KTH
Jing Gong, Stefano Markidis
KTH Royal Institute of Technology
Sweden
https://www.pdc.kth.se

About KUNGLIGA TEKNISKA HÖGSKOLAN (KTH)

The KTH Royal Institute of Technology, established in 1827, is one of Europe's top schools for science and engineering, graduating one-third of Sweden's undergraduate and graduate engineers in the full range of engineering disciplines. The PDC Center for High Performance Computing at the KTH provides leading HPC services to Swedish academics.

EPCC
Luis Cebamanos, David Henty
University of Edinburgh
Edinburgh, Scotland
www.epcc.ed.ac.uk

About EPCC

The University of Edinburgh is one of the world's leading research universities. Through its supercomputer centre, EPCC, it is the project coordinator of the CRESTA project. EPCC manages a collection of HPC systems including ARCHER, the UK's national high-end computing system.

Collaborative Research into Exascale
Systemware, Tools & Applications
cresta-project.eu

About CRESTA

CRESTA (Collaborative Research into Exascale Systemware, Tools & Applications) is a collaborative research effort funded by the European Union exploring how to meet the exaflop challenge. The project has two integrated strands: one focused on enabling a key set of co-design applications for exascale, the other focused on building and exploring systemware for exascale platforms.

Cray Inc.
Harvey Richardson, Alistair Hart
www.cray.com

About Cray

As a global leader in supercomputing, Cray provides highly advanced systems and solutions and world-class service and support to government, industry and academia. Cray technology enables scientists and engineers to not only meet existing and future simulation and analytics challenges but achieve remarkable breakthroughs by accelerating performance, improving efficiency and extending the capabilities of their most demanding applications.

CRESTA

AUTO-TUNING OPENACC DIRECTIVES within the NEK5000 application to fully exploit GPU architectures

Applications need to be tuned to get the best performance from a High Performance Supercomputer. This can be a time-consuming process to do by hand, with potentially thousands of parameter combinations to explore. Auto-tuners can speed this process up significantly, but existing methods are not well suited to HPC applications.

To solve this problem, the CRESTA project designed a Domain Specific Language (DSL) for auto-tuning parallel applications. This was then used to optimise the performance of NekBone (a standalone benchmark for the Nek5000 parallel CFD code) on a Cray XC30 system accelerated with Nvidia K20X GPUs. The results were impressive, giving a 200% speed-up that consistently outperformed the best hand-tuning efforts of a domain specialist.

A GPU version of NekBone was developed using the new OpenACC accelerator directives that are implemented by the Cray compilers. OpenACC directives accept a wide variety of parameters controlling exactly how the parallelisation is performed, and their settings can have a significant effect on performance.

