EPJ Web of Conferences will be set by the publisher
DOI: will be set by the publisher

Speeding Up and Parallelizing the GARFIELD++

A. Sheharyar[1,a], O. Bouhali[1], and A. Castaneda[1]

[1] *Texas A&M University at Qatar*

Abstract. Garfield++ is a toolkit for the simulation of particle detectors that use gas and semi-conductors as sensitive medium. It takes enormous amount of time to complete the simulation of complex scenarios such as those involving high detector voltages, gases with large gains, or electric field meshes with large number of elements. We observed that most of the simulation time is being consumed in finding the correct element in the electric field mesh. We optimized the element search operation and achieved significant boost in the speed up. In addition, We added the parallel computing support in the toolkit to simulate multiple events simultaneously over multiple machines. In this paper, we present our approach of speeding up the computations and benchmark results.

1 Introduction

The use of Micro-Pattern Gas Detectors (MPGDs) for both applied and fundamental science has substantially increased during the last few years. For instance in the field of High Energy Physics (HEP) in which there is a constant demand for faster detection devices, MPGDs have become an excellent candidate for this task. In addition to a fast signal response, MPGDs provide high rate capability, unprecedented spatial resolution, large sensitivity area, operational stability and radiation hardness. The design of a MPGD can vary according to the experimental needs, two of the most popular ones are the Gas Electron Multiplier (GEM) [1] and the Micro-Mesh Gaseous Structure (MICROMEGAS) [2]. The role of MPGDs in HEP experiments is fundamental, as an example, the CMS experiment at the CERN Large Hadron Collider (LHC) is preparing for an upgrade of its muon detection system with the installation of new chambers based on GEM technology [3], given the large scale of such experiment, designing and producing large area gas detectors to preserve the desirable performance remains a challenge. Due to several experimental factors the performance of MPGDs can degrade, some of those include: changes in the detector geometry due to imperfections in the production or produced during the installation or operation, the choice of the sensitive material (which commonly consist on the mixture of two gases, one of them being a noble gas), the corresponding high voltage applied between the detector layers, among others. It can be costly and not feasible to study each one of these parameters in-situ, this is why Monte Carlo simulation represents a powerful tool. In order to study these parameters GARFIELD++ is used, which is the C++ version of the GARFIELD toolkit [4], this software can handle the simulation of detectors that use a gas mixture or a semi-conductor material as a sensitive medium.

1.1 Signal formation in a GEM detector

A GEM is a gaseous ionization detector, commonly used in particle physics. In the operation of a GEM detector a high energy particle (i.e. muons produced in the proton-proton collisions at the LHC) traverse the detector volume, due to interaction (collisions) with atoms and molecules in the sensitive medium (gas mixture) an electron is released via ionizing radiation, such electron is accelerated due to the presence of an uniform electric field, the electron is guided from the "drift" to the "amplification" region which consists of small semi-conical holes in a thin polymer sheet where an even more intense electric field is created, due to further interaction with other atoms and molecules more electrons are released producing an "avalanche" effect. Electrons in this avalanche are collected in the readout just after passing the amplification region (in the case of a single-GEM configuration) or are further amplified in a sequence of additional drift and amplification regions (which is the case for a triple-GEM detector configuration). Figure 1 (left) shows a typical electron avalanche simulated in GARFIELD++ using a triple-GEM detector configuration.

