POP CoE: Free Performance Assessments for the HPC Community

Bernd Mohr, Jülich Supercomputing Centre

EU H2020 Centre of Excellence (CoE)

1 December 2018 – 30 November 2021

Grant Agreement No 824080

Performance Optimisation and Productivity

A Centre of Excellence in HPC

- Promotes best practices in parallel programming
  - Improving Parallel Software can add a lot of value:
    Reduced expenditure, faster results, novel solutions
  - **The POP Methodology** - a systematic approach to performance optimization building a quantitative picture of application behavior
- Free services for all EU academic and industrial codes and users
  - Suggestions on improving code performance, described in a ***Performance Assessment***
  - Practical help with code refactoring through a ***Proof of Concept***

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Teratec

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

JÜLICH Forschungszentrum | JÜLICH SUPERCOMPUTING CENTRE

HLRS

nag®

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

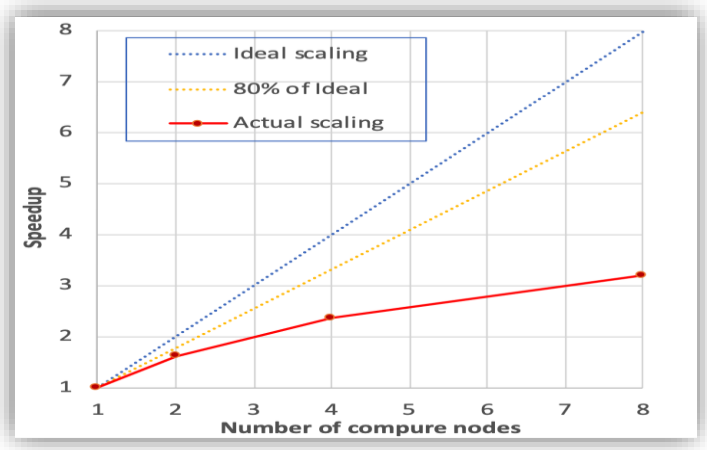
RWTH AACHEN UNIVERSITY

For further information, visit: https://www.pop-coe.eu pop@bsc.es @POP_HPC youtube.com/POPHPC

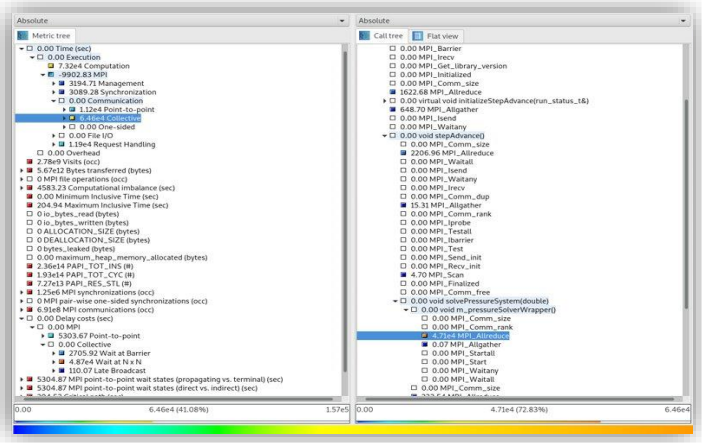
Parallel Performance is hard to understand

How do we measure the performance of our parallel programs?

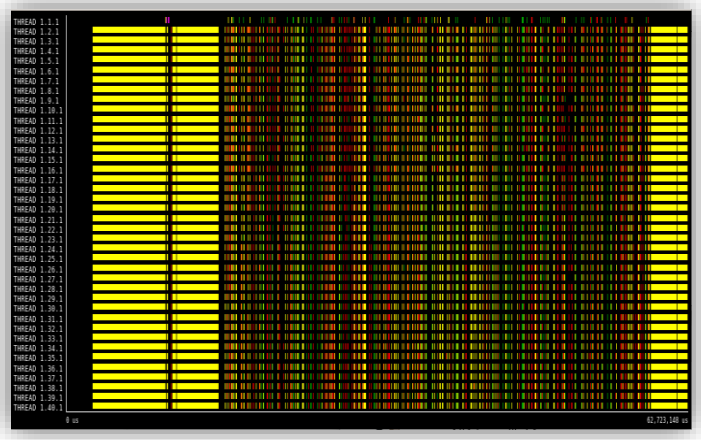
- Traditional speed-up and efficiency plots?
- Profiling & tracing with performance tools?
  - Tracing is powerful, but potentially generates overwhelming amount of data



