# Performance Analysis and Optimisation Services at your Fingertips!

or

Why you should use our Services for your HPC code!

## Contact

|  |  |
|---|---|
| Twitter: | @POP_HPC |
| LinkedIn: | POP group |
| You Tube: | POP HPC |
| Web: | https://www.pop-coe.eu |
| Email: | pop@bsc.es |

For POP webinars and newsletter subscription see website

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

Performance Optimisation and Productivity

**Imprint**

| | |
|---|---|
| Editor: | Bernd Mohr<br>(Forschungszentrum Jülich GmbH) |
| Graphic Design: | Nadine Daivandy<br>(Forschungszentrum Jülich GmbH) |
| Photos: | POP project (page 2)<br>Graphics provided by customers, used with permission |
| Copyright: | POP Project<br>March 2018 |

Performance Optimisation and Productivity

A Centre of Excellence in Computing Applications

The POP Centre of Excellence is funded from October 2015 to March 2018.

# ROI Examples

## Application Savings after POP Proof-of-Concept

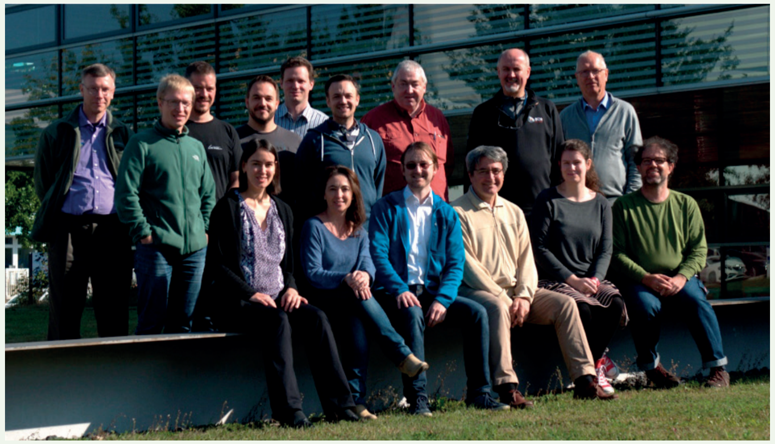
- POP PoC resulted in 72 % faster-time-to-solution
- Production runs on ARCHER (UK national academic supercomputer)
- Improved code saves €15.58 per run
- Yearly savings of around €56,000 (from monthly usage data)

## Application Savings after POP Performance Plan

- Cost for customer implementing POP recommendations: €2,000
- Achieved improvement of 62 %
- €20,000 yearly operating cost
- Resulted in yearly saving of €12,400 in compute costs ⇨ ROI of 620 %

# The POP Team

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



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

HLRS

nag®

JÜLICH Forschungszentrum | JÜLICH SUPERCOMPUTING CENTRE

RWTH AACHEN UNIVERSITY

Teratec

# Services Provided

The **Performance Optimisation and Productivity (POP) Centre of Excellence in Computing Applications** provides performance optimisation and productivity services for **academic AND industrial code in all domains!**

The services are **free of charge** to research organisations, SMEs, ISVs and companies in Europe!

**What are the main performance issues of your code?**

? **Performance Audit Service**

- Primary service
- Small effort (typically 1 month)
- Customer receives report

**Determine root causes of issues found and quantify approaches to address them!**

! **Parallel Application Performance Plan**

- Follow-up on the audit service
- Longer effort (typically 1-3 months)
- Customer receives report

**Perform experiments and code changes to show effect of proposed optimizations**

✓ **Proof-of-Concept**

- Follow-up on the performance plan service
- 6 months effort
- Customer receives software demonstrator

# Customer Satisfaction

**Performance Audits**
(73 customers)

- Over 90 % very satisfied or satisfied with service
- About half of the customers signed-up for a follow-up service

