GeantV – Adapting simulation to modern hardware

Sofia Vallecorsa for the GeantV team

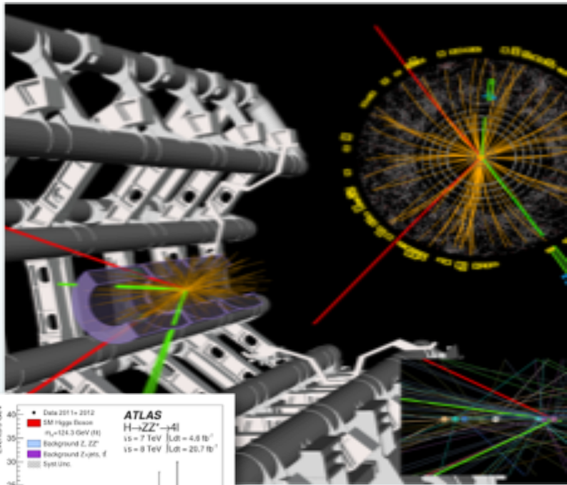
PASC 2017

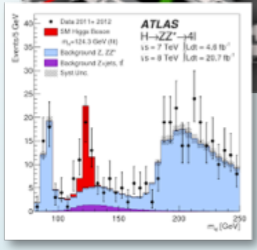
Outline

- Introduction
- The GeantV approach
  - Portability
  - Vectorisation and geometry navigation
  - Data layout and memory optimisations
  - Scalability
  - Towards a HPC friendly application
- A Deep Learning engine for fast simulation
  - Generative adversarial networks for calorimeter shower
- Summary and plans

Monte Carlo Simulation for HEP…

- Detailed simulation of subatomic particles is essential for data analysis, detector design
  - Understand how detector design affect measurements and physics
  - Use simulation to correct for inefficiencies, inaccuracies, unknowns.
  - The theory models to compare data against.

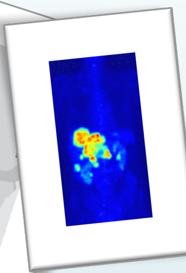


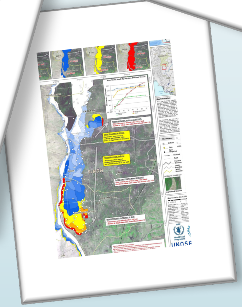


A good simulation demonstrates that we understand the detectors and the physics we are studying

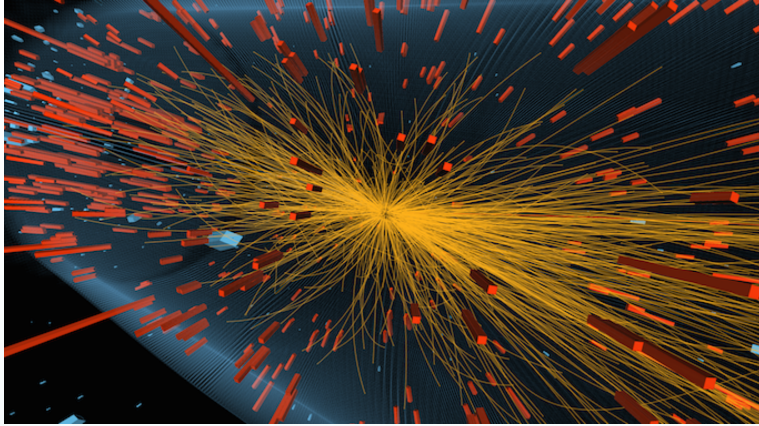
…and for the rest of humanity…

- Medical applications
  - MRI scan (supra conducting magnet)
  - PET scan (scintillators)
  - Proton beam therapy
- Industrial radioscopy
- Radioprotection





The problem



- Complex physics and geometry modeling
  - Some physics process are extremely rare!
- Heavy computation requirements, massively CPU-bound
- Already now more than 50% of WLCG power is used for simulations

WLCG Worldwide LHC Computing Grid

200 Computing centers in 20 countries: > 600k cores

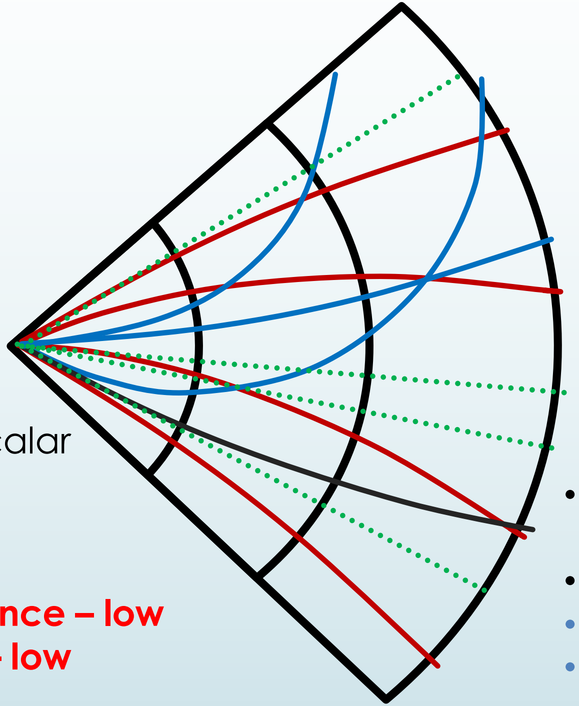
@CERN (20% WLCG): 65k processor cores ; 30PB disk + >35PB tape storage

By 2025 with the High Luminosity LHC run we will have to run simulation 100x faster!

Parallelism in simulation

Classical simulation

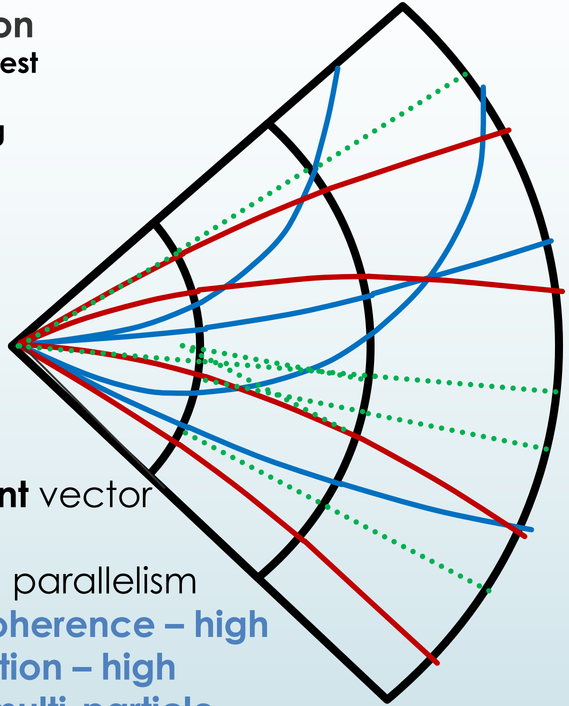
hard to approach the full machine potential



- **Single event** scalar transport
- Embarrassing parallelism
- **Cache coherence – low**
- **Vectorization – low (scalar auto-vectorization)**

