Romanian Reports in Physics, Vol. 63, No. 4, P. 1018–1025, 2011

ELEMENTARY PARTICLE PHYSICS

DATA ANALYSIS AND 3D EVOLUTION IN HIGH ENERGY PHYSICS USING GRAPHIC PROCESSOR[*]

ALEXANDRU JIPA[1], CIPRIAN-MIHAI MITU[2], MIHAI NICULESCU[2], SORIN-ION ZGURA[2]

[1] University of Bucharest, Faculty of Physics, P.O.Box MG-11, RO 077125 Bucharest-Magurele, Romania
E-mail: jipa@brahms.fizica.unibuc.ro
[2] Institute of Space Science, Magurele, P.O.Box MG-23, RO 077125, Bucharest-Magurele, Romania,
E-mail: cmitu@spacescience.ro; mihai@spacescience.ro; szgura@spacescience.ro

Received November 4, 2010

Abstract. One of the main challenges in High Energy Physics (HEP) is to make fast analysis of high amount of experimental and simulated data. For example, the amount of data generated at Large Hadron Collider (LHC) is estimated to reach 1 PetaByte/year. The time taken to analyze the data and to obtain fast results depends on high computing power. We present a new approach in data analysis from HEP using Graphics Processor Units (GPU). A graphical user interface application was developed for data analysis and visualization from Ultra relativistic Quantum Molecular Dynamics (UrQMD) simulated data. The increase GPU performance makes possible both the online analysis of data and also the real-time visualization of the interaction in 3D.

Key words: nuclear physics, UrQMD, GPU, data analysis, anisotropic flow, parallel computing.

1. INTRODUCTION

The most important goal of the heavy-ion physics in the ultra-relativistic energy domain is the search for the phase transition between hadronic matter and the quark-gluon plasma (QGP). This new state of matter is believed to be created at approximate 1 microsecond after the Big Bang. To describe this state of nuclear matter it is necessary to analyze a huge volume of information of physical observables characterizing various emitted particles in ultra relativistic nuclear collisions.

The time taken to analyze the data and to obtain fast and reliable results depends on high performance computing. Most common solution to solve this problem is to use large arrays of computer clusters such GRID or GRID-like computing solutions or with local parallel computing facilities.

[*] Paper presented at the National Conference on Physics, 23–25 September 2010, Iaşi, Romania.

During the last a new technology has emerge in the parallel processing field. This technology allows the processing data to be done on graphic processor.

In this paper, we present an application that computes several interesting observables for heavy-ion physics using the GPUs.

2. ANISOTROPIC FLOW IN HEAVY-ION COLLISIONS

One of the signals of QGP formation is the collective flow which develops in heavy ion collisions as a result of the almond shape of the participant region in the transverse plane in peripheral nucleus-nucleus collisions (Fig.1).

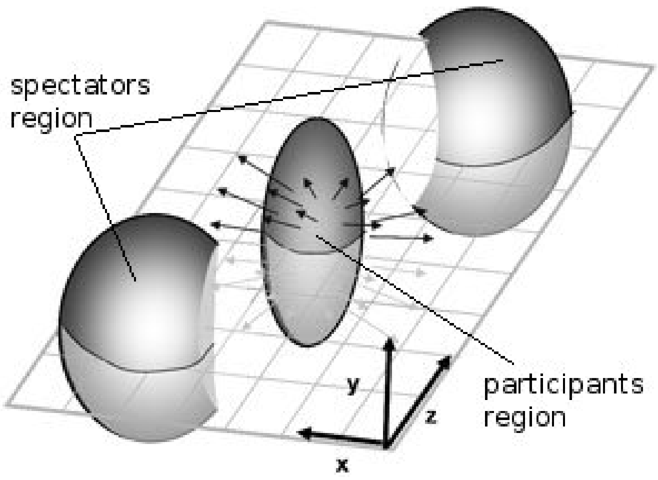


Fig. 1 – Geometry of the nucleus-nucleus interaction.

