

GeantV: from CPU to accelerators

Philippe Canal¹

Fermi National Accelerator Laboratory

P.O. BOX 500, Batavia IL, USA

E-mail: pcanal@fnal.gov

Federico Carminati

CERN

CH-1211 Geneva 23, Switzerland

E-mail: Federico.Carminati@cern.ch

Andrei Gheata

CERN

CH-1211 Geneva 23, Switzerland

E-mail: Andrei.Gheata@cern.ch

Sandro Wenzel

CERN

CH-1211 Geneva 23, Switzerland

E-mail: sandro.wenzel@cern.ch

Soon Yung Jun

Fermi National Accelerator Laboratory

POBOX 500, Batavia IL, USA

E-mail: syjun@fnal.gov

Guilherme Amadio

UNESP - Universidade Estadual Paulista

São Paulo, Brazil

E-mail: guilherme.amadio@cern.ch

Ananya Ananya

IIT- Indian Institute of Technology, Indore

Khandwa Road, Simrol, Indore, Madhya Pradesh 453552, India

E-mail: ananya.ananya@cern.ch

John Apostolakis

CERN

CH-1211 Geneva 23, Switzerland

E-mail: John.Apostolakis@cern.ch

Anmol Arora

IIT- Indian Institute of Technology, Indore

¹Speaker

Khandwa Road, Simrol, Indore, Madhya Pradesh 453552, India

E-mail: anmol.arora@cern.ch

Marilena Bandieramonte

CERN

CH-1211 Geneva 23, Switzerland

E-mail: marilena.bandieramonte@cern.ch

Abhijit Bhattacharyya

Bhabha Atomic Research Centre

Trombay, Mumbai, Maharashtra 400085, India

E-mail: abhijit.bhattacharyya@cern.ch

Calebe De Paula Bianchini

UNESP - Universidade Estadual Paulista

São Paulo, Brazil

E-mail: bianchini@ncc.unesp.br

Rene Brun

CERN

CH-1211 Geneva 23, Switzerland

E-mail: Rene.Brun@cern.ch

Daniel Elvira

Fermi National Accelerator Laboratory

POBOX 500, Batavia IL, USA

E-mail: elvira@fnal.gov

Mihaela Gheata

ISS - Institute of Space Science

409, Atomistilor Street Magurele, Ilfov, 077125, Romania

E-mail: Mihaela.Gheata@cern.ch

Ilias Goulas

CERN

CH-1211 Geneva 23, Switzerland

E-mail: Ilias.Goulas@cern.ch

Rogério Iope

UNESP - Universidade Estadual Paulista

São Paulo, Brazil

E-mail: rogerio.iope@gmail.com

Guilherme Lima

Fermi National Accelerator Laboratory

POBOX 500, Batavia IL, USA

E-mail: lima@fnal.gov

Ajit Kumar Mohanty

Bhabha Atomic Research Centre

Trombay, Mumbai, Maharashtra 400085, India

E-mail: ajit.mohanty@cern.ch

Tatiana Nikitina

Gangneung-Wonju National University
123 Jibyeon-dong, Gangneung, Kangwon, South Korea
E-mail: tatiana.nikitina@cern.ch

Mihaly Novak

CERN
CH-1211 Geneva 23, Switzerland
E-mail: mihaly.novak@cern.ch

Witold Pokorski

CERN
CH-1211 Geneva 23, Switzerland
E-mail: Witold.Pokorski@cern.ch

Alberto Ribon

CERN
CH-1211 Geneva 23, Switzerland
E-mail: Alberto.Ribon@cern.ch

Raman Sehgal

Bhabha Atomic Research Centre
Trombay, Mumbai, Maharashtra 400085, India
E-mail: raman.sehgal@cern.ch

Oksana Shadura

National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Peremohy Ave, 37, Kyiv, 03056, Ukraine
E-mail: oksana.shadura@cern.ch

Sofia Vallecorsa

Gangneung-Wonju National University
123 Jibyeon-dong, Gangneung, Kangwon, South Korea
E-mail: Sofia.Vallecorsa@cern.ch

