Ahmet Oguz
ECE 60827
CUDA Programming Part 1
January 27, 2025

For this assignment we will be working with CUDA programming for GPU acceleration. For the first part, we will be implementing Single-precision A · X Plus Y (SAXPY). For the second part, we will be implementing Monte Carlo estimation of the value of π.

In Figure 1, we have our SAXPY GPU activities as reported by the nvprof CUDA profiler with the varying vector sizes. We see that as the vector size increases the saxpy_gpu kernel time % decreases, while the amount of time used for the memcpy from host to device and device to host increases. This makes sense, as the vector size increases, the amount saxpy_gpu kernel will benefit from the parallelism will also increase, resulting in memcpy dominating the GPU activity.

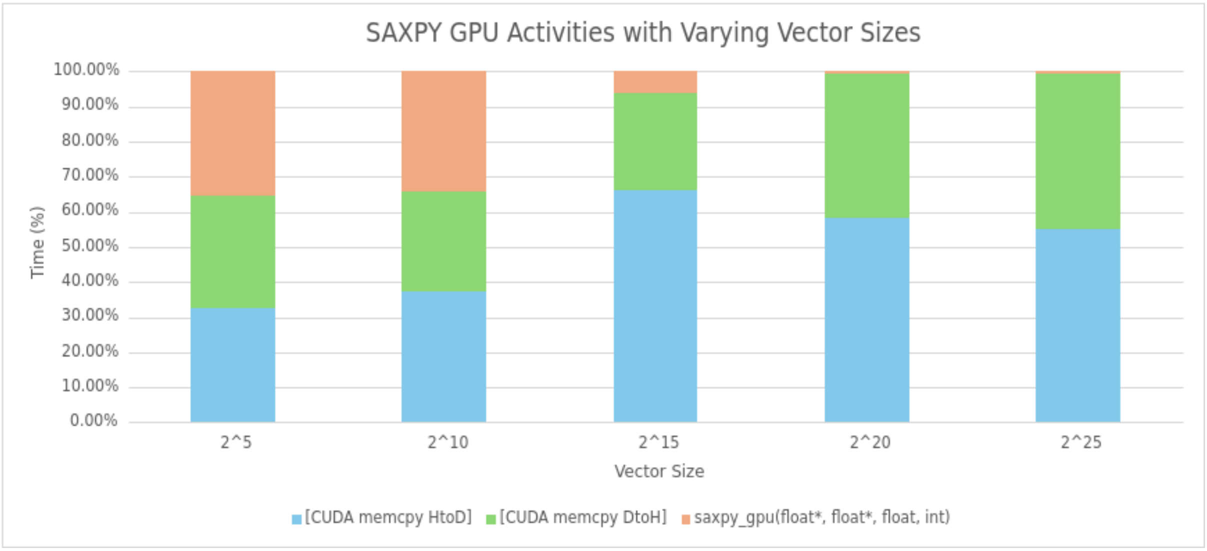

Figure 1

In Figure 2, we have our Monte-Carlo π estimation GPU activities with varying sample sizes. We see that, while the generatePoints is the major portion of the execution time, as the sample size decreases, the CUDA memcpy time (%) increases. This makes sense, since the amount the kernel will have to compute with the increased sample size is going to have it dominate the GPU activity. At lower sample sizes, since the amount of computation necessary will be lower, memcpy will have a higher impact on GPU activity.