Direct and elliptic flow is defined as the first and second harmonic coefficient $v1$ respectively $v2$ of an azimuthal Fourier expansion of the azimuthal distribution of produced particles with respect to the reaction plane and carry information on the early stages of the collision. [1]

$$\frac{d^3N}{p_T dp_T dy}(p_T, y, \Phi) = \frac{1}{2\Pi}\frac{d^2N}{p_T dp_T dy}[1+2v_1(p_T,y)\cos(\Phi)+2v_2(p_T,y)\cos(2\Phi)+\ldots], \quad (1)$$

where φ is the azimuthal angle between the transverse momentum of the particle and the reaction plane and p_T and y is the transverse momentum and the rapidity, respectively. The first harmonic or direct flow is defined as:

$$v_1 = <\cos\Phi> = <\frac{p_x}{p_T}>. \quad (2)$$

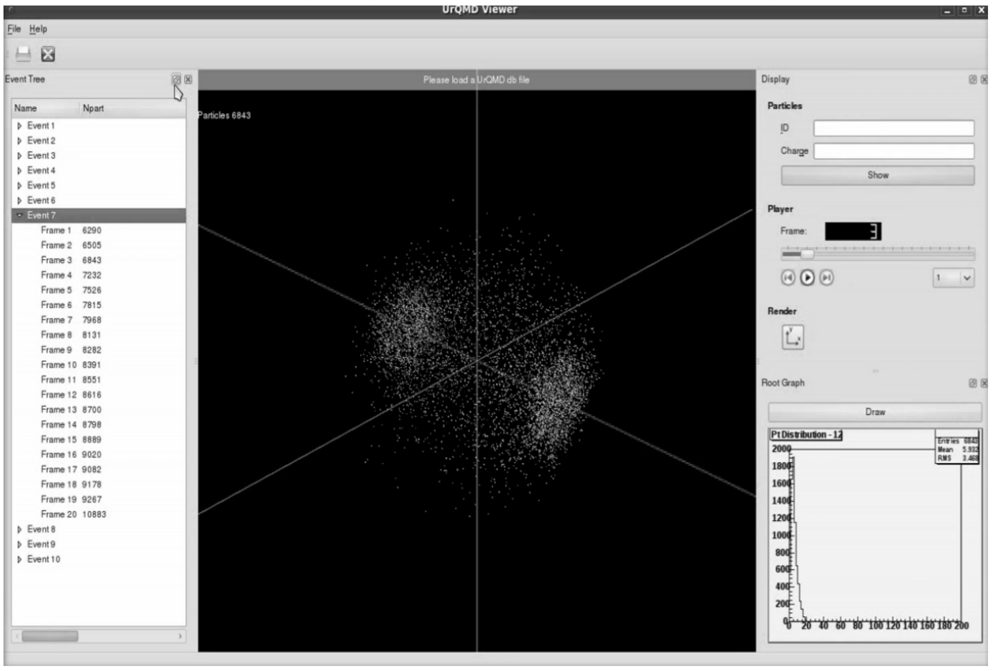
The second harmonic or elliptic flow measures the eccentricity of the particle distribution in the momentum space:

$$v_2 = < \cos 2\Phi > = < \frac{p_x^2 - p_y^2}{p_T^2} >. \tag{3}$$

In this coordinate system, z axis is along the beam and impact parameter b is calculated along the x axis, making $p_T = \sqrt{p_x^2 + p_y^2}$.

3. APPLICATION OVERVIEW

We developed an application which allows the user to visualize in 3 dimensions (3D) the particles generated by UrQMD [2] (Ultra relativistic Quantum Molecular Dynamics) simulations and run a variety of algorithms on the simulation data. The increase GPU (Graphical Processor Unit) performance makes possible both the analysis of data and also the real-time visualization of the interaction.



Name	Npart
Event 1	
Event 2	
Event 3	
Event 4	
Event 5	
Event 6	
Event 7	
Frame 1	6290
Frame 2	6505
Frame 3	6843
Frame 4	7232
Frame 5	7526
Frame 6	7815
Frame 7	7968
Frame 8	8131
Frame 9	8282
Frame 10	8391
Frame 11	8551
Frame 12	8616
Frame 13	8700
Frame 14	8798
Frame 15	8889
Frame 16	9020
Frame 17	9082
Frame 18	9178
Frame 19	9267
Frame 20	10883
Event 8	
Event 9	
Event 10	

Fig. 2 – Application main window. On left side is the event tree. On center the 3D visualization of the interaction (Au-Au at sqrt (s_{NN})=200GeV). On right side is the evolution P_T distribution over all particles in real-time. Also, at the top right side is the particle filter.

