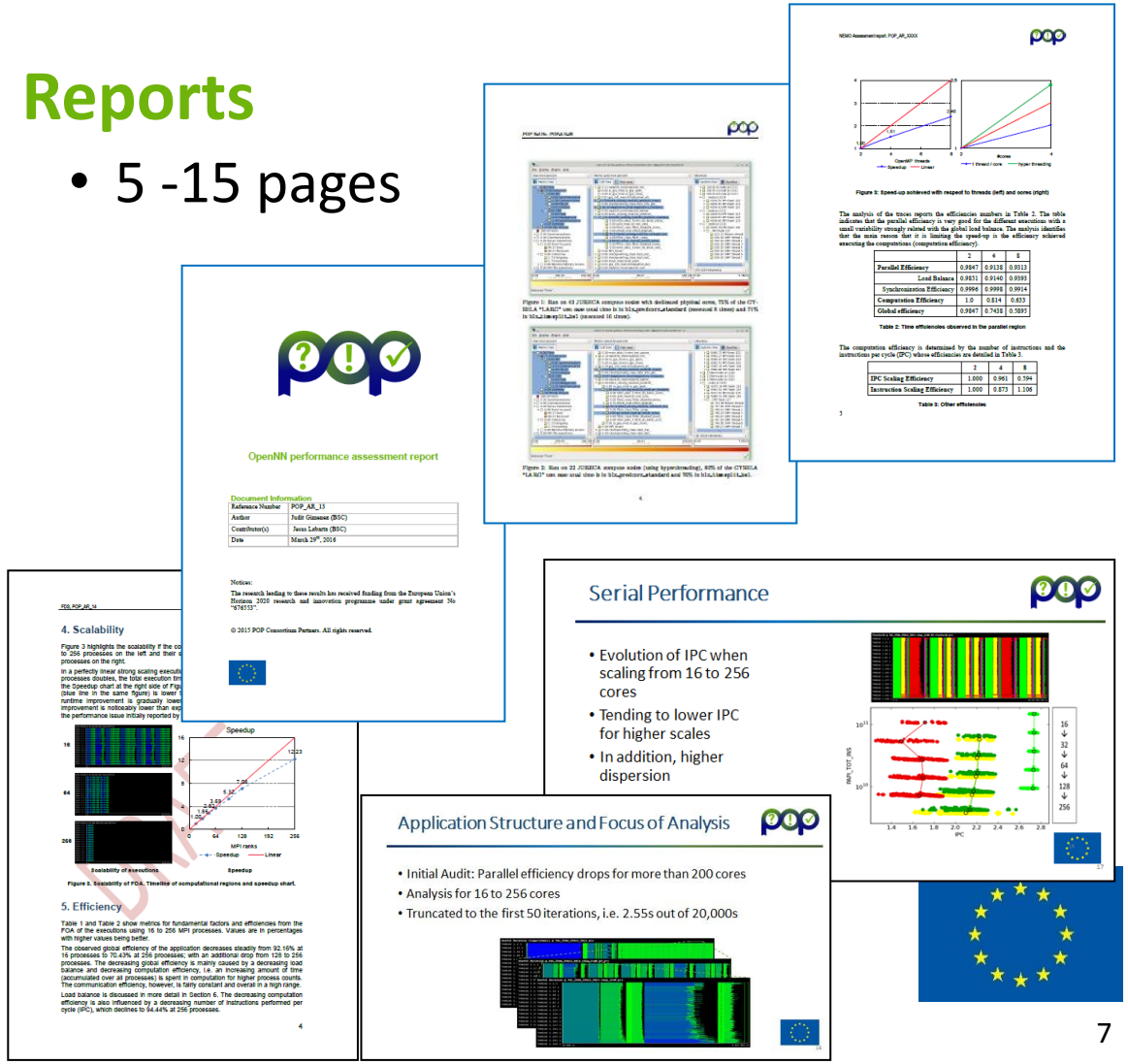
- **Code developers**
  - Assessment of detailed actual behavior
  - Suggestion of more productive directions to refactor code
- **Users**
  - Assessment of achieved performance on specific production conditions
  - Possible improvements modifying environment setup
  - Evidences to interact with code 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 (Feb 2017)

- **Services**
  - Completed/reporting: 54
  - Codes being analyzed: 16
  - Waiting user / New: 15
  - Cancelled: 7