Yang Zhang

Karlsruhe Institute of Technology - KIT
76131 Karlsruhe, Germany
E-mail: yang.zhang@student.kit.edu

The GeantV project aims to research and develop the next generation simulation software describing the passage of particles through matter, targeting not only modern CPU architectures, but also more exotic resources such as GPGPU, Intel® Xeon Phi, Atom or ARM, which cannot be ignored any more for HEP computing. While the proof of concept GeantV prototype has been mainly engineered for CPU threads, we have foreseen from early stages a bridge for such accelerators, materialized in the form of architecture/technology specific backend templates. This approach allows to abstract out not only basic types such as scalar/vector, but also to formalize generic computation kernels using transparently library or device specific constructs based on several industry leading libraries. While the main goal of this approach is performance and access to functionality, this comes as bonus with the insulation of the core application and

algorithms from the technology layer, allowing our application to be long term maintainable and versatile to changes at the backend side. The talk will present the early results of basket-based GeantV geometry navigation on the Intel© Xeon and Intel© Xeon Phi KNC architecture, as well as the work done for making the transport NUMA aware. We will present a detailed scalability and vectorization study conducted using Intel performance tools, as well as our preliminary conclusions on the use of accelerators for GeantV transport. We will also describe the current work and preliminary results for using the GeantV transport kernel on GPUs.

*38th International Conference on High Energy Physics
3-10 August 2016
Chicago, USA*

POS (ICHEP2016) 177

1. Vertical scaling for particle transport simulation

The computing GRIDs [1,2] has allowed the large scale horizontal scaling of the most important HEP computing tasks including simulation, reconstruction and analysis on hundreds of thousands of concurrent jobs taking advantage of distributed computing resources. However measurements [3] have shown that HEP code was in general significantly underutilizing the current hardware due to lack of instruction and data locality and lack of use of features such as instruction level parallelism and SIMD, leaving room for very large potential improvements.

GeantV introduces the concept of vectorized processing in particle transport [4], enabled by a *basketizing* procedure that feeds vectors of particles to floating-point intensive geometry and physics algorithms. The goal is to leverage vector units available on modern processors and to optimize the use of caches by using geometry and physics locality as much as possible [5,6]. The project aims to provide a long-term solution for particle transport simulation working in general purpose CPU, accelerators such as Xeon Phi and GPU, and HPC environments.

2. GeantV in native and offload mode

GeantV's design is based on creating baskets of filtered (eg. located in the same geometry volume or having similar physics properties) tracks and dispatching them to processing units. These can be threads of the same process, different processes or arbitrary co-processors acting as accelerators. The track data is passed to a broker that can collect as many tracks as deemed efficient for the co-processor and taking care of shipping the data to and from the device, as shown in Figure 1. This mode can only be efficient if the data transfer back and forth is asynchronous and concurrent in order to mitigate the data transfer latency.

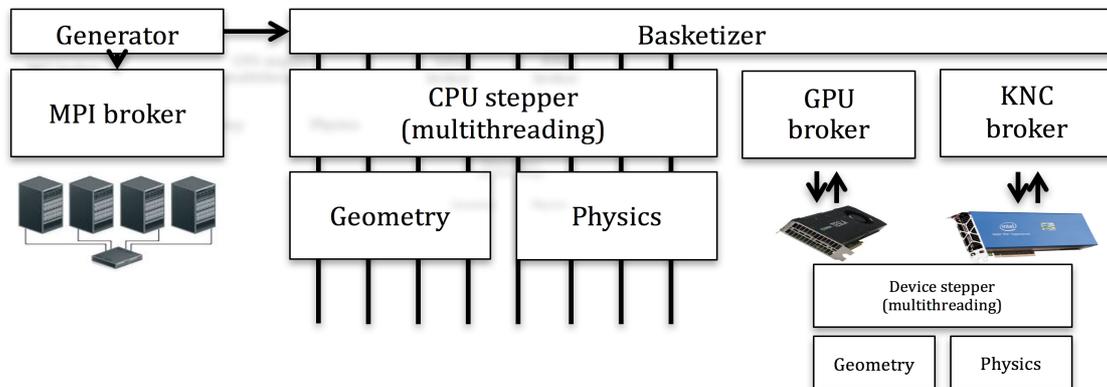


Figure 1. *GeantV* scheduling in native and offload modes. The coprocessor brokers can work in ‘fine grain’ mode, processing a set of baskets and sending transported tracks back to host, or MPI mode, fetching data from an event or file server.

