OP2-Clang: A source-to-source translator using Clang/LLVM LibTooling

Gábor Dániel Balogh[1], Gihan R. Mudalige[2] and István Zoltán Reguly[1]

[1]Pázmány Péter Catholic University, Information Technology and Bionics

[2]The University of Warwick, Department of Computer Science

Domain Specific Languages or Active Library frameworks have recently emerged as an important method for gaining performance portability, where an application can be efficiently executed on a wide range of HPC architectures without significant manual modifications. Embedded DSLs such as OP2, provides an API embedded in general purpose languages such as C/C++/Fortran. They rely on source-to-source translation and code refactorization to translate the higher-level API calls to platform specific parallel implementations. OP2 that provides a high-level API for the solution of unstructured-mesh computations can generate a wide range of parallel implantations for execution on architectures such as CPUs, GPUs, distributed memory clusters and heterogeneous processors making use of radical platform specific optimizations. Compiler tool-chains supporting source-to-source translation of code written in mainstream languages currently lacks the capabilities to carry out such wide-ranging code transformations. Clang/LLVM's Tooling library (LibTooling) has long been touted as having such capabilities but have only demonstrated its use in simple source refactoring tasks.

In this talk we introduce OP2-Clang, a source-to-source translator based on LibTooling, for OP2's C/C++ API, capable of generating target parallel code based on SIMD, OpenMP, CUDA and their combinations with MPI. OP2-Clang is designed to significantly reduce maintenance, particularly making it easy to be extended to generate new parallelizations and optimizations for hardware platforms. In this research, we demonstrate its capabilities including (1) the use of LibTooling's AST matchers together with a novel strategy that use parallelization templates or skeletons to significantly reduce the complexity of generating radically different and transformed target code and (2) the generation of optimized kernels for computation loops with indirections for two industrial representative unstructured-mesh applications. We determine the current limits of LibTooling to support the source-to-source translations in eDSLs such as OP2 and provide recommendations for its future development to facilitate code-generation requirements of such frameworks.