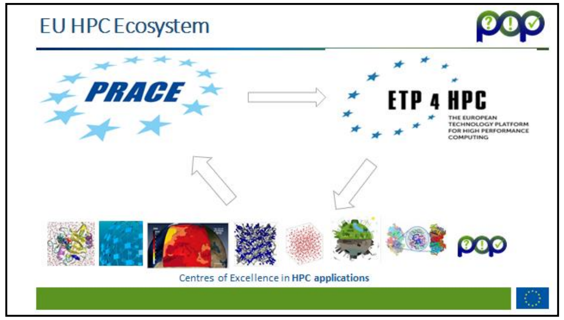
Parallel Performance Analysis and Tuning as a Service

EU H2020 Centre of Excellence (CoE)

1 October 2015 – 31 March 2018

Grant Agreement No 676553

POP CoE



- A **Centre of Excellence**
  - On **Performance Optimisation and Productivity**
  - Promoting **best practices in parallel programming**
- Providing **Services**
  - Precise understanding of application and system behaviour
  - Suggestion/support on how to refactor code in the most productive way
- **Horizontal**
  - Transversal across application areas, platforms, scales
- **For (your?) 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 behaviour

  → Not clear most productive direction of code refactoring

- Important to maximize efficiency (performance, power) of compute intensive applications and productivity of the development efforts

What?

- Parallel programs, mainly MPI/OpenMP
  - Although also CUDA, OpenCL, OpenACC, Python, …

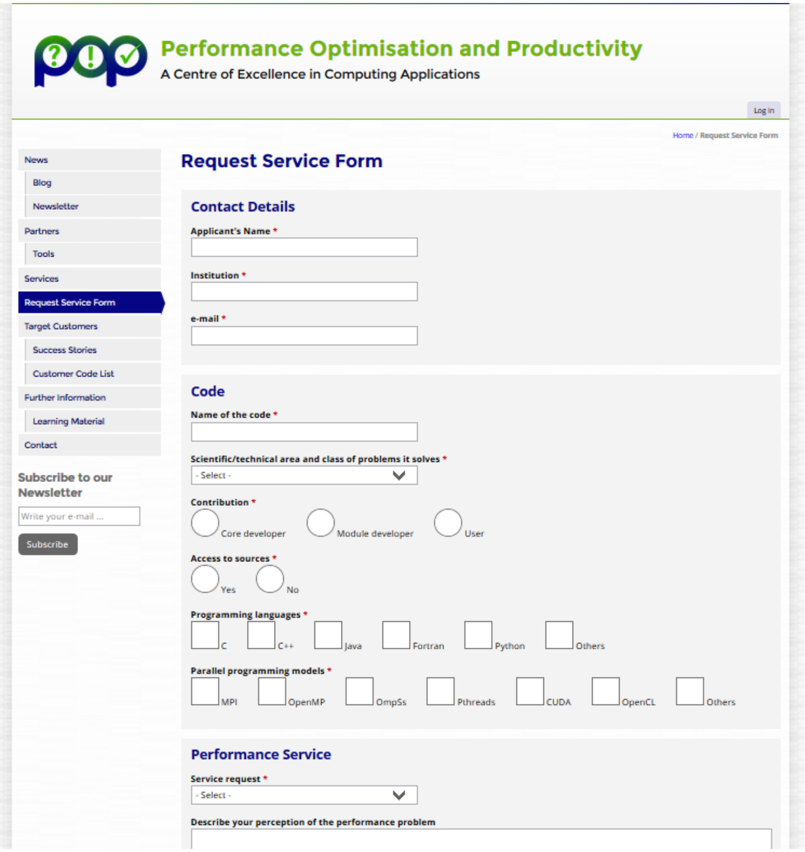
The process …

When?

October 2015 – March 2018

How?

- Apply
  - Fill in small questionnaire describing application and needs https://pop-coe.eu/request-service-form
  - Questions? Ask pop@bsc.es
- Selection/assignment process
- Install tools @ your production machine (local, PRACE, …)
- Interactively: Gather data → Analysis → Report