Speedup plot



Cube, perf. metrics per routines/call stack, data collected by Scalasca/Score-P



Paraver, timeline view of program execution, data collected by Extrae

Difficult to know where to start and what to look for

Main Problem: Lack of quantitative understanding of the actual behavior of a parallel application

A Solution:

The POP Metrics

Simple but extremely powerful idea

- Devise a simple set of performance metrics (efficiencies, [0:1]) using values easily obtained from the trace data
- Where low values indicate **specific** causes of poor parallel performance

These metrics then are used to understand

- What are the causes of poor performance
- What to look for in the trace data
- Besides, the metrics provide a common ground for discussing performance issues
  - Between developers, users and analysts

POP MPI Parallel Efficiency Metrics

Global Efficiency

Describes how well your parallel application scales.

GE = PE × Comps

Parallel Efficiency

Describe how well the execution of your code is working in parallel

PE = LB × CommE

Computational Scaling

Describe how well the computational load of an application scales with the number of processors/threads.

CompS

Communication Efficiency

Reflects the loss of efficiency by communication.

CommE = max (CT/TT) = SerE × TE

Load Balance

Measures how well work is distributed to threads/processes in the application.

LB = average (CT) / max (CT)

Transfer Efficiency

Describes the loss of efficiency due to actual data transfer time.

TE = TT on ideal network / TT

Serialisation Efficiency

Describes efficiency loss due to dependencies, waiting in MPI calls, ets.

SerE = max(CT / TT on ideal network)

Key

CT Computation Time

TT Total Time

Ideal network data transfer takes zero time

Instruction Scaling

Compares the total number of instructions executed for different numbers of threads/processes.

IPC Scaling

Compares how many instructions per cycle (IPC) are executed for different numbers of threads/processes.

Frequency Scaling

Compares the processor frequency for different numbers of threads/processes.