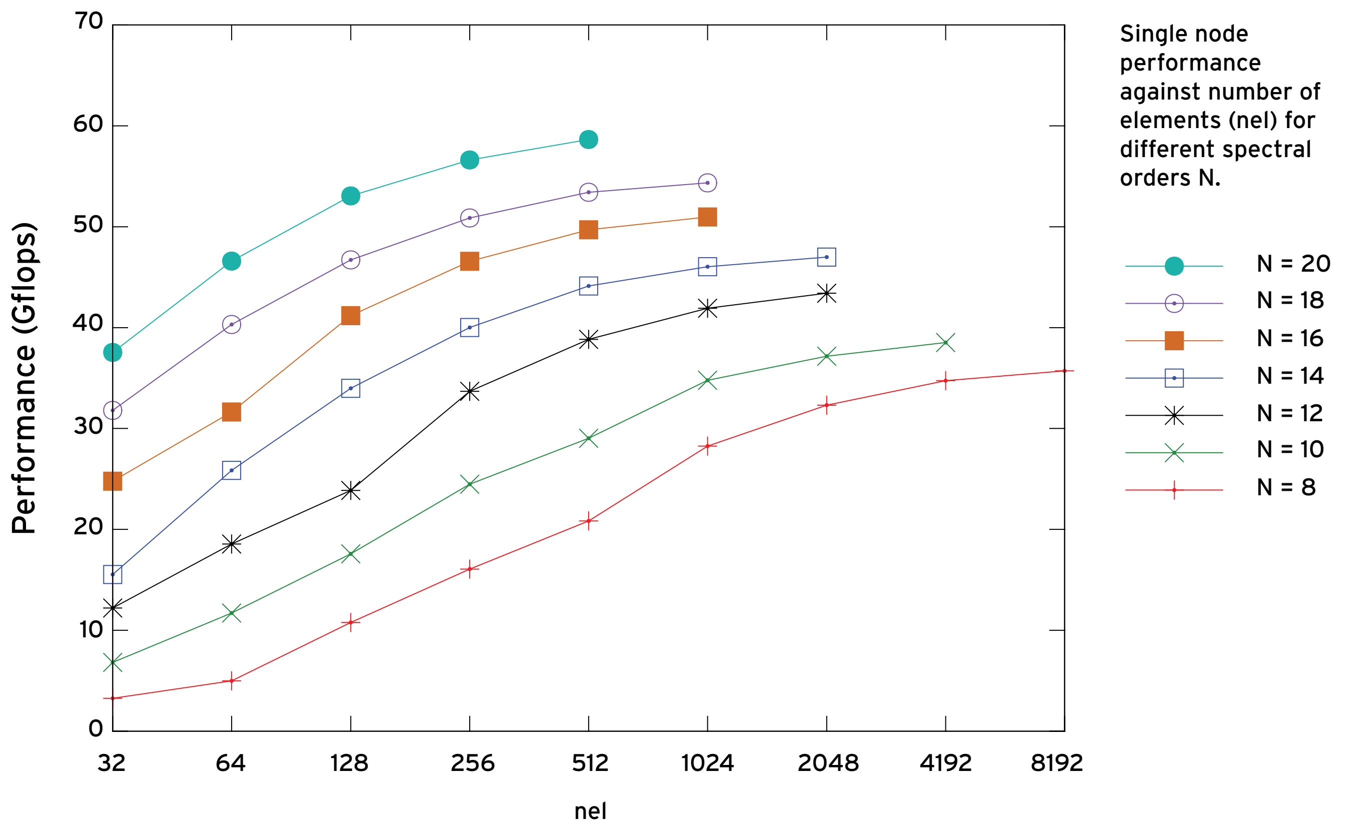
NekBone was restructured so that it called a small number of kernel functions, each of which could be implemented in OpenACC and individually auto-tuned using CRESTA's DSL. We worked closely with the Cray compiler team during this phase. For example, we identified cases where kernels performed very differently when run in isolation compared to being run in the main code. The compiler team suggested additional directives to solve the problem in the short term while they updated the compiler based on our test cases.

The NekBone results were very encouraging. Across a wide range of representative cases, the auto-tuning increased the performance of the code by around 200% compared to the default OpenACC settings. We also compared to an OpenACC version of NekBone that was designed and hand-tuned by an expert developer. For representative problem sizes, the auto-tuned version always performed within a few percent of the hand-tuned version, and outperformed it by over 15% for the largest systems.

This work has recently been extended to the full Nek5000 application. Although this is much more of a challenge for auto-tuning, now working with 100,000 lines of code compared to 42,000 for NekBone, the results are very positive. For example, we see performance improvements from auto-tuning of around 32% compared to the best hand-tuned implementation.

> The results were impressive, giving a 200% speed-up that consistently outperformed the best hand-tuning efforts of a domain specialist.