Services provided by the CoE

? Parallel Application Performance Audit ⇨ Report

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

! Parallel Application Performance Plan ⇨ Report

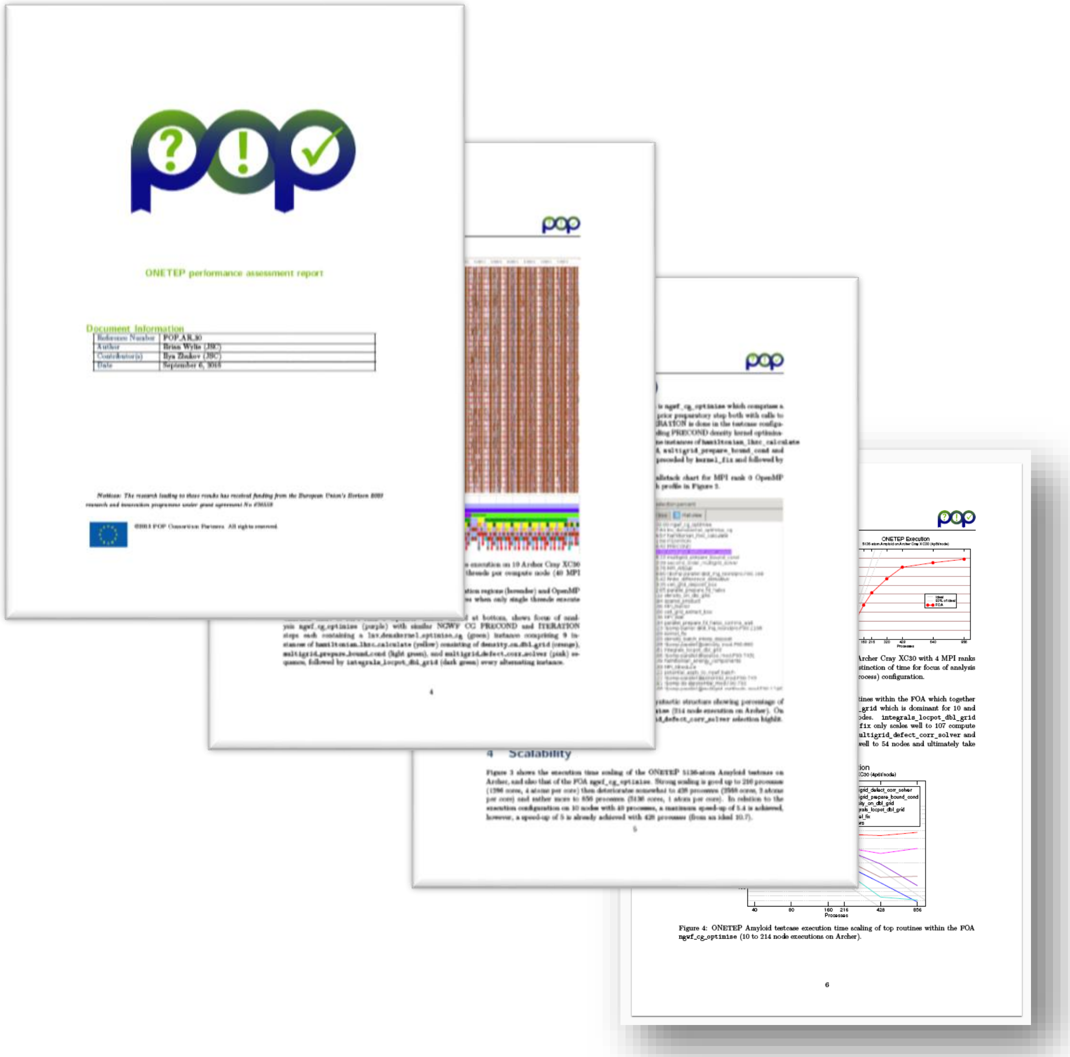
- Follow-up on the audit service
- Identifies the root causes of the issues found and qualifies and quantifies approaches to address them
- Longer effort (1-3 months)

✓ Proof-of-Concept ⇨ Software Demonstrator

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

Outline of a Typical Audit Report

- Application Structure
- (if appropriate) Region of Interest
- Scalability Information
- Application Efficiency
  - E.g. time spent outside MPI
- Load Balance
  - Whether due to internal or external factors
- Serial Performance
  - Identification of poor code quality
- Communications
  - E.g. sensitivity to network performance
- Summary and Recommendations



Efficiencies (WIP!)

