Grant Pedersen CSCI 5593

Report #1, February 5, 2019

**Report on “NVIDIA Tensor Core Programmability, Performance & Precision”**

Companies are pushing for Artificial Intelligence based applications using deep learning and machine learning techniques that require heavy matrix multiplication. This causes companies to construct new hardware to speed up the computation of matrix multiplication using GPUs. The authors investigate the performance of a new microarchitecture for GPUs called Tensor Core. Tensor Core performs matrix-multiply-and-accumulate on 4x4 matrices per clock cycle. This means multiplying two 4x4 FP16 matrices and sum the multiplication product of FP32 matrix to an accumulator; this utilizes mixed precision operations. They target the product called NVIDIA Tesla V100, which uses the Volta GPU microarchitecture or Tensor Core to test the accuracy of precision of matrix multiplication for floating point numbers and performance. They provide three test cases to measure the performance of Tesla V100 and how the applications of the GPU can be used in high performance computing (HPC) applications and how large matrix multiplication and batch matrix multiplication compare in speed.

The authors employ three methods to program a matrix-multiply-and-accumulate on Tensor Cores. First is using CUDA Warp Matrix Multiply Accumulate (WMMA), second CUTLASS and third cuBLAS General Matrix to Matrix Multiplication (GEMM); the authors also investigate large matrix or batch matrix multiplications performance [1]. WMMA API allows for large matrix multiplication only and can calculate 16x16 matrices. WMMA allows for fragmentation of GPU registers to store input and accumulator matrices while storing results of fragmentation into the GPU’s global memory; execution is done on the threads or CUDA Warp and for only large matrices not batches [1]. By performing this operation, they can tile each part of a matrix onto a thread block and exploit the shared memory in the GPU for fast computations. Second method, NVIDIA CUTLASS, used CUDA C++ library for GEMM operations to also support tiling of matrices and now exploits the software pipelining to aid in the memory latency. Lastly, NVIDIA cuBLAS uses GEMM calls on Tensor Cores, while still performing non-batched matrix multiplications. The other approach uses batched matrix multiplications in context of an HPC application, and found the GEMM works best with large matrices, but not well with batch sizes. The authors saw that solving small matrix multiplications in parallel was the best in performance for Tensor Cores, so they developed a batched GEMM to be solved by Tensor Cores.

Authors found that deep learning neural network models can work well with some precision loss with mixed-precision matrix multiplication, which means less computation, but higher loss of accuracy in precision for mixed-precision operations; however, HPC are in need for higher precision, meaning more computation for more accurate precision. In performance the authors achieved 83 Tflops/s from the cuBLAS method. Batched WMMA proved that shared memory reduces memory traffic in Tensor Cores and performs at 4 Tflops/s [1].

**Reference**

[1] S. Markidis, S. W. D. Chien, E. Laure, I. B. Peng, and J. S. Vetter, “NVIDIA Tensor Core Programmability, Performance & Precision,” *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2018, doi: 10.1109/IPDPSW.2018.00091