A Performance Evaluation of Kokkos & RAJA using the TeaLeaf Mini-App

Matt Martineau*, Simon McIntosh-Smith*, Mike Boulton*, Wayne Gaudin†, David Beckingsale§
*Dept. of Computer Science, University of Bristol (m.martineau@bristol.ac.uk), †AWE, §LLNL

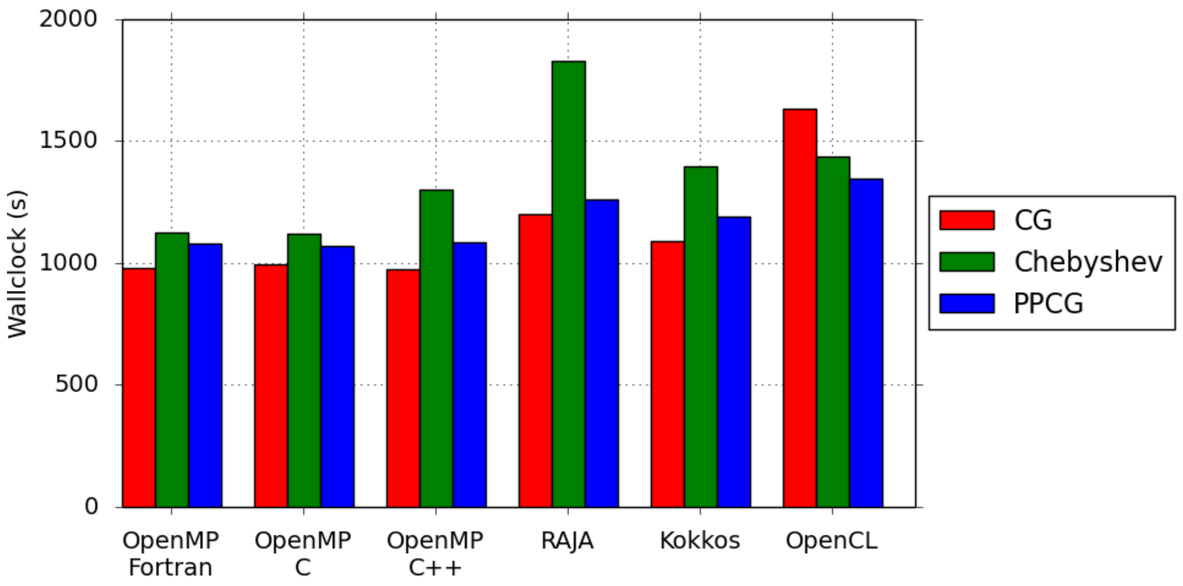
Introduction

TeaLeaf is an open source heat conduction 'mini-app', that is part of the Mantevo benchmark suite from Sandia [1]. It is designed to enable HPC experiments free from the burden of using a fully functional scientific program. The application is memory bandwidth bound and hosts three double precision iterative sparse matrix solvers: Conjugate Gradient (CG), Chebyshev, and Preconditioned Polynomial CG (PPCG). Through extensive testing of a number of new TeaLeaf 2D ports, we have been able to demonstrate the performance profile of a range of diverse programming models across modern HPC devices. In particular, this research focussed upon two new programming models: Kokkos (Sandia National Laboratories) and RAJA (Lawrence Livermore National Laboratories), which leverage template meta-programming and lambda functions respectively, to improve ease of development and long term functional portability. We show here that TeaLeaf RAJA demonstrates promising performance on CPUs and TeaLeaf Kokkos is competetively performance portable.