The generated electric field is modeled by an external package (ANSYS) which is a finite element solver. The 3D electric field modeling is described by a collection of tetrahedral elements building a mesh as it is shown in Figure 1 (right). The size of the elements in the mesh is a configurable in the simulation and it has a direct impact on the accuracy of the results and the time of the simulation, the finer the size of the elements the longer the simulation time, this fact combined with the complexity of the processes described in the simulation may lead to a total

[a] e-mail: ali.sheharyar@qatar.tamu.edu

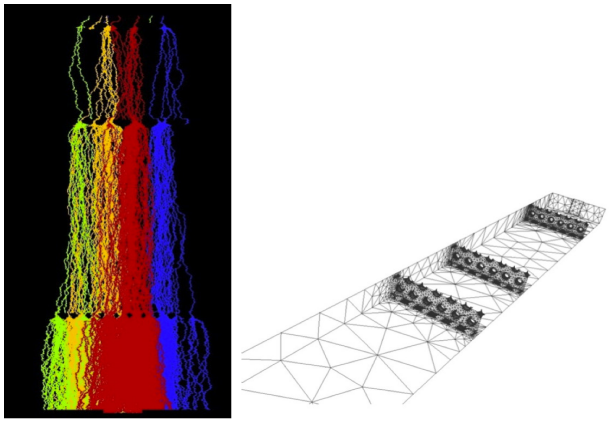

Figure 1. Left: Visualization of the simulated avalanche development for seven primary electrons in a triple-GEM chamber starting from the drift volume. Right: Electric field mesh generated by ANSYS for a triple-GEM detector

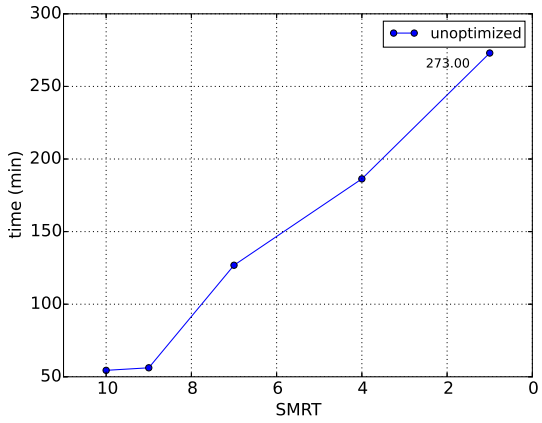

Figure 2. Simulation time in GARFIELD++ as a function of the size of the electric field mesh using a triple-GEM detector configuration.

simulation time of several days or even weeks, depending on the input parameters (high voltage, gas mixture, size of the mesh, number of events). Given the statistical nature of the related processes to get reliable results a sample of few thousand events should be collected, this fact makes the timely production of results very difficult to achieve. Figure 2 shows the simulation time as a function of the size of the electric field mesh for five different configurations and using as benchmark a single-GEM detector. It is clear how the simulation time drastically increases as the size of the mesh is reduced, such results were produced running GARFIELD++ in a serial mode, in fact, the two last points in the graph are extrapolations due to the jobs being aborted by exceeding the time limit commonly imposed by the cluster administrators. In this paper we propose several methods to optimize the algorithms and reduce the simulation time allowing for a fast generation of results without affecting the accuracy of the same.

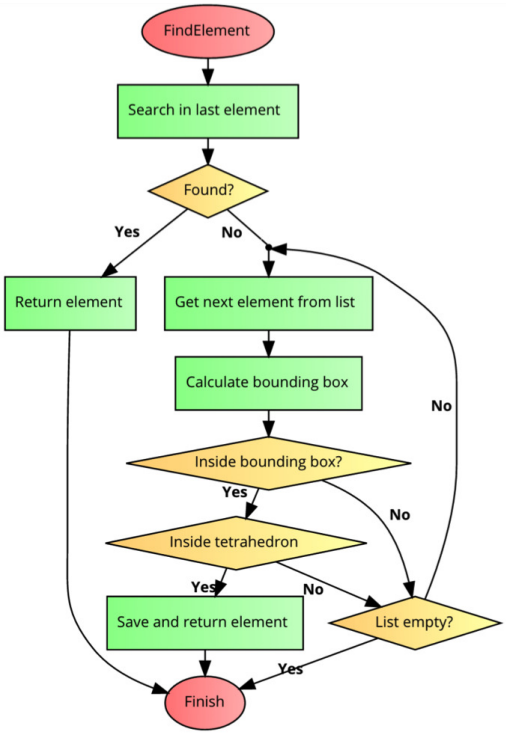

Figure 3. Workflow representing the different steps in the FindElement algorithm.

