Programming Massively Parallel Processors

Lab 1

Abhinav Kannan
Haotian Liu

a. How many times is each element of the input matrices loaded during the execution of the kernel?

Each element of the input matrix is loaded once into the shared memory. Once all the elements are loaded, all the computation is done of the shared memory.

b. What is the memory-access-to-floating-point-computation ratio in each thread? Consider a multiply and addition as separate operations, and ignore the storing of the result. Only global memory loads should be counted towards your off-chip bandwidth.

In the code we load the entire M matrix and the entire N matrix to shared memory. Since the matrix size is 16, there will be 256 loads for M and 256 loads for N. But, we use 256 threads in the block. So, each thread will be loading one element from the matrix and since the half warp has 16 threads, the memory accesses are coalesced. Totally, there are **512 loads from the global memory**, ignoring store operations.

Each thread will compute one element of the output matrix. So, each thread reads one row of shared memory M, one column if shared memory N, multiply them and add all the partial sums into the final value. Therefore, each thread performs 2 operations (one multiply and one add) 16 times (since each thread accesses one row and one column which has 16 elements). Therefore, each thread performs 16 * 2 = 32 floating point computations. The total floating point computations in the block is 256 * 16 * 2 = 8192.

Therefore,

Memory-access-to-floating-point ratio per thread = 2 / 32 = **1 / 16**. (Each thread loads two elements from memory. One from M and one from N).

Memory-access-to-floating-point ratio for block = 512 / 8192 = 1 / 16.