Contribution: Yitong Li mainly finish the exercise 1 and 2 and Guoqing Liang mainly finish the exercise 2 and 3.

Code Link：[CHU-2002/DD2360HT23 (github.com)](https://github.com/CHU-2002/DD2360HT23)

### Exercise 1

1. Assume X=800 and Y=600. Assume that we decided to use a grid of 16X16 blocks. That is, each block is organized as a 2D 16X16 array of threads. How many warps will be generated during the execution of the kernel? How many warps will have control divergence? Please explain your answers.

Because the grid size is 16X16 = 256, for X=800 and Y=600, considering the kernel function, the block number should be ⌈800/16⌉ \* ⌈600/16⌉= 1900. So the number of threads should be 1900X16X16 = 486400.

So the number of warps is 486400/32 = 15200.

Assuming the GPU use row-major order, for the blocks y from 601 to 608, *Divergences* occur.So the number is 4\*800/16 = 200. (In these blocks, only half of warps get *Divergences*)

2. Now assume X=600 and Y=800 instead, how many warps will have control divergence? Please explain your answers.

Just as the first question. X from 601 to 608 *Divergences* occur.

Still assuming the GPU use row-major order, the number is 800/16 \* 256/32 = 400. (In these blocks, all blocks get *Divergences*)

3. Now assume X=600 and Y=799, how many warps will have control divergence? Please explain your answers.

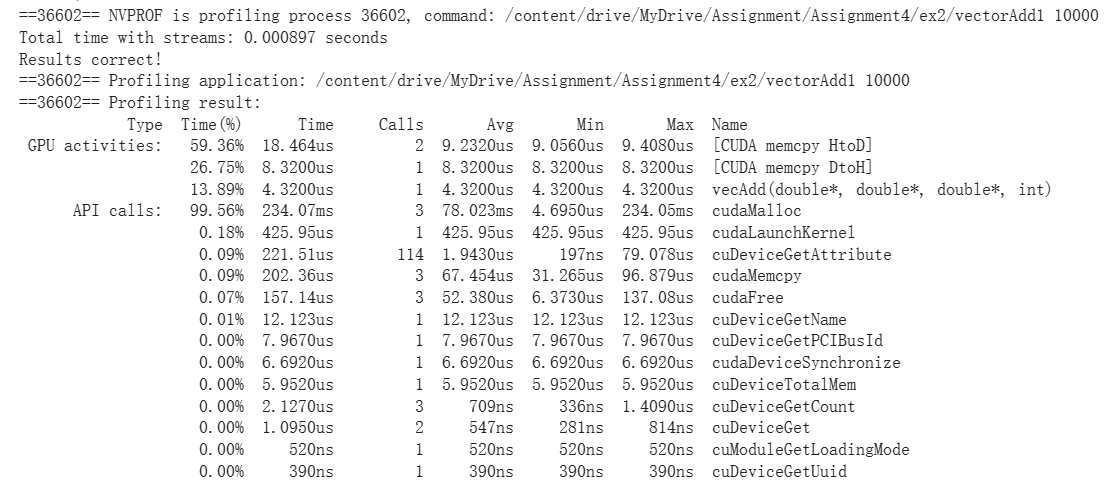
For X = 600 and Y = 799, *Divergences* occur in threads get X from 601 to 608, Y in 800.

Still assuming the GPU use row-major order, for warps, the last row and blocks in question 2 will all lead to *Divergences.* So the number will be 800/16 \* 256/32 + ⌈600/16⌉ -1= 438.