- **By type**
  - Audits: 68
  - Plan: 11
  - Proof of concept: 6

- **Reports**
  - 5 -15 pages

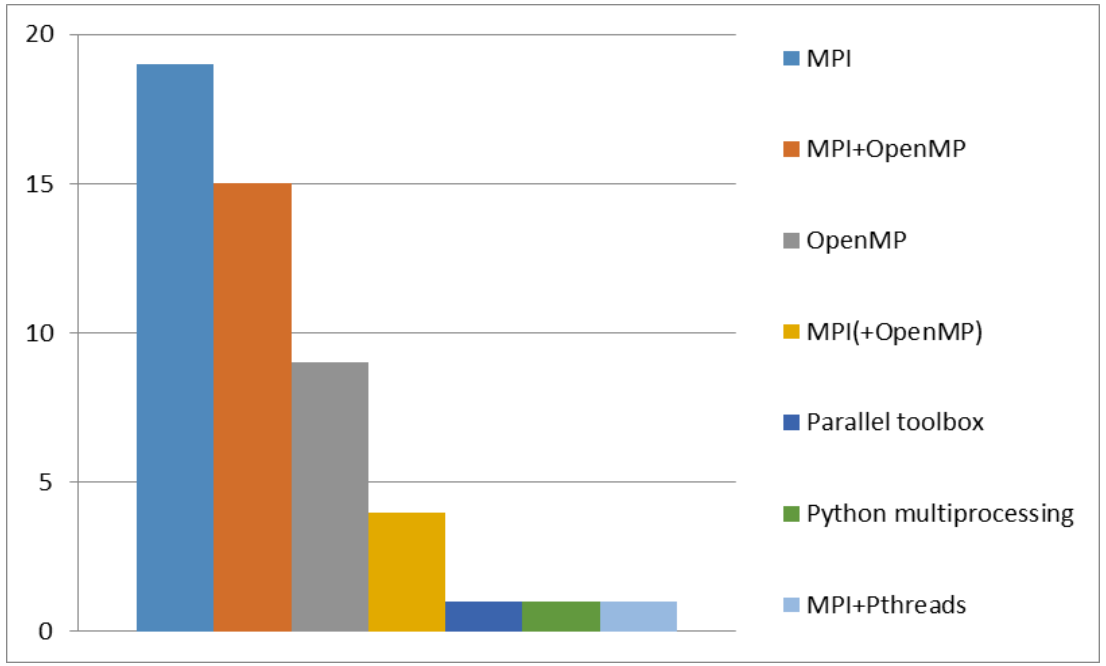


Other activities

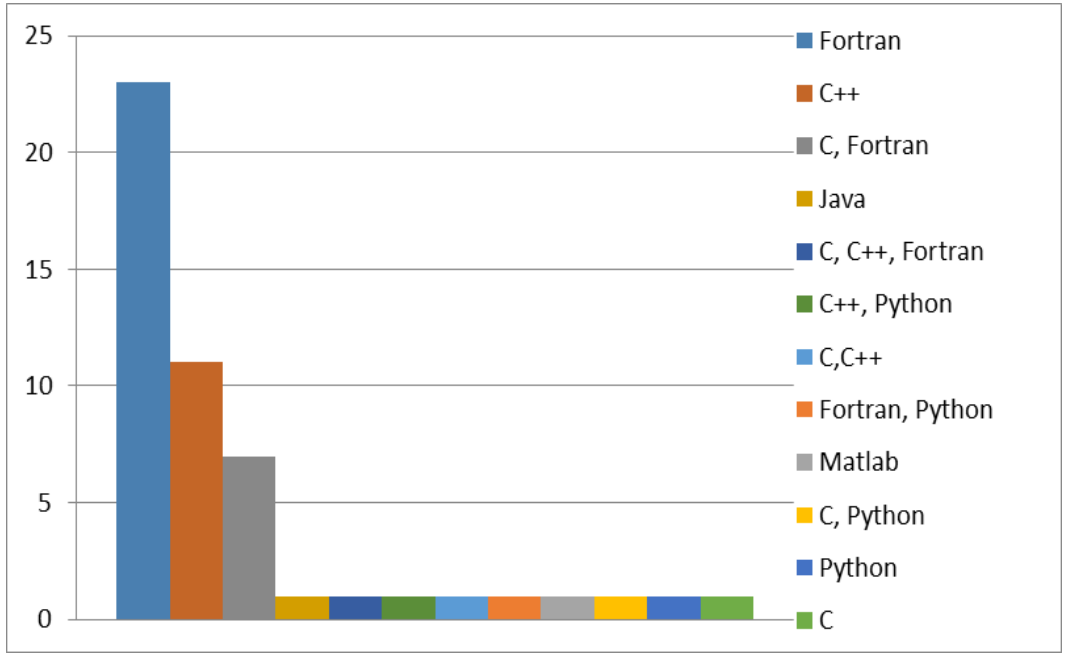
- **Promotion and dissemination**
  - Market and community development
  - Dissemination material and events
- **Customer advocacy**
  - Gather customers feedback, ensure satisfaction, steer activities
- **Sustainability**
  - Explore business models
- **Training**
  - Best practices on the use of the tools and programming models (MPI + OpenMP)
    - Lot of interest … customers want to learn how to do it themselves