- The following metrics are used in a POP Performance Audit:

CT = Computational time
TT = Total time

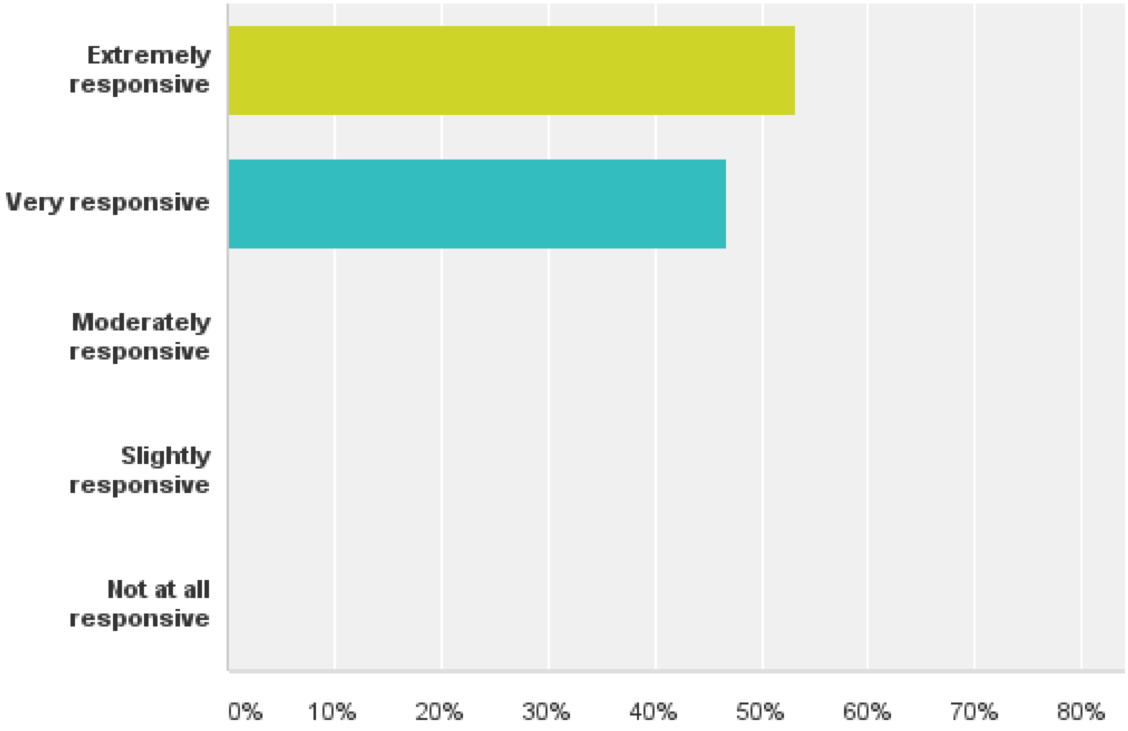
- Global Efficiency (GE): GE = PE * CompE
  - Parallel Efficiency (PE): PE = LB * CommE
    - Load Balance Efficiency (LB): LB = avg(CT)/max(CT)
    - Communication Efficiency (CommE): CommE = SerE * TE
      - Serialization Efficiency (SerE): SerE = max (CT / TT on ideal network)
      - Transfer Efficiency (TE): TE = TT on ideal network / TT
  - Computation Efficiency (CompE)
    - Computed out of IPC Scaling and Instruction Scaling
    - For strong scaling: ideal scaling -> efficiency of 1.0
- Details see https://sharepoint.ecampus.rwth-aachen.de/units/rz/HPC/public/Shared%20Documents/Metrics.pdf