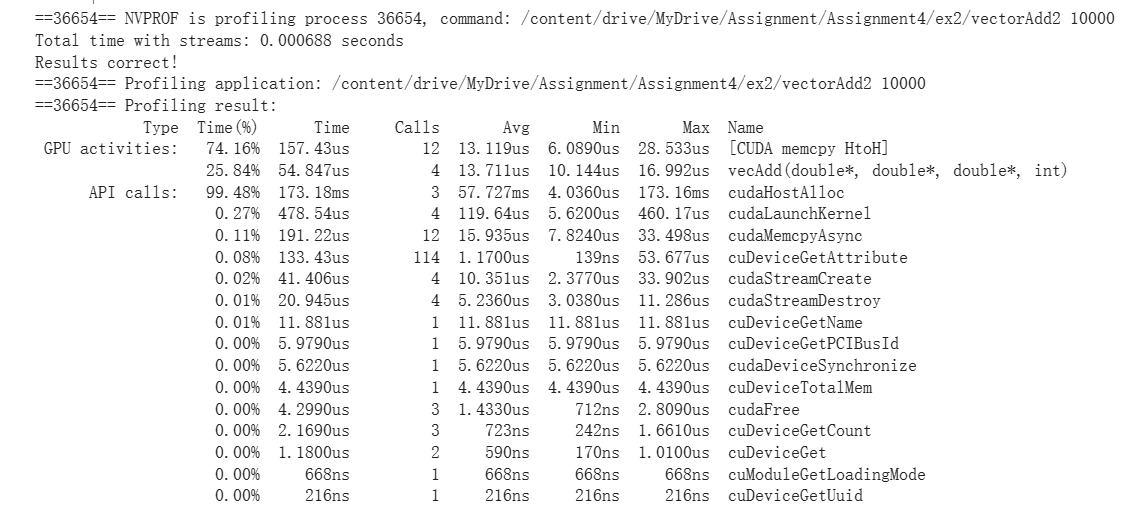
### **Exercise 2 - CUDA Streams**

1.Compared to the non-streamed vector addition, what performance gain do you get? Present in a plot ( you may include comparison at different vector length)

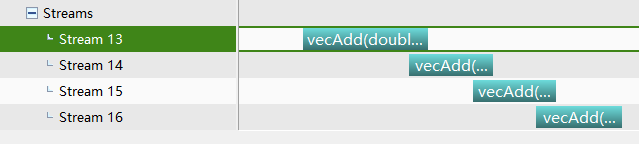
**Non-streamed vector addition**



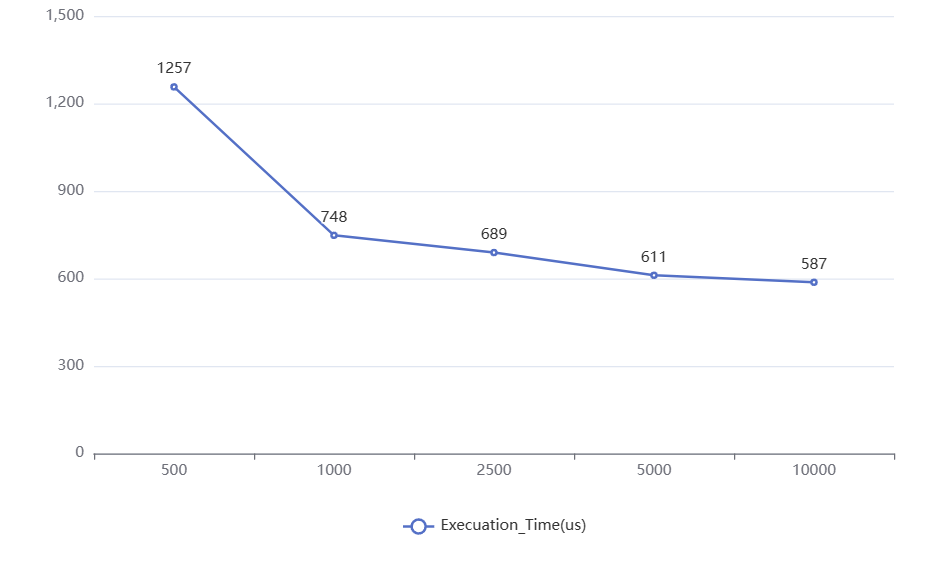
Streamed vector addition



2.Use nvprof to collect traces and the NVIDIA Visual Profiler (nvvp) to visualize the overlap of communication and computation.