As a front-end to the user, the application uses Qt [3] libraries to create the GUI (graphical user interface library) and user interaction (Fig.2). The input data, obtained from a SQLite3 [4] database file, is fetched and loaded on the graphic device. After this, the data is processed on GPU using CUDA (Compute Unified Device Architecture) [5]. The processed data is output in a OpenGL [6] (Open Graphics Library) viewport and in ROOT's [7] graphs or histograms. Using widgets, one can select/filter events, frames or particles that are going to be processed (Fig. 3). Also, another feature is to render all frames in an event continually, such that one sees the evolution of particles as a fluid animation. Just like movie player, with buttons to play/pause or go forward, backward.

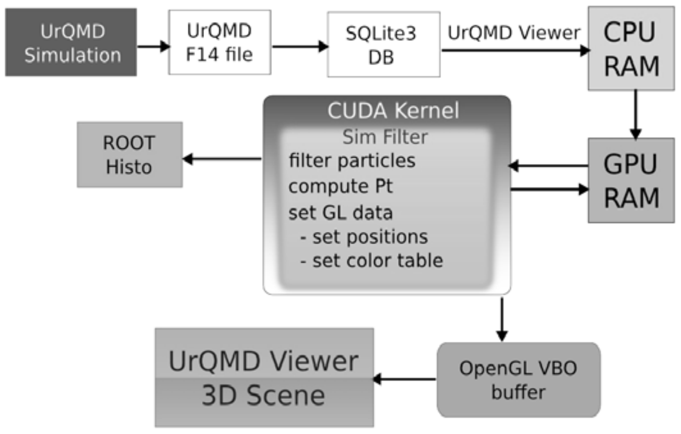


Fig. 3 – The general program flow.

3.1. GRAPHICAL USER INTERFACE

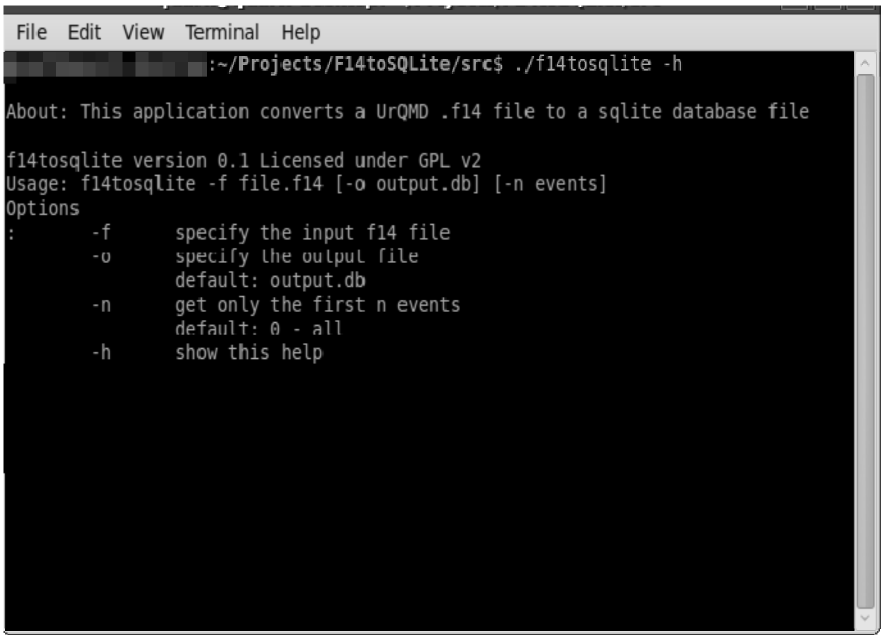
We wanted the application to be user-friendly, thus we choose a GUI – Qt to create our application. Qt is a C++ cross-platform application and UI framework. It's libraries came helpful in more ways than just the GUI. Qt libraries are very well documented and provide easy access for database files and OpenGL.

3.2. INPUT DATA

The simulated data from UrQMD is not well organized, meaning the files generated doesn't allow a great control over the data. One can not fetch the interest data, for example, from a specific event or a time frame from the evolution of the interaction without going through all the previous events, frames or particles. Our answer, was to transform this generated file into the most obvious solution, a database. More precisely, a SQLite3 database. SQLite is a SQL (Structured Query Language) database engine that does not require a server to be run on.