This workflow has been tested successfully on Kepler cards using CUDA7. The same approach was initially foreseen for the Intel® KNC device broker, but it was dropped due to lack of efficient asynchronous data transfers when using offload mode.

To understand the performance features on this architecture for some of our main software components, targeting a full *GeantV* deployment on the future Intel® KNL, we thus rely on running in native mode. A custom tuning of the scheduling parameters to optimize the workflow depending on the device architecture will be necessary to take full advantage of those platforms.

To allow compiling the same code on both host and device modes, GeantV uses the concept of backend interfaces [7, 8]. Since the main differences across platforms are related to data types, template specialization can be used to run a common sources code across different technologies, including scalar and SIMD vector mode.

To adapt to the KNL Xeon Phi native mode, we are enhancing the scheduler to provide better scalability and NUMA awareness. On the GPU side, the CUDA backend already allows running a full transport step but currently the rescheduling still has to be done on the host. We do not exclude the possibility to eliminate this limitation in future.

3. Physics and geometry benchmarks on accelerators

We have investigated in a first step the performance of GeantV geometry and physics benchmarks on Kepler K20 and Intel® Xeon Phi C0PRQ-7120 (KNC) cards. We have tried to understand the performance limits on such devices for elementary algorithms such as geometry intersection for different solids or simple physics models.

Table 1. Overall performance for particles traversing a simplified detector, compared to ROOT v5.34.17. We compare the speedup of the vector version compared to the scalar version of ROOT running on the same architecture.

Architecture	16 particles	1024 particles
Intel® Ivy-Bridge (AVX)	2.8	3.7
Intel® Haswell (AVX2)	3.0	5.0
Intel® Xeon Phi (Intel® IMCI)	4.1	4.8

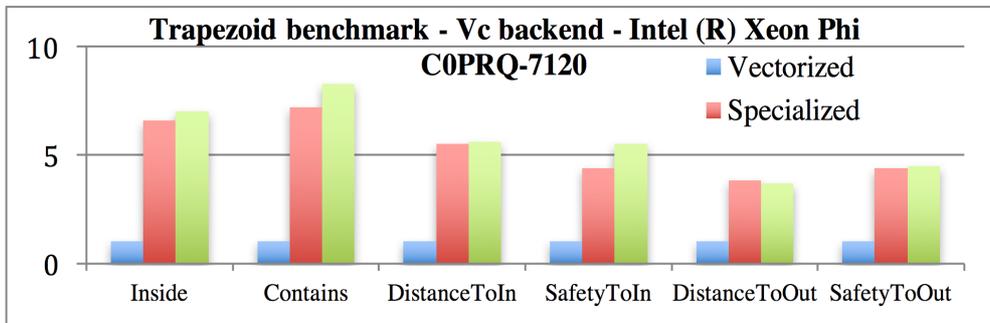


Figure 2. Relative time normalized to the vectorized case for different trapezoid shape intersection algorithms (Xeon®Phi C0PRQ-7120 P).

For the geometry case, we have tested the speedup compared to the implementation in Geant4 [9] and ROOT [10]. Table 1 presents some of the results combining both scalar optimizations and vector gains for a simplified detector setup using a simple navigation. Standard intersection benchmarks for solids such as boxes, or trapezoids are showing very good speedup of often more than a factor of 4 on KNC (Figure 2) For the most complex algorithms such as the polycone or polyhedron intersection, the vector gains are reduced (sometimes less than x2) due to their multiple branches, which do not map well to vector operations, In such complex cases speedup might come from vectorizing inner loops on surfaces or from better

algorithms. Figure 3 presents the performance for the different shape algorithms on the K20, showing how the die can be saturated with large track containers running in a single kernel.

