Heidelberg University
Institute for Computer Engineering (ZITI)

Master of Science Computer Engineering

GPU Computing

Exercise 4

gpucomp03

Benjamin Maier
Daniel Barley
Laura Nell

Due date December 1st, 09:00

4.1 Reading

Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU

Because of the massive amount of generated data, it is important for throughput computing applications that handle with these data to have a reasonable amount of data prallelism. Such applications can be realized for CPUs or GPUs. Because of their different architectures, they perform different for different throughput computing workloads and have different possibilities for optimizations. Thus, 14 throughput computing kernels with different characteristics are compared based on their execution on the Intel Core i7 CPU and the Nvidia GTX280 GPU.

The performance of the several kernels depends highly on their requirements and the corresponding architecture of the devices like bandwidth limitations, compute flops, cache size or synchronization.
Typically the highest performance is achieved when a suitable number of multiple threads are used per core.
Nevertheless, the performance gap between CPU and GPU can be a lot decreased (down to an average of 2.5X) by indvidual optmizations of software and hardware on both devices. Main optimizations for CPUs leading to performance improvements are multithreading, cache blocking and reorganization of memory accesses for SIMDfication. Optimizations that lead to GPU performance improvements are minimizing global synchronization and using local shared buffers. Besides that, the key hardware architecture features which lead to an imporved performance are identified.

The main characteristics of performance differences still fit for todays architectures, while the overall performance and functionality of CPUs and GPUs has actually improved. Thus, we accept the paper and its content.

NVIDIA Tesla: A Unified Graphics and Computing Architecture

To enable flexible, programmable graphics and high-performance computing, NVIDIA has developed the Tesla scalable unified graphics and parallel computing architecture, which extends the functionality of traditional graphics pipelines. The goal for that architeture was to unify vertex and pixel processors in one architecture.
The developed architecture itself along with the resulting (parallel) workflows is described before an overview of the parallel programming model in C with CUDA tools is given.

There are many innovations in the Tesla architecture, including the execution of scalar instructions or the SIMT processor architecture, which enables executing threads grouped into several warps and increases the efficiency a lot.
Regarding the programming model, the Tesla architecture introduces cooperative thread arrays or CUDA thread blocks to ensure the correct execution of a large number of concurrent threads.
The Tesla architecture was developed for scalability also with large problem sizes. Thus, GPUs using this architecture deliver high performance also for demanding parallel-computing and graphics applications.

Nvidia was one major driver in the development of GPUs and corresponding architectures as highly parallel general purpose computers. Thus, also the Tesla architecture had, at this time, some great new improvements for graphics processing and lead to further innovations. We therefore fully accept the paper and its content.

4.2 Shared Memory Analysis - Basis

4.3 Shared Memory Analysis - Conflicts

4.4 Matrix Multiply - CPU Sequential Version

4.5 Willingness to present

Hereby, we declare our will to present the results presented in the former sections.