TeaLeaf 2D Performance Results

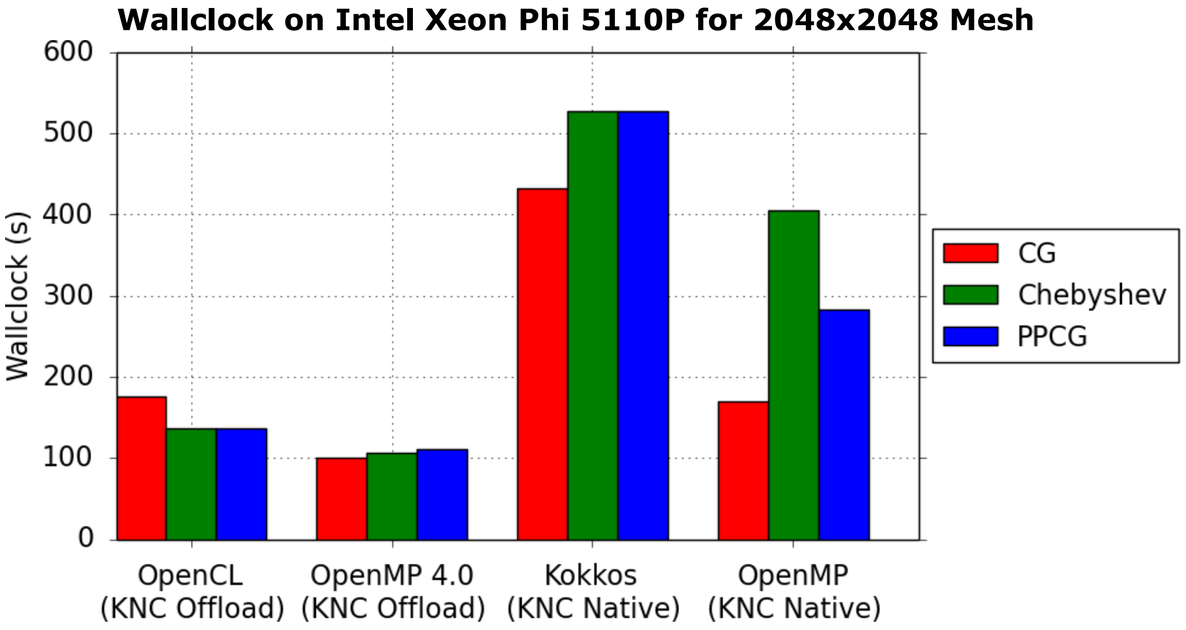
RAJA is only currently implemented for CPUs, but Kokkos can execute on CPUs, NVIDIA GPUs, and supports native compilation for the Intel Xeon Phi Knights Corner (KNC). The net effort required to complete each port varied significantly, with RAJA and OpenMP allowing the fastest development time. The CPU and KNC testing was performed on the Blue Crystal supercomputer at the University of Bristol, and the GPU testing was performed on the Swan supercomputer owned by Cray Inc. Core logic and configurations were conserved across ports to reduce experimental bias, and a mesh size of 4094x4096 was preferred because it represents the point of mesh convergence.

Wallclock on Intel Xeon E5-2670 (Sandybridge) for 4096x4096 Mesh



All CPU implementations were compiled with the Intel compiler suite (version 15.0), and ran on all 16 cores. For CG and PPCG, RAJA demonstrates only a small performance penalty (less than 15% over the baseline OpenMP), and Kokkos achieves performance within 5% of the OpenMP C++ implementation.

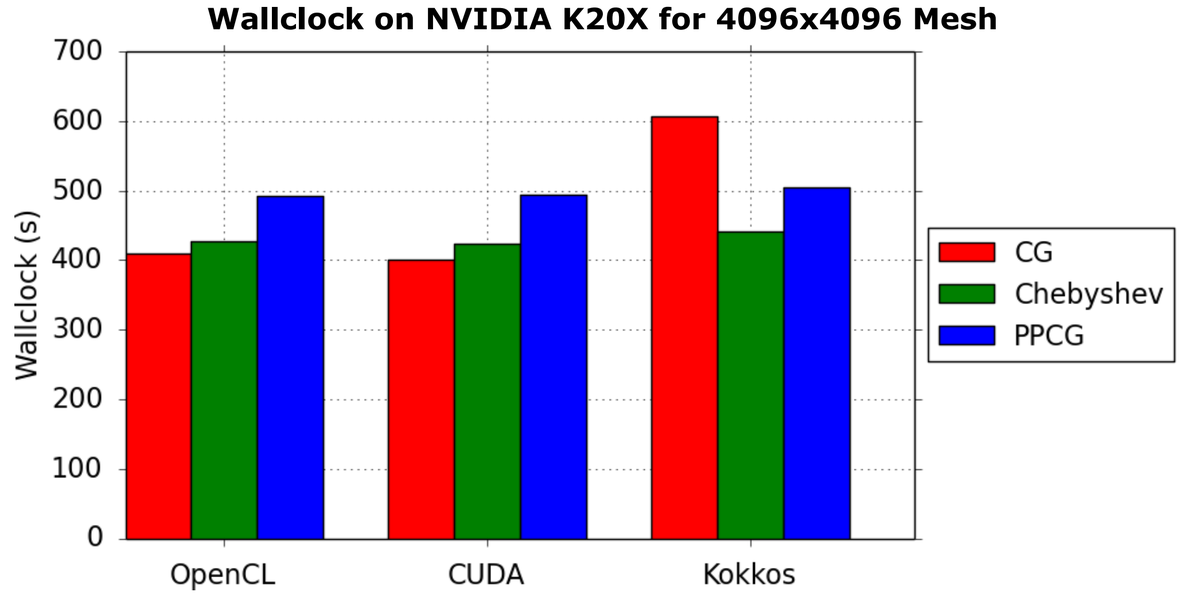
Wallclock on Intel Xeon Phi 5110P for 2048x2048 Mesh



The Kokkos port suffers from poor performance on the KNC which, through collaboration with Sandia, has been isolated as an issue with the native compilation of conditions in the loop body that handle halo exclusion. A solution using hierarchical parallelism provided by Sandia greatly improved the performance on the KNC to within 5% of the OpenMP native port, and sped up the CG solver by 10% on the GPU.