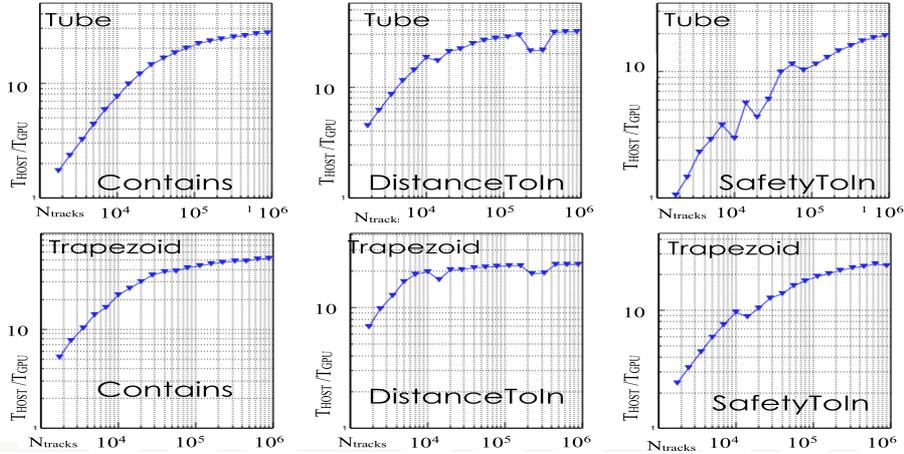


Figure 3. Performance of box navigation algorithms on K20 GPU compared to scalar mode on CPU and compared to equivalent ROOT/USolids algorithms. The speedup is computed as $T_{\text{scalar}}/T_{\text{vector/GPU}}$.

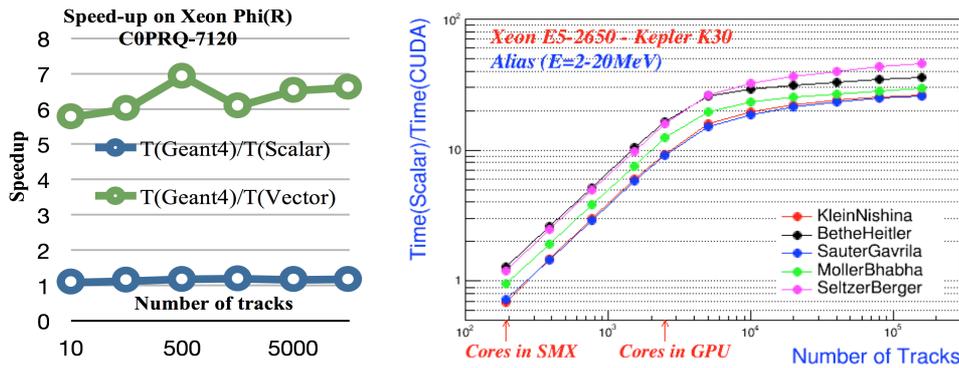


Figure 4. Vectorized Compton process on KNC (left). Performance of the alias sampling method for different process cross-sections (right).

The left side of Figure 4 presents the performance one of the vectorized physics model (Compton) algorithm on KNC card, which uses the vector unit of the accelerator close to its ideal performance, while the scalar version has basically the same performance as the corresponding Geant4 one. We have tested alias sampling on different electromagnetic models' distributions (having quite different profiles) on the Kepler card [11]. The performance profile keeps the same shape but can vary with as much as 50% for different distributions.

4. Performance of combined algorithms on accelerators: the X-ray benchmark

An important question is the relevance of vector optimizations of elementary algorithms for the complete simulation workflow: is it possible to maintain the performance improvements or will the processing steps connecting different vectorized algorithms may represent scalar bottlenecks thus reducing gains?

We have tried to understand the vector optimization procedure by running a composite benchmark on the Intel® Xeon Phi C0PRQ-7120 card. The standard Xeon-based benchmark uses a part of the full detector geometry and runs multi-particle propagation traversing through

it as shown in Figure 5. This benchmark focus on just the geometry navigation part of the simulation task and creates an “X-ray” image. The gray level depends on the number of different volumes traversed by each track. We looked at both overall scalability and throughput, and in-depth VTune profiling to understand the bottlenecks. As a result we have clearly identified the particle relocation algorithm as one of the important scalar bottlenecks.