**Performance Plans**
(11 customers)

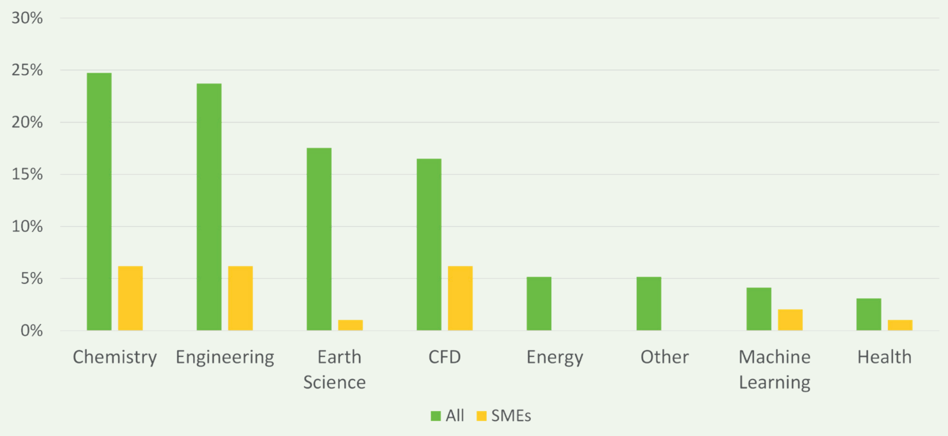
- About 90 % very satisfied or satisfied with service
- All customers thought suggestions were precise and clear and 70 % plan to implement the suggested code modifications
- About 2/3 plan to do use the POP services again

**Proof-of-Concepts**
(8 customers)

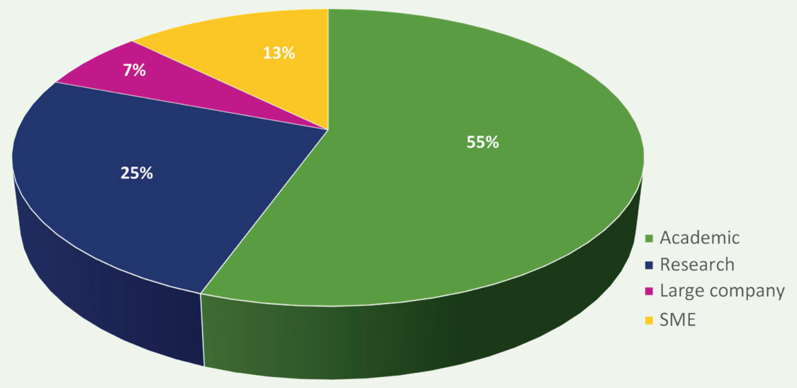
- All customers very satisfied or satisfied with this service
- Over 80 % plan to implement further code modifi-cations or complete the work of the POP experts

# Performance Audit and Plan Statistics

## Application Sectors



## Customer Types



* *Based on data collected for 161 POP Performance Audits*

# Customer Quotes

*“The audit of the VAMPIRE code has been **extremely helpful** in identifying the hot spots and specific areas to focus on performance improvements. Preliminary results suggest this may give a **factor of 2 performance improvement** on modern CPUs. I would **highly recommend** the service for the speed and usefulness of the audit.”*

– Richard Evans, VAMPIRE developer

*“High performance computing is an extremely interesting topic to our application. The POP project has helped Artelnics to **speed-up Neural Designer up to 5 times**, when compared to the serial version. And we can still improve a lot more by implementing MPI processing in computer clusters.”*

– Dr Roberto Lopez, CEO Artelnics

*“POP analysis elegantly reveals in detail how our application’s algorithm is running on HPC architectures. It is an **extremely useful** optimisation tool! Our POP contact was very knowledgeable and enthusiastic. **An excellent service!**”*

– Joseph Parker, GS2 Developer

# GraGLeS2D Success Story

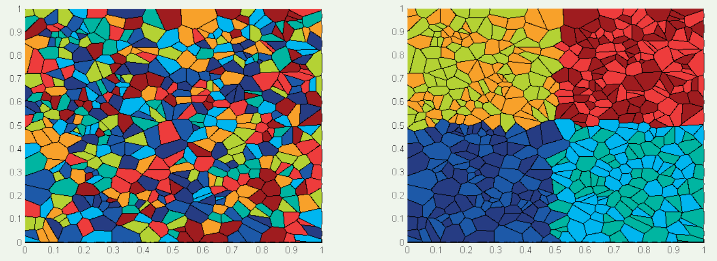
## POP Proof-of-Concept study leads to 10X performance improvement for customer

The Institute of Physical Metallurgy and Metal Physics of RWTH Aachen University (IMM) develops a code for the simulation of microstructure evolution in polycrystalline materials, called GraGLeS2D. The OpenMP parallel code is designed to run on large SMP machines in the RWTH compute cluster with 16-sockets and up to 2 TB of memory. After a POP performance audit of the code, several performance issues were detected and a performance plan on how these issues could be resolved was set up.

