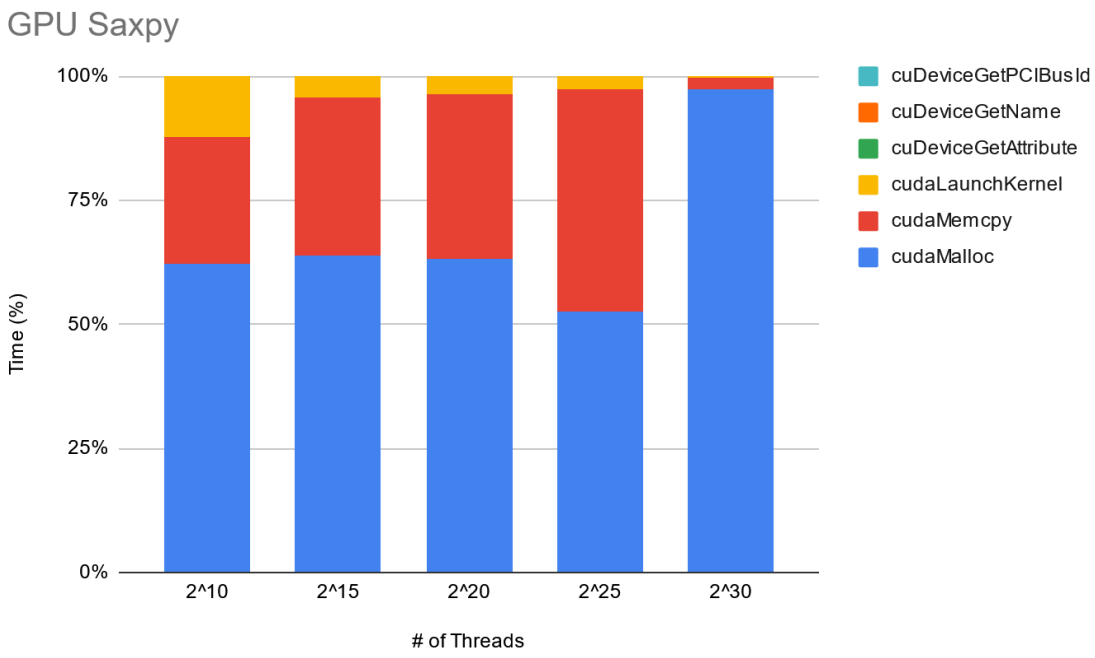
ECE 60827 - Karson Shields Cuda Assignment 1
Part A: GPU Saxpy

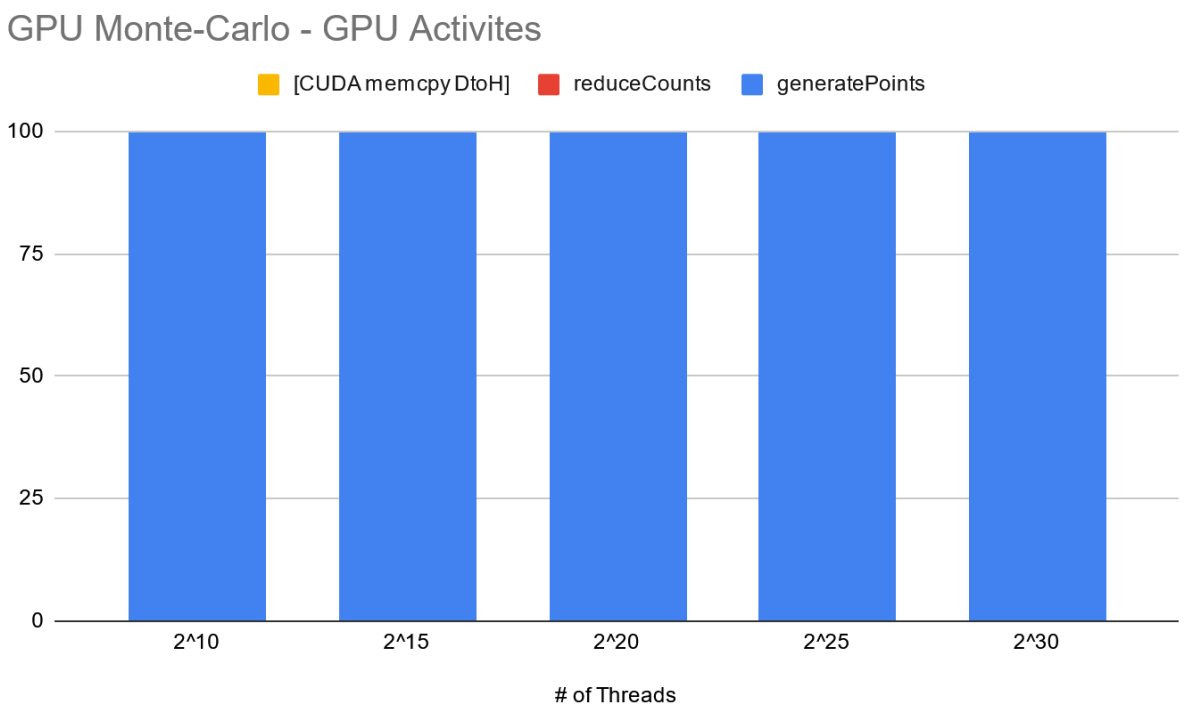


GPU Saxpy (Time %)												
# of Threads	cudaMalloc	cudaMemcpy	cudaLaunchKernel	cuDeviceGetAttribute	cuDeviceGetName	cuDeviceGetPCIBusId	cudaFree	cuDeviceGetCount	cuDeviceGet	cuModuleGetLoadingMode	cuDeviceGetUuid	cuDeviceTotalMem
2^10	62.29	25.46	12.15	0.08	0.01	0	0	0	0	0	0	0
2^15	63.69	31.93	4.28	0.09	0.01	0	0	0	0	0	0	0
2^20	63.09	33.32	3.49	0.08	0.01	0	0	0	0	0	0	0
2^25	52.52	44.94	2.47	0.06	0.01	0	0	0	0	0	0	0
2^30	97.35	2.46	0.19	0	0.01	0	0	0	0	0	0	0

