PROCEEDINGS OF SCIENCE

The GAP Project - GPU for Realtime Applications in High Level Trigger and Medical Imaging

Marco Corvo, Giovanni Di Domenico, Massimiliano Fiorini[*†]

Università degli Studi di Ferrara and INFN Sezione di Ferrara, Italy

Gianluca Lamanna, Jacopo Pinzino, Marco Sozzi

INFN Sezione di Pisa, Italy

Matteo Bauce, Silvia Capuani, Stefano Giagu, Marco Rescigno, Marco Palombo

INFN Sezione di Roma and Università di Roma "La Sapienza", Italy

Andrea Messina

Università di Roma "La Sapienza", Italy, and CERN, Switzerland

The paper describes the GAP Project, whose objective is the deployment of Graphic Processing Units (GPUs) in real-time applications, ranging from trigger selection in high energy physics experiments to medical imaging reconstruction. The final goal of the project is to demonstrate that GPUs have a positive impact in applications that differ for rate, bandwidth, and computational demand, but have a common approach of solving complex problems in real-time using parallel architectures. The relevant aspects under study are the analysis of the system latency, the optimisation of the computational algorithms and the integration with data acquisition systems. As a benchmark application for high-level trigger algorithms in HEP experiments we consider the ATLAS Muon trigger case. In particular we discuss how specific algorithms can be parallelised and thus benefit from the implementation on the GPU architecture, in terms of increased execution speed and more favourable dependency on the complexity of the analysed events. Such improvements are particularly relevant for the foreseen LHC luminosity upgrade where highly selective algorithms will be crucial to maintain a sustainable trigger rate with the many multiple pp interactions per bunch crossing. GPUs can provide a feasible solution also to accelerate the reconstruction of medical images. We discuss the implementation of new computational intense algorithms boosting the performances of Nuclear Magnetic Resonance and Computed Tomography. The deployment of GPUs can significantly reduce the processing time, making it suitable for the use in realtime diagnostic.

Technology and Instrumentation in Particle Physics 2014,
2-6 June, 2014
Amsterdam, the Netherlands

[*]Speaker.
[†]E-mail: fiorini@fe.infn.it

PoS(TIPP2014)422

1. The Use of GPUs in Trigger Systems for High Energy Physics Experiments: the ATLAS Muon Trigger case

High Level Trigger (HLT) systems, in particular those of LHC experiments, offer a complex environment in terms of rate, bandwidth and latency. Typical values are latencies from ms up to seconds, input rates of few hundred Hz, and bandwidths of hundreds of GB/s. HLT systems are nowadays implemented as customized software algorithms executed on farms of commodity PCs. The LHC upgrade with the consequent increase of luminosity and pile-up, poses new challenges for the HLT systems in terms of rates, bandwidth and signal selectivity. To exploit more complex algorithms aimed at better performances, higher computing capabilities and new strategies are required. Moreover, given the tendency of the computing industry to move away from the current CPU model towards architectures with high numbers of small cores well suited for vectorial computation, it is becoming urgent to investigate the possibility to implement higher level of parallelism in the HLT software. The GAP project is studying the deployment of GPUs for the HLT in LHC experiments. Specifically, we are focusing on the muon HLT of the ATLAS experiment at CERN [1]. The ATLAS trigger system is organised in 3 levels [2]. The first-level trigger is built on custom electronics, while the second-level and the event-filter are implemented in software algorithms executed by a farm PCs. For the forthcoming Run II of LHC, the second-level and the event-filter will be merged in a single HLT stage.

We are exploring the potential improvements attainable in the near future by deploying GPUs in the ATLAS muon trigger algorithms. The present algorithms are implemented as approximated solutions of complex primitives; the high computing capabilities of GPUs would allow the use of refined algorithms with higher selection efficiency, and thus to maintain the sensitivity to interesting physics signals even at higher luminosity.

To exploit GPUs to execute selection algorithms at the HLT aiming at offline-like performance requires to address two major points. The first one regards how to integrate a new computing device, with its own software libraries, in a complex framework as the ATLAS trigger infrastructure is. The second one has to do with the identification of new, highly parallel algorithms well suited to exploit the GPUs architecture.