2 Optimization methods

The GARFIELD++ source code elements and operations were studied in order to find the one that takes the most of the simulation time. It was found that the so-called "FindElement" operation takes around 90% of the total execution time. The FindElement method basically search for elements in the tetrahedral mesh in order to find the position of the electron inside the electric field. As it is originally structured this method will search for compatible elements with the electron position, if such element is not found the search will start from the initial element in a sequential way, instead of searching for compatible elements around it, which clearly is a non-optimized implementation of this element searching algorithm. Figure 3 shows the workflow of the FindElement method which was later optimized to reduce the simulation time. Three optimization methods were studied and they are described in the following sub-sections.

2.1 Caching of bounding boxes

The original FindElement operation scans through all tetrahedrons (or elements) sequentially in order to find the corresponding tetrahedron that contains the given 3D point. This point-inside-tetrahedron test is an iterative and computationally expensive operation involving several matrix and vector operations. In order to speed up the computation, the FindElement operation calculates the axis-aligned bounding box (a cuboid) of the tetrahedron and checks if the point lies in it. The point-inside-cuboid operation is computationally inexpensive and sim-

ple as compared with the point-inside-tetrahedron operation. However, It was found that the bounding boxes were computed in every function call – each calculation requires 9 min and 9 max operations. In the optimized FindElement operation, the bounding boxes are computed when mesh is loaded initially when the simulation starts.

2.2 Element search using Tetrahedral Tree structure

As mentioned earlier, the FindElement operations scans through all tetrahedrons in a linear fashion to find the correct tetrahedron that contains a given 3D point. The order of scanning the tetrahedrons is arbitrary and depends on the way the mesh is stored on disk. In the optimized version, a spatial index is built based on the work by De Floriani et al. [5]. The tetrahedrons are arranged spatially using a *tetrahedral tree* that utilizes the *octree* as the underlying data structure. The *Octree* recursively subdivides the initial cubic domain (containing all tetrahedrons) into eight blocks. The octree has been selected instead of the kd-tree data structure as it produces tree of smaller depth hence reducing the number of tree nodes to be visited to reach the desired tetrahedron.

2.3 Search through neighbors

As the electron moves through the electric field or mesh, its new position is expected to be closer to the previous one. Based on this behavior, the non-optimized GARFIELD++ keeps track of the tetrahedron found in the last operation and runs the point-in-tetrahedron test. If the test is successful, then it skips the scanning through all elements and it simply returns the last tetrahedron. Otherwise, it scans through all elements sequentially. It was observed that if the new position does not lie in the last element, then it is most likely to be in one of the neighboring elements. The optimized FindElement operation search through the neighboring elements if the point-in-tetrahedron test fails for the last found element. Note that the neighbors of all tetrahedrons are precomputed initially when the simulation starts.

3 Results

The results of each of the optimization methods presented in section 2 are illustrated for a single- and triple-GEM configurations in Figure 4 and 5 respectively, in which the speed up factor is defined as the ratio of the simulation time using the standard GARFIELD++ to the simulation time using any of the optimization methods.

4 Introduction to a parallelized version of GARFIELD++ (pGARFIELD)

Based on a Message Passing Interface (MPI) architecture, the GARFIELD++ toolkit was adjusted to be used in parallel mode. Running with a master node which does random generation, the workload is distributed to and gathered back from the slave nodes. The performance was