To verify the proposed optimization steps, POP experts and the code developer at IMM implemented these steps in close collaboration as the first proof-of-concept study done in POP. The optimizations include:

- The use of a memory allocation library optimized for multi-threading.
- Reordering the work distribution to threads to optimize for data locality between neighboring cells. (see Figure below)
- Algorithmic optimizations in the convolution algorithm.
- Code restructuring to enable vectorization in parts of the computation.

Initial work/data distribution over sockets (left) compared to the optimized distribution (right).

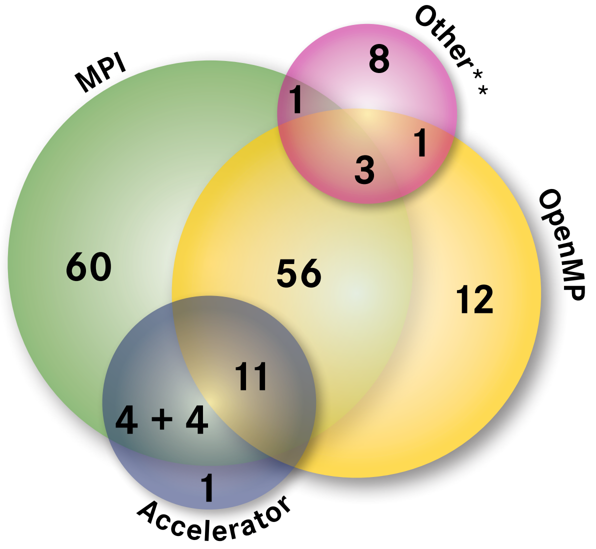


*Initial work/data distribution over sockets (left) compared to the optimized distribution (right).*

After these optimization steps were implemented, a significant performance improvement was achieved. For the hotspot of the application, the convolution region, the **speedup going from 1 to 16 sockets is about 15 instead of 6** as it was before the optimization. Overall, the runtime of this region was **improved by a factor of more than 10X**. The proof-of-concept verified that the planned optimizations indeed resulted in significantly better code performance.