To interact with the ATLAS online software we decided to use a client-server structure called *APE* (Accelerator Process Experiment) [3]. Such an infrastructure is being developed within the ATLAS experiment precisely to allow the integration of new devices and independent software kernels with the ATLAS software. It works by exchanging messages and data structures between the client and the server. The client-server structure of the *APE* interface is particularly well suited for our application because it make the development of the new algorithms to be executed on the GPUs completely independent of the ATLAS software infrastructure, moreover it can be easily expanded to several devices. The requirements that the *APE* interface needs to satisfy are: a) the ability to perform server-client communications and data access with a limited and well defined latency such as not to introduce a sizeable overhead latency if compared with the trigger latency requirement; b) be easy to interface with the ATLAS software.

As a first study case, we have taken the trigger algorithm that computed the calorimetric isolation of a given track, later referred to as *muIso*. The algorithm is intended to increase the purity of the muon trigger candidates by requiring that the energy released in the calorimeter in a given

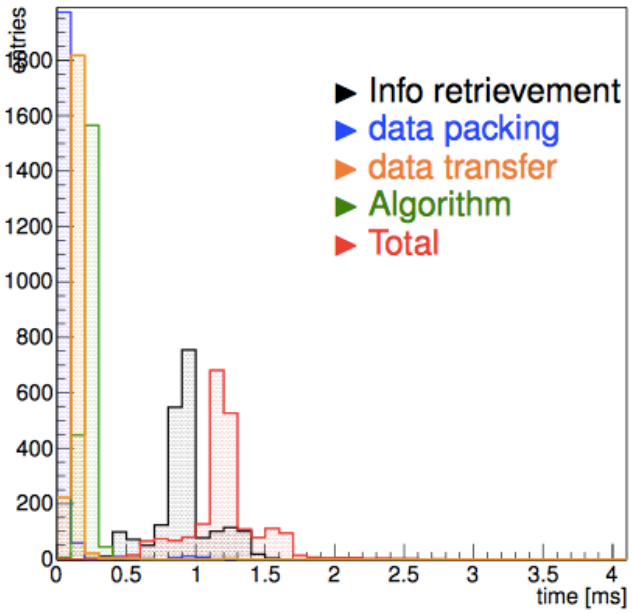

Figure 1: Time sharing of the different steps necessary to execute the *muIso* algorithm on a GPU using the *APE* interface

cone around the muon direction is below a given threshold, typically of the order of a GeV. The algorithm itself is rather simple and proceeds as follow: for a give track direction a cone in the $\eta - \phi$ plane is defined and a loop is performed on the calorimeter cells within the cone, the total energy is then computed and compared with the threshold.

Figure 1 shows the time sharing of the different steps necessary to perform the algorithm. This test shows that a typical muon trigger algorithm can successfully be implemented on GPUs deploying the *APE* interface by adding an overhead time of about 0.3 ms necessary for the operation previously listed as 2, 3, 5. This is a small fraction of the total time necessary to executed the *muIso* algorithm of about 1.2 ms. Having demonstrated that a standard ATLAS trigger algorithm can be effectively executed outside the ATLAS trigger infrastructure without significant latency degradation allows to implement more efficient, and thus computing intense, algorithms tailored on the GPUs parallel architecture.

2. The Use of GPUs for Imaging Reconstruction

2.1 Reconstruction of NMR images

In the reconstruction of NMR images, we are working on the implementation of a parallel algorithm for post-processing and reconstruction of brain NMR diffusion weighted (DW) images by using the kurtosis method [4]. The peculiar anomalous diffusion contrast due to biological water in brain tissues provides more specific and complementary information than conventional DW maps (based on a Gaussian diffusion model) for the diagnosis of several brain pathologies.

Specifically, a complex non-linear relation describing the DW-MR signal attenuation has to be fitted to experimental dataset voxel-by-voxel by using the Levenberg-Marquardt algorithm. This

algorithm is based on an iterative numerical optimisation procedure that minimises the sum of squared model residuals. Currently this kind of reconstruction is so expensive in terms of hardware and time that is still far from being used in medical diagnosis, even if has been recently recognised its enormous diagnostic power.