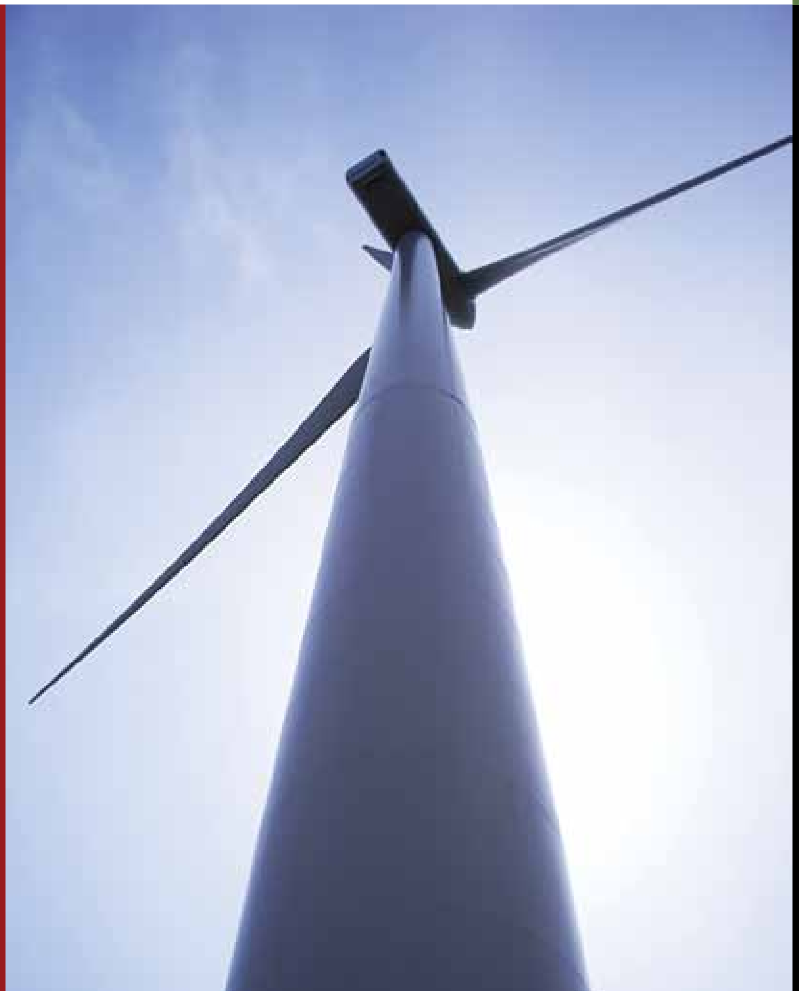
Graph of performance for various problem sizes



Single node performance against number of elements (nel) for different spectral orders N.

OpenACC co-design with GROMACS

The porting of Nek5000 to exploit the GPUs on ORNL's Cray XK7 Titan system demonstrates how the OpenACC programming model exploits system software (compilers and runtime libraries) to accelerate HPC applications in a productive and portable manner. With the GROMACS code, CRESTA sought to complete the co-design loop: porting key kernels with OpenACC and comparing their performance with the pre-existing and highly-optimised CUDA versions. The result was a surprising success, driving the development of three significant new OpenACC features in the Cray Compilation Environment (CCE) that allow users to: manage register use, access CUDA functions, and optimise cache configuration. These new features brought the OpenACC performance close to that of the high-performance CUDA in GROMACS but also, with the latest CCE release, are now benefiting Cray users worldwide.



CRESTA Auto-tuning DSL

The Auto-tuning DSL was developed to enable application developers to tune many aspects of applications that have a direct bearing on performance. This covers: build parameters, algorithm choice, specific parameters relating to application source (for example OpenACC directive clauses), and runtime parameters. The DSL can be used for global tuning configuration or embedded in application source. Our mockup auto-tuner was used for this work and it implements many features of the DSL and is easy to use. It uses exhaustive search to determine the best run from the set defined by the tuning parameter values. Of particular note for the Nek5000 tuning was the feature to define tuning scenarios which partition the search space, for example to target different algorithms or problem size regimes.

Particularly useful for the Nek5000 tuning was the feature to define tuning scenarios which partition the search space.