Laboratory 3: Report

Piotr Harmuszkiewicz

Juan Salmerón Moya

In this laboratory, we try different methods for avoiding memory page faults so programs can run faster in a CUDA enivoronment.

The first method is memory prefetching. By prefetching data (using the cudaMemPrefetcAsync function) we can drastically reduce the time our programs take as we evade a lot of page faults. By prefetching, we store in memory the next consecutive pages of the ones we initialize with. The program vector_add_standard loses 149ms in memory page faults. In vector_add_prefetch_gpu this time is not lost, so vector adding takes 2ms instead of 151ms. The method cudaDeviceSynchronize takes less time, and Host to Device takes 34ms instead of 44ms.

The second method is to use strides when initializing data. Instead of loading data in a sequential fashion, we can do it by a grid at a time. By doing this, we eliminate overhead from scheduling and retiring blocks. There can be considerable efficiency gains in simple kernels by doing so. The initWith takes 2ms of GPU time in vector_add_prefecth_gpu_init_gpu but by doing so, cudaDeviceSynchronize and cudaMemPrefetchAsync take even less time, falling from around 22ms each to 5 or 3 ms.