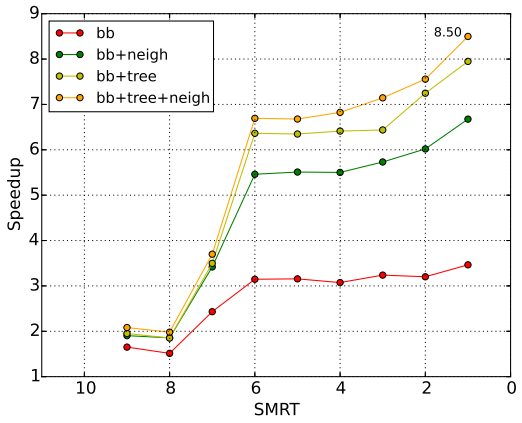

Figure 4. Speed up factor as a function of the size of the electric field mesh using a single-GEM detector configuration and comparing the performance of the different optimization methods

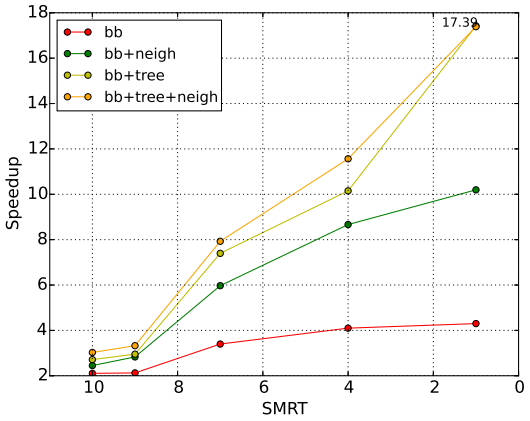

Figure 5. Speed up factor as a function of the size of the electric field mesh using a triple-GEM detector configuration and comparing the performance of the different optimization methods

studied on the HPC cluster at Texas A&M University at Qatar showing a good performance. The speed up factor reached is presented as a function of the number of nodes in Figure 6

5 Conclusions

Several optimization methods were studied to reduce the simulation time in GARFIELD++, using a single- and triple-GEM detector configurations results were obtained as a function of the size of the electric field mesh. It was shown that the simulation time is reduced by the use of each individual method, and it decreases drastically when a combination of them is applied. The largest reduction was observed by combining the three methods and using a triple-GEM detector configuration, in which, a reduction factor of 17 was achieved. This reduction was obtained without affecting the accuracy of the results. The algorithm improvements have been partially included in

the official GARFIELD++ repository, while the remaining parts will be implemented soon. In addition to the optimization methods a parallelized version of GARFIELD++ (pGARFIELD) was introduced, allowing for an even further reduction in the simulation time.

6 Acknowledgements

This work was supported by the Qatar National Research Fund under the project NPRP-5-464-1-080

References

[1] F. Sauli, Nucl. Instrum. Meth. A **805**, 2-24 (2015). doi:10.1016/j.nima.2015.07.060

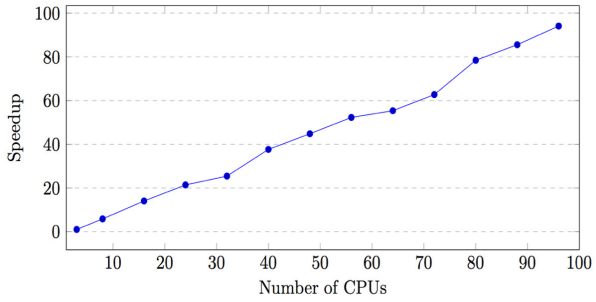

Figure 6. Speed up factor using a parallelized version of GARFIELD++ (pGARFIELD), the simulation time is reduced as the number of nodes is increased.

[2] G. Charpak, J. Derre, Y. Giomataris and P. Rebourgeard, Nucl. Instrum. Meth. A **478**, 26 (2002). doi:10.1016/S0168-9002(01)01713-2

[3] D Abbaneo, et al. JOURNAL OF INSTRUMENTATION **9**, 10 (2014). http://dx.doi.org/10.1088/1748-0221/9/10/C10036

[4] Rob Veenhof http://garfieldpp.web.cern.ch/garfieldpp

[5] De Floriani, Leila and Fellegara, Riccardo and Magillo, Paola, Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems, 2010