# Performance Audit and Plan Statistics

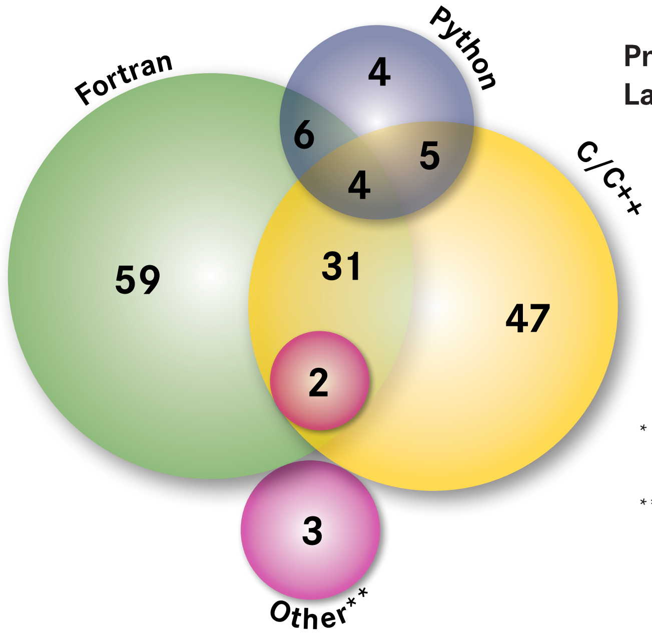
## Programming Models used



* Based on data collected for 161 POP Performance Audits

** MAGMA
Celery
TBB
GASPI
C++ threads
MATLAB PT
StarPU
GlobalArrays
Charm++
Fortran Coarray

## Programming Languages used



* Based on data collected for 161 POP Performance Audits

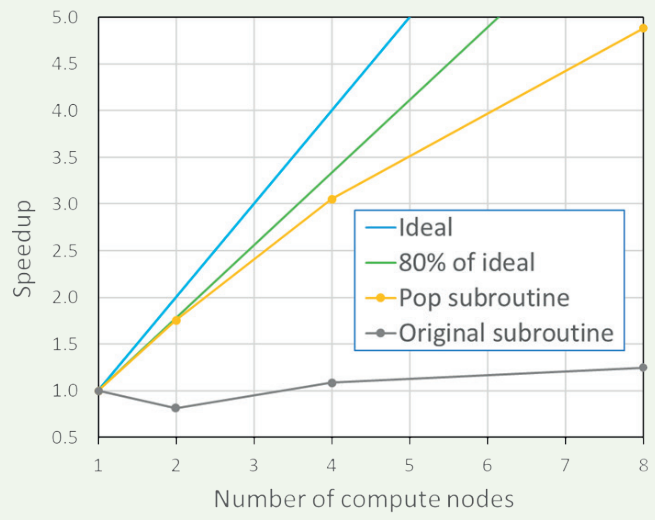
** TCL
Matlab
Perl
Octave
Java

# BAND Success Story

## Big Performance Improvements for SCM's ADF Modeling Suite

BAND is part of SCM's renowned ADF Modeling Suite, a set of powerful tools used by academic and industrial research chemists, and written in Fortran with MPI parallelisation. ADF wanted to know if POP could help improve parallel performance.

After a POP Audit and two Performance Plans, which analysed various components of BAND, a POP Proof of Concept focussed on improving performance of complex matrix multiplications. The earlier work had determined that for multiplication of small matrices the parallel scaling was limited by underperformance of BLAS/PBLAS routines coupled with a large percentage of time within MPI data transfer.



The Proof of Concept identified and implemented a range of improvements, which included overlapping computation with communication, improved use of BLAS which **doubled the speed of computation**, and reorganising the algorithm to reduce the amount of data communicated via MPI. **The optimised subroutine showed four times speed up**, compared to the original code, on eight 36-core compute nodes.

# Ateles Success Story

## Runtime for fluid dynamics code reduced by nearly 50 %

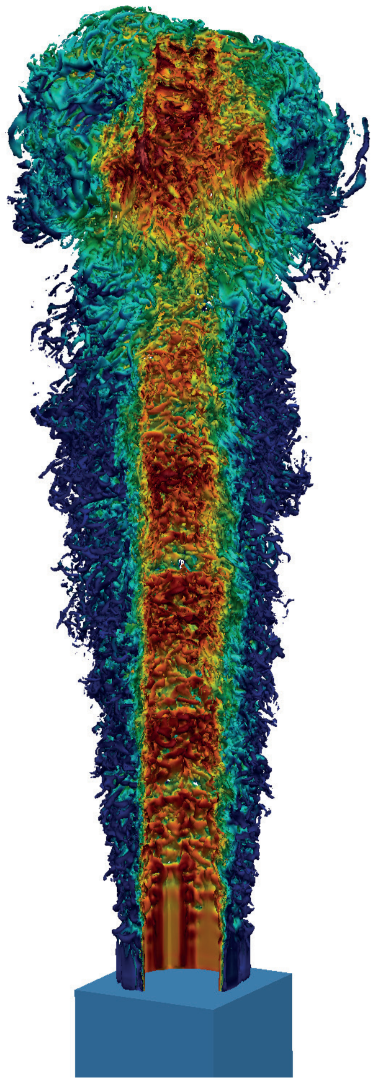
The Institute for Simulation Techniques and Scientific Computing of the University of Siegen develops a fluid dynamic code called Ateles. The code is written in Fortran and had already shown good performance on HPC systems in the past. The code was now extended by new features which were analysed within a POP performance audit. Several issues related to the serial code performance were identified and a performance plan on how these issues could be resolved was set up.

In a proof-of-concept study the POP experts verified, that the proposed code optimizations lead to a significant performance improvement.

The optimizations included:

- Inlining of very short functionswith high call rates
- Parameter and variable redefinitions that allow the reduction of expensive CPU operations like divisions

With these optimizations applied to the real code, we measured for the provided test case a **performance increase of nearly 50 %** and the customer was able to confirm a substantial performance improvement for his production runs.



