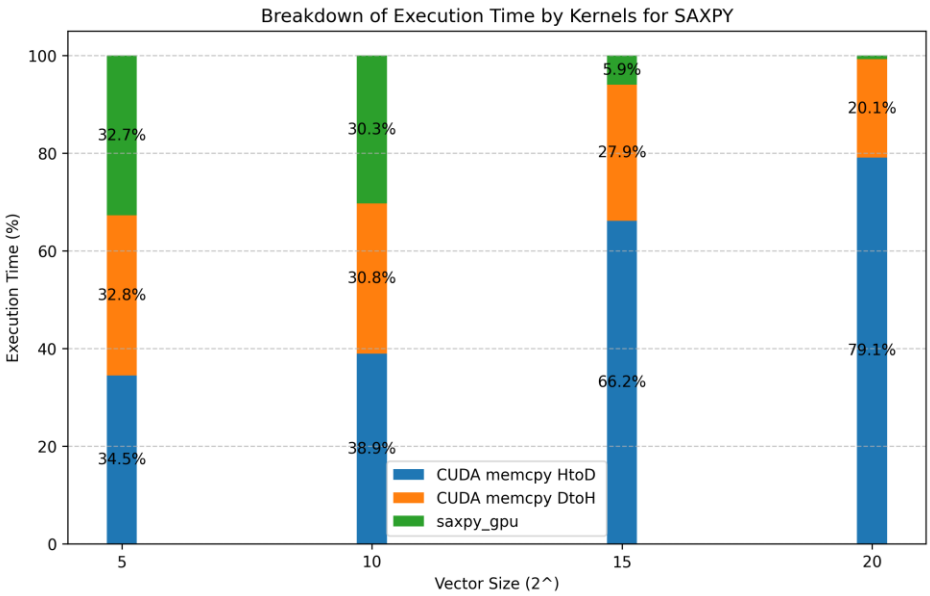
Program Assignment 1

John Cho

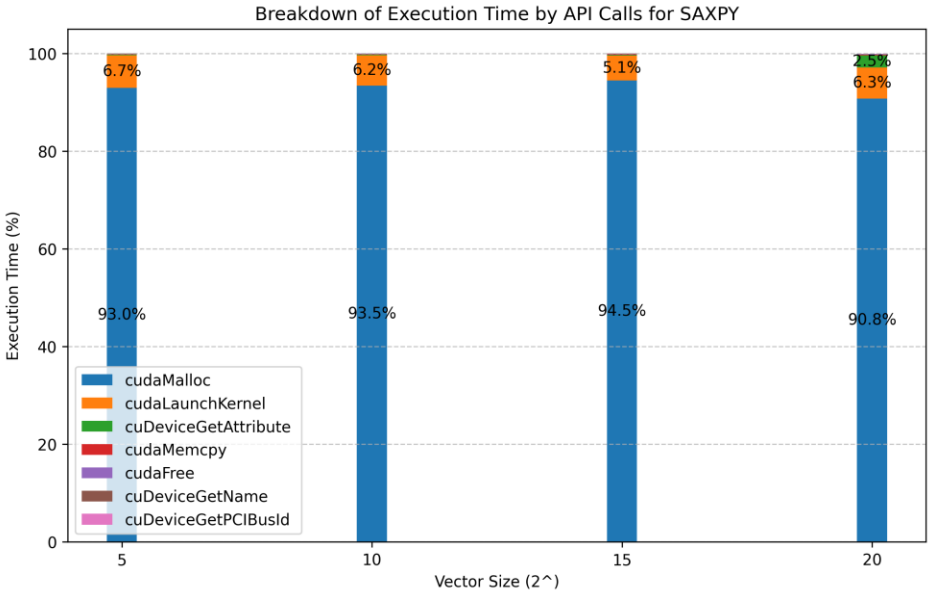
Parallel Programmable Architecture

January 31, 2025

Part 1: SAXPY



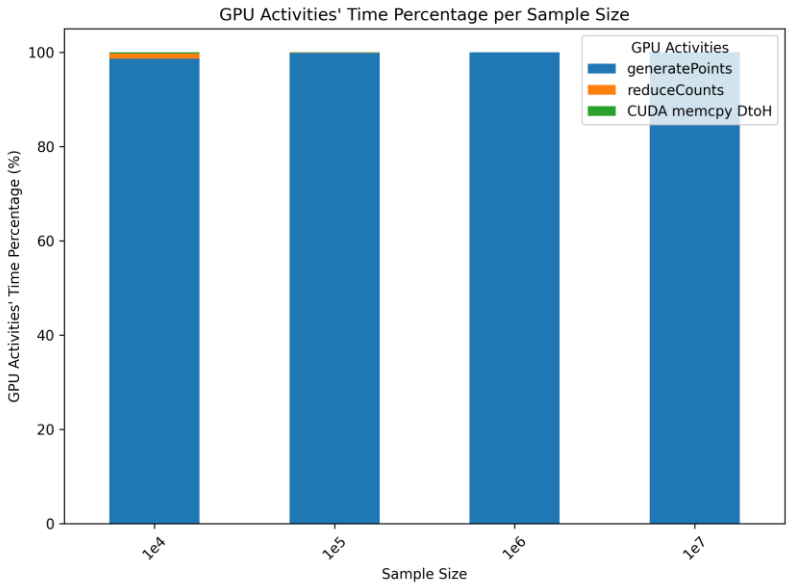
From the graph, I observed that as vector sizes increase, the time required to copy data from the host to the device also increases. However, the execution time for SAXPY on the GPU decreases.