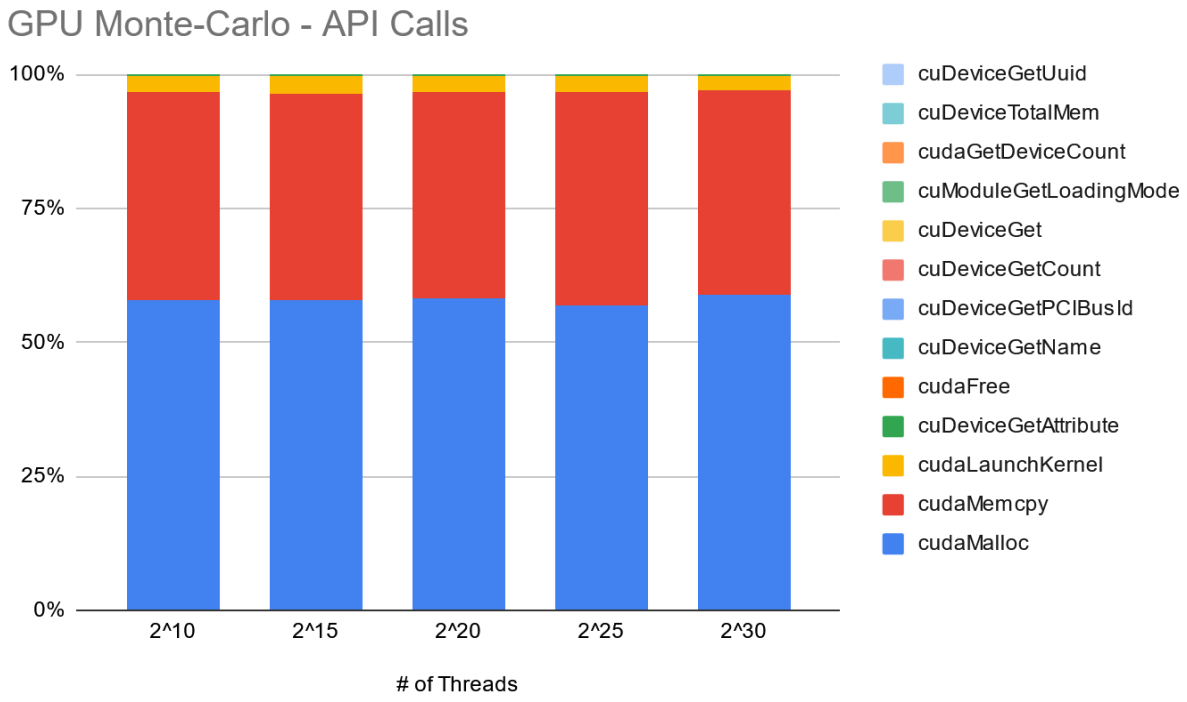
When running GPU Saxpy, while increasing the # of threads, the main observation I made is that it is memory bound and ramps quite quickly in the amount of memory time. The main API calls that take up execution time are the memory commands and the kernel launch itself. The rest of the API calls are negligible in execution time, thus depicting that majority of execution

time is in memory operations.

Part B: GPU Monte-Carlo Estimation of Pi



GPU Monte-Carlo - GPU Activities (Time %)			
# of Threads	generatePoints	reduceCounts	[CUDA memcpy DtoH]
2^10	99.99	0	0
2^15	99.99	0	0
2^20	99.99	0	0
2^25	99.99	0	0
2^30	99.99	0	0



GPU Monte-Carlo - API Calls (Time %)												
cudaMalloc	cudaMemcpy	cudaLaunchKernel	cuDeviceGetAttribute	cudaFree	cuDeviceGetName	cuDeviceGetPCIBusId	cuDeviceGetCount	cuDeviceGet	cuModuleGetLoadingMode	cudaGetDeviceCount	cuDeviceTotalMem	cuDeviceGetUuid
57.76	38.83	3.26	0.08	0.06	0.01	0	0	0	0	0	0	0
58.01	38.41	3.23	0.26	0.08	0.01	0	0	0	0	0	0	0
58.09	38.56	3.19	0.08	0.07	0.01	0	0	0	0	0	0	0
56.93	39.85	3.06	0.08	0.07	0.01	0	0	0	0	0	0	0
59.01	37.91	2.92	0.08	0.07	0.01	0	0	0	0	0	0	0

While running GPU Monte-Carlo, the main observation I notice is that the amount of execution time spent in memory is fairly consistent across the # of threads, indicating that this is a compute bound algorithm. The amount of time spent in the GPU is all on generating the points, with the other functions being negligible on execution time despite being important functions on the algorithm. In regards to the API calls, the time spent in memory has slight variations in the different functions being called, but overall the time spent is fairly consistent.