

Comparison of Parallelisation Approaches, Languages, and Compilers for Unstructured Mesh Algorithms on GPUs

G. D. Balogh, I. Z. Reguly, and G. R. Mudalige

Efficiently exploiting GPUs is increasingly essential in scientific computing, as many current and upcoming supercomputers are built using them. To facilitate this, there are a number of programming approaches, such as CUDA, OpenACC and OpenMP 4, supporting different programming languages (mainly C/C++ and Fortran). There are also several compiler suites (clang, nvcc, PGI, XL) each supporting different combinations of languages. In this study, we take a detailed look at some of the currently available options, and carry out a comprehensive analysis and comparison using computational loops and applications from the domain of unstructured mesh computations. Beyond runtimes and performance metrics (GB/s), we explore factors that influence performance such as register counts, occupancy, usage of different memory types, instruction counts, and algorithmic differences. Results of this work show how clang's CUDA compiler frequently outperform NVIDIA's nvcc, performance issues with directive-based approaches on complex kernels, and OpenMP 4 support maturing in clang and XL; currently around 10% slower than CUDA.