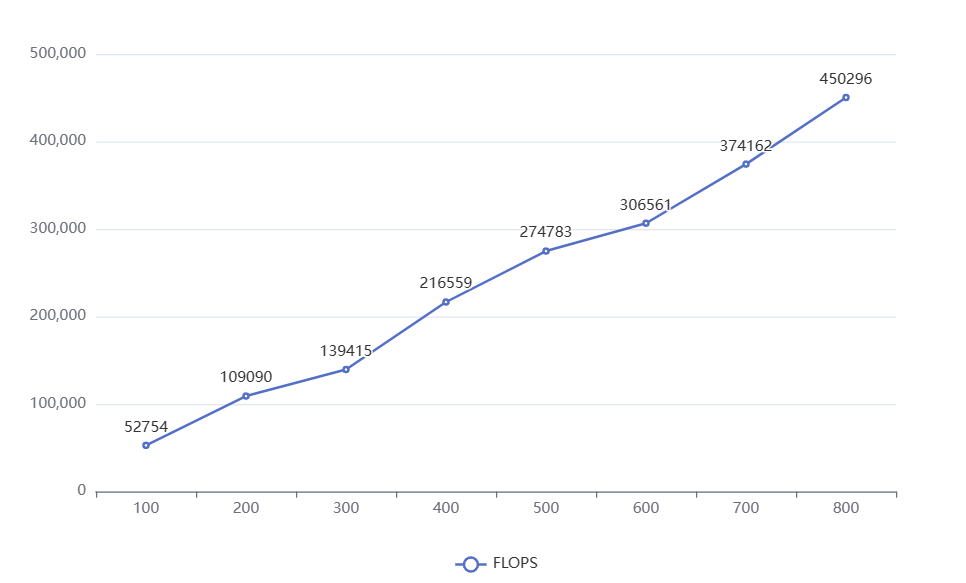
3.What is the impact of segment size on performance? Present in a plot ( you may choose a large vector and compare 4-8 different segment sizes)



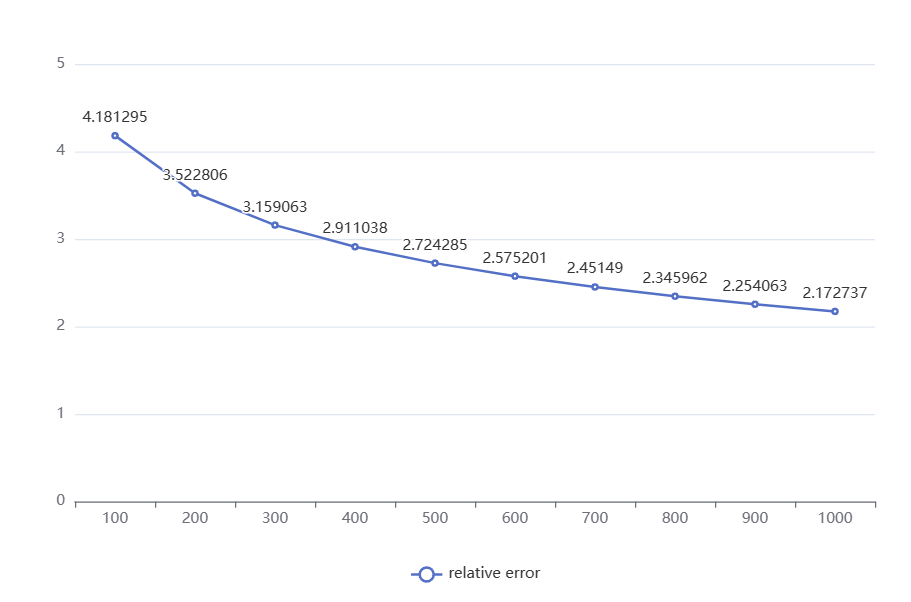
As segment Size increase, the performance of Streamed vector addition is getting better.

### **Exercise 3 - Heat Equation with using NVIDIA libraries**

1.Run the program with different dimX values. For each one, approximate the FLOPS (floating-point operation per second) achieved in computing the SMPV (sparse matrix multiplication). Report FLOPS at different input sizes in a FLOPS. What do you see compared to the peak throughput you report in Lab2?

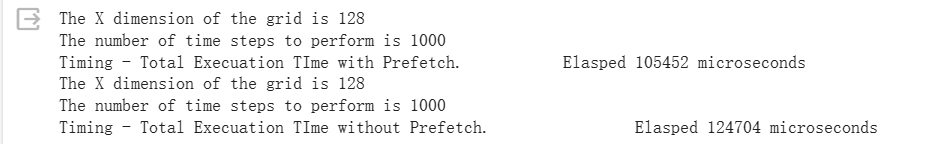


As dimX increases, the size of the matrix and the number of non-zero elements grow, which cases more GPU cores can be used simultaneously and increases FLOPS

2.Run the program with dimX=128 and vary nsteps from 100 to 10000. Plot the relative error of the approximation at different nstep. What do you observe?

As nsteps increases, the relative error decreases, indicating a more accurate approximation with a higher number of steps.

3.Compare the performance with and without the prefetching in Unified Memory. How is the performance impact?



Prefetching in Unified Memory can enhance performance when dimx is not large. However, when dimx become larger, the performance will be reduced.This maybe because prefetching can affect the efficiency of GPU memory and leads to an imbalance in memory bandwidth utilization or a decrease in cache hit rate