A program was written to transform the generated file into a SQLite3 database file, *f14tosqlite*. After running the program over an .f14 file, one obtains a database structured in tables: *events, frames, particles*. All data from f14 file is contained in these tables. One can read this database with any software that is capable of reading SQLite3 files.

The usage is simple, entering f14tosqlite -h gives (Fig.4)



```
[redacted]:~/Projects/F14toSQLite/src$ ./f14tosqlite -h

About: This application converts a UrQMD .f14 file to a sqlite database file

f14tosqlite version 0.1 Licensed under GPL v2
Usage: f14tosqlite -f file.f14 [-o output.db] [-n events]
Options
:       -f      specify the input f14 file
        -o      specify the output file
                default: output.db
        -n      get only the first n events
                default: 0 - all
        -h      show this help
```

Fig. 4 – Output from f14tosqlite.

Using Qt's SQL classes SQLDatabase, SQLQuery it is easy to interrogate the database and fetch only the interested data.

3.3. PROCESSING DATA

When one wants fast analysis and real-time animation, one chooses parallel processing. Having this in mind, we put move all processing on the best and cheapest parallel unit one can afford: GPU parallel processing. A recent technology developed by NVIDIA named CUDA allows one to harness the power of the GPU device using parallel processing.

The programming execution model on GPU is SIMT (Single Instruction Multiple Thread) and is described in Fig. 5. This means, only one function (running in multiple threads or instances) can run on the GPU at one time processing the data. In CUDA terminology, this function is named a kernel.

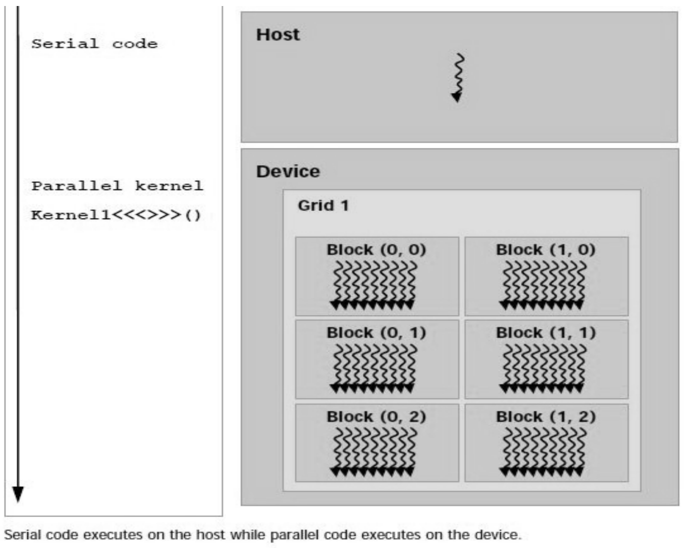


Fig. 5 – Execution model on GPU is SIMT (Single Instruction Multiple Threads).

After querying database for the interested data, and loading it in GPU, depending on the one's action, a CUDA kernel is launched accordingly. The kernel above, takes in particles information (x,y,z,px,py,pz,E,...) and outputs the computed values of: transverse momentum, direct and elliptic flow parameters, pseudo-rapidity and OpenGL's necessary data to render the particles.

The representation of the processed data can be seen in OpenGL viewport (particles as spheres) and parameters in ROOTs graphs and histograms.

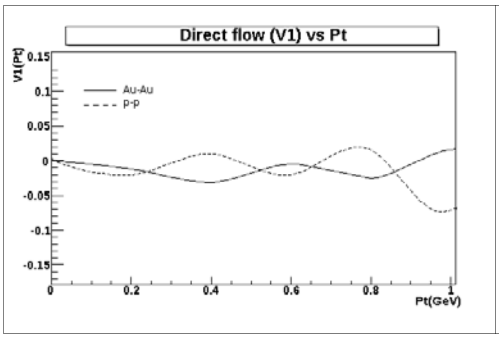
4. RESULTS

4.1. SIMULATION AND DATA ANALYSIS

We have simulated 1000 p-p events and 1000 Au-Au events at $\sqrt{s_{NN}}$ =200 GeV with UrQMD Monte Carlo simulation code. The impact parameter for Au-Au collisions varied from 0 fm to 12 fm. For Au-Au interaction we used the hydrodynamics model implemented in UrQMD. Data analysis of v_1 and v_2 was made in the pseudo-rapidity range [–2, 2] (Fig.6).

$$\eta = \frac{1}{2}\ln\left(\frac{E + p_z}{E - p_z}\right), \tag{4}$$

where η – pseudorapidity, E – total particle energy, p_z – particle momentum along z axis.