GeantV simulation

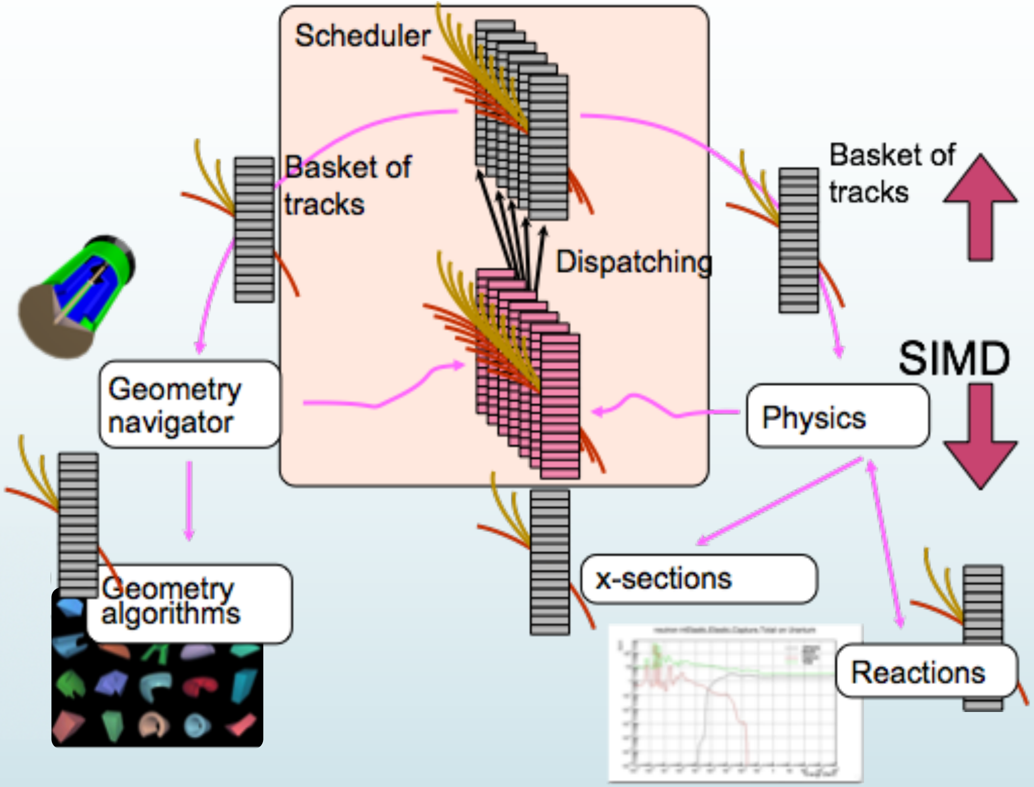
profits at best from all processing pipelines



- **Multi-event** vector transport
- Fine grain parallelism
- **Cache coherence – high**
- **Vectorization – high (explicit multi-particle interfaces)**

GeantV approach: boosting vectors

- Transport particles in vectors ("baskets")
  - Filter by geometry volume or physics process
- Keep "(re-) basketizing" overhead under control
- Abstract vector types to achieve portable vectorization

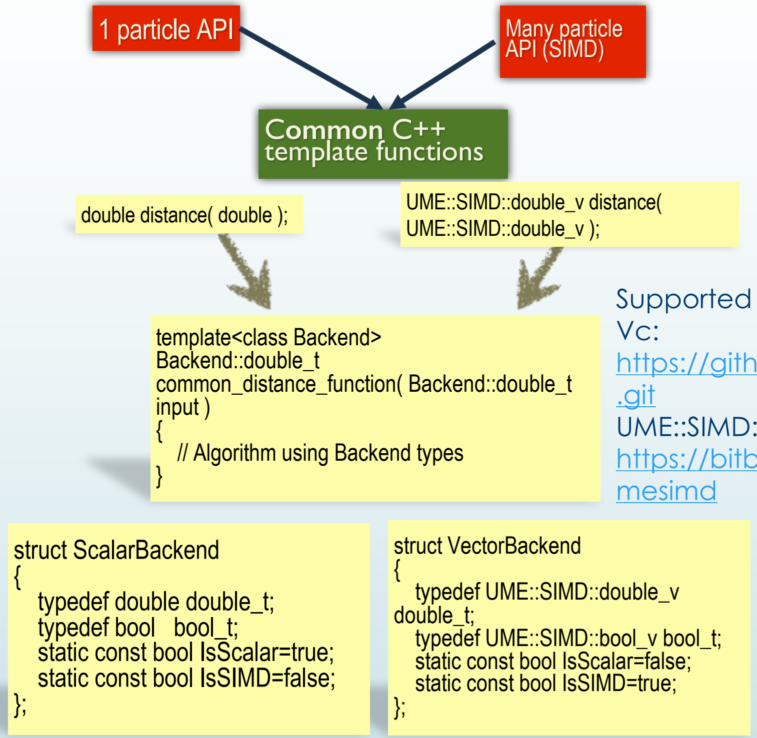


Aim for a 3x-5x faster code, understand hard limits for 10x

Portable performance

Long-term maintainability of the code

- Write one single version of each algorithm
- Platform specialization via **C++ templates** and low level optimised libraries
- **Backend**: (trait) struct encapsulating standard types/properties for “scalar, vector, GPU
  - Makes information injection into template function easy



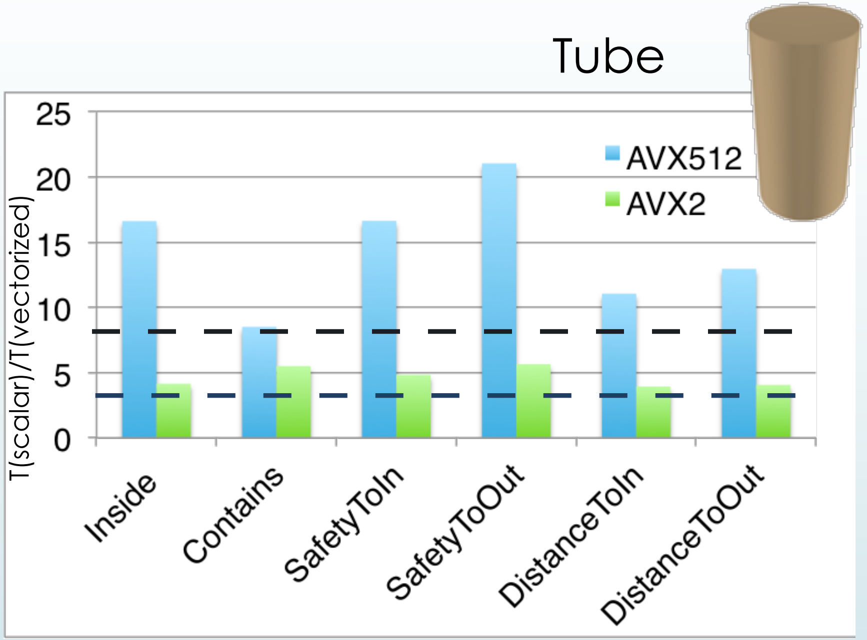
Supported SIMD backends:
Vc: https://github.com/VcDevel/Vc.git
UME::SIMD: https://bitbucket.org/edanor/umesimd

AVX512 aware

Vectorized geometry

- GeantV uses [VecGeom](), vectorized geometry library
  - Vectorized APIs for shape primitives
  - Vectorized APIs for navigation
- Measure speed-up for single shapes
  - Super-linear speedup for some methods on KNL
  - Compiler and algorithms effects

