Annals of Nuclear Energy xxx (2014) xxx–xxx

Contents lists available at ScienceDirect

Annals of Nuclear Energy

journal homepage: www.elsevier.com/locate/anucene

Recent developments in Geant4

Makoto Asai [a], Andrea Dotti [a], Marc Verderi [b], Dennis H. Wright [a,*], The GEANT4 Collaboration

[a] SLAC National Accelerator Laboratory, Stanford, CA, USA
[b] IN2P3 Laboratoire Leprince-Ringuet, Paris, France

ARTICLE INFO

Article history:
Received 29 April 2014
Accepted 12 August 2014
Available online xxxx

Keywords:
Monte Carlo
Multi-threading
Physics modeling
GUI

ABSTRACT

During the last two to three years extensive development of the GEANT4 simulation toolkit has occurred, which will culminate in a major release at the end of 2013. This development includes the adoption of multi-threading, the extension and improvement of physics models, improvements to the geometry modeler with a revised implementation of most geometrical primitives, advances in visualization and graphical user interfaces (GUI), the move from GNU make to CMake and the extension and automation of the GEANT4 testing suite. The reasons for and implementation of multi-threading will be discussed. Certain electromagnetic and hadronic model extensions will also be discussed along with their effects on calorimeter results. Finally, visualization and GUI improvements will be highlighted along with the new configuration, build and testing systems.

1. Introduction

Perennial demands for improved physics modeling, the need for faster and more efficient simulations, and the desire to keep pace with modern hardware and software trends have all led to numerous improvements in the GEANT4 toolkit (Agostinelli et al., 2003; Allison et al., 2006) over the past two to three years.

Faster simulations can be achieved by exploiting the trend toward increasing numbers of cores per chip. In GEANT4 this is done by adopting multi-threading. We discuss the resulting changes to the kernel, the geometry package and other GEANT4 categories, which are extensive enough to warrant a major release of GEANT4 by the end of 2013.

Requests to improve physics simulations, in particular for hadronic showers, have come mostly from the high energy physics community. Hence the concentration on parton string and cascade models which has improved the agreement of simulations with calorimeter data. Significant input from the space and medical communities has led to improvements in low and medium energy models. We cover here electromagnetic and hadronic processes and note that improvements in physics models almost always imply increased computing time. It is hoped that these increases can be offset by employing more cores via multi-threading.

Algorithms of most geometrical primitives utilized by the geometry modeler have been deeply reviewed in the last couple of years. We will describe the major improvements and new features introduced.

To aid in the interpretation of simulated results, GEANT4 offers an array of visualization options and graphical user interfaces, most of which have seen significant advances recently. A few of these are highlighted below.

Finally we discuss the upgrade of build and testing tools. The move from GNU make to CMake has resulted in a more versatile configure and build system and a more powerful developer environment. An extensive testing suite has been developed with a number of online tools for running, monitoring and validating results of system tests.

2. The Geant4 kernel

2.1. Incorporating Geant4-MT in the Geant4 production release

To make efficient use of multi-core processors and reduce the memory footprint of the simulation we have developed a version of GEANT4 which uses multi-threading (MT for short) to share a substantial part of data between threads. Two GEANT4-MT prototypes were released in 2011 and 2012. In the next major release the multi-threaded code will be merged into the main source code. By design the memory savings in GEANT4-MT (Dong et al., 2010, 2012) are obtained by sharing the majority of the geometry descriptions and the tables of the physics processes among the threads; threads are otherwise independent. Each thread is responsible for simulating one or more full events, thus implementing event-level parallelism. Measurements performed with the

* Corresponding author. Address: SLAC National Accelerator Laboratory, 2575 Sand Hill Road, Menlo Park, CA 94025, USA. Tel.: +1 650 926 4743.
E-mail address: dwright@slac.stanford.edu (D.H. Wright).

http://dx.doi.org/10.1016/j.anucene.2014.08.021
0306-4549/

GEANT4-MT prototype demonstrate (Figs. 1 and 2) that this approach scales well with the number of threads.

To ensure the long-term maintainability of the code, and to allow the code to evolve towards newly emerging technologies, GEANT4-MT is based on the POSIX thread standard. This has allowed the code to be ported to different operating systems, including Mac OS X, and also to newly emerging technologies, such as the Intel Xeon Phi co-processor. This latter aspect is interesting since these co-processors allow the use of a large number of threads, and enable testing of the scalability of the system. We have reached an efficiency in linearity of more than 80% with about 200 threads on Intel Xeon Phi (Ahn et al., 2013).

An important constraint and driving consideration in the design of GEANT4-MT is that the effort required for application developers to port an application to GEANT4-MT is small. Our criterion is that this extra effort should be a small fraction of the total development effort. In addition, the effort required to maintain the required code modifications should be small. For a simple or standalone GEANT4 application, i.e. one which does not depend on a large external framework, the effort to make the first port of an application should be of the order of a few hours' work.