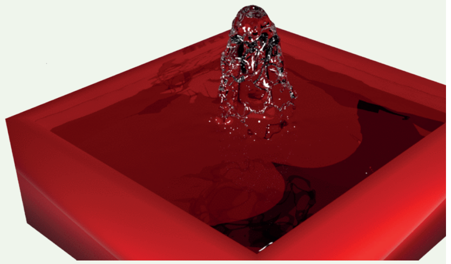
# sphFluids Success Story

## Insights into computer graphics code for fluids led to a factor of 6 improvement

The computer animation department of the Stuttgart Media University, in cooperation with the Visualisation Research Centre of the University of Stuttgart, develops a Smoothed Particle Hydrodynamics solver to simulate fluids for computer graphics applications called sphFluids. The Code is written in C++ and was mainly developed as a cross-platform desktop application, which is parallelized with OpenMP.

The sphFluids code supports the most common pressure as well as viscosity models. Additionally, various approaches to model surface tension are integrated. More details can be found in the paper „Evaluation of Surface Tension Models for SPH-Based Fluid Animations Using a Benchmark Test“.

The sphFluids code underwent a POP performance audit, which identified several issues related to the sequential computational performance.
The good information exchange with the POP experts during the study helped the code developers to identify critical parts in their application.



One of the issues found was code regions with low instruction per cycle (IPC) values. Several causes for this were pointed out including:

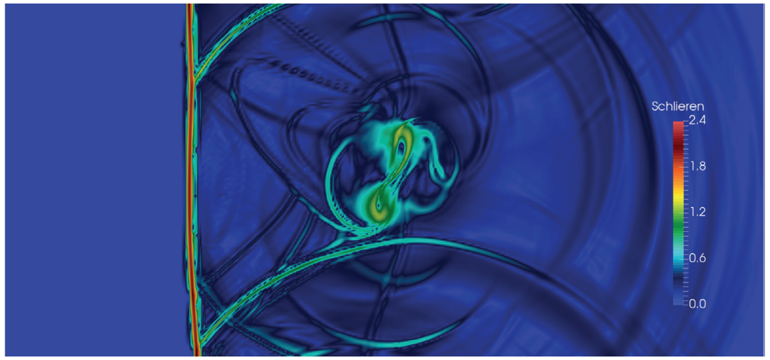
- Definitions of variables in inner loops
- Unnecessary operations caused byindirections in the code design
- Non-inlined functions
- Cache misses, due to memory calls

Based on the audit, the code developers could optimize the identified parts in the code by, e.g., inlining very short functions that were used frequently or, regarding the cache misses, reorder the particle processing order. These modifications improved the performance of the code by about 100 %. Furthermore, they identified similar issues in other parts of the code and reviewed the overall code design. The developers came to the decision to completely rewrite the simulation code. Using the insights gained from the POP-Experts, they could optimize the simulation performance further, which led to an **overall performance improvement up to 500 % - 600 %**, depending on the scenario and pressure model used.

# zCFD Success Story

## 3x Speed Improvement for Zenotech’s zCFD Computational Fluid Dynamics Solver

zCFD by Zenotech is a density based finite volume and Discontinuous Galerkin (DG) computational fluid dynamics (CFD) solver for steady-state or time-dependent flow simulation. It decomposes domains using unstructured meshes. It is written in Python and C++ and parallelised with OpenMP and MPI.



POP conducted a Performance Audit to identify potential areas for improvement. This identified that the code was spending a surprisingly large amount of time executing in serial and that one particular OpenMP loop was suffering from load imbalance. POP also noted that the CPU frequency was being lowered when the code was run on the maximum number of threads (12 for the machine used in the Audit).

As a result, Zenotech made a number of changes to the code:

- Parallelising serial portions of code.
- Improving load balance.
- Removing OpenMP regions that were being created on multiple threads.
- Memory management modifications.
- Changing execution environment settings to boost CPU performance.

For the test case used in the study, these improvements meant the code ran 1.65x faster on 12 threads. When Zenotech applied the modified code to a test case that was 100x larger, they **observed a 3x performance improvement over the old code** on 12 threads. The average cycle time fell from 3,253ms to 1,185ms, which corresponds to going from 10.4 GFlop/s to 30.6 GFlop/s for a single Broadwell socket.