From an API perspective, cudaMalloc accounts for a significant portion of execution time. However, when the vector size reaches 2^20, cudaMalloc API calls are reduced by 90%, while cuDeviceGetAttribute API calls increase to 2.5%. I assume this occurs because, as the data size grows, retrieving data attributes takes longer.

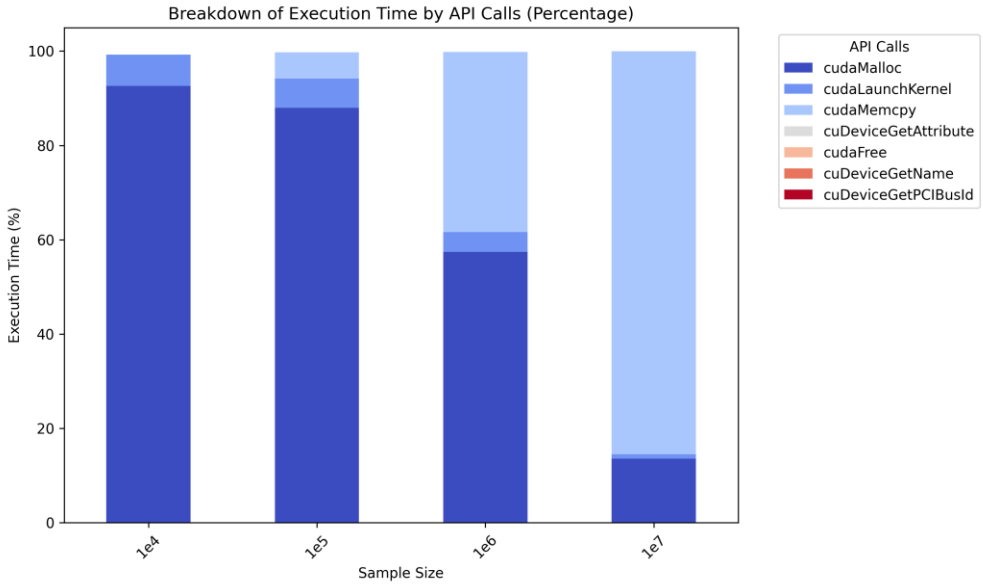
Part 2: Monte-Carlo Pi

1) Sample size vs. Execution Time



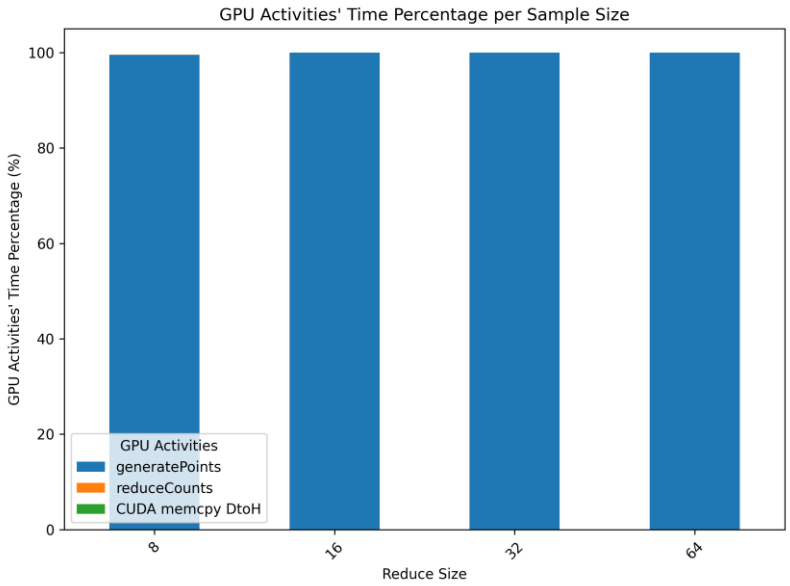
As the sample size increases, the percentage of time spent generating points also increases. I think this is because, the Montecarlo pi is the computer intensive program, therefore, the proportion of time spent on other GPU activities, such as reduce Counts.

Regarding API calls, as the sample size grows, the percentage of execution time for cudaMalloc decreases, while cudaMemcpy increases.



2) Reduce Size vs. Execution time

When the reduction size increases, the time spent on generate points also rises, while the proportion of cuda memcpy decreases. However, since it's almost 99 percent of the total execution time, the difference is not easily visible in the graph.



From the API call perspective, cudaMalloc occupies the largest percentage at a reduction size of 16. Beyond this point, its contribution decreases, while cudaMemcpy takes up a larger share.