The process of porting a GEANT4 application to multi-threading involves the inspection and potentially the adaptation of a small set of methods, which are the responsibility of an application developer. The key parts that require intervention are classes in which the user examines the internal state of the simulation at the end of every step, track or event, and records observables which will be output (*SensitiveDetector*).

GEANT4-MT has been integrated with the main GEANT4 code base and will become publicly available with Version 10. For a detailed description of multi-threading capabilities in GEANT4 and benchmarks performed so far see Ahn et al. (2013)).

2.2. Geometry

The GEANT4 geometry modeler has undergone a deep review in different areas. The integration of multi-threading capabilities into the toolkit and the stringent requirement to allow for sharing of all geometrical information in a multi-threaded run, has required the adaptation of a few key classes to clearly separate the read-only part from data which can change during the generation of an event, and therefore should be treated privately by each single working thread. See Ahn et al. (2013)) for details.

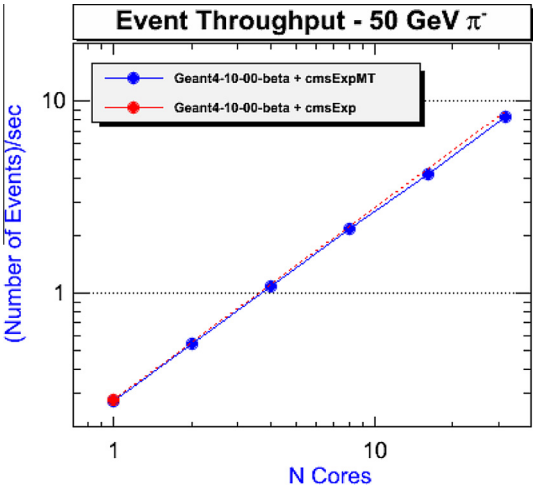

Fig. 1. Linearity of speed-up as a function of the number of threads. For reference, results obtained with the sequential version of GEANT4 are shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

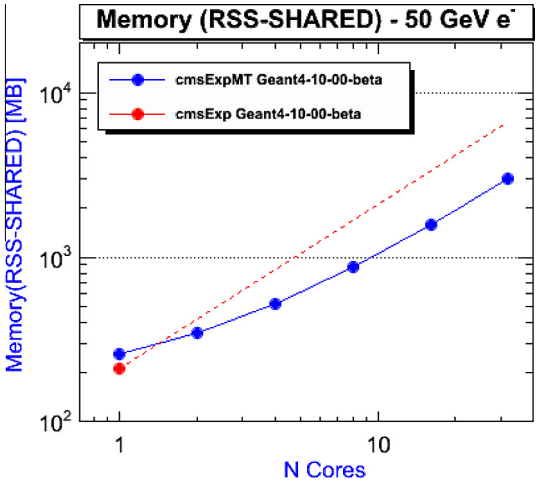

Fig. 2. Linearity of memory use as a function of the number of threads. For reference, results obtained with the sequential version of GEANT4 are shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In addition to the modifications strictly related to thread safety, several new features and improvements have been introduced and are planned for the geometry for the next major release of GEANT4.

It is now possible to define volumes with material in parallel geometry layers. The volumes created can complement or overlap the standard geometry setup used for tracking. All geometries and all geometrical regions are now scanned for their own material. This new capability, *layered mass geometry*, was introduced in release 9.5 and allows for the definition of one or more additional layers for the mass geometry. This is an enhancement of the existing functionality for parallel specialized geometries (Apostolakis et al., 2008), which has been in use for scoring, fast simulation, shower parameterization and event biasing. Interesting applications could be the definition of sampling calorimeters, where the geometry could be layered according to different levels of detail for different particle types, or in medical physics applications like brachytherapy (Asai et al., 2012).

An important effort was begun in the last couple of years, based on the AIDA (AIDA, 2012) initiative, to write a new software library for the modeling of geometrical primitives (solids), starting from the existing implementations in GEANT4 and in the Root geometry package (Gayer et al., 2012).

This work, which is now nearly completed, consists in reviewing at the algorithmic level most of the primitives and provides an enhanced, optimized and well-tested implementation to be shared among software packages. In most cases considerable performance improvement was achieved. For example, the time required to compute intersections with the tessellated solid was dramatically reduced with the adoption of spatial partitioning for composing facets into a 3D grid of voxels.

Such techniques allow speedup factors of a few thousand for relatively complex structures having of order 100K to millions of facets, which is typical for geometry descriptions imported from CAD drawings (see Fig. 3). Consequently, it is now possible to use tessellated geometries for tuning the precision in simulation by increasing the mesh resolution, something that was not possible before. This development has been included in the GEANT4 distribution since release 9.6.

In addition to a full set of highly optimized primitives and a tessellated solid, the library includes a new "multi-union" structure implementing a composite set of solids (several or many) to be placed in 3D space. This differs from the simple technique based on Boolean unions, with the aim of providing excellent scalability

Please cite this article in press as: Asai, M., et al. Recent developments in Geant4. Ann. Nucl. Energy (2014), http://dx.doi.org/10.1016/j.anucene.2014.08.021