POP Performance Monitoring Tools

Developing open-source tools

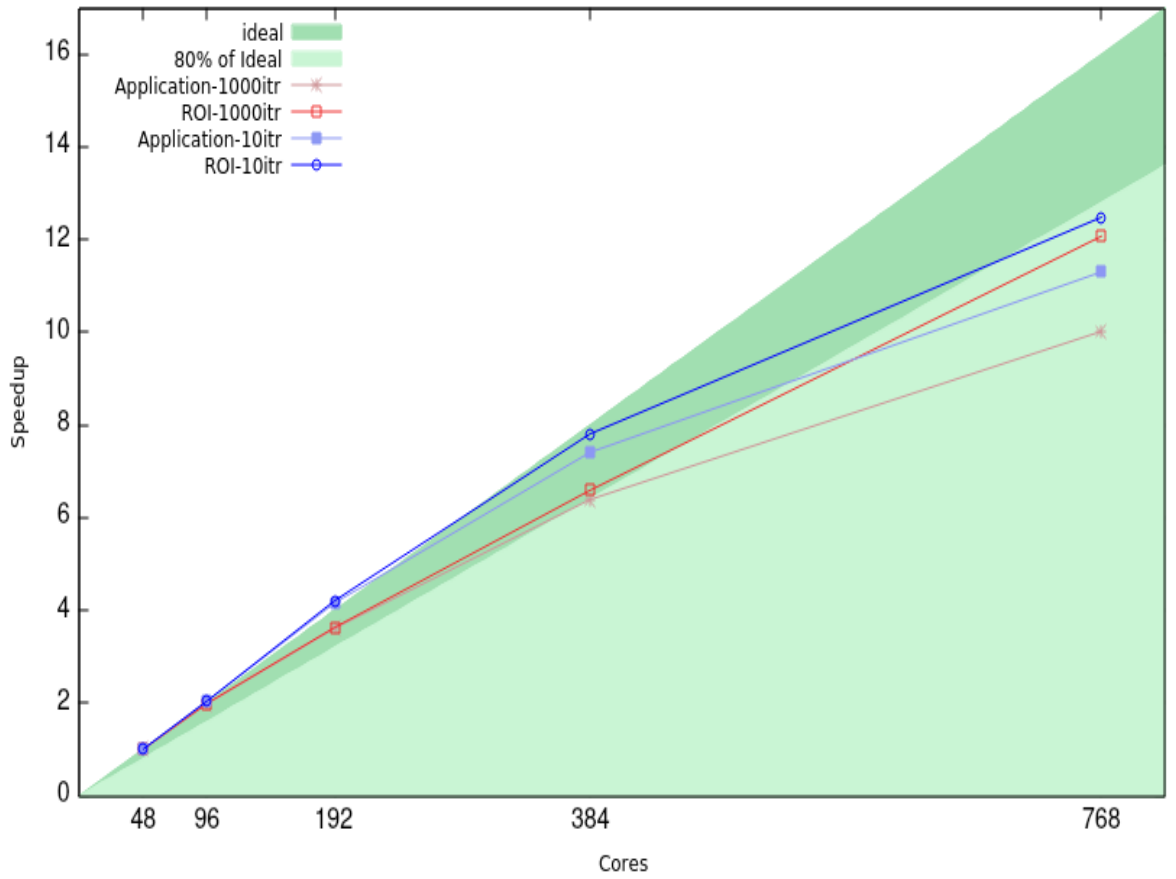
- Extrae (tracing), Paraver (visualisation) & Dimemas
  - https://tools.bsc.es
- Score-P (profiling and tracing), Scalasca (Post Processing) & Cube (visualisation)
  - https://score-p.org, https://scalasca.org
- MAQAO: synthetic reports and hints with a focus on core performance
  - http://www.maqao.org
- PyPOP: automated generation of POP metrics from Extrae traces
  - https://github.com/numericalalgorithmsgroup/pypop

For more help on how to use these tools and calculate the POP metrics

- https://pop-coe.eu/further-information/learning-material
- https://pop-coe.eu/further-information/online-training

Example 1: A Computational Fluid Dynamics Code

- Code: **C++, MPI**
- Platform: MareNostrum-IV(@BSC)
  - Dual Intel Xeon Platinum 8160 Skylake 48-core nodes
- Performance data collected using ***Score-P/Scalasca***
  - Using compiler instrumentation filter and hardware counters
- Scale:
  - ***48-768 cores (1-16 nodes)***



Example 1- POP Metrics

Number of cores	48	96	192	384	768
Global Efficiency	0.93	0.94	0.93	0.84	0.76
↳ Parallel Efficiency	0.93	0.91	0.87	0.77	0.68
↳ Load balance	0.99	0.98	0.98	0.97	0.95
↳ Communication Efficiency	0.94	0.92	0.89	0.79	0.72
↳ Serialisation	0.95	0.94	0.92	0.85	0.81
↳ Transfer efficiency	0.99	0.99	0.97	0.94	0.89
↳ Computational Scaling	1.00	1.03	1.07	1.09	1.12
↳ Instruction Scaling	1.00	0.99	0.97	0.95	0.92
↳ IPC Scaling	1.00	1.05	1.10	1.18	1.27
↳ Frequency Scaling	1.00	1.00	1.00	0.98	0.96

