USC Viterbi
School of Engineering
Ming Hsieh Department of Electrical and Computer Engineering

Seminar

T2S: Programmatic Control of a Compiler for Productive Performance: Accelerating Tensor Computing across Spatial and Vector Architectures

Hongbo Rong - Intel

Friday, November 12, 2021 - 5:30-6:30 PM

THH 119 or prasannazoom.github.io

Abstract: The death of Moore's Law and the rapid development of AI have led to the booming of accelerators for tensor computations. Spatial architectures like FPGAs and vector architectures like GPUs are commonly used for creating such accelerators. FPGAs are well known to be difficult and very time-consuming to program for performance. And it takes time to engineer performance for each important compute on each architecture.

We propose a new programming methodology, T2S (Temporal to Spatial), to make it easier to program FPGAs. A programmer specifies a temporal definition and a spatial mapping. The temporal definition defines the functionality to compute, while the spatial mapping defines how to decompose the functionality and map the decomposed pieces onto a spatial architecture. The specification precisely controls a compiler to actually implement the spatial mapping, including many generic, strategic loop and data optimizations. Consequently, high performance is expected with substantially higher productivity.

We further extend the methodology to heterogeneous systems including both FPGAs and GPUs. This brings two big advances over our original approach. First, building high-performance user-managed caches for tensors is usually a big hurdle for average software programmers. We make this easy by allowing programmers to specify a data flow across an *abstract memory system*. Second, unlike CUDA and OpenCL, our programming model *explicitly combines SIMT and SIMD*, *and realizes SIMD (multi-dimensional vectorization) by specifying architecture-agnostic multi-dimensional systolic arrays*. The compiler generates optimized hardware/software systolic arrays on FPGAs/GPUs. The explicit SIMD gives programmers convenient control in utilizing critical hardware resources (e.g. DSPs on FPGAs and SIMD units on GPUs).

We have prototyped our methodology for Intel FPGAs and GPUs. We implemented several high-profile kernels that are expressed in tensors and with different compute patterns (SGEMM, 2-D convolution, Capsule convolution, PairHMM, and LU matrix decomposition) across 2 generations of FPGAs (Arria 10 and Stratix 10) and 3 generations of GPUs (GEN 9.5, GEN 12, and a research GPU), and compared them with state-of-the-art ninja implementations, and machine peaks, when available. Our designs provide *throughput on par with or better than state-of-the-art comparison points*, achieve *close to 100% DSP and PE efficiencies on both FPGAs, and 75%-91% theoretical peak throughput on all GPUs*, for all kernels where such measurements are applicable. A nearly full DSP/PE efficiency on an FPGA indicates that the compute engine we generate is kept busy with useful operations without pipeline stalls (due to, e.g. slow memory operations). *This performance is achieved with a very compact, high-level code specification* – all kernels have only several tens of lines of code, and can be written in under half an hour. These results confirm that our methodology can achieve high performance and programmer productivity on a range of tensor operations, and performance can be made largely portable across spatial and vector architectures.

Bio: Hongbo is a language and compiler expert at the Parallel Computing Lab of Intel. His current research focuses on performance programming of heterogeneous systems. His previous works on compilers won Best Paper Awards at CGO 2004 and CGO 2014, and a Best Paper Finalist at SC 2016. The idea of this talk, T2S, won Intel's Research Velocity Challenge in 2017. Previously he has worked on various static and dynamic compilers at Microsoft and Intel for ten years.

Host: Professor Viktor Prasanna