WP4 – Audit characterization

Code



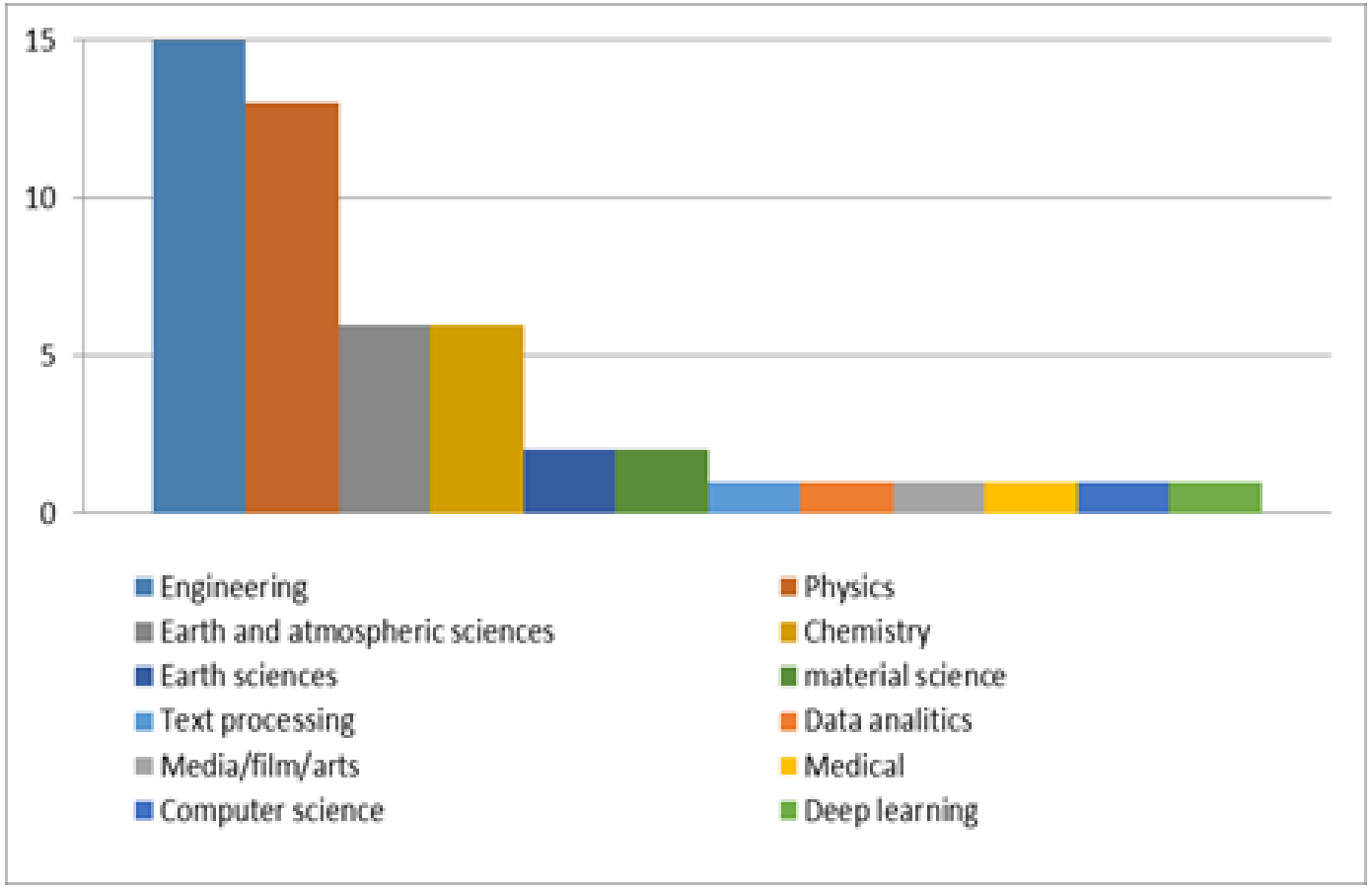
- **Parallel programming model**
  - 77% MPI or MPI+X
  - 17% pure OpenMP
  - Few from new paradigms



