Comparison of CUDA and OpenCL

This document is written as an investigation of the differences between using CUDA or OpenCL to write GPGPU application. The investigation focuses on the differences in performance and ease of use. Finally, CUDA is recommended as a basis for further development.

Performance comparison

In order to compare the performance differences between CUDA and OpenCL, a simple matrix multiplication algorithm was implemented in both CUDA and OpenCL. These implementations where based on examples provided by NVIDIA ans AMD. In order to establish a baseline, to which the CUDA and OpenCL results could be compared, additional implementations of the matrix multiplication algorithm was made, as both a naive serial implementation in C and a highly optimized implementation using the Automatically Tuned Linear Algebra Software (ATLAS) implementation of BLAS. Finally, a highly optimized CUDA implementation was made using the cuBLAS library.

The test algorithm multiplies two square matrices of size NxN. This is an interesting problem to use for performance benchmarking for a number of reasons:

- Matrix multiplication is often used as a subroutine in more advanced mathematical algorithms.
- Matrix multiplication can be parallelized over a large number of computational cores, making it suitable for GPGPU programming.
- The mathematics of matrix multiplication is trivial, making it an easy to understand example problem.

The four implementations where tested on a desktop computer running 64-bit Ubuntu 13.04 with the following hardware:

Type	Hardware
Processor	Intel Core i7-2600K CPU @ 3.40GHz x 8
Graphics	GeForce GTX 560 Ti/PCIe/SSE2
Memory	7.8 GiB

The results are presented in Figure 1:

We see that the naive serial implementation quickly becomes unusable, due to a

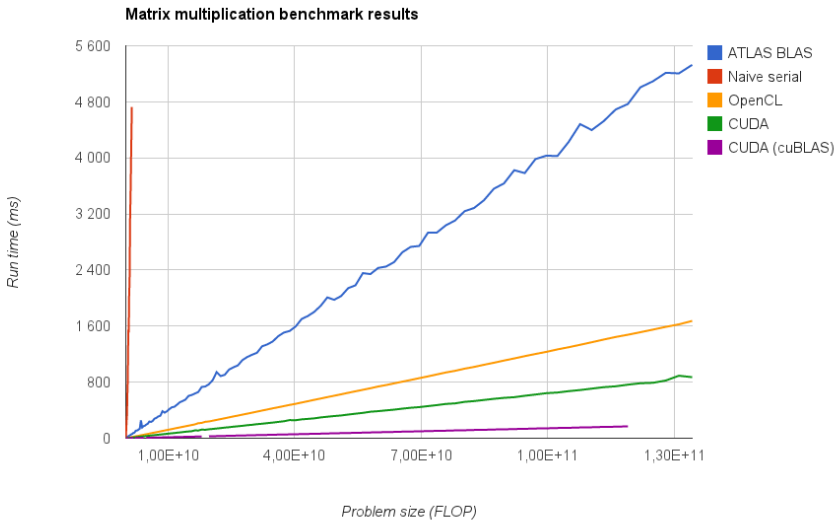

Figure 1: Matrix multiplication benchmark results

rapid increase in run time. The improvement gained by using ATLAS BLAS is very large compared to the naive implementation, although it cannot keep up with the run times achieved by the CUDA and OpenCL implementations.

The difference between CUDA and OpenCL is quite small, compared to the naive and BLAS implementations, but the CUDA implementation is on average about twice as fast as the OpenCL implementation. This is quite a big difference, and this could be related to all tests being run on a NVIDIA graphics card. It might also have been caused by different quality between the NVIDIA and AMD examples.

Looking at the results for the cuBLAS implementation, we can also see the impact of using a highly optimized library for GPGPU programming. The cuBLAS implementation is faster than using the basic CUDA example, indicating that proper use of libraries can be very beneficial.

It is also important to note that this is a very small test. In order to be able to conclude if CUDA is indeed faster than OpenCL, one would have needed to implement a wide selection of algorithms and test them on several different hardware configurations. Although this test is non conclusive regarding this question, the results seem to support several older investigations, concluding that CUDA is faster than OpenCL. One notable example being A Comprehensive Performance Comparison of CUDA and OpenCL by Janbin Fang et al.

Documentation

When we where installing CUDA and OpenCL, and implementing our test algorithms, we where reliant on the online documentation available for the two GPGPU frameworks. Our subjective experience was that finding good documentation for CUDA was a lot easier than for OpenCL. In order to investigate this, we made a series of queries for pages related to CUDA and OpenCL on Google, Google scholar and Stackoverflow.com (a popular programming QA site). The results are shown in the following tables (all data from 16. Jan 2014).

Query	No of Stackoverflow.com results
Tagged OpenCL	1935
Tagged CUDA	6137
Open search OpenCL	5818
Open search CUDA	16174

Query	No of Google results
opencl paralell programming	~322000
cuda paralell programming	~558000
opencl gpgpu	~558000
cuda gpgpu	~816000
opencl programming	~875000
cuda programming	~2790000

Query	No of Google Scholar results
opencl paralell programming	~7480
cuda paralell programming	~17100
opencl gpgpu	~5230
cuda gpgpu	~13500
opencl programming	~8160
cuda programming	~22700

Conclusion

In all our tests CUDA outperformed OpenCL. Although our tests were very limited in scope, they support the opinion that currently, CUDA is faster and better documented than OpenCL. If you don not need the portability offered by OpenCL, we would recommend using CUDA.