Linköping University
IDA Department of Computer and Information Sciences
PELAB
Prof. Dr. Christoph Kessler

LINKÖPING UNIVERSITY

Master Thesis Project Proposal (30hp):

Skeleton programming with SkePU and MLIR

High-level parallel programming aims to abstract challenging aspects of parallel and heterogeneous systems for non-expert programmers. Algorithmic skeletons is an interface approach based on computational patterns, such as map, reduce, and stencil operations. These patterns can be instantiated by providing a custom operator ("user-function"), which is then applied to a supplied dataset in parallel according to the particular pattern semantics. Skeleton programming framworks and libraries such as SkePU implement skeletons as C++ constructs and provides "backends" for parallelism in multi-core CPUs, GPU accelerators, and multi-node clusters. The skeletons are typically provided as libraries, or in the case of SkePU, as a framework with both library and a custom pre-compiler toolchain. The pre-compiler is a source-to-source compiler based on LLVM clang. It performs a rather light-weight source code transformation; in particular, generates platform-specific code variants from the user-functions that can be used with the different SkePU back-ends. SkePU is a long-term open-source effort at PELAB, Linköping University.

MLIR is a rather new, customizable, multi-level intermediate representation (IR) within the LLVM open-source compiler infrastructure. It has become very popular in both academic and industry research and development in recent years. MLIR allows to develop custom, domain-specific as well as device-specific, IR dialects at various levels of abstraction, which are translated into other IR dialects at higher or lower level of abstraction by raising and lowering passes, respectively, eventually lowering down to the LLVM IR from which target code can be generated. Directly controlling the compiler's lowering process for SkePU-specific code constructs is expected to enable new optimization opportunities and provide more independence from the vendor-specific compilers employed in the existing SkePU backends.

Task Use the sequential SkePU interface (and header-only implementation) to generate MLIR of existing dialects (e.g., "linalg", "affine", "gpu", "acc") in order to generate optimized and/or parallelized IR. This is then evaluated against the current, textual-replacement-based SkePU backend with regards to performance and portability. A comparison of the two with regards to future extensibility and maintainability should also be conducted.

Remark This is an exploratory, research-oriented thesis project. It provides a perfect preparation for compiler-related work in academia or industry.
More information is available from us on request.

Prerequisites A completed course in compiler construction with labs, such as TDDE66. TDDD56 Multicore and GPU Programming; Advanced Programming in C++; some familiarity with Linux and Cmake.

Contact Christoph Kessler, August Ernstsson (`first.last@liu.se`)

Open thesis projects: `https://www.ida.liu.se/~chrke55/exjobb/open-exjobb-projects.shtml`