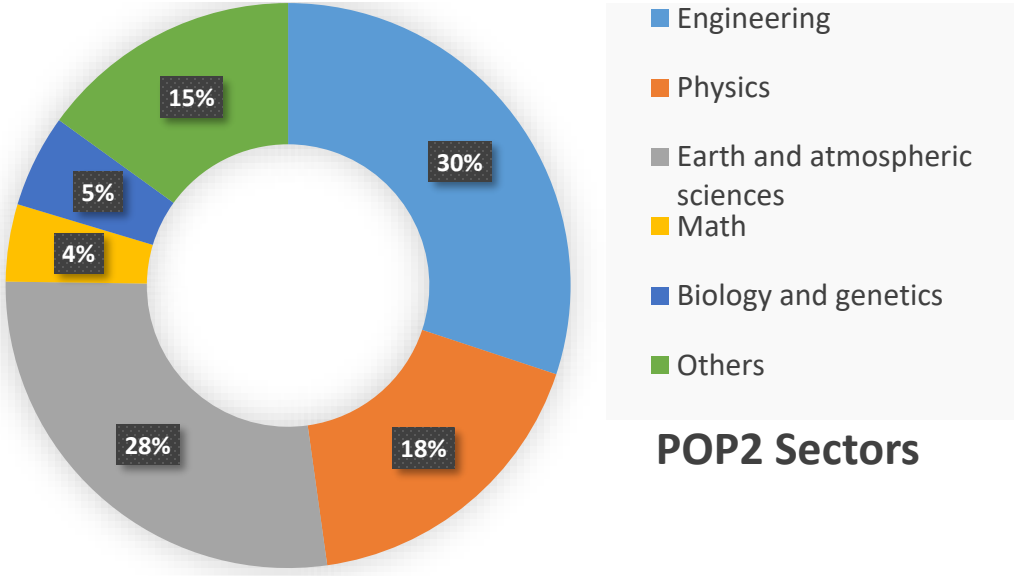
- We immediately see that **Serialisation** is the main factor that limits the scalability
- Efficiency values are between 0 to 1, and
  - metric values above 0.8 represent acceptable performance

Some Success Stories

- More than 350 services since 2015 across all domains
  - e.g. engineering, earth & atmospheric sciences, physics, biology and genetics
- See ⇨ https://pop-coe.eu/blog/tags/success-stories

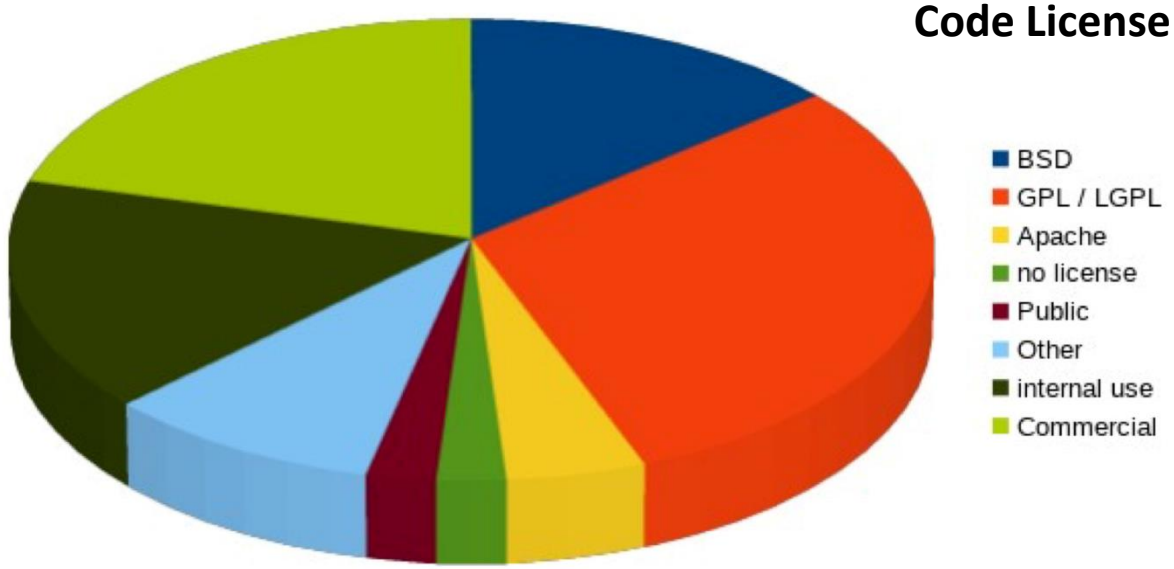
- Performance Improvements for SCM's ADF Modeling Suite
- **3x Speed Improvement** for zCFD Computational Fluid Dynamics Solver
- **25% Faster time-to-solution** for Urban Microclimate Simulations
- **2x performance improvement** for SCM ADF code
- Proof of Concept for BPMF leads to around **40% runtime reduction**
- POP audit helps developers **double their code performance**
- **10-fold scalability improvement** from POP services
- POP performance study improves performance **up to a factor 6**
- POP Proof-of-Concept study leads to **nearly 50% higher performance**
- POP Proof-of-Concept study leads to **10X performance improvement** for customer