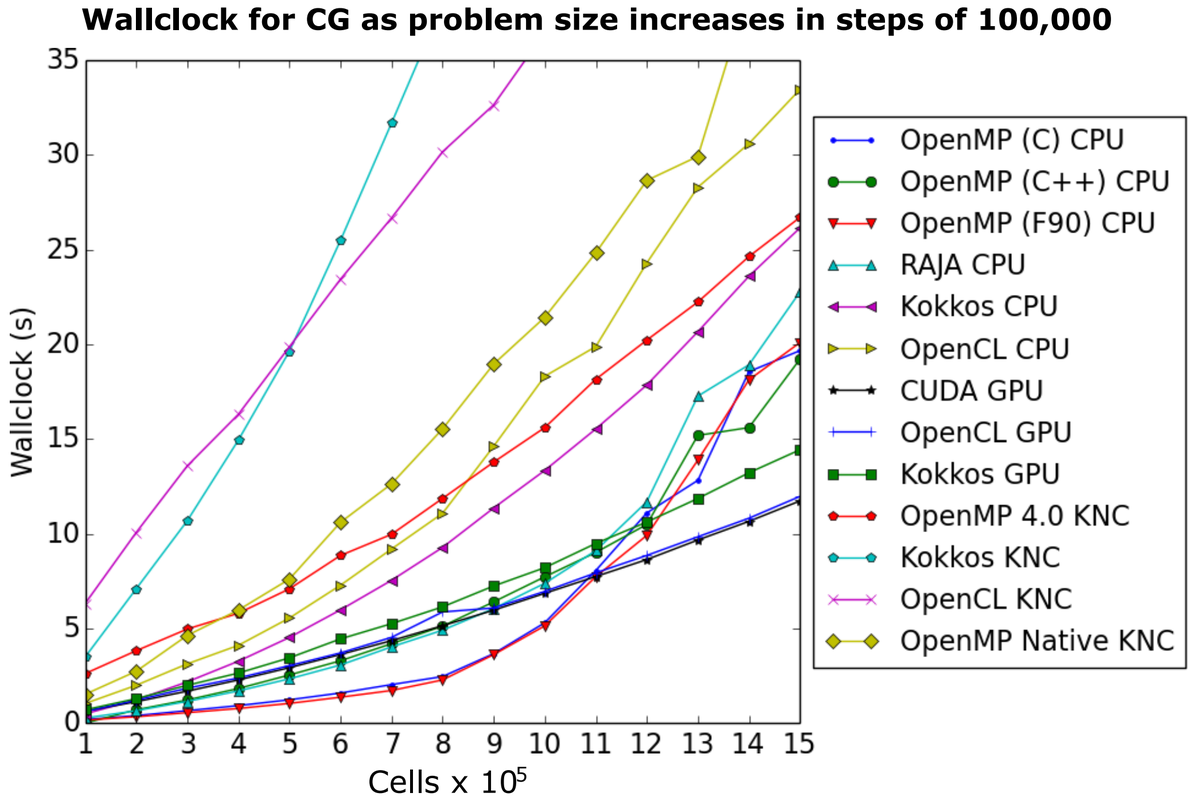
Contrary to those improvements, the performance was reduced by around 30% for the Chebyshev and PPCG solvers on the GPU and so the solution has not been included in the presented results.

Wallclock on NVIDIA K20X for 4096x4096 Mesh



The ports were compiled with the GNU compiler suite (4.9.3) and the CUDA toolkit (7.0.28). Kokkos exhibits impressive NVIDIA GPU performance for the Chebyshev and PPCG solvers, emitting kernels that perform as well as our hand optimised CUDA implementation.

Wallclock for CG as problem size increases in steps of 100,000



This graph shows that the OpenMP and RAJA ports dominate performance until the point that CPU fast cache is saturated, when the high bandwidth devices begin to take over.

Conclusion

TeaLeaf is a valuable research tool that has been used to successfully investigate the performance of a number of modern parallel programming models. The current RAJA implementation requires a similar development effort to OpenMP, and we have shown it exhibits competetive performance on the CPU. Its reliance on C++11 lambdas has limited its functional portability, but this will be improved upon before public release. Kokkos uses C++ templates, demanding extensive re-development of C or Fortran codes, and represents a greater up-front development effort than OpenMP or RAJA. However, Kokkos provides abstractions that reduce some of the complexities inherent with porting to CUDA and OpenCL. Importantly, we have demonstrated with TeaLeaf that codes using Kokkos can expect impressive performance portability between CPUs and NVIDIA GPUs. Collaboration with Sandia found that it is possible to attain good native performance on KNC, using hierarchical parallelism, but this solution represents a trade-off, exposing additional layers of parallelism and increasing complexity, while improving KNC performance and reducing performance of some solvers on the GPU.

University of BRISTOL

UOB HPC

[1] Sandia National Laboratory: The Mantevo project home page. http://mantevo.org/

[2] M. Boulton, S. McIntosh-Smith: Optimising sparse iterative solvers for many-core computer architectures. UK Many-Core Developer Conference (UKMAC), Dec. 2014.

[3] UK Mini-App Consortium (UK-MAC) TeaLeaf page: http://uk-mac.github.io/TeaLeaf/