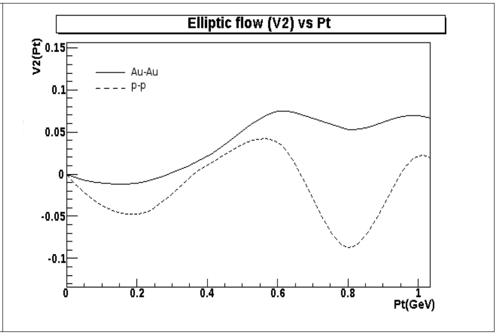


Fig. 6 – Direct flow and elliptic flow vs Pt for pp and AuAu interaction (impact parameter 6 fm < b < 9 fm) for charged particles.

For each event we have simulated time evolution of the interaction between time interval 0 and 200 fm/c. Reconstruction of the direct and elliptic flow was done for p_T<1 GeV/c because of the good agreement between experimental data and UrQMD simulations [8].

The obtained v1 and v2 are found to have large errors at higher P_T especially due to low statistics.

The shape of v1 is of special interest because it has been identified as a possible QGP signature [9]. It exhibits a characteristic wiggle, whereby directed flow changes sign at mid-rapidity.

The observed v2 depends on the amount of rescatterings during the expansion of the fireball. Hydrodynamics assumes zero mean free path, or equivalently, infinitely strong rescattering. Therefore, it provides practical upper limits for the anisotropy of particle distribution [10].

4.2. PERFORMANCE TESTS

The data analysis was done on a NVIDIA graphic board GeForce 9600 GT (8 processors) with 1GB DRAM at 900MHz and 64 cores at 1.6GHz. A comparative test was done on cluster of 16 CPUs CORE 2 DUO at 2.2GHz and 2GB RAM per core. The benchmark reports are presented in Table 1.

Table 1

The comparative benchmarked interactions

Interaction	GPU (T(ms)/MB)	CPU (T(ms)/MB)
p-p	30	4
Au-Au	37	6

5. CONCLUSIONS

In this paper, we presented a new way to analyze data from UrQMD simulations. We performed analysis on anisotropic flow in pp and AuAu interaction at $\sqrt{s_{NN}} = 200$ GeV.

The application we developed allows one to make online analysis and see in real time the way parameters change. The possibility to visualize the 3D evolution of the nuclear interaction may prove as an useful tool in education.

The benchmark tests show that a single graphic board is close in performance with a cluster of CPUs at a much lower maintenance and price.

Because the parallel programming model on GPU is very different than on CPU, this approach makes the implementation of algorithms from heavy-ions a slow process.

Acknowledgements. This work was supported in part by UEFISCDI Romania, Project Registration Code PN-II-RU-TE-2011-3-0279 and the European Commision under EU FP7 project HP-SEE (under contract number 261499).

REFERENCES

1. A.M. Poskanzer and S.A. Voloshin, Phys. Rev., **C 58**, 1671 (1998).
2. H. Petersen, J. Steinheimer, G. Burau, M. Bleicher and H. Stöcker: Phys. Rev., **C 78**, 044901 (2008).
3. *** http://qt.nokia.com/
4. *** http://www.sqlite.org/
5. *** http://www.nvidia.com/object/cuda_home_new.html
6. *** http://www.opengl.org/
7. Rene Brun and Fons Rademakers, Nucl. Inst. & Meth. in Phys. Res., **A 389**, 81–86 (1997).
8. L.V. Bravina, K. Tywoniuk, E.E. Zabrodin, G. Burau, J. Bleibel, C. Fuchs, Amand Faessler, Physics Letters, **B 631**, 109–117 (2005).
9. Gang Wang, Nuclear Physics, **A 774**, 515–518 (2006).
10. P. Huovinen, P.F. Kolb, U. Heinz, arXiv:nucl-th/0104020v1.