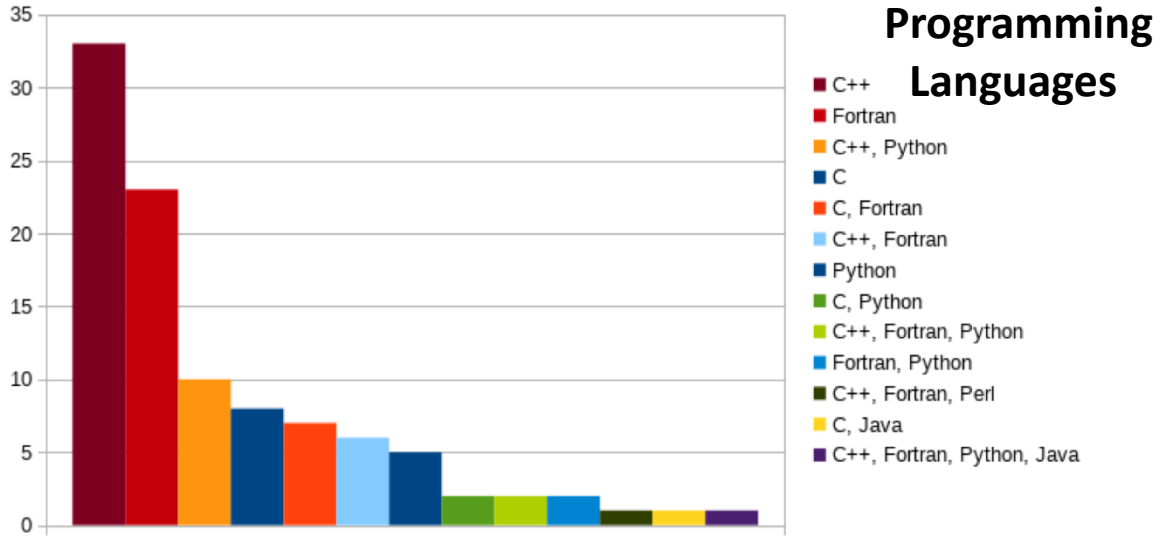
POP2 Services & HPC Codes



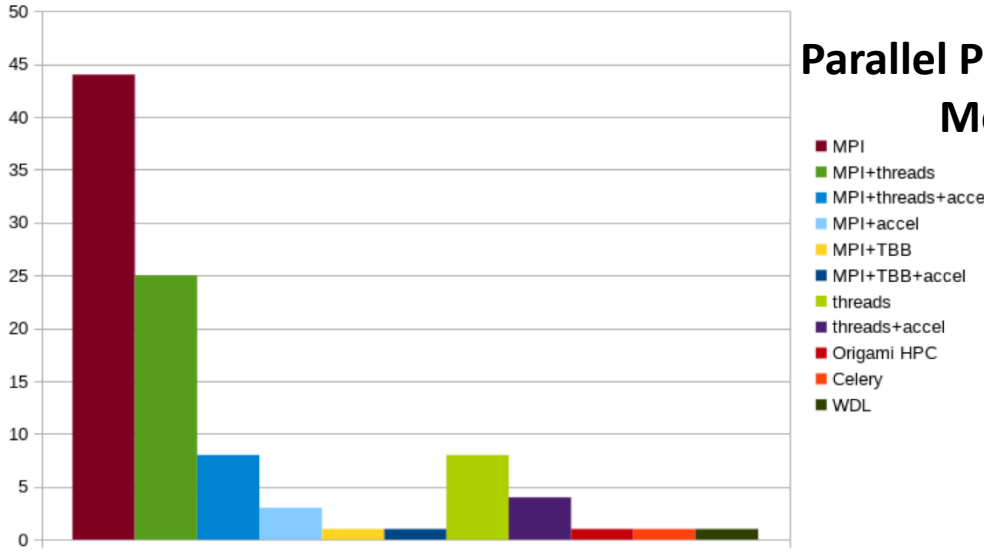
POP2 Sectors

Code License





Programming Languages



Parallel Programming Models

Online Content

POP Website

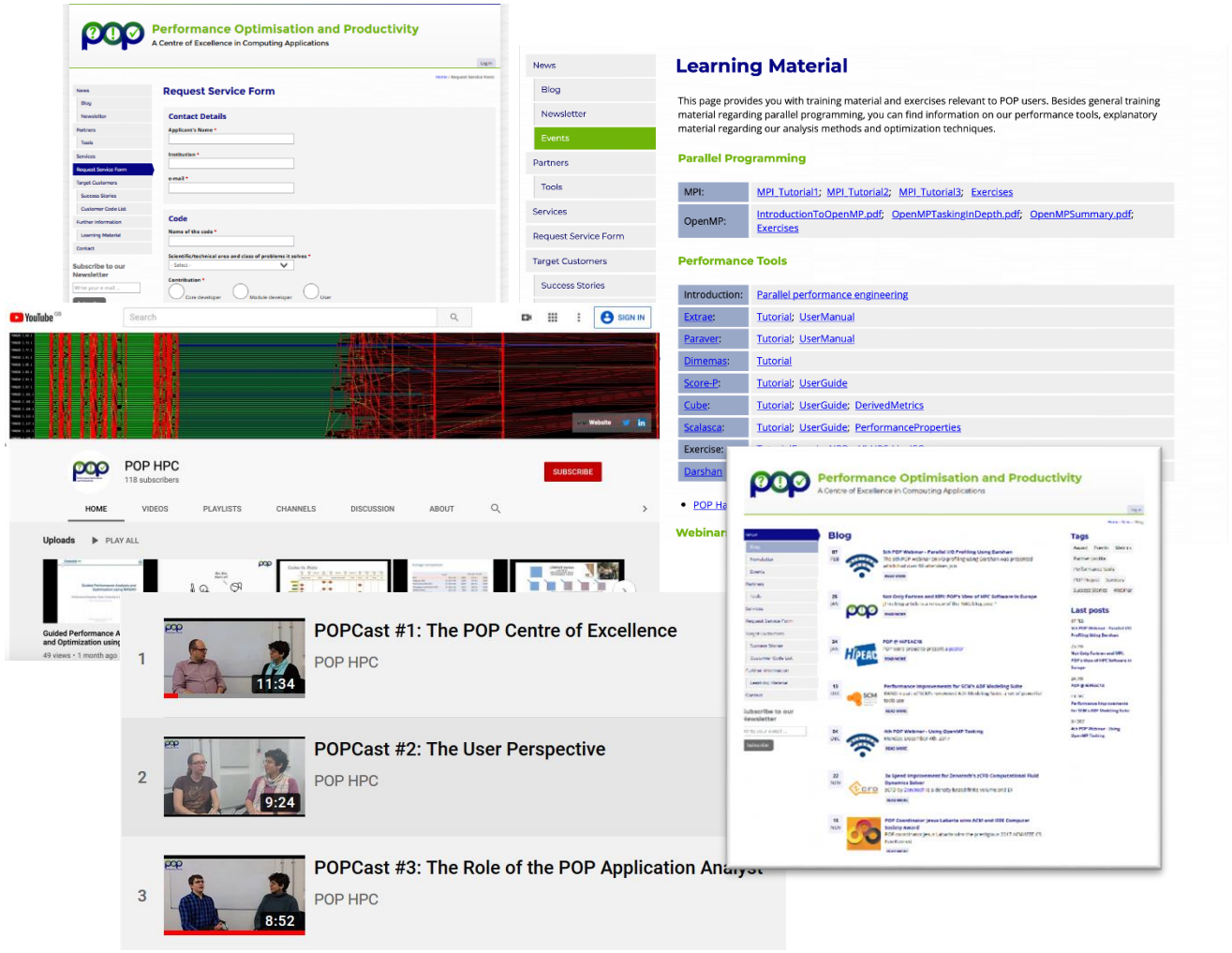
www.pop-coe.eu

- All you need to access POP services
  - **https://pop-coe.eu/services**
- Blogs
- Newsletter
  - subscribe and see past issues

YouTube Channel

https://www.youtube.com/pophpc

- Past Webinars
- POPCasts



