Introduction to CUDA and OpenCL
Lab 7 report
30.04.2020

Introduction

This time we were looking at the OpenCL library for computation acceleration. We will be comparing the implementation of two simple programs introduced in CUDA and implement third one comparing its execution time.

Device Query comparison

Device query give us a list of devices on out machine with some of their properties. Here are links to code for the OpenCL and CUDA implementations:

- https://github.com/HandsOnOpenCL/Exercises-Solutions/blob/master/Exercises/Exercise01/Cpp/DeviceInfo.cpp
- https://github.com/HandsOnOpenCL/Exercises-Solutions/blob/master/Exercises/Exercise01/C/DeviceInfo.c
- https://github.com/NVIDIA/cuda-samples/blob/master/Samples/deviceQuery/deviceQuery.cpp

Both of these programs give us information about devices on our machine that can be used, but nvidia gives us more information. It gives us information about texturering and other parameters that are useless for OpenCL and only usable by CUDA. However, OpenCL can give us information and be used on devices other than NVIDIA graphics since it gives us platforms which can mean that other graphics platfor vendors can appear.

Vector addition time and implementation comparison

Here we have implementation of vector addition in OpenCL:

- https://github.com/HandsOnOpenCL/Exercises-Solutions/tree/master/Exercises/Exercise02/C
- https://github.com/HandsOnOpenCL/Exercises-Solutions/tree/master/Exercises/Exercise03/Cpp
- https://github.com/NVIDIA/cuda-samples/blob/master/Samples/vectorAdd_nvrtc/vectorAdd.cpp
  https://github.com/NVIDIA/cuda-samples/blob/master/Samples/vectorAdd_nvrtc/vectorAdd_kernel.cu
- and streams addition from previous labs.

Compared to the CUDA implementation, OpenCl its less readable at first sight. Below we can see the difference in execution times of OpenCL addition of two vectors (C and Cpp implementation) and CUDA sample of vector addition without managed memory and prefetch, with them, and then with streams.