Tube

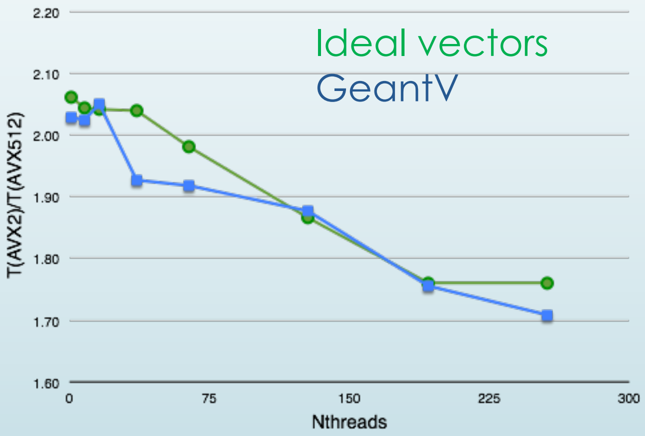


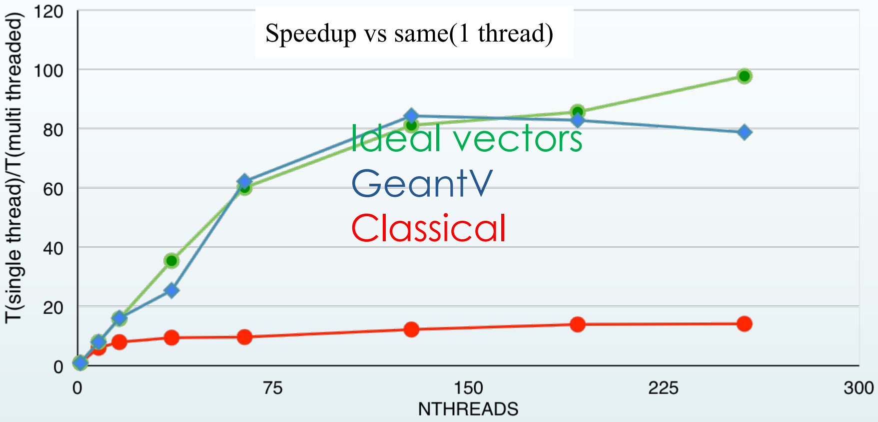
Intel® Xeon Phi™ CPU 7210 @ 1.30GHz, 64 cores

Geometry navigation on Intel Xeon Phi

Intel Xeon Phi 7210 @1.30 Hz – 64 cores

- Testing geometry navigation performance wrt classical approach
  - X-Ray scan of a simple toy detector geometry

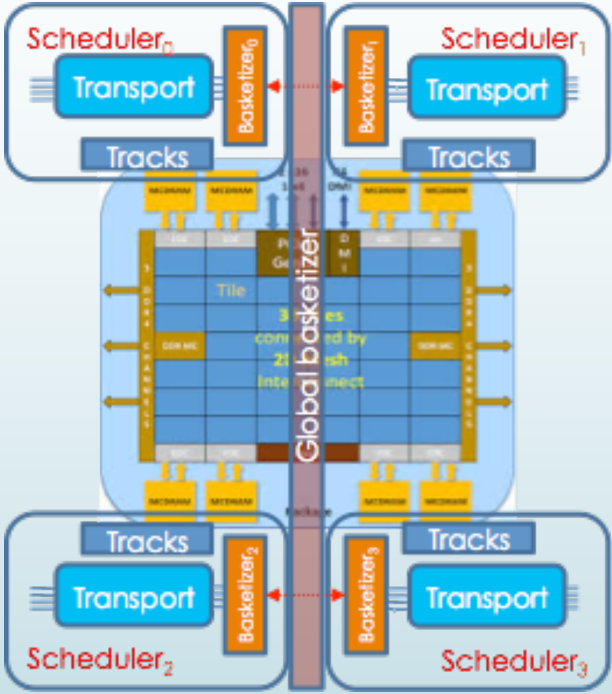




- High vectorization intensity achieved for AVX2 and AVX512 builds on KNL
- AVX512 brings the extra 2x speedup

Data layout and memory optimization

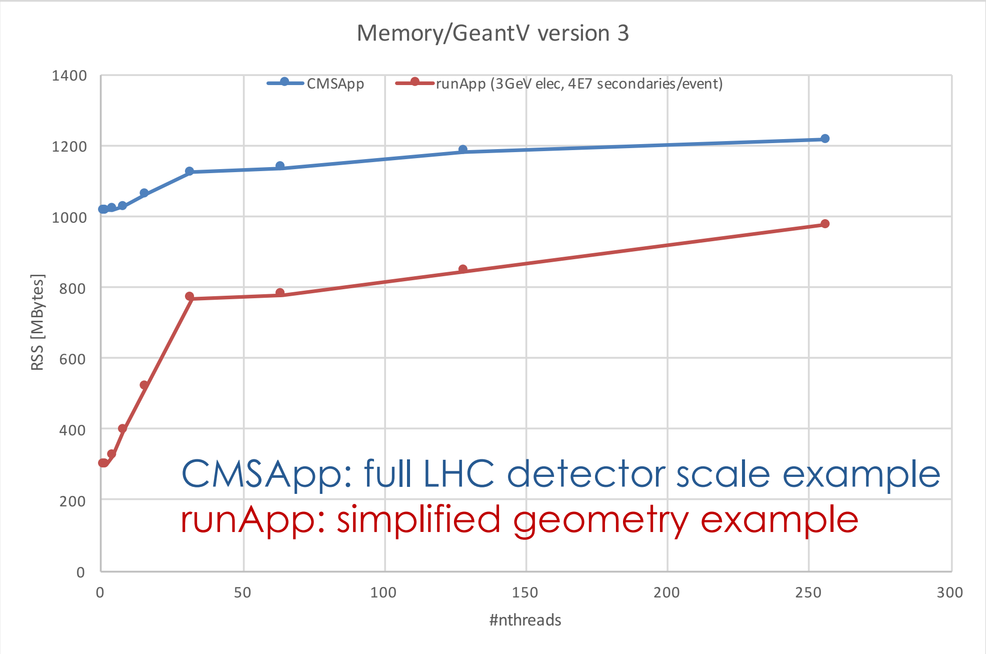
- Reducing overheads for scatter/gather, reshuffling, concurrency
  - Smart AOS/SOA usage
- Improve locality
  - Thread-local data
  - NUMA-aware allocation of resources, relying on topology discovery (libhwloc)
  - Minimize communication between NUMA nodes