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

WP4 – Audit characterization

Code

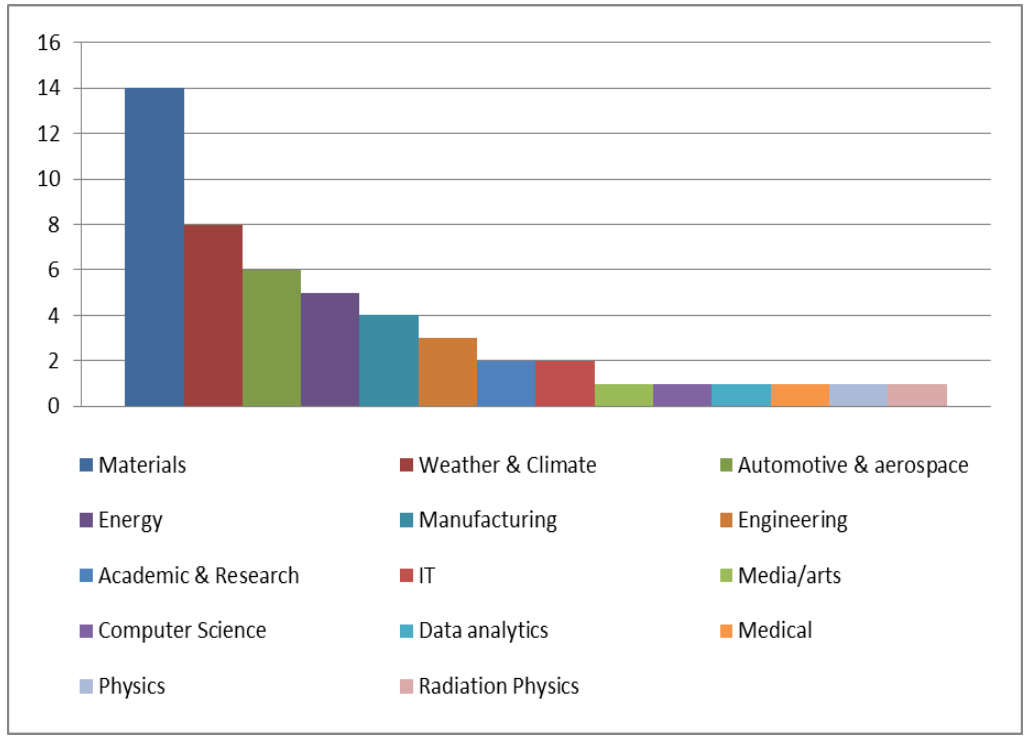


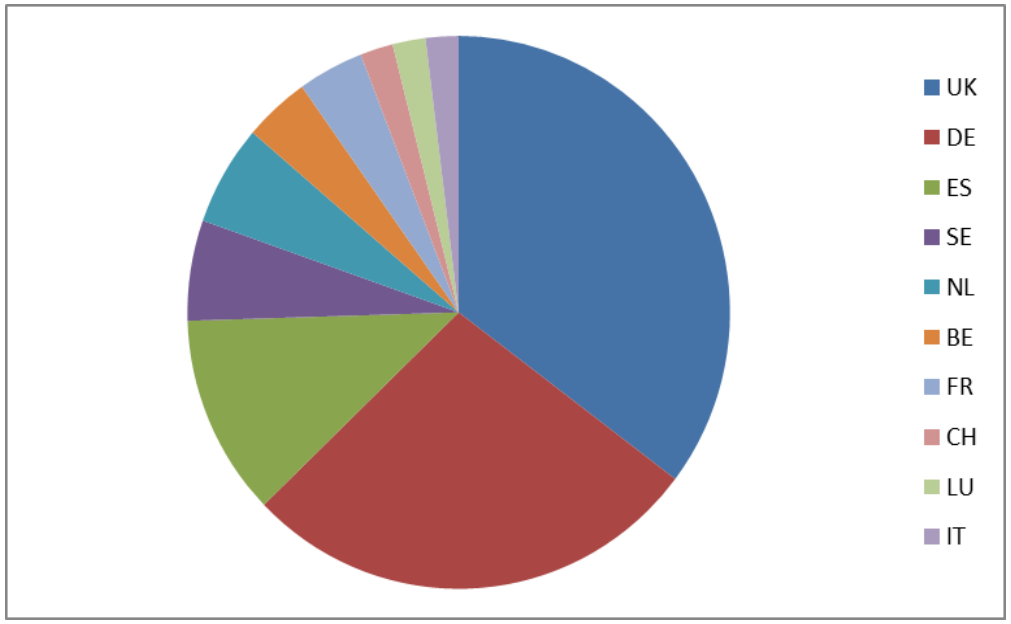
- **Scientific/technical area**
  - Dominated by Engineering and Physics
  - 90.5% of the requests from traditional HPC sectors
  - But also some requests on Data analytics, Deep learning, Medical, Media film, Text processing

WP4 – Audit characterization

User profile

- **Company /department sector**
  - 26.4% request from the materials sectors while only 3.7% of the codes classified as material by the user