Summary

POP Performance Metrics

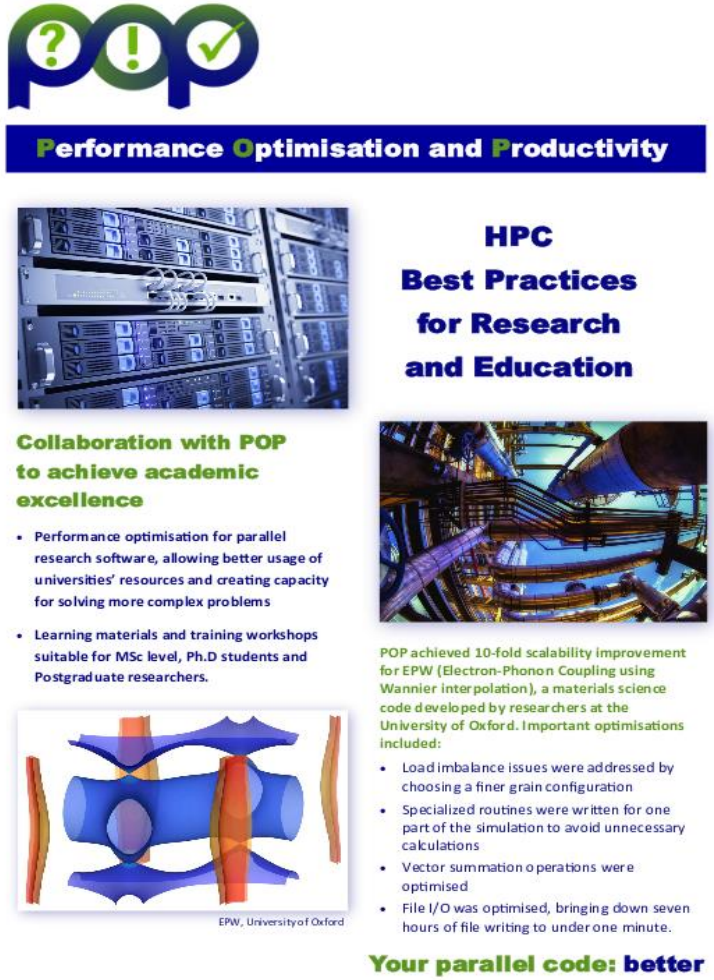
- Build a quantitative picture of application behavior
- Allow quick diagnosis of performance problems in parallel codes
- Identify strategic directions for code refactoring
- So far metrics for **MPI**, **OpenMP** and **Hybrid** (OpenMP + MPI) codes

POP works

- Across application domains, platforms, scales
- With (EU) academic and industrial customers including code developers, code users, HPC service providers and vendors
  - To apply for a POP service go to https://pop-coe.eu/services

POP CoE

- Promotes **best practices in parallel programming**
- Encourages a systematic approach to performance optimization
- Facilitates and invests in training HPC experts



Performance Optimisation and Productivity

A Centre of Excellence in HPC

Contact:

- https://www.pop-coe.eu
- pop@bsc.es
- @POP_HPC
- youtube.com/POPHPC

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 676553 and 824080.