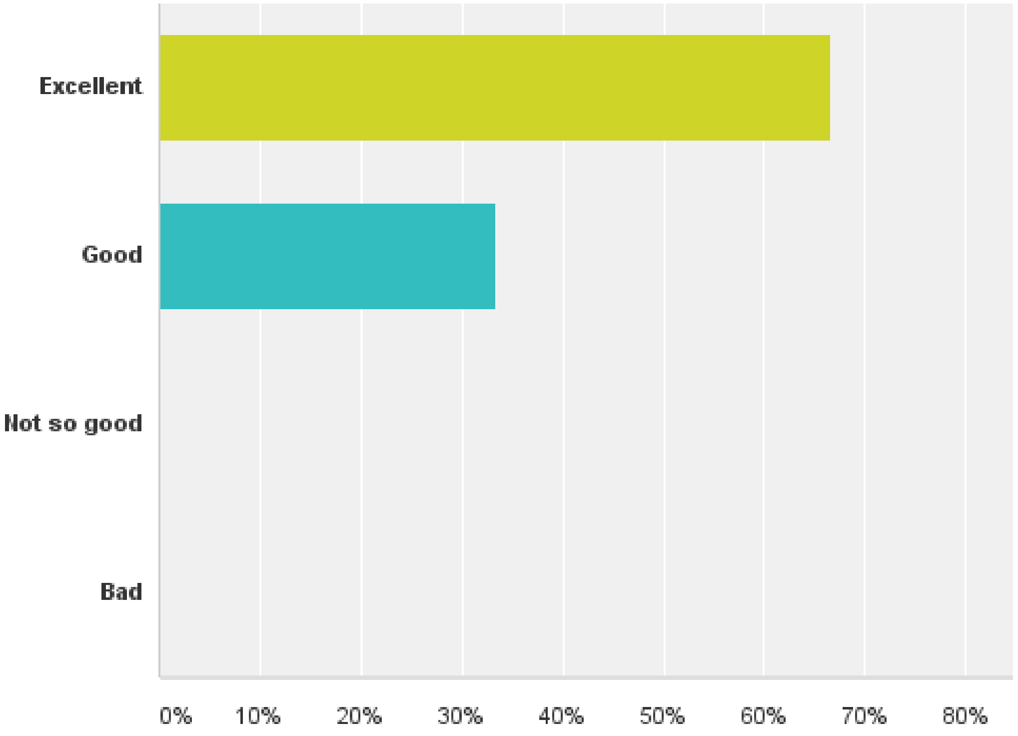
POP Users and Their Codes

Area	Codes
Computational Fluid Dynamics	DROPS (RWTH Aachen), Nek5000 (PDC KTH), SOWFA (CENER), ParFlow (FZ-Juelich), FDS (COAC) & others
Electronic Structure Calculations	ADF (SCM), Quantum Expresso (Cineca), FHI-AIMS (University of Barcelona), SIESTA (BSC), ONETEP (University of Warwick)
Earth Sciences	NEMO (BULL), UKCA (University of Cambridge), SHEMAT-Suite (RWTH Aachen) & others
Finite Element Analysis	Ateles (University of Siegen) & others
Gyrokinetic Plasma Turbulence	GYSELA (CEA), GS2 (STFC)
Materials Modelling	VAMPIRE (University of York), GraGLeS2D (RWTH Aachen), DPM (University of Luxembourg), QUIP (University of Warwick) & others
Neural Networks	OpenNN (Artelnics)

Customer Feedback (Sep 2016)

- Results from 18 of 23 completed feedback surveys (~78%)





- How responsive have the POP experts been to your questions or concerns about the analysis and the report?
- What was the quality of their answers?

Best Practices in Performance Analysis

- **Powerful tools …**
  - Extrae + Paraver
  - Score-P + Scalasca/TAU/Vampir + Cube
  - Dimemas, Extra-P
  - Commercial tools (if available)

- **… and techniques**
  - Clustering, modeling, projection, extrapolation, memory access patterns,
  - … with extreme detail …
  - … and up to extreme scale

- **Unify methodologies**
  - Structure
    - Spatio temporal / syntactic
  - Metrics
    - Parallel fundamental factors: Efficiency, Load balance, Serialization
    - Programming model related metrics
    - User level code sequential performance
  - Hierarchical search
    - From high level fundamental behavior to its causes

- **To deliver insight**
- **To estimate potentials**

Proof-of-Concept Examples

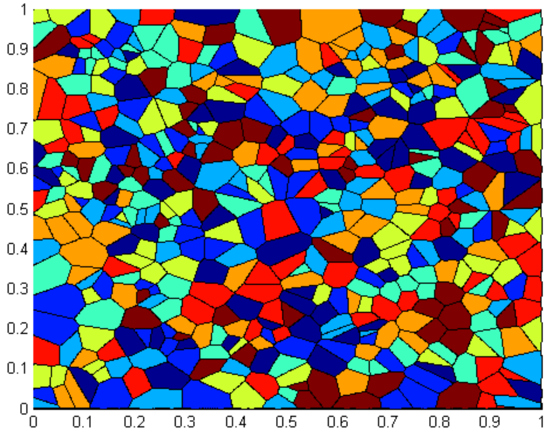
GraGLeS2D – RWTH Aachen

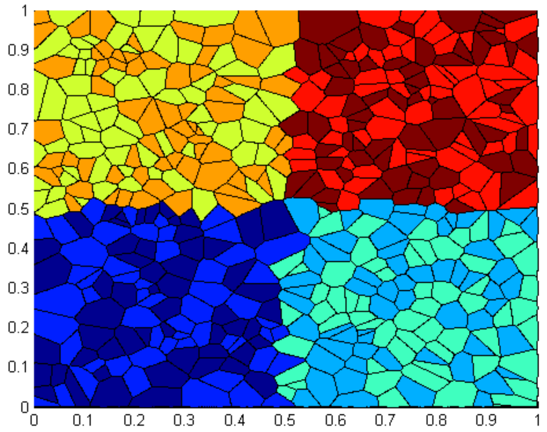
- Simulates grain growth phenomena in polycrystalline materials
- C++ parallelized with OpenMP
- Designed for very large SMP machines (e.g. 16 sockets and 2 TB memory)

