Optimizing GEMM routine with Data Preloading on ARM ThunderX2

Enoch Jung and Jaeyoung Choi

School of Computer Science and Engineering, Soongsil University, Seoul 06978, Republic of Korea

Corresponding author (enochjung@soongsil.ac.kr)

General Matrix-Matrix Multiplication (GEMM) is an important computation routine in linear algebra, demanding highly optimized implementations. The Blocked matrix multiplication algorithm is commonly used in the implementation of optimized GEMM routines. It partitions matrices into blocks in order to enhance data reuse, and it performs multiplication operations between these blocks. Optimizing GEMM routines on a target CPU involves two main tasks: searching for an appropriate block size and implementing an optimized multiplication routine between blocks (micro-kernel). Research on automatically optimizing the block size has been conducted extensively, but research on automatically generating optimized micro-kernels is scarce. The reason is simple: the performance should be higher than of existing kernels. The micro-kernel takes more than 95% of the overall matrix multiplication time, and of the importance, it is usually handcrafted by experts with deep knowledge of processor architecture. This implies that the research is developing algorithms that can automatically generate micro-kernel with higher performance than expert's one, even without knowledge of the architecture.

This research introduces an approach for automatically optimizing single-core DGEMM (Double-precision GEMM) routine by applying various optimization strategies to the micro-kernel. We categorize these strategies into four general approaches and use an auto-tuner to generate optimized micro-kernel by measuring their effectiveness of each strategy on the target CPU. Among the applied strategies, 'data preloading,' which loads data (or submatrices) into registers directly from cache or memory ahead of their actual requirement, shows promising performance, warranting further analysis. We conducted auto-tuning of the DGEMM routine on the ARM-based server processor Marvell ThunderX2. The performance achieved on a single core of the ThunderX2 is 16.456 Gflops, representing 100.969% of the performance compared to ArmPL's 16.298 Gflops. Since the performance improvement is solely attributed to the optimization of the micro-kernel, we anticipate that our methodology can also enhance the performance of other libraries.

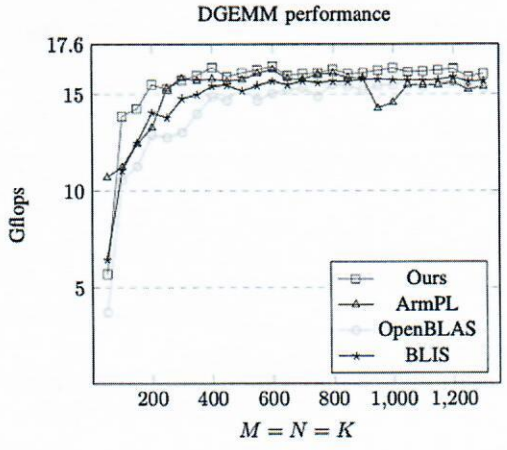

Figure 1. Single-core DGEMM performance comparison of our tuned DGEMM routine and other BLAS libraries on the ThunderX2 processor

Acknowledgments This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2023-00321688).