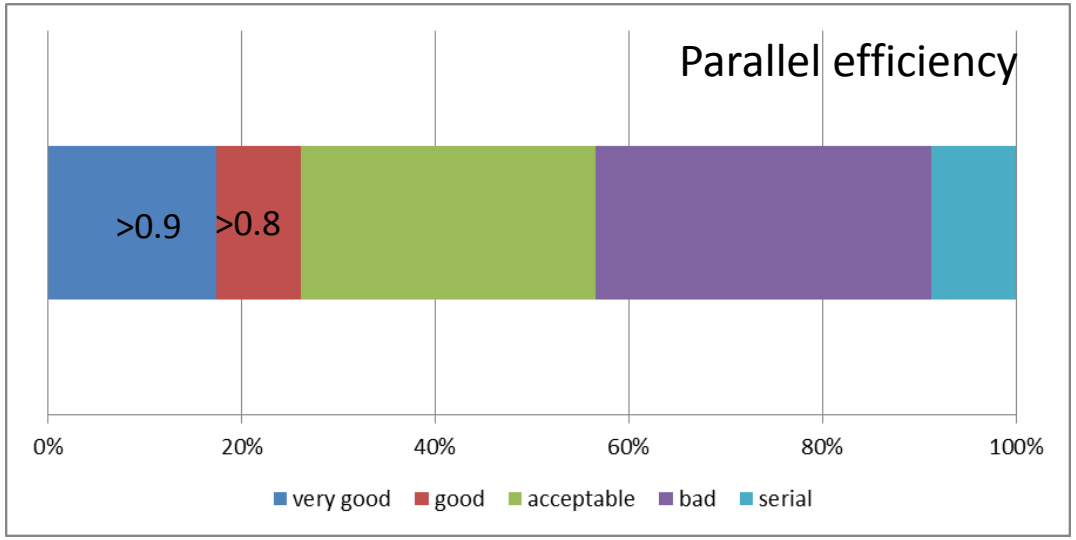


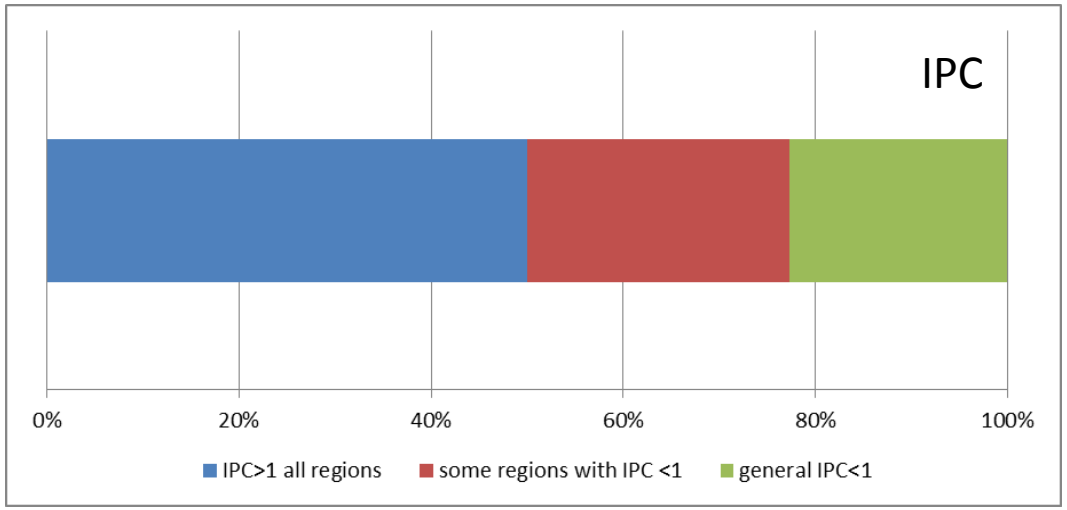
- **Country**
  - 23% requests from countries outside the consortium
  - 33.9% UK, 26.3% DE, 13.2% ES, 3.6% FR

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)

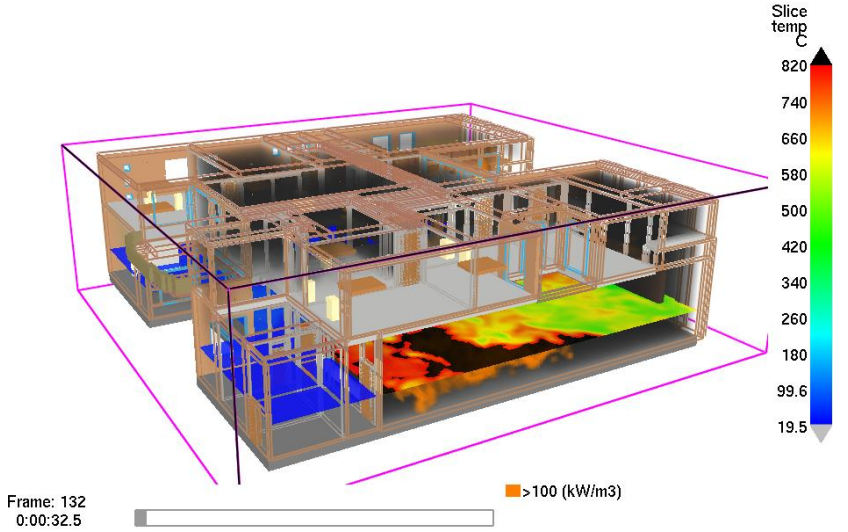


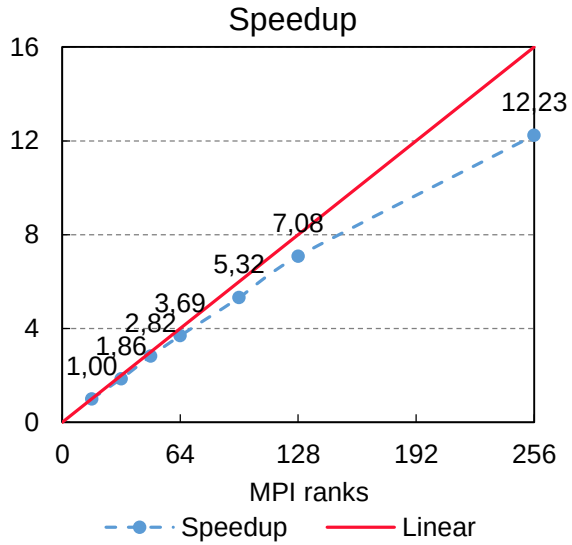


- **Serial performance (IPC)**
  - 44% have IPC >1 for all regions
  - Others may benefit from a serial performance improvement
    - 24% general IPC < 1