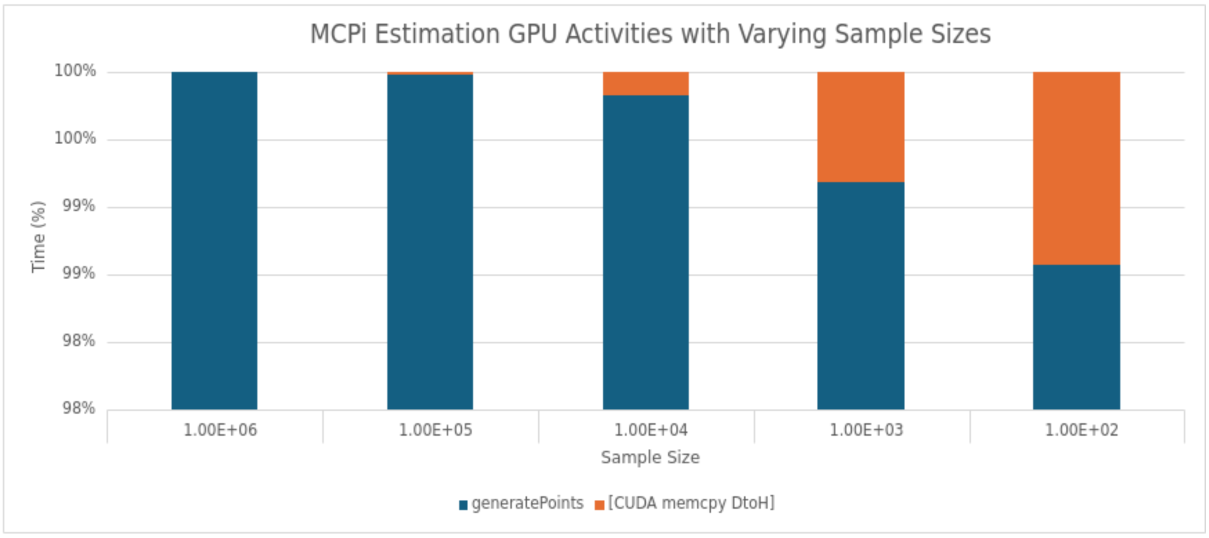

Figure 2

In Figure 3, we have our Monte-Carlo π estimation GPU activities with varying generated thread counts. We see that the number of generated threads does not affect the GPU activities. This is also expected considering the time % of different GPU activities wouldn't be affected, but rather the overall execution time the program takes would be expected to differ.

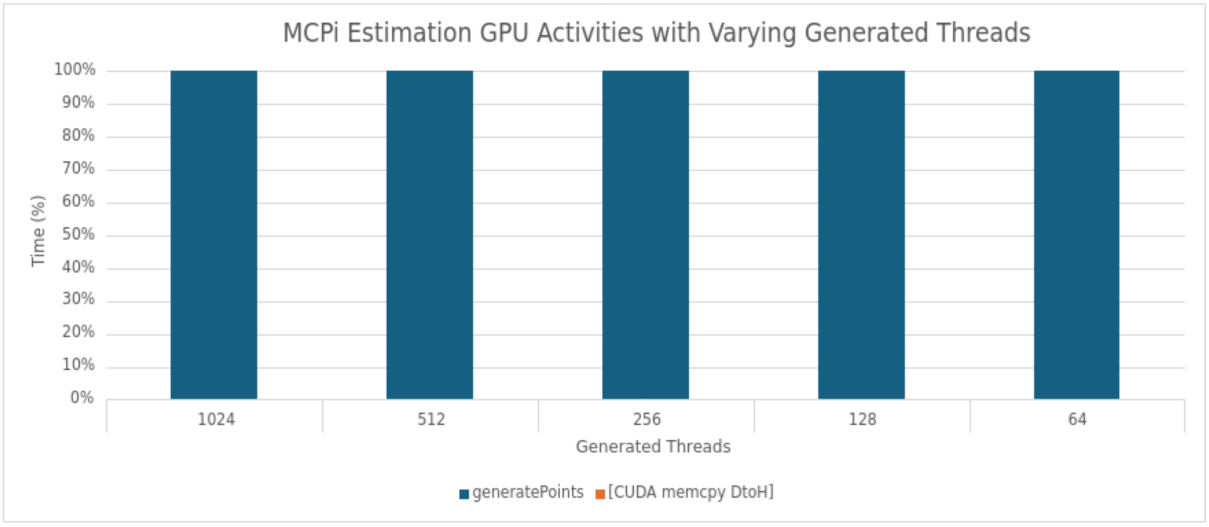

Figure 3