Fast Sparse General Matrix-Matrix Multiplication on GPU with Low Memory Usage
Yusuke Nagasaka, Akira Nukada, Satoshi Matsuoka
Tokyo Institute of Technology

Proposal: Fast and Memory Saving SpGEMM Algorithm

Flow of new algorithm
1. Count the number of intermediate products of each row
2. Group rows by the number of intermediate products
3. Count the number of non-zero elements of each row of output matrix for all groups with CUDA kernels and streams
4. Set row pointers of output matrix to store in CSR by Thrust scan
5. Group rows by the number of non-zero elements
6. Calculate the output matrix for all groups with CUDA kernels and streams
   a. Calculate values and column indices on hash table
   b. Shrink table to hold only non-zero elements
   c. Sort by column index in ascending order

Two Way Thread Assignment and Memory Access

WARP/ROW: One warp for each row of A, one thread for each non-zero of A and B

TB/ROW: One thread block for each row of A, one warp for each non-zero of A, one thread for each non-zero of B

Performance Evaluation

Single Precision: Speed up is up to $\times 4.0$

Double Precision: Speedup is up to $\times 3.3$

Maximum Memory Usage

Single Precision

Double Precision

10.3% reduction on average

14.3% reduction on average

Conclusion and Future work

We propose the novel sparse general matrix-matrix multiplication algorithm which is designed for reducing both memory overhead and execution time on GPU. Our algorithm achieves speedups of up to $\times 4.0$ in single precision and $\times 3.3$ in double precision compared to existing fast SpGEMM libraries. For future work, we will apply our technique to the preconditioner such as AMG method and real-world applications.

Reference