Case study: FDS Audit

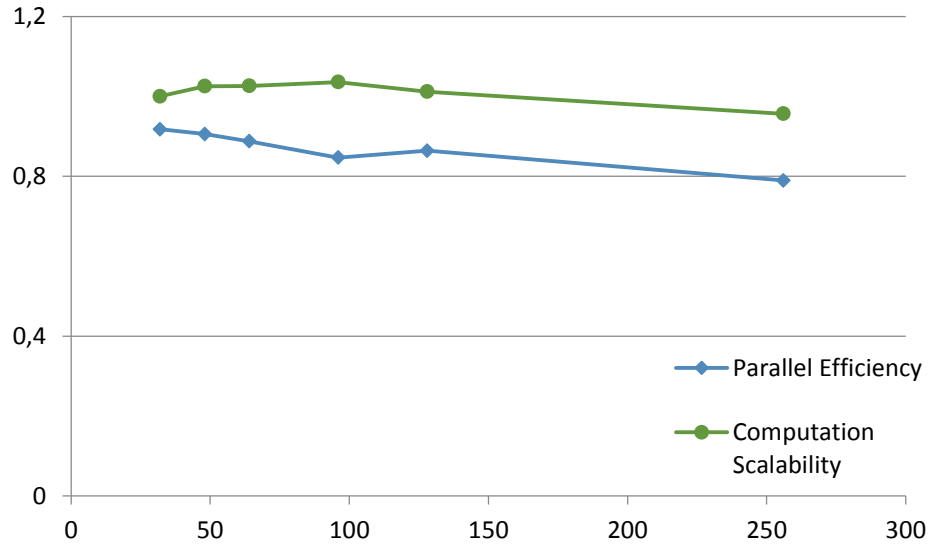
- User: Spanish SME
- Code: FDS (Fire dynamics simulation)
  - Simulates fire and smoke development in structures
- Code Area: Engineering
- Performance Audit:
  - Parallel efficiency drops for more than 200 cores
  - Evaluate efficiency running @ MareNostrum

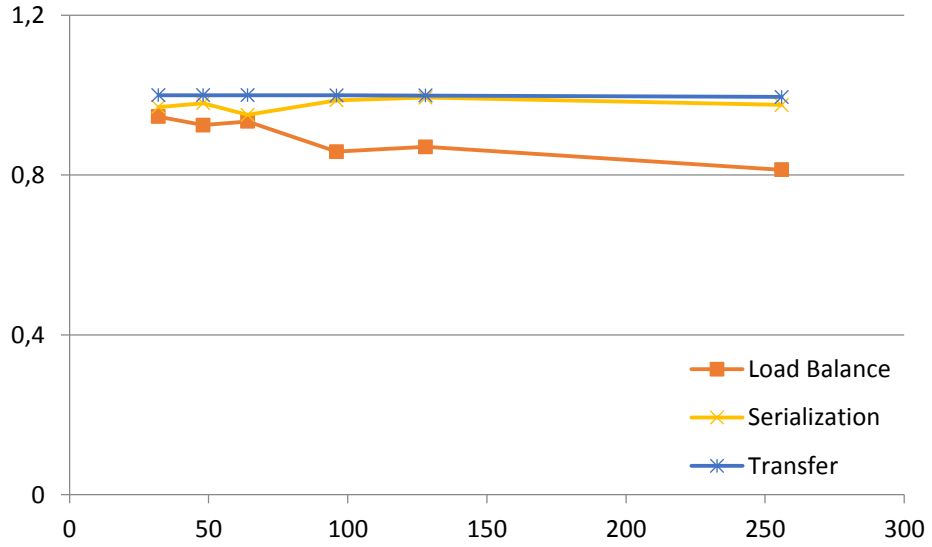




FDS Efficiency Analysis

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

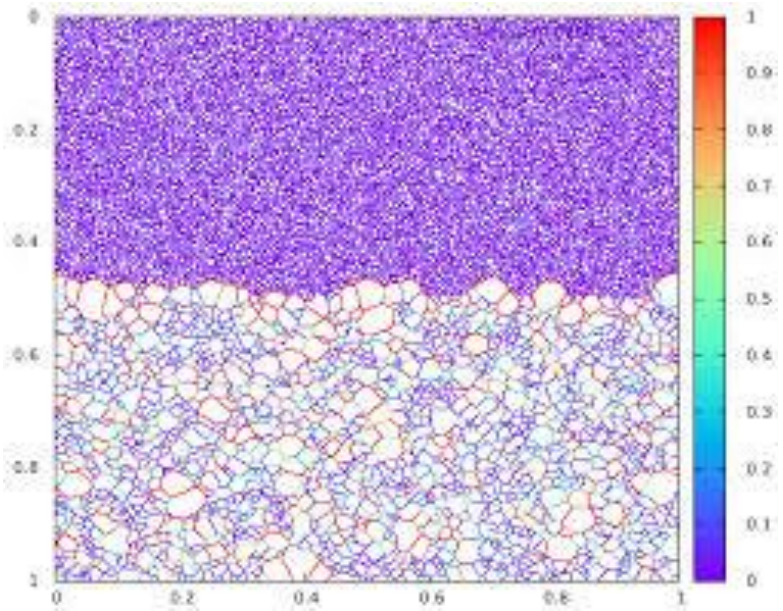




- Efficiencies still good at that scale
- Main lose of efficiency: unbalanced amount of work
- In MN3 a XYZ decomposition would improve balance and improve 20%