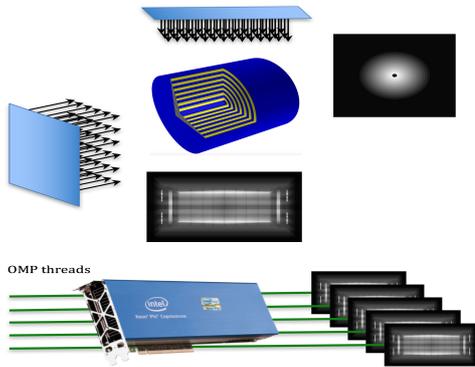


Figure 5. The X-Ray benchmark tests geometry navigation in real detector geometry, which is one of the main components of *GeantV*. The name comes the fact that one takes a detector geometry module and shoots virtual rays (with starting points disposed in a grid) along a given direction and with a given resolution (input parameters). Each ray is propagated from boundary to boundary using the *VecGeom* navigator, and the number of crossings is counted until the volume is exited on the other side. The model for parallelism uses *OpenMP* to dispatch different scans to different threads.

For our optimization purpose, we have used a simplified geometry setup emulating concentric cylindrical tracking detectors, where crossing volume boundaries are predictive. We attached a basketizer to each of these detectors, aiming to form vectors of tracks of fixed size before invoking geometry navigation algorithms. This reproduces to some extent the actual re-basketizing schema used in *GeantV*, including the rundown phase when the partially filled baskets have to be processed at the end of the simulation.

The setup is presented in Figure 5. The exercise’s objective was to saturate the processing pipelines of a Xeon Phi card. To achieve this goal we steered multithreading using OpenMP, parallelizing on the loop of X-ray scans to be performed: each thread had to complete full scans over a fixed-size grid of 1024x1024 pixels. We have tested both compact and balanced OpenMP thread scheduling, using an increasing number of threads, up to 4 threads per physical core of the Xeon Phi card. We have measured both scalability and the image throughput, comparing them with the same benchmark done on the IvyBridge dual processor machine hosting the KNC card.

5. Results and conclusions

The benchmark was run with both compact and balanced OMP modes. In balanced mode it showed an excellent scalability close to the ideal case up to the physical core count. The balanced model converges towards the compact model as all cores are filled. Running Xeon Phi fully saturated brought the maximum benefit for this application.

To understand the vectorization features of the model, we have run the benchmark and studied three different cases: *Scalar case*: Simple loop over pixels, one ray per pixel transported with scalar code. *Ideal vectorization case*: vectors with N times the **same** ray, transported with vector code. This is not realistic as tracks will always be different, but this was useful to be used as a reference for the maximum achievable vectorization. *Filtering/regrouping case*: vectors of tracks grouped per geometry volume and regrouping tracks to the appropriate baskets as

particles are entering the next volume (similar to the full GeantV approach). Transport is triggered when vectors reach a given size.

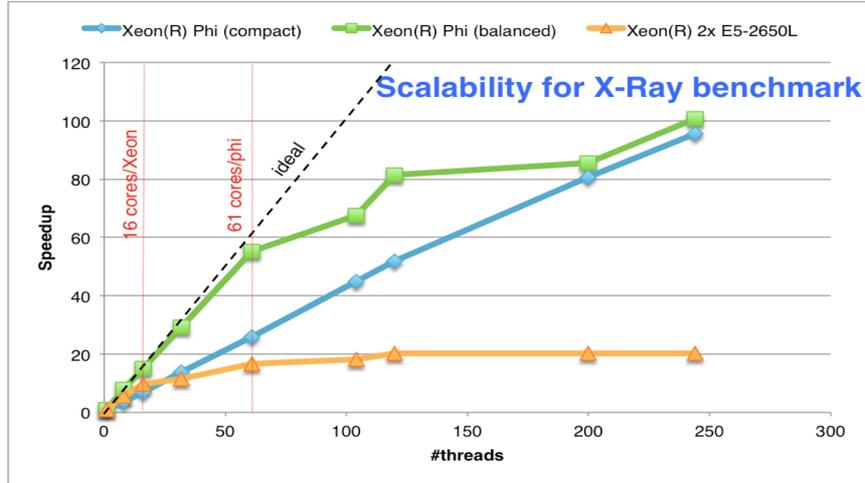


Figure 6. X-ray benchmark scalability on Xeon Phi and IvyBridge for compact and balanced thread allocation. The Xeon server shows unexpected benefits even running beyond its hyper-threading regime, hinting to vectorization as main responsible.