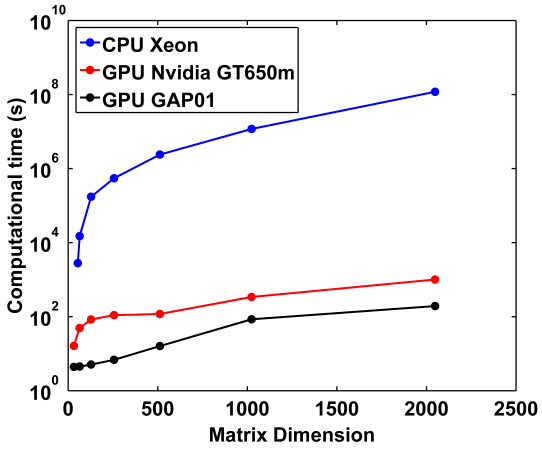

Figure 2: The figure shows the performance attainable as a function of the number of pixels for a simulated NMR image.

The reconstruction of a typical brain scan implies the solution of about 10^7 equations. The image is in fact usually subdivide in 128×128 pixels and 10-40 slices with at least 15 diffusion gradient directions. For the case of the non linear fit to the kurtosis tensor, the reconstruction time on a conventional CPU increases dramatically up to several hours. To estimate the potential speed-up factor if the algorithm would be executed by a GPU, we have performed studies on simulated images. Figure 2 reports the total computing time to reconstruct a realistic image on a Intel Xeon CPU, on a Nvidia GT650m and on a Nvidia GTX TITAN (GAP01) GPUs showing that a significant improvement can be attained by running on GPUs.

2.2 3D cone-beam CT reconstruction algorithm on GPU

Fast 3D cone-beam reconstruction is required by many application fields like medical CT imaging or industrial non-destructive testing. For that reason, researchers work on hardware optimised 3D reconstruction algorithms. The GAP research group has implemented a Feldkamp-Davis-Kress (FDK) algorithm for cone-beam CT (CBCT) on GPU by using NVIDIA CUDA SDK version 5.5. The FDK algorithm is described by the formula:

$$f(x,y,z) = \frac{1}{2}\Sigma_{n=1}^{N}\left[\frac{1}{U_n^2(x \times cos\theta_n + y \times \sin\theta_n, z)} \times V(u,v) \times g_n(u,v) * h(u)\right]$$

where $g_n(u,v)$ is the acquired projection, $V(u,v)$ represents the cosine weighting function, $h(u)$ is the ramp filter and $U_n(u,v)$ represents the back-projection weighting function. We have divided our implementation into three steps: the cosine weighting, the convolution with ramp filter and the weighted back-projection, so to optimise each single step in CUDA. The implemented algorithm

has been tested on 256^3 and 512^3 datasets for two different GPUs: a GTX-680 with 4 GB and a GTX-Titan with 5 GB, both with Kepler architecture. The results for the execution time are reported in table 1.

Step	256^3		512^3	
	GTX-680	**GTX-Titan**	**GTX-680**	**GTX-Titan**
Cosine weighting	1.5	1.4	4.0	3.9
Ramp filter	15.0	13.5	56.4	51.2
Weighted backprojection	4.4	3.3	12.5	8.4
Total	20.9	24.2	72.9	63.5

Table 1: Execution time (in seconds) for FDK algorithm on two NVIDIA GPUs

3. Conclusion

We have presented preliminary results on novel applications of GPUs in a real-time environment attained by the GAP group. We discussed two main domains of interest: the trigger systems for HEP experiments and the reconstruction of medical images. Concerning HEP trigger systems, we described the implementation of a muon trigger algorithm on a GPU for the ATLAS experiment by using the *APE* interface based on a client-server structure. We showed that the exchange of information between the ATLAS trigger infrastructure and the GPU does not introduce significant latency and thus it opens the road to more computing intense and selective algorithms to be implemented on GPUs. Finally, we discussed the impact of deploying GPUs for the reconstruction of medical images. In particular, we have presented the NMR images obtained for a specific reconstruction algorithm based on a nonlinear analysis of the kurtosis tensor for the diffusion attenuation and the timing results of CUDA implementation of FDK algorithm on CT datasets.

Acknowledgment

The GAP project is partially supported by MIUR under grant RBFR12JF2Z “Futuro in ricerca 2012”.

References

[1] The Atlas Collaboration, JINST **3** (2008) S08003.

[2] The Atlas Collaboration, JINST **3** (2008) P08003.

[3] APE - Accelerator Process Experiment, Atlas Software and Computing Workshop, Oct 2013

[4] J.H. Jensen, J.A. Helpern, NMR Biomed; **23** (7): 698-710, 2010.