Performance Optimisation & Productivity (POP) CoE
JSC highlights

Michael Knobloch, Marc Schlütter, Brian Wylie & Ilya Zhukov

Introduction

The *Performance Optimisation and Productivity Centre of Excellence in HPC* (POP) is funded by the European Commission to provide **free of charge** performance assessments to uncover inefficiencies and their causes in HPC parallel applications.

From its start in October 2015, POP has analysed over 200 application codes, including several on HPC computer systems at JSC using a variety of languages and parallelization paradigms for CPUs and GPUs. As well as highlights on this poster, assessments have included JURASSIC, NEST, ParFlow, pySDC & TerrSysMP.

Methods & Tools

Open-source performance measurement and analysis tools based on runtime summarisation and event tracing developed by members of the POP consortium are used along with other available tools: in particular

scalasca using CUBE and Score-P with Vampir.

POP uses a defined methodology to fully understand issues affecting performance of parallel applications by calculating the following metrics:

- **Global Scaling Efficiency** - Overall performance
  - **Parallel Efficiency** - Efficiency of parallelisation strategy
    - **Load Balance Efficiency** - Effectiveness of work distribution
    - **Communication Efficiency**
      - **Serialisation Efficiency** - Dependencies between processes
      - **Transfer Efficiency** - Effect of data transfer
  - **Computation Scaling** - Scaling of computational load
    - **Instructions Scaling** - Increase in computational work
    - **IPC Scaling** - How computational resources are used

Highlight: Nekbone

Proxy mini-app for Nek5000 incompressible CFD code conjugate gradient solver in Fortran+C using MPI and OpenMP on JUQUEEN Blue Gene/Q. Excellent weak scaling, however, synchronization of OpenMP threads and MPI neighbour ranks impact performance at all scales.

Racks	$\frac{1}{2}$	1	2	4	8	16	28
Processor distribution	8x8x8	16x8x8	16x16x8	16x16x16	32x16x16	32x32x16	32x32x28
Processes	512	1024	2048	4096	8192	16384	28672
Threads	32768	65536	131072	262144	524288	1048576	1835008
Global efficiency	0.67	0.67	0.67	0.67	0.67	0.67	0.65
- Parallel efficiency	0.67	0.67	0.67	0.67	0.67	0.67	0.65
- - Load balance efficiency	0.96	0.96	0.96	0.96	0.96	0.96	0.95
- - Communication efficiency	0.70	0.70	0.70	0.70	0.70	0.70	0.69
- - - Serialization efficiency	0.70	0.70	0.70	0.70	0.70	0.70	0.69
- - - Transfer efficiency	1.00	1.00	1.00	1.00	1.00	1.00	1.00
- Computation efficiency	*1.00*	1.00	1.00	1.00	1.00	1.00	0.97
- - Instructions scaling	*1.00*	1.00	1.00	1.00	1.00	1.00	1.00
- - IPC scaling	*1.00*	1.00	1.00	1.00	1.00	1.00	0.98

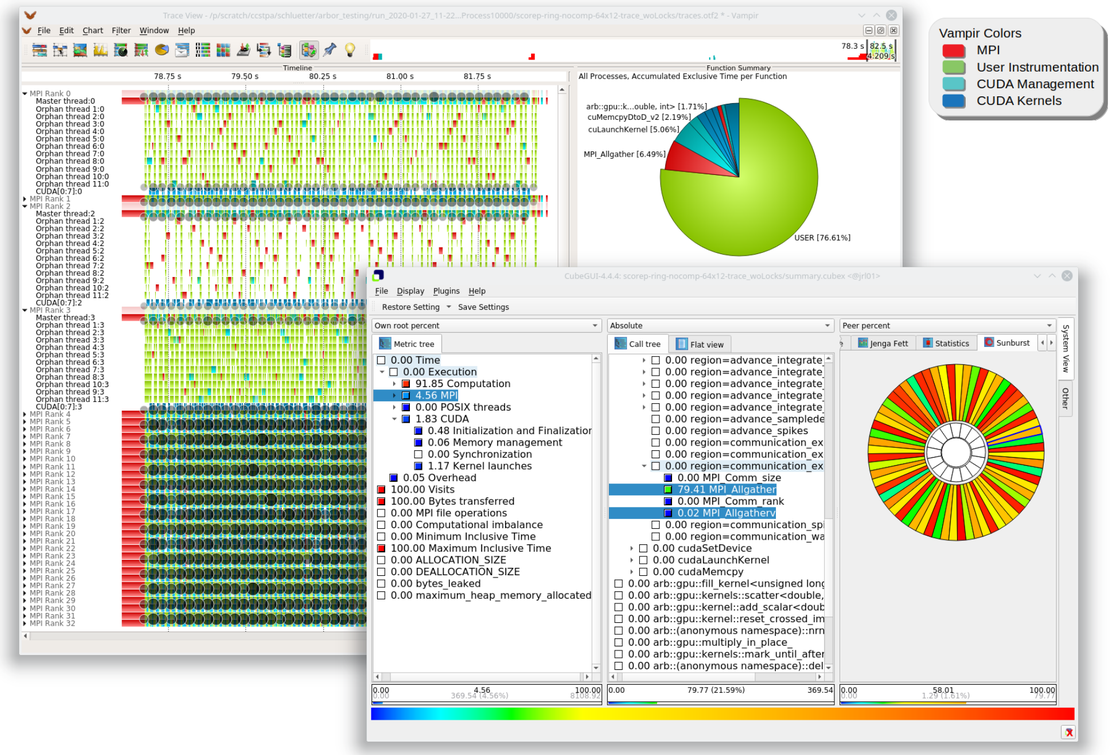
HPC Tunathon (25-29 May 2020)

Five-day workshop at JSC dedicated to assisting qualified teams of application developers with scaling and optimising their codes on CPUs and GPUs of JUWELS in preparation for future exascale systems. Prior POP assessment of efficiency and scalability highly recommended.

http://www.fz-juelich.de/ias/jsc/2020/HPC-tuna

Highlight: Arbor

Neural network simulator developed by JSC SimLab Neuroscience and CSCS in C++ using std::thread (orphan Pthreads) with MPI and CUDA. Initial assessment on JUWELS compute nodes, follow-up in progress with quad-GPU nodes.



Vampir event trace visualisation & CUBE profile of Arbor on JUWELS GPU nodes

Serial post-processing step after MPI collective communication shows insufficient overlap with computation on other threads.

Highlight: SOMA

Soft matter simulator developed by Universität Göttingen in C (and CUDA) using MPI and OpenMP. Initial assessment on JUWELS compute nodes.

Vampir event trace visualisation of SOMA on 64 nodes of JUWELS

Parallel efficiency only 0.34 due to large chunks of OpenMP parallel computation with significant internal load imbalance (0.46), followed by MPI global synchronization (0.85) for serialized writing of intermediate analytics results.

https://pop-coe.eu

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