5 Kernels were implemented: C++ Version , CUDA Grid Stride (No Prefetching/No Cooperative Group), CUDA Grid Stride (Prefetching/ No Cooperative Group), and the following two are utilizing cooperative group with and without prefetching. This was done as due diligence to see the how prefetching and cooperative group syncing affects the performances of the kernels. Generally all kernels utilize the same structure of performing a block per block parallelization of getting the full dotproduct. In order to achieve accuracy and avoid any race conditions atomics and a reduction method was done.

Performance of the parallelized versions were slower than the base C version for various reasons such as the threads taking time to synchronize, the absence of shared memory and due to time constraints limitations on the use of cooperative groups. The implementation of cooperative groups mostly focused on utilizing its function to access the current thread block and synchronizing that thread block. This implementation on kernels made it faster than kernels that didn't have that function after reduction and storing on partial sums. The implementation of prefetching was also done to prepare the GPU device for the amount of data being processed. The implementation of these two features yielded better performances than the kernels that didn't and only implemented one or the other.

Due to time constraints the improvements needed to make the parallelization programs faster than the base C program is to utilize the prefetch functionality the shared memory feature that cuda has should be used. The use of shared memory will contain the prefetched values and shared memory will also ensure that the threads will work together cutting down the time with reduction and synchronization.

References:

Using Shared Memory in CUDA C/C++ | NVIDIA Technical Blog. (2022, August 21). NVIDIA Technical Blog. https://developer.nvidia.com/blog/using-shared-memory-cuda-cc/#:~:text=Summary,mechanism%20for%20threads%20to%20cooperate.

Boosting Application Performance with GPU Memory Prefetching | NVIDIA Technical Blog. (2023, June 12). NVIDIA Technical Blog. https://developer.nvidia.com/blog/boosting-application-performance-with-gpu-memory-prefetching/

What Every Computer Scientist Should Know About Floating-Point Arithmetic. (n.d.). https://docs.oracle.com/cd/E19957-01/806-3568/ncg_goldberg.html

CUDA Floating Point. (n.d.). https://docs.nvidia.com/cuda/floating-point/index.html