Performance studies

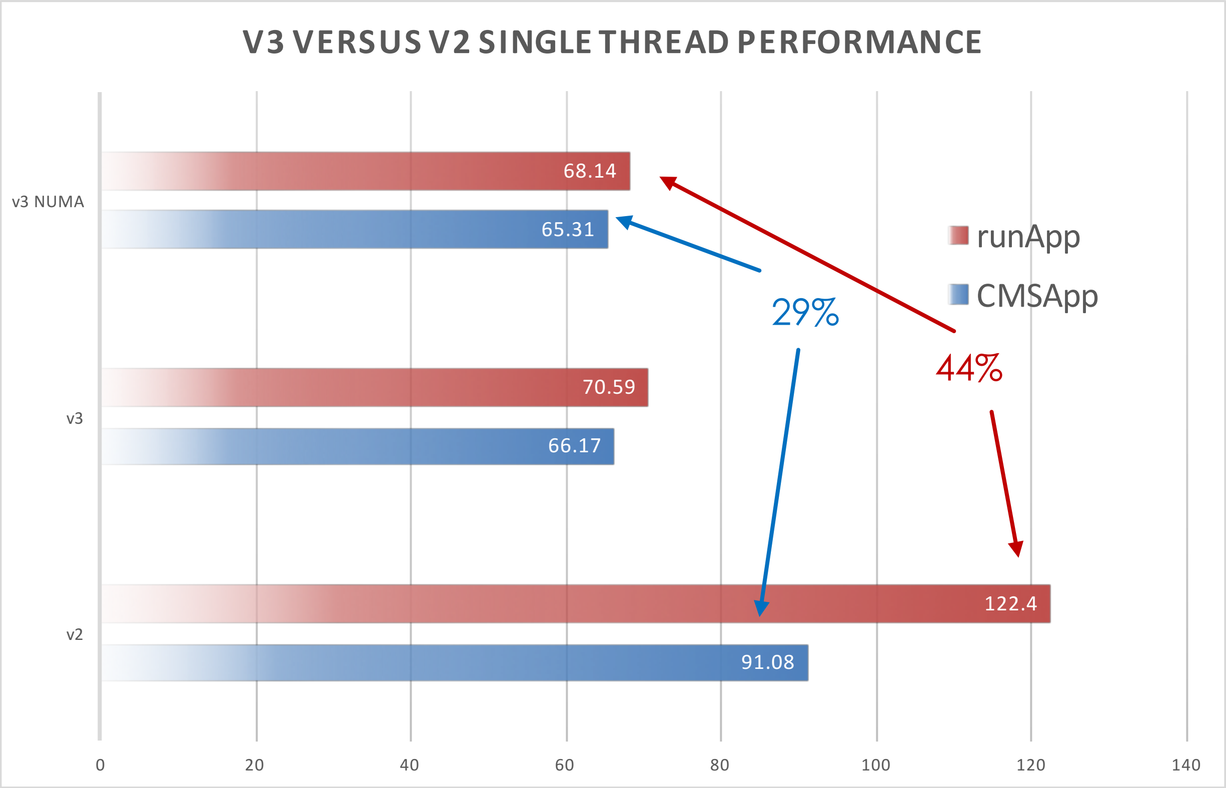
Memory control

- Simulation of secondary particles can be a problem for memory management
  - Higher generation secondaries flushed with priority
  - Very good behavior even for high number of threads/secondaries



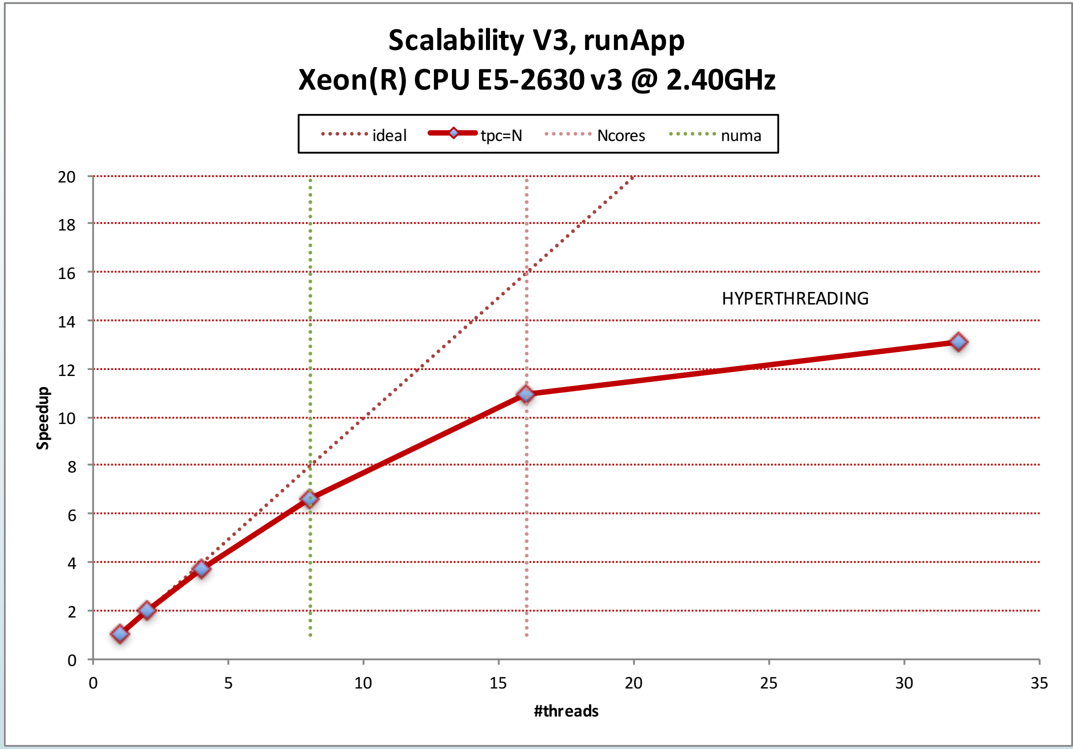
Single thread performance

- Relevant improvements in single and multi-threaded mode
  - Increase in locality
  - Removal of SOA gather/scatter overheads
  - NUMA awareness



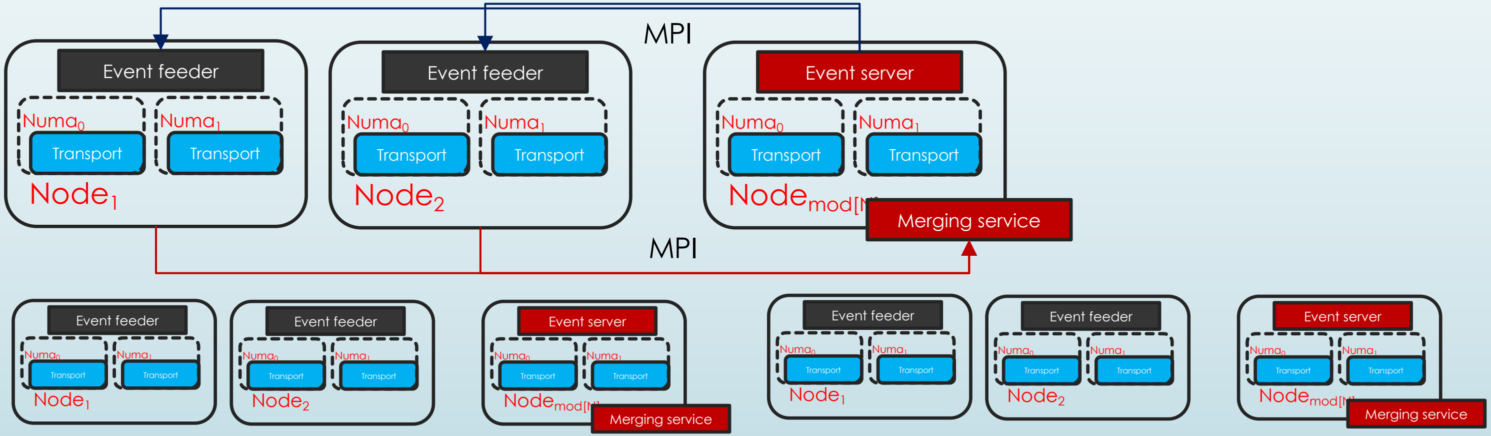
Scalability

- Not as good as expected
- No obvious hotspots
- Memory operations still high in the profile, we expect picture to improve when having a more balanced scenario with more (vector) work on physics side.
- **Studying scaling on Intel Xeon Phi**



GeantV plans for HPC environments

- Standard mode (1 independent process per node)
  - Always possible, no-brainer
  - Possible issues with work balancing (events take different time)
  - Possible issues with output granularity (merging may be required)
- Multi-tier mode (event servers)
  - Useful to work with events from file, to handle merging and workload balancing
  - Communication with event servers via MPI to get event id's in common files



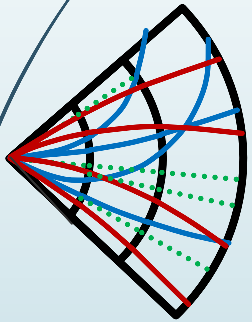
Summary – Part 1

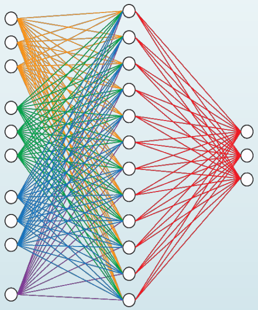
- A big effort to modernize simulation code and exploit at best modern hardware
- GeantV already delivers part of the expected performance
  - Demonstrating portability of our backend approach, no algorithmic line changed!
  - Excellent vector performance showing that the code should better be vectorized
  - Smart memory management and data locality further improve performance
  - Benchmarking on Intel Xeon Phi