Figure 7 shows that while some of the vector efficiency is lost compared to the ideal case, the realistic case gains up to 4.5, with vector starvation showing up beyond the core count.

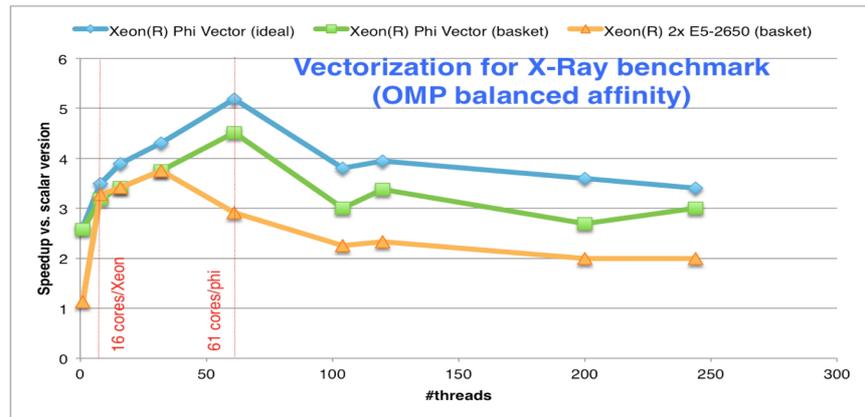


Figure 7. Vectorization performance on Xeon Phi (KNC) for the basketized X-ray benchmark, compared to idealized performance of the same setup and with IvyBridge. The speedup is defined as $T_{\text{scalar}}/T_{\text{vector}}$ for the same number of threads. The decreasing slope at large number of threads is not an absolute loss in performance but a relative one (i.e. the scalar version scales better than the vector).

In parallel with the GeantV steering framework advancements, the development of vectorized algorithms based on our approach using backends is well advanced in case of geometry and in an initial phase in case of physics. Both geometry and physics benchmarks demonstrate good performance so far on co-processors. Development plans for the next phase aim to provide full built-in accelerator awareness for GeantV-based simulation on both GPGPU and KNL processors.

References

- [1] M Lamanna 2004 *N.I.M. in Phys. Res.* **A 534**(1):1-6.
- [2] Pordes, R., et al. 2007. The Open Science Grid . *Journal of Physics: Conference Series* 78:012057.
- [3] S Jarp, A Lazzaro, J Leduc and A Nowak 2012 *Comp. High Energy and Nucl. Phys. (CHEP)* <https://indico.cern.ch/event/149557/session/6/contribution/486>
- [4] Apostolakis J, Brun R, Carminati F and Gheata A 2012 *J.Phys.: Conf. Ser.* **396** 022014
- [5] <http://iopscience.iop.org/1742-6596/396/2/022014>
- [6] Apostolakis J, Brun R, Carminati F, Gheata A and Wenzel S 2014 *J. Physics: Conf. Ser.* **513** 052006
- [7] J Apostolakis et al 2015 *J. Phys.: Conf. Ser.* **608** 012003
- [8] Wenzel S 2014 Towards a high performance geometry library for particle-detector simulation *16th Intl. workshop on Adv. Comp. and Analysis Techniques in phys. Res. (ACAT)*
- [9] de Fine Licht J 2014 First experience with portable high-performance geometry code on GPU *GPU Computing in High Energy Physics*
- [10] S Agostinelli et al 2003 *Nuclear Instruments and Methods* **A 506** (53pp)
- [11] <http://root.cern.ch>
- [12] G Amadio et al 2016 Electromagnetic Physics Models for Parallel Computing Architectures *17th Intl. workshop on Adv. Comp. and Analysis Techniques in phys. Res. (ACAT)*