# ADF Success Story

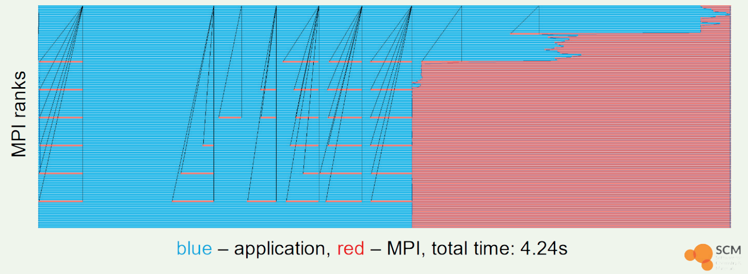
## Load balance of flagship computational chemistry application improved leading to factor of two runtime reduction

ADF is the flagship code from Software for Chemistry and Materials (SCM) company based in The Netherlands. It is a computational chemistry application which uses density functional theory calculations to predict the structure and reactivity of molecules.

A POP Audit and Performance Plan were carried out on their new Hartree-Fock exchange implementation which is an important new feature of the application. The application uses MPI and shared memory within a node to parallelise the problem.

The main issue located was the load imbalance due to unequal distribution of work, there was also low computational scalability but that was found to be an artefact of the time cores spent idle waiting to be distributed work. The communication efficiency was found to be good and did not need further investigation.

A recommendation was made to improve the load balancing algorithm with an expected performance improvement of a factor of two for good balance.



*Original timeline of load balance for Hartree-Fock exchange, communication lines in black*

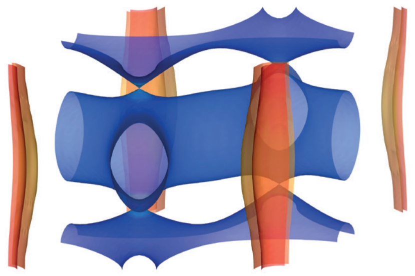
On 128 cores the section of imbalanced work took 4.24s for 45 atoms. Dynamic load balancing was implemented by the SCM developers with a dedicated dispatcher process to farm out the work to all other cores. This reduced the runtime to 1.992s which is a **performance improvement of over 2 times**, as was estimated in the POP Performance Plan.

# EPW Success Story

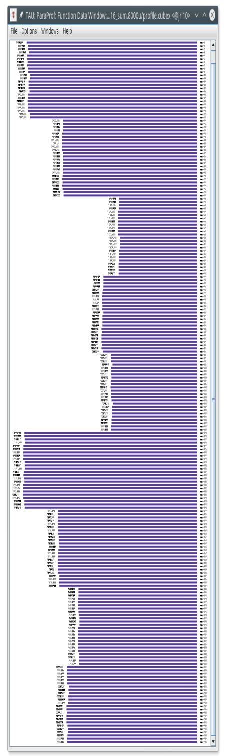
## University of Oxford code scalability improved 10-fold

EPW (Electron-Phonon using Wannier interpolation) is a materials science DFT code distributed in the Quantum ESPRESSO suite. It is Fortran code parallelised with MPI. Developers from the University of Oxford requested a POP performance audit of an unreleased version of the code that was still in development, to be tested with a GaN polar wurtzite crystal dataset on the ARCHER Cray XC30 computer at EPCC.

The initial audit of 48 processes identified a variety of load imbalance issues, and excessive time in the ephwann simulation phase. This became the focus of a subsequent POP performance plan, where the developers specialized routines to avoid unnecessary calculation and optimize vector summations.



Using a finer uniform grid reduced load imbalance and this revised version was 60 % faster and could be used for larger execution configurations with 240 MPI processes.



Unfortunately, overall performance was disappointing, with writing the final simulation results having grown to dominate execution time. The figure shows a histogram of the writing time varying by MPI process on nine compute nodes. Although the amount of data is not large (around 50MB of formatted text), it was a bottleneck inhibiting scaling and larger simulations. A POP proof-of-concept investigation was pursued which replaced file writing concurrently by all processes with serial writing only by rank zero. **This reduced writing time from over seven hours to under one minute**, and now a negligible component of EPW execution.