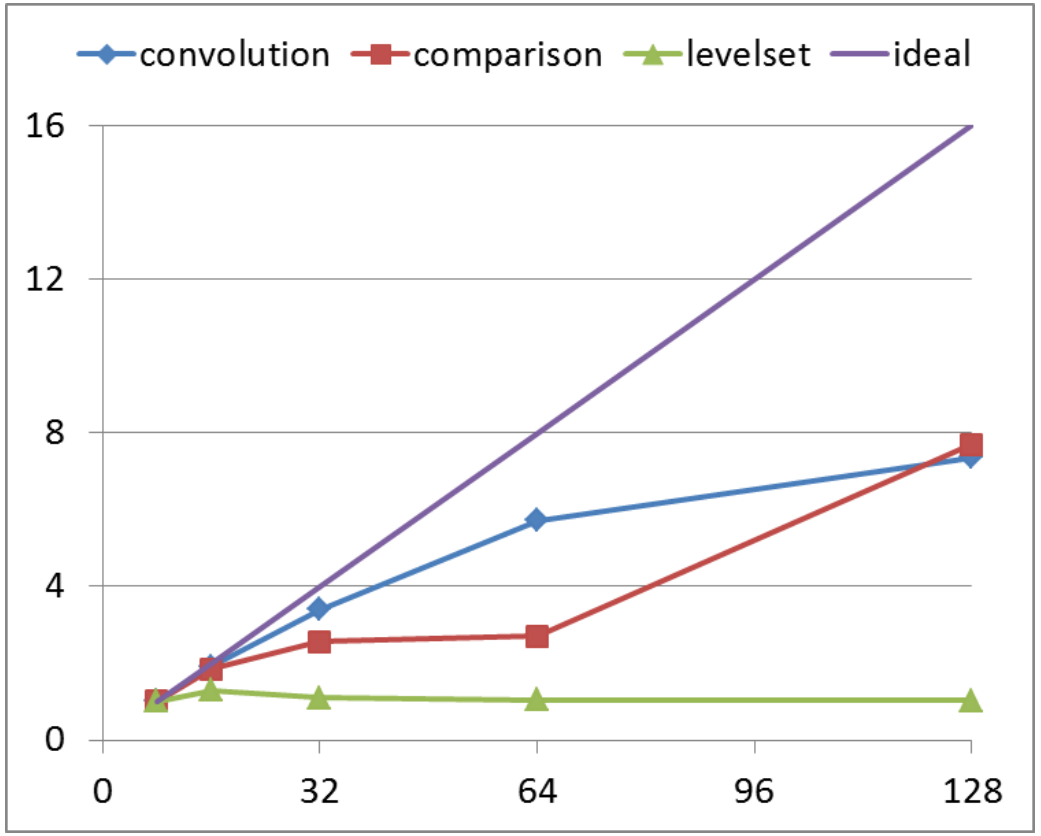
Case study: GraGLeS2D Audit

- 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



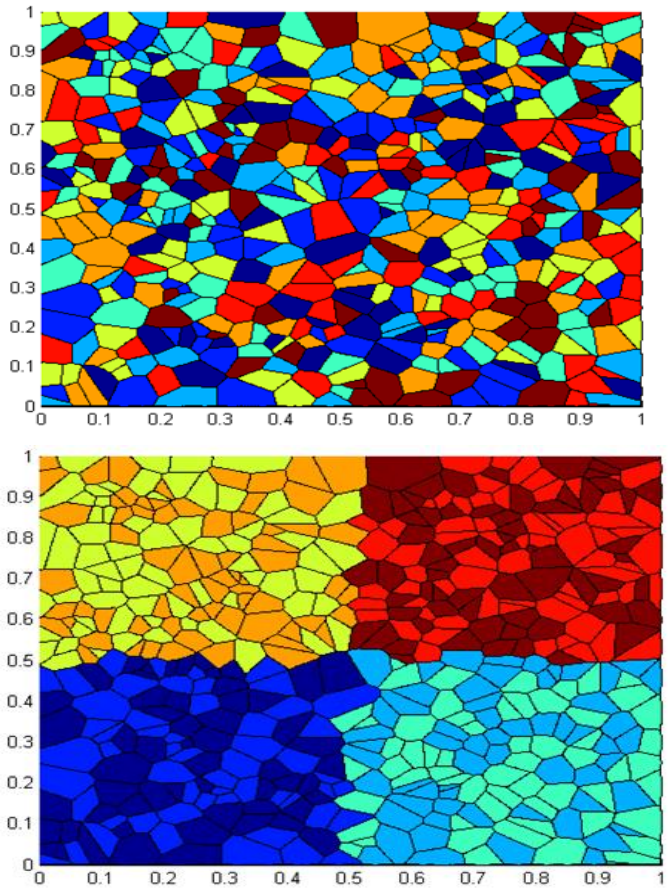
GraGLeS2D Audit Analysis

- 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



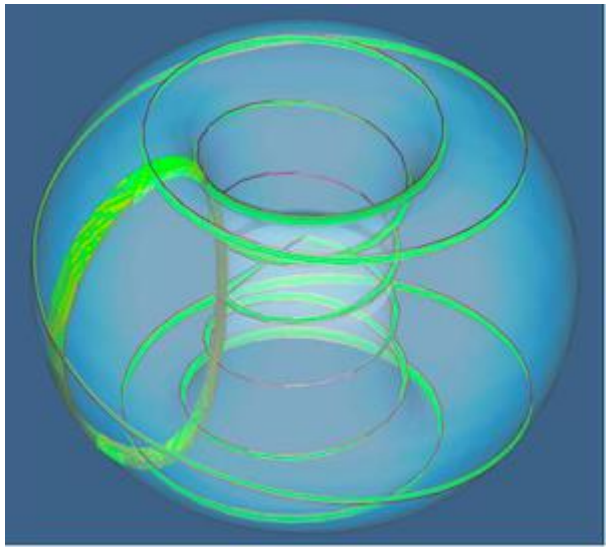
GraGLeS2D Proof of Concept

- 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



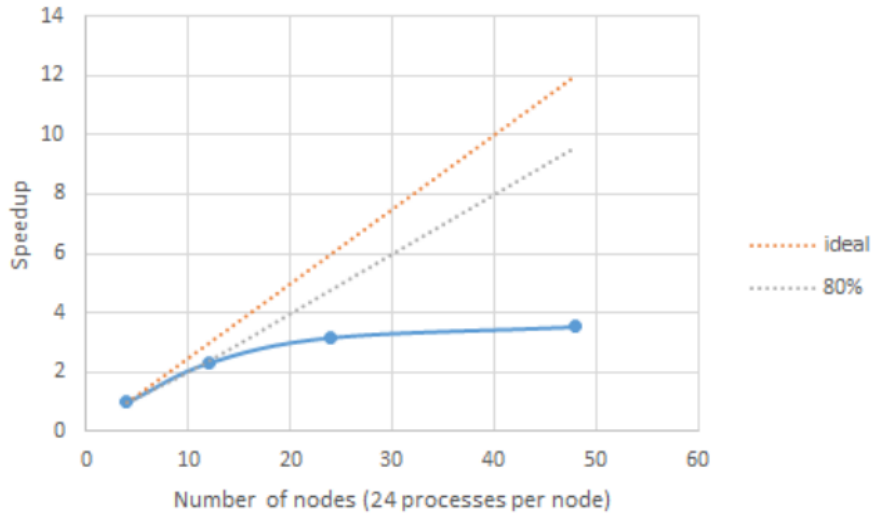
Case study: GS2 Audit

- User: UK national fusion laboratory (core developer) ITER project
- Code: GS2
  - Simulates low-frequency turbulence in magnetized plasma
- Code Area: Physics
- Performance Audit:
  - Code has strong scaling up to ~2000 cores. Want to confirm /identify bottleneck to improve scalability



GS2 Efficiency Analysis

- Analysis of MPI + SHMEM version for 4 – 48 nodes @ Archer



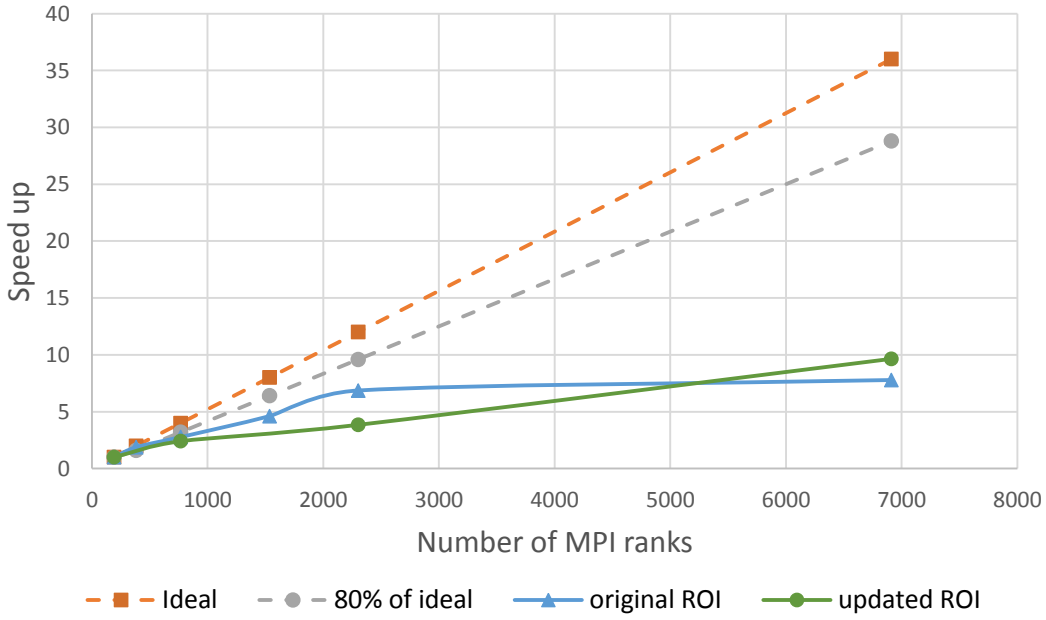
Nodes	4	12	24	48
Global Efficiency	47.3%	36.8%	25.2%	14.0%
↰ Computational Scalability*	100.0%	84.9%	67.6%	41.2%
↰ IPC Scalability*	100.0%	100.3%	93.9%	83.1%
↰ Instructions Scalability*	100.0%	85.4%	76.1%	53.9%
↰ Parallel Efficiency	47.4%	43.4%	37.2%	34.0 %
↰ Load Balance	81.1%	78.9%	76.7%	76.3 %
↰ Comm. Efficiency	58.4 %	55.0%	48.5 %	44.5%

- Efficiencies bad even with 4 nodes (96 cores)
- Main loss of efficiency: communication efficiency
- Main problem for scaling: code replication

Performance Plan proposed and being implemented

GS2 Performance Plan

- Analysis on larger production input set, MPI only
  - Frequent redistribution of data -> poor communication efficiency
- Evaluating EPCC improvement
  - Improved scaling but still far from 80% of ideal
  - Load imbalance: potential for ~50% performance improvement
  - Large reduction in data transferred but still inefficient due to dependencies
- Considering to apply a PoC



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.