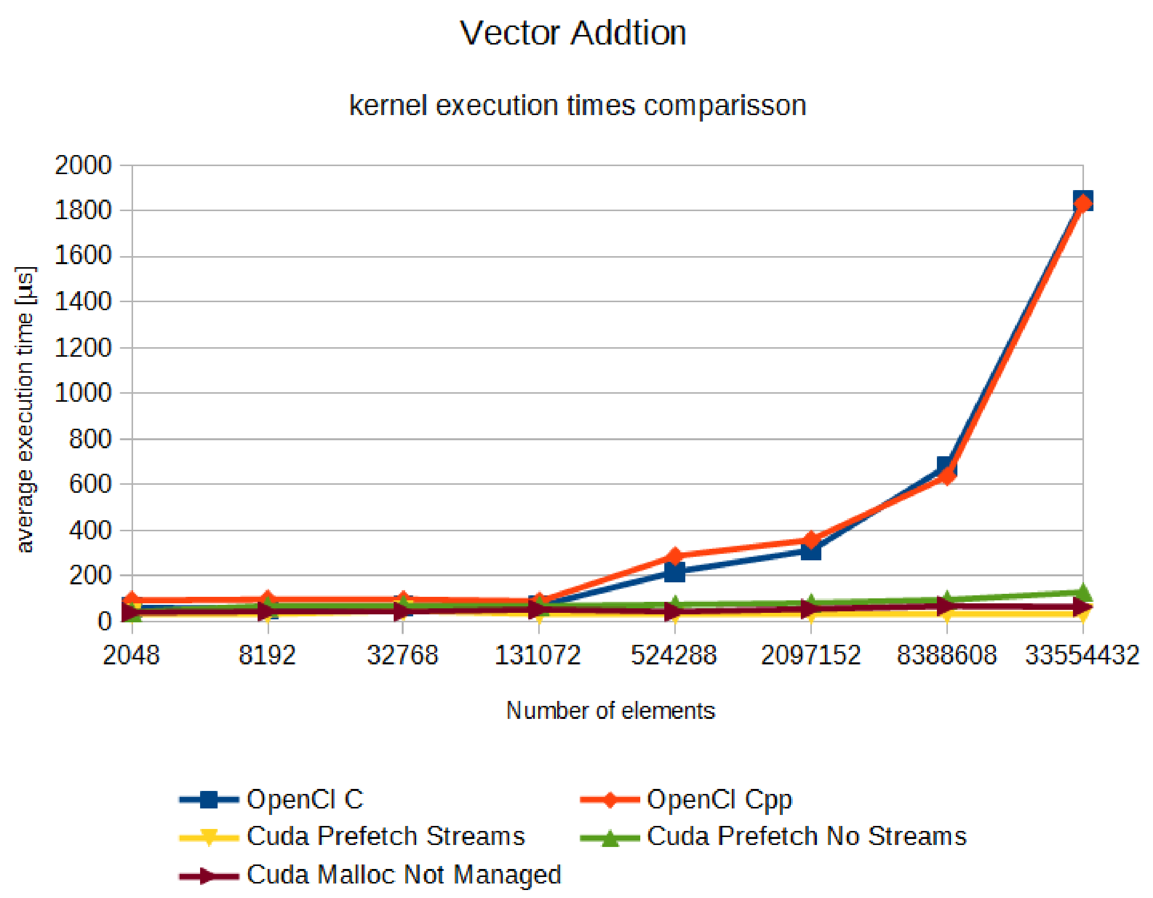

Chart 1. Shows average kernel execution time for vector addition of different number of elements, 8 executions averaged of each number.

In image above we see that the execution of OpenCL kernel for high number of elements is much slower than for CUDA.

Cascade addition of four vectors

In this exercise we were to give our input to the vector addition program use above to make a cascading vector addition program: A+B=C → C+E=D → D+G=F.

The code is included in two file next too this reports. The result of execution of this programs and their counter part in CUDA are given below.

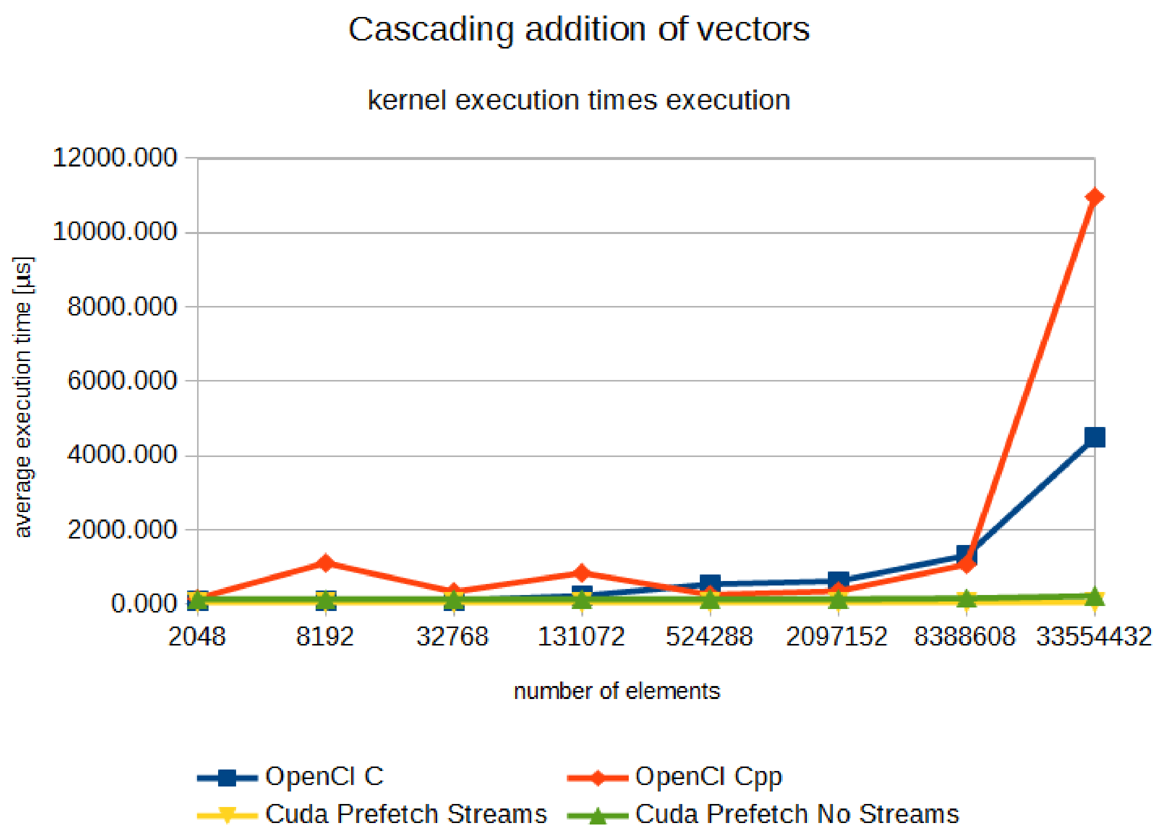

Chart 1. Shows average kernel execution time for chain vector addition of different number of elements, 8 executions averaged of each number.

In image above we see that as before the average execution time of OpenCl program is much slower with higher number of elements.

Conclusion

From the examples that we used, the OpenCl is slower, and less flexible that the CUDA. The code is harder to understand at first glance and the kernels have to read as string which is a detriment. All in all the only plus of the library is that it can be used on any graphics platform with drivers, and doesn't need specialized compiler.