High-Level Support Projects

Current Projects

ECHO-3DHPC: Advancing the performance of astrophysics simulations

AstroLab contact: Luigi Iapichino
Application partner: Matteo Bugli (CEA Saclay, France)
Project partner: Fabio Baruffa (Intel)

In this project we improved the parallelization scheme of ECHO-3DHPC, an efficient astrophysical code used in the modelling of relativistic plasmas. With the help of the Intel Software Development Tools, like Fortran compiler and Profile-Guided Optimization (PGO), Intel MPI library, VTune Amplifier and Inspector we have investigated the performance issues and improved the application scalability and the time to solution. The node-level performance is improved by 2.3x and, thanks to the improved threading parallelisation, the hybrid MPI-OpenMP version of the code outperforms the MPI-only, thus lowering the MPI communication overhead.

![Graph showing speed-up at node level for baseline and optimized code versions.]

Parallel speed-up at node level (OpenMP-only) for the baseline and optimized code versions.

More details: see article on Intel Parallel Universe Magazine 34, p. 49. ArXiv version here.

Project Template

Finalized Projects

Optimizing the TARDIS parallel performance (2016/17)

AstroLab contacts: Vytautas Janauskas, Stephan Hachinger
Application Partner: Wolfgang Kerzendorf (ESO, Garching near Munich)

TARDIS is a numerical code for Monte-Carlo simulations of supernova spectrum formation. It serves as a tool to analyse the conditions generating the spectra of observed supernovae, i.e. to trace back the supernova structure from the observations.
With this ADVISOR 2016 project, our aim was to obtain an overview of possible performance bottlenecks (and of discovery strategies for bottlenecks in the future), and first steps towards an optimised TARDIS code. The code is largely parallelised with OpenMP, and job-farming techniques (e.g. with MPI) are used to perform ensemble runs on larger machines.

In the context of ADVISOR 2016 and LRZ Astro-Lab, a comprehensive standard profiling/scaling test of the TARDIS code was performed. The profiling brought out no obvious, easy-to-resolve bottlenecks or problems, but was very valuable for planning algorithmic and conceptual improvements (e.g. an improved convergence strategy) for TARDIS v2.0.

More details: see TARDIS on GitHub, TARDIS on readthedocs, TARDIS v1.0.1 via DOI.

Project Template