View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by GSI Repository

Track finding and fitting on GPUs, first steps toward a software trigger

M. Al-Turany[1], *A. Herten*[2], *and A. Rybalchenko*[1]
[1]GSI, Darmstadt, Germany; [2]FZJ, Juelich, Germany

The graphics processing units (GPUs) have evolved into high performance co-processors that can be easily programmed with common high-level language such as C, Fortran and C++. Todays GPUs greatly outpace CPUs in arithmetic performance and memory bandwidth, making them the ideal co-processor to accelerate a variety of data parallel applications. For the online processing (i.e: Software triggers and online event selection with high data rates as in CBM [1] and PANDA[2] experiments) GPUs present an attractive solution.

Online applications include high level processing, which requires floating point operations. However, the widely used FPGA (Field Programmable Gate Array) does not have such capabilities. The users have to program in the low-level hardware description language (HDL). GPUs, on the contrary, are programmable with high-level languages and meanwhile provide support even for double precision. In order to evaluate GPU solution in a practical way, an algorithm based upon conformal mapping and Hough transform was implemented on GPUs using NVIDIA CUDA (Compute Unified Device Architecture). The algorithm is tested with the PANDA central tracker simulated data. The results of the same algorithm are compared with CPU and FPGA implementations.

To evaluate the run-time and the precision of the algorithm, the GPU implementation was compared against CPU implementation. Used hardware included NVIDIA GeForce GTX 480 GPU and Intel Xeon W3505 CPU clocked at 2.53 GHz. Already with a trivial GPU implementation it was possible to decrease the run-time of the algorithm by over a 100 times. After applying various memory optimization techniques, GPU configuration optimizations and program flow optimizations the performance was further improved, with total run-time decrease of around 200 times. The precision of the results is comparable within 5% to the CPU solution.

	CPU	GPU
tracks segments in first step	1615	1609
After first iteration	158	151
After last iteration	14	14

Table 1: Comparison of number of tracks found for the same event when analyzed on CPU or GPU

Compared to the FPGA, the GPU implementation is about 30% faster. However, one should keep in mind that on both systems we still have a huge space for improvement. With the new release of the CUDA compiler (CUDA 5.0) the same code runs 20% faster on the same hardware. This is an important illustration of the current state of the GPU and GPU compiler technology, which, although already offering promising performance, is still at an early development stage and further improvements are likely to happen.

The newest GPU generation from NVIDIA (so called Kepler architecture [3]) includes various new hardware improvements and allows more functionality on the GPU side. The new features have the potential to further improve the performance of track finding and fitting. The new features (e.g: GPUDirect [4] and ”Dynamic Parallelism” [3]) will have a huge impact on the whole design of the software running on GPUs, moreover, it will releases the host CPU almost completely and the whole fork could be done on the GPU.

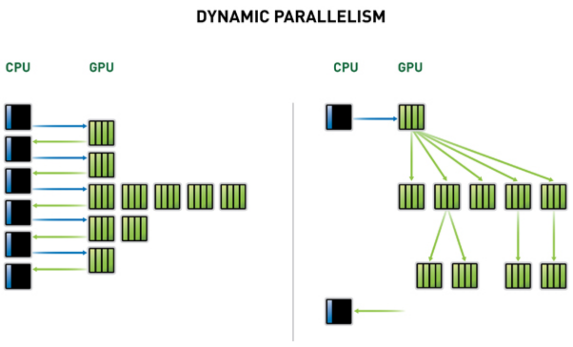

Figure 1: Dynamic Parallelism on Kepler GPU dynamically spawns new threads by adapting to the data without going back to the CPU, greatly simplifying GPU programming and accelerating a broader set of popular algorithms. (From nvidia.com)

As an alternative to the traditional Hough transform, an adaptive approach needs to be considered, which holds the potential to greatly reduce the size of the data to analyze. Although it reduces the data parallelism of the algorithm, it also frees up GPU resources for more task parallelism, which now can be implemented very efficiently with Dynamic Parallelism.

References

[1] https://www.gsi.de/work/forschung/cbmnqm/cbm.htm
[2] http://www-panda.gsi.de/
[3] http://www.nvidia.com/object/nvidia-kepler.html
[4] https://developer.nvidia.com/gpudirect