- Key audit results:
  - Good load balance
  - Costly use of division and square root inside loops
  - Not fully utilising vectorisation in key loops
  - NUMA specific data sharing issues lead to long times for memory access

GraGLeS2D – RWTH Aachen

- Improvements:
  - Restructured code to enable vectorisation
  - Used memory allocation library optimised for NUMA machines
  - Reordered work distribution to optimise for data locality

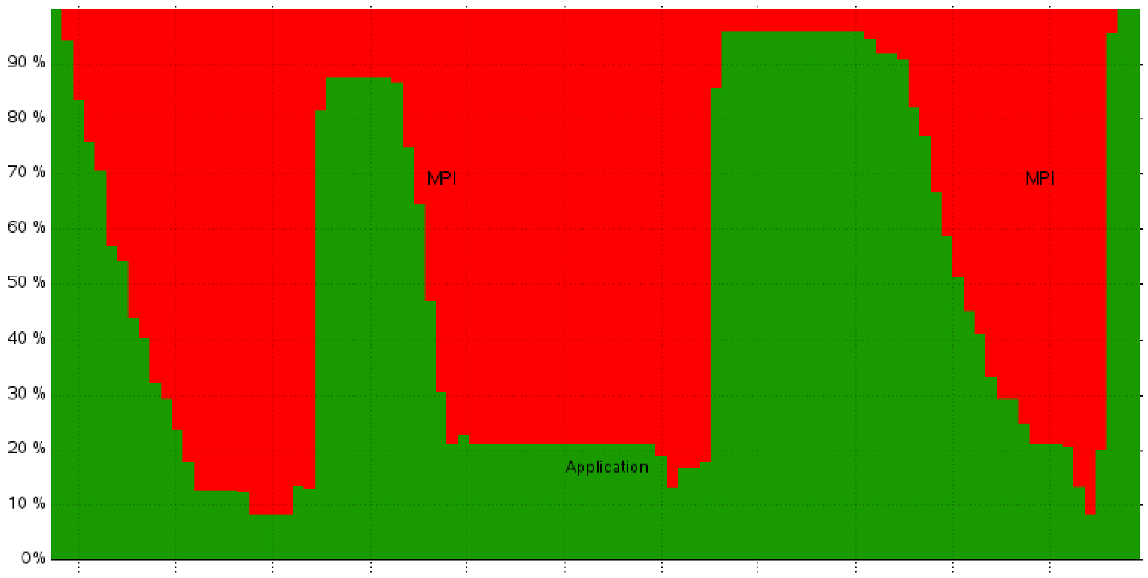




- Speed up in region of interest is more than 10x
- Overall application speed up is 2.5x

Ateles – University of Siegen

- Finite element code
- C and Fortran code with hybrid MPI+OpenMP parallelisation
- Key audit results:
  - High number of function calls
  - Costly divisions inside inner loops
  - Poor load balance
- Performance plan:
  - Improve function inlining
  - Improve vectorisation
  - Reduce duplicate computation



Ateles – Proof-of-concept

- Inlined key functions → 6% reduction in execution time
- Improved mathematical operations in loops → 28% reduction in execution time
- Vectorisation: found bug in gnu compiler, confirmed Intel compiler worked as expected

- 6 weeks software engineering effort
- Customer has confirmed “substantial” performance increase on production runs

Sustainability

- H2020 CoE's are supposed to sustain themselves after some point
  - Proposals had to include a business plan
- Current plan: 3 sustainable operation modes
  - Pay-per-service
  - Service subscriptions
  - Continue as non-profit organisation (broker for free + payed services)
- Requires to have more industrial rather than academic/research customers
  - Experience so far
    - Typically require NDA ⇨ delays services by months
    - No access to code/computers ⇨ guide (inexperienced) customer to install tools + measure ⇨ delays services by months

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.