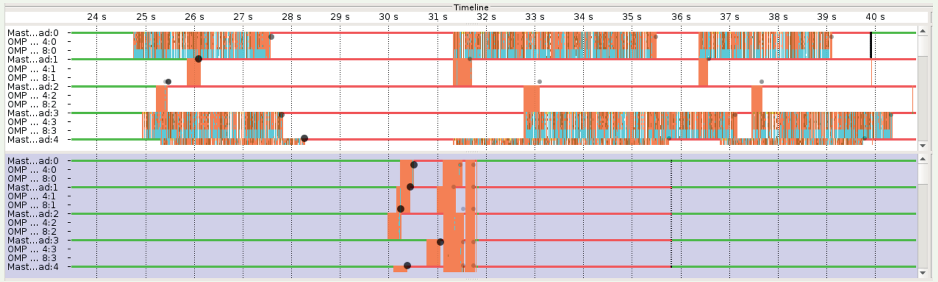
**The final code scales well with 85 % parallel efficiency for 960 MPI processes**, supporting larger simulations. These POP reports helped support EPW readiness to productively utilize additional larger allocations of computational resources.

# k-Wave Success Story

## Open-source acoustic simulation code runtime halved

k-Wave is an open-source toolbox for time domain acoustic and ultrasound simulations in complex and tissue-realistic media. Simulation functions are based on the k-space pseudospectral method.

POP was requested by developers from Brno University of Technology to audit the C++ version parallelised with MPI+OpenMP executing on the Salomon supercomputer hosted by IT4Innovations in the Czech Republic. A configuration of 32 dual-processor Intel Xeon compute nodes was used running 64 MPI processes each with 12 OpenMP threads. The 3D domain decomposition employed (4x4x4 process arrangement) was discovered to suffer from poor performance with large amounts of both MPI and OpenMP synchronization time arising from major load imbalance.



The figure shows an extract of the time-line visualization, showing the three FFTW phases for one timestep of the first four MPI processes. Originally (top with white background), the interior processes (ranks 1&2) wait in MPI communication (red) for the much slower exterior processes (ranks 0&3) where many more small and poorly-balanced parallel loops have lots of OpenMP synchronization time (cyan). Although the exterior MPI processes have fewer grid cells, the OpenMP-parallelized FFTs from the FFTW library are much less efficient as they have a larger FFT base.

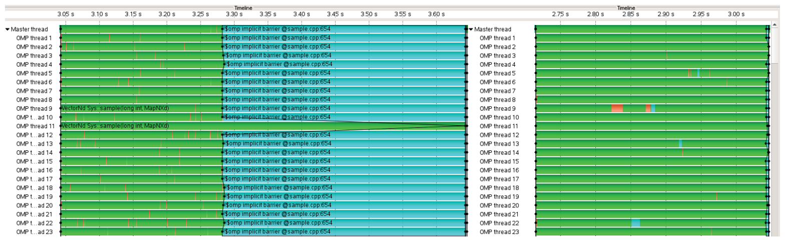
With this insight, the developers were quickly able to apply a periodic domain with identical halo zones for each MPI rank (lower time-line with lilac background), with the result that the **execution is now more than twice as fast**. Both versions of the code are compared in the POP performance audit.

# BPMF Success Story

## Data analysis code used to predict movie ratings improved by around 40 %

Modelling complex data sets is a major problem today. An example here is prediction of compound-on-target-activity in chemogenomics from the ChEMBL data set with more than 2 Million compound records. The compound-on-target-activity study at large scale is an extremely important question in the process of discovering new drugs, which is currently addressed in the ExCAPE project. The Bayesian Probabilisitc Matrix Factorization (BPMF) is an efficient method to solve these kind of problems. The BPMF code was analysed in a POP Audit and Performance Plan service activity. While the BPMF code had already shown scalability over several 100 nodes and also good efficiency on the node level, POP experts could still identify points for improvement.

So a follow-up Proof of Concept study was performed together with the customer. During the study several points were addressed. Besides optimization of the linear algebra computations and improvement of the selection process for optimized algorithms inside BMBF, the main and most challenging issue was load balance. Due to the nature of the problems solved with BPMF, the datasets include very inhomogeneous data, which result in load balance problems in the parallelization. BPMF therefore comes with a hybrid MPI+OpenMP parallelization. The still existing load balance problem found was at the lower OpenMP level. Here a single level OpenMP parallelization was used on the node level.



The POP experts now implemented a second nesting level and also made use of OpenMP tasks, which solved the load balance problem. Originally the load balance of the problematic code part was 42.5 % – after the modifications 98.9 %. Finally, the improvements made in this POP Proof of Concept were evaluated with three different datasets **achieving speedups between 1.6 and 1.8 that correspond to runtime reductions between 38 and 44 %.**