Deep Learning for fast simulation in GeantV

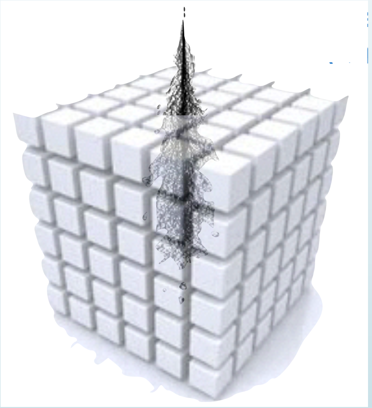
Going beyond 10x: fast simulation

- In the best case scenario GeantV will give 10x speedup → not enough
- A certain percentage of events will have to be simulated using "faster approaches" → fast simulation
- Improved, efficient and accurate fast simulation based on DL techniques





Test on most time consuming detectors: calorimeters

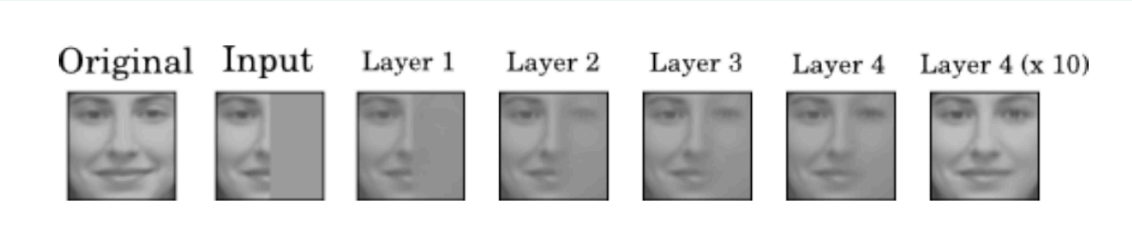


DL for calorimeter simulation

Generative models (Generative Stochastic Networks, Variational Auto-Enconders, Generative Adversarial Networks, ..) can be used for simulation

- Realistic generation of samples
- Use complicated probability distributions
- Optimize multiple output for a single input
- Can do interpolation
- Work well with missing data

https://arxiv.org/pdf/1605.05396.pdf

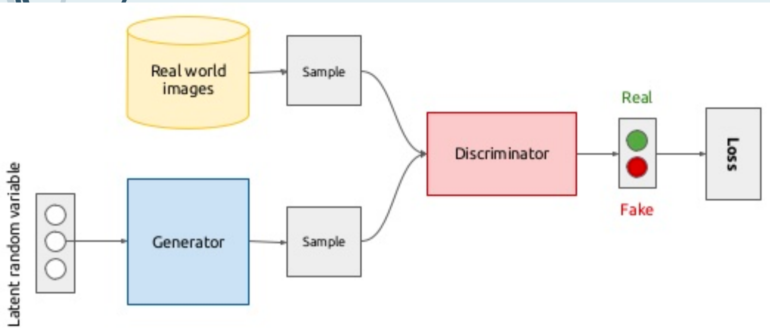


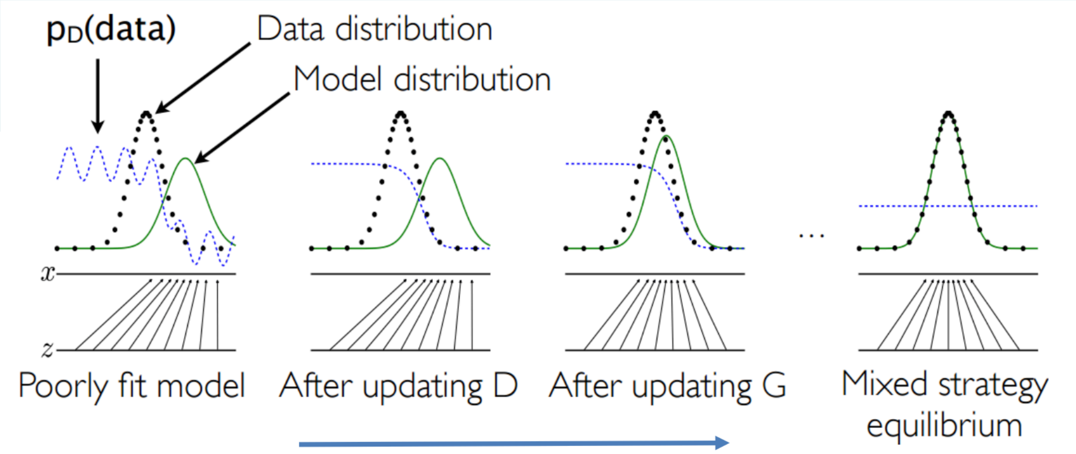
Ranzato, Susskind, Mnih, Hinton, IEEE CVPR 20

Generative adversarial networks

Simultaneously train two models:

- G(z) captures the data distribution
- D(x) estimates the probability that a sample came from the training data rather than G
- Training procedure for G(z) is to maximize the probability of D(x) making a mistake

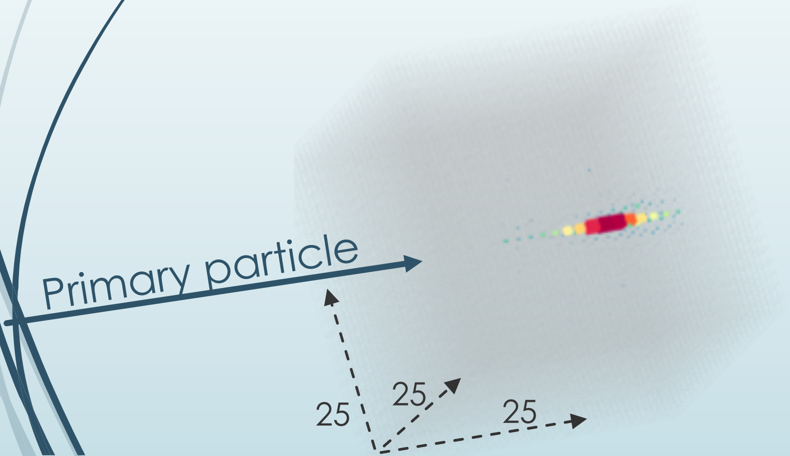


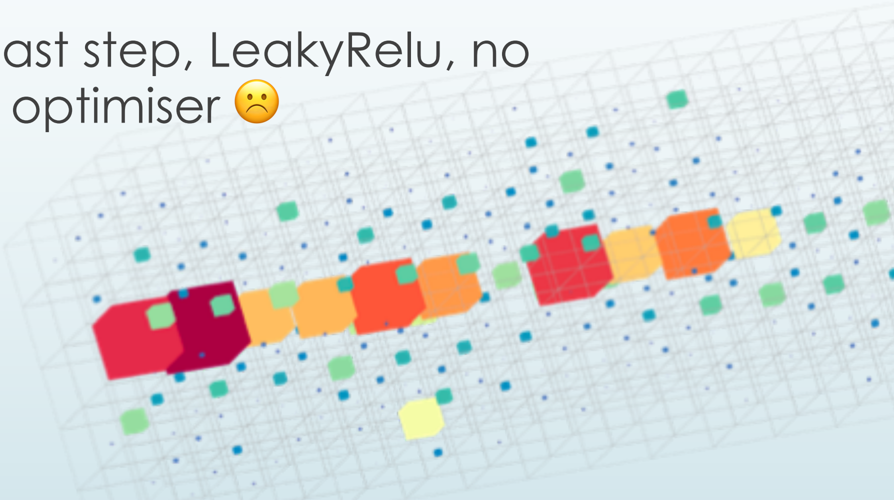


3dGAN for particle detectors

Data is essentially a 3D image

- Generator and Discriminator based on 3D convolutions
- Explored several “tips&tricks”
  - No batch normalisation in the last step, LeakyRelu, no hidden dense layers 😀, Adam optimiser ☹️





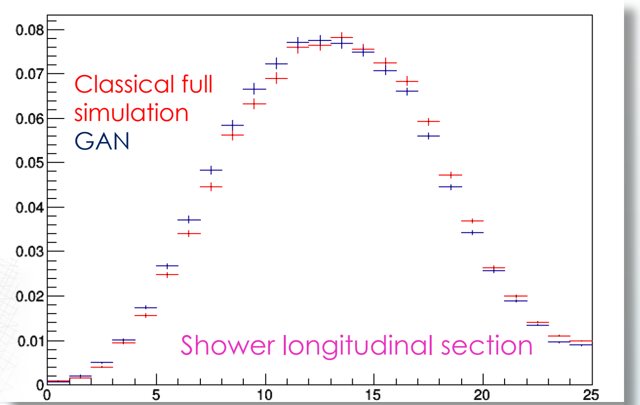
Geant4 π shower in LCD calorimeter

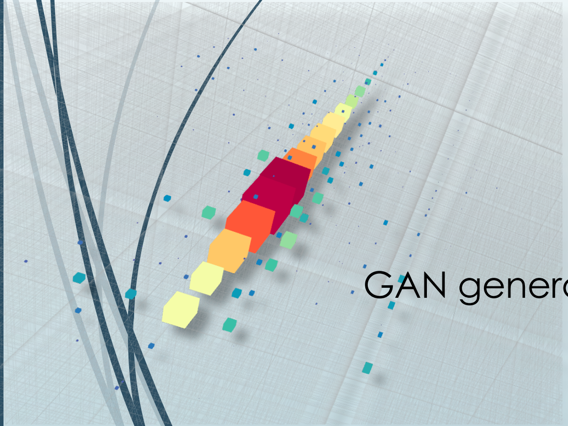
https://github.com/tpmccauley/ispy-hepml

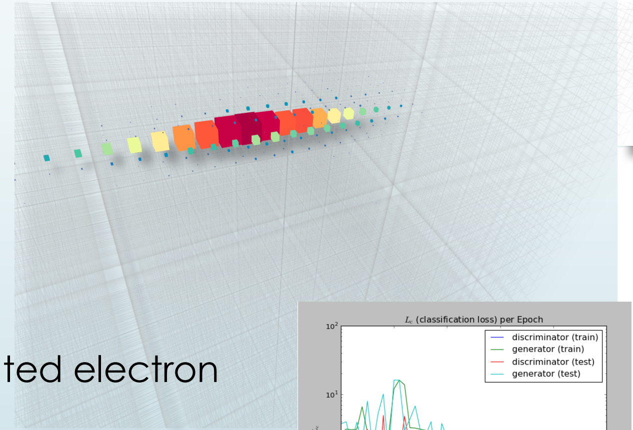
Some generated images

- First results look very promising!
- Qualitative results show no collapse problem

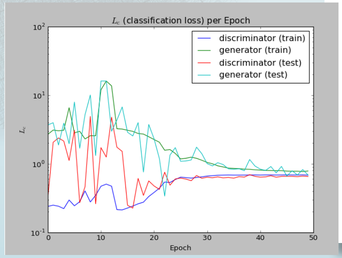
100 GeV electrons

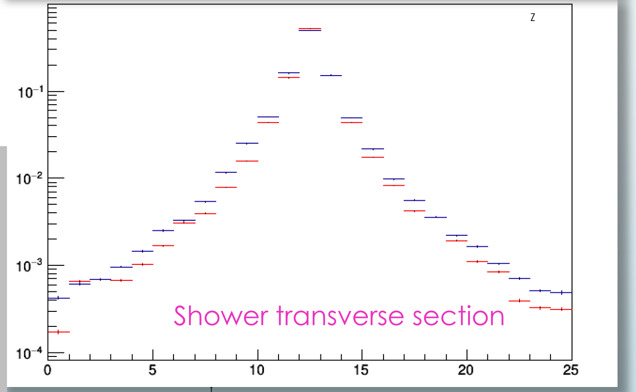






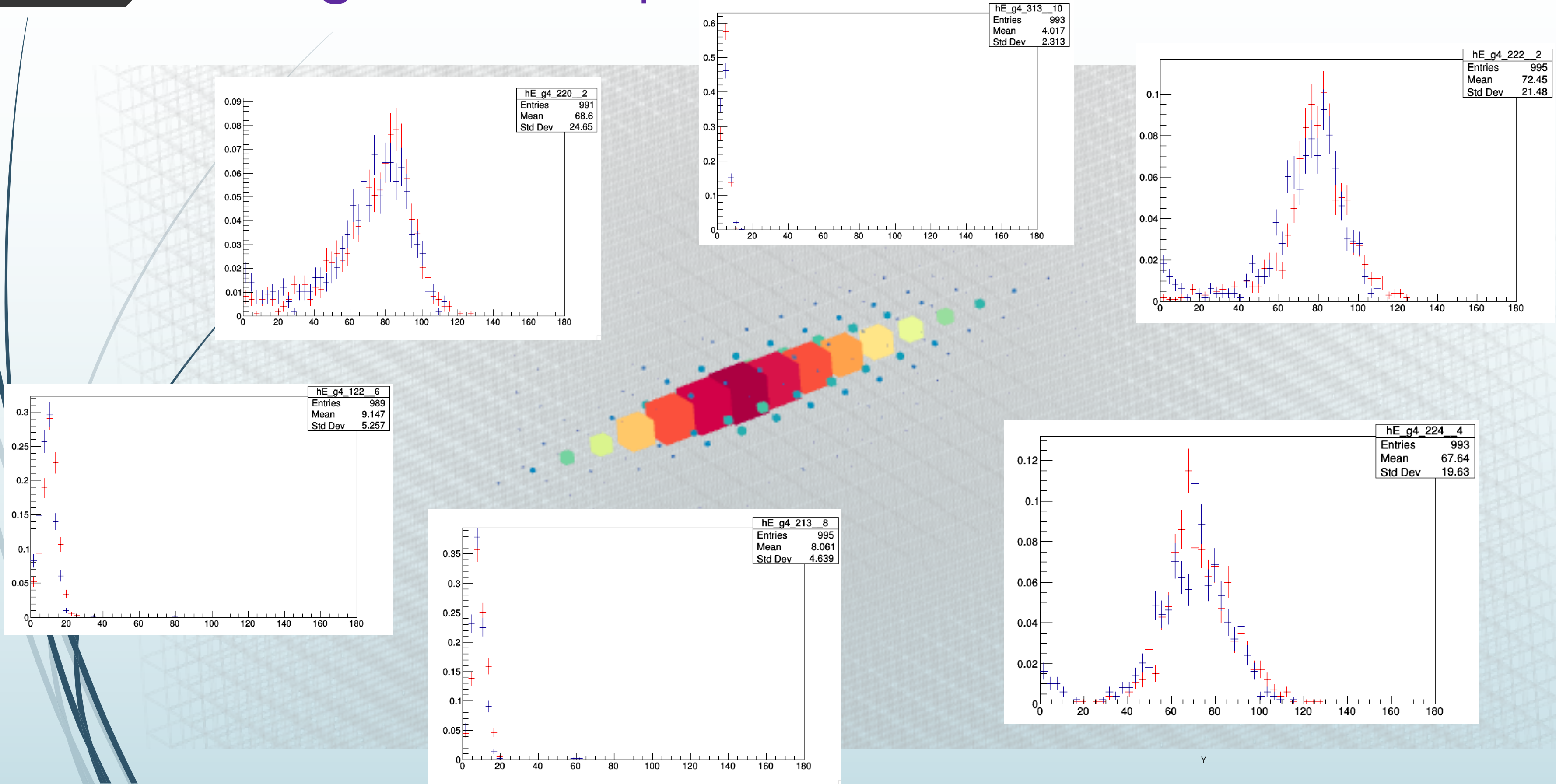
GAN generated electron





Preliminary

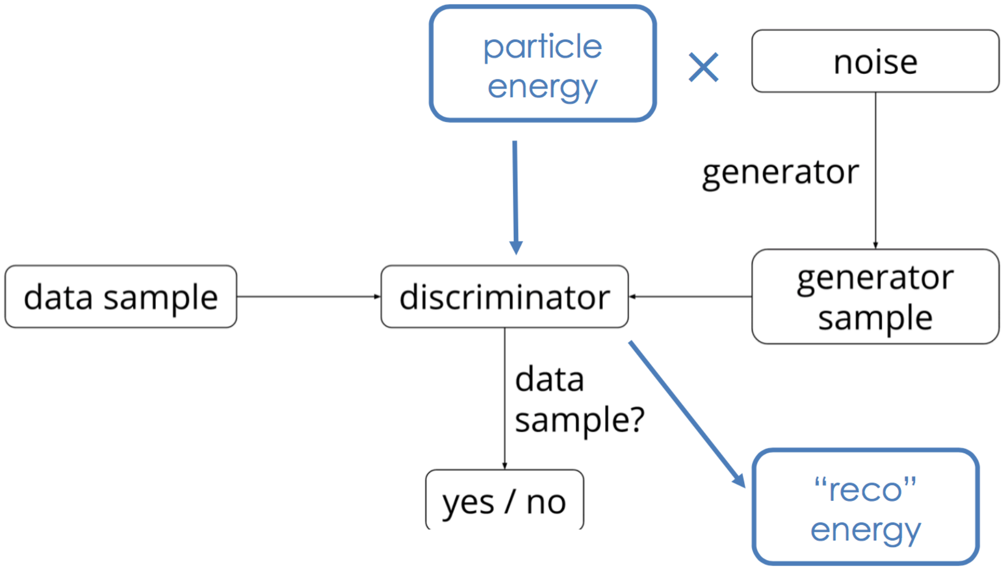
Single cell response



Conditioning on energy

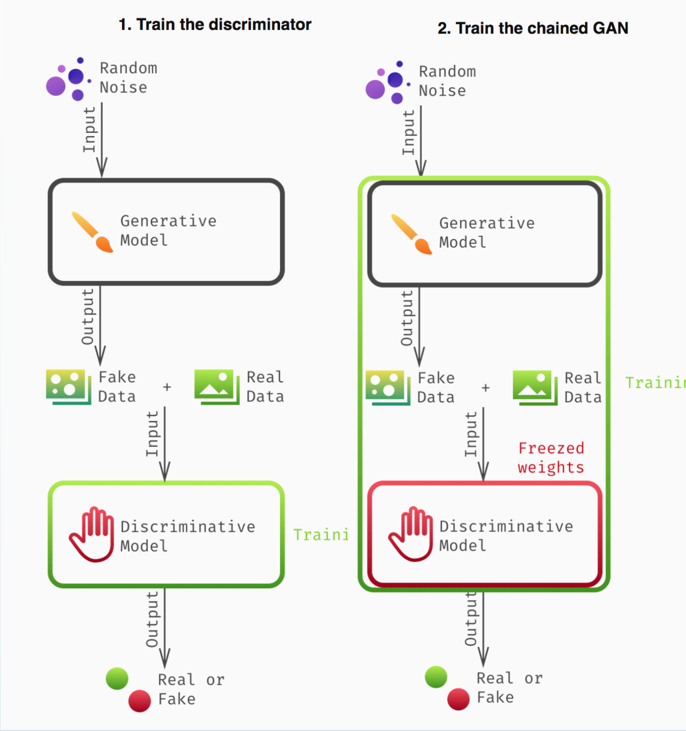
Training the generator and the discriminator using initial particle energy

- Discrete energy slices to test interpolation and extrapolation
- Test continuous spectrum
- Add other variables (primary entry point, angle, etc..)



Training time and multi-node scaling

- 3D GAN are not "out-of-the-box" networks
  - Complex training process
- Training time cannot be a bottleneck
  - Depending on the use case retraining might be necessary
  - Hyper-parameters scan and meta-optimization
  - Including additional variables will increase complexity
- Thanks to a collaboration with CINECA, Italy and Intel, we will **test multi-node scaling** on a cluster of Xeon Phi interconnected with Intel Omni-Path



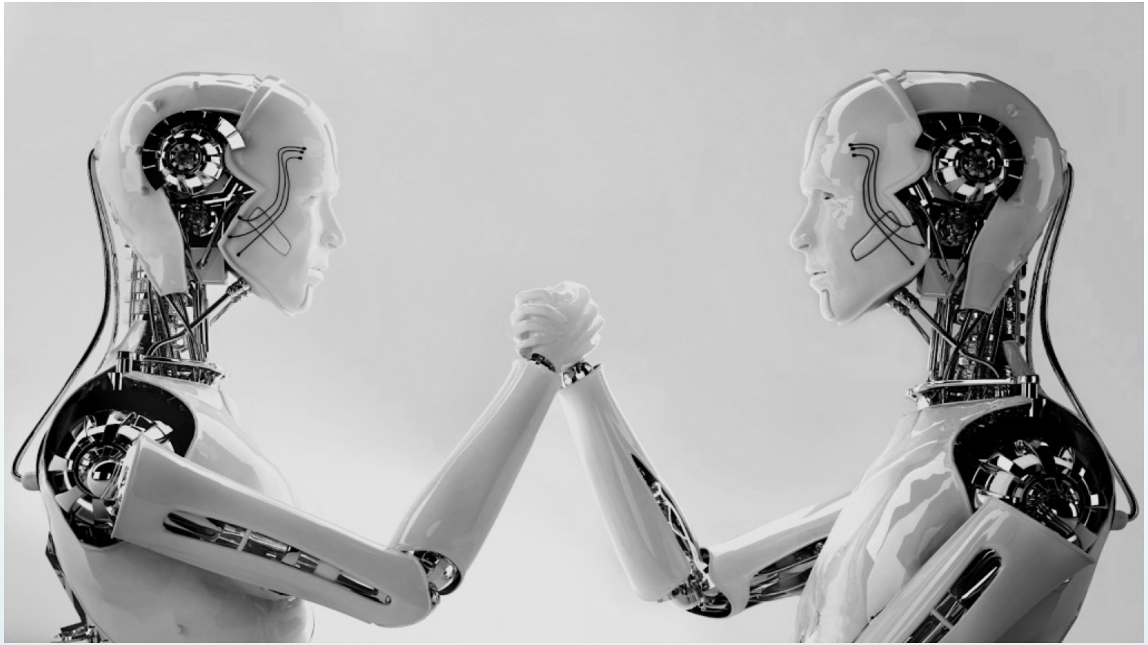
http://www.rricard.me/machine/learning/generative/adversarial/networks/2017/04/05/gans-part1.html

Summary

arXiv:170x.xxx

- One of the first 3D GAN implementations and results are very promising!
  - Detailed assessment of current performance and "resource costs" (training time/training samples)
  - Optimization, scaling and comparison to other models
- Looking forward to new software & hardware solutions!
  - Next-generation Intel Xeon "Skylake" and Intel Xeon Phi "Knights Mill"
  - Test inference dedicated hardware (integrated FPGA solution) Intel DLIA
- Prototype interface and ML proof of concept in GEANTV beta

Thank you!



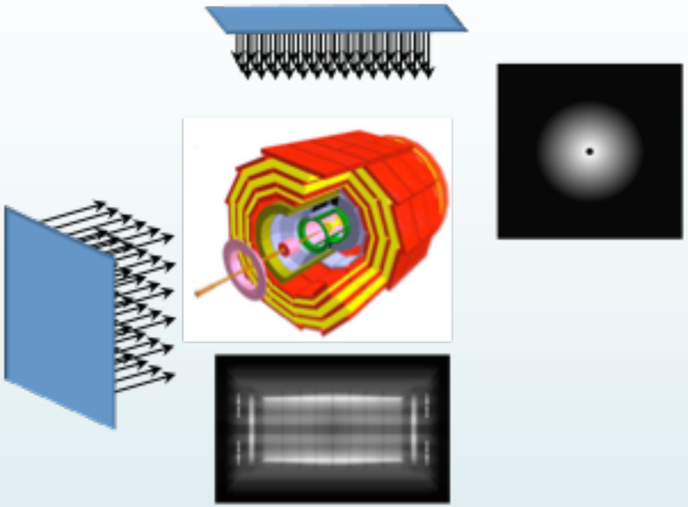
Questions?

References

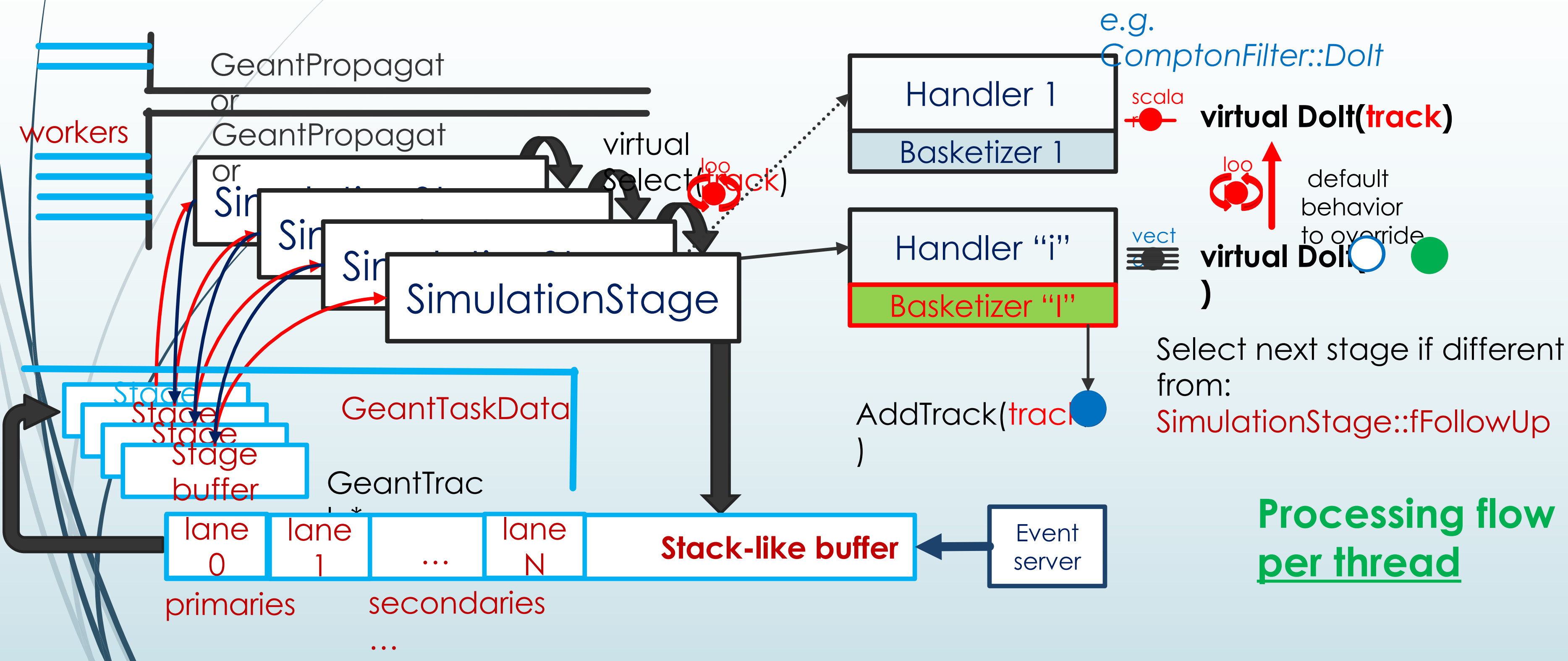
- Goodfellow et al. 2014
- Conditional GAN, arXiv: 1411.1744
- Deep Convolutional GAN, arXiv:1511.06434
- Auxiliary Classifier GAN, arXiv:1610.0958

Geometry: navigation benchmark

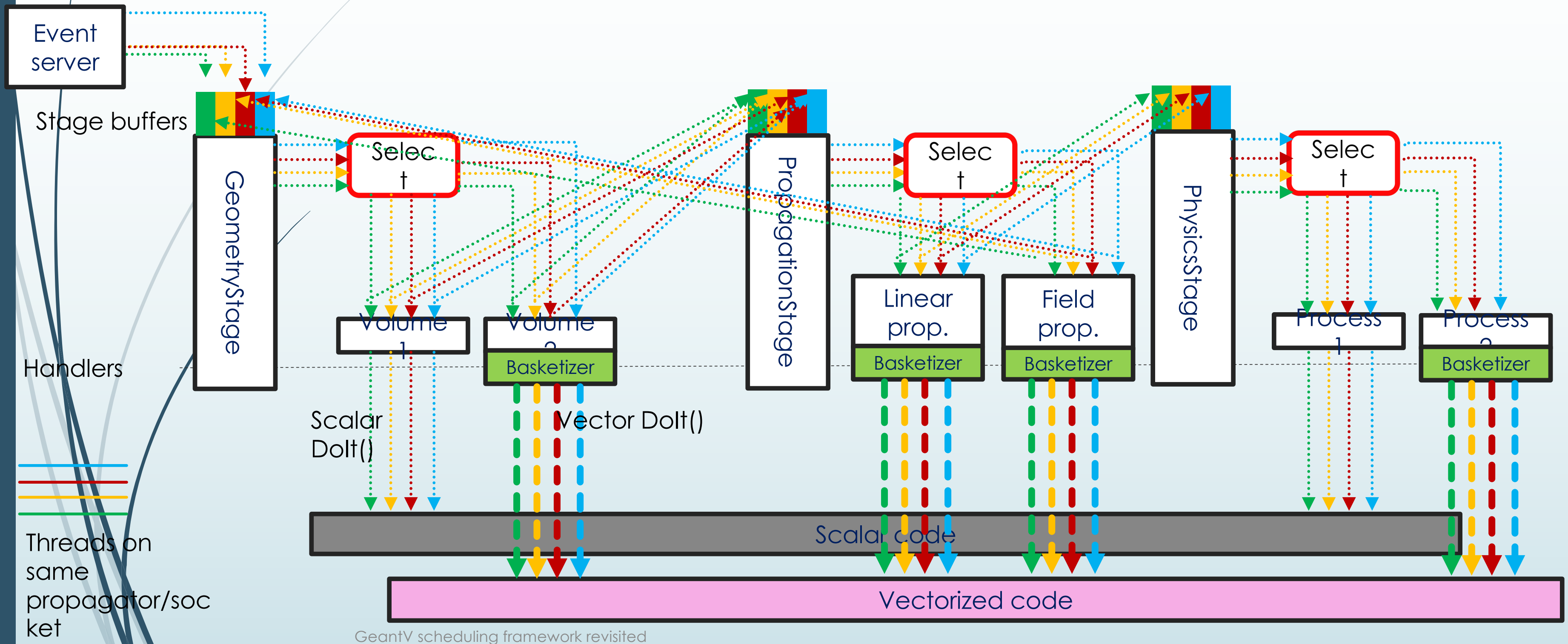
- X-Ray scan of a simple toy detector geometry
  - Concentric set of tubes emulating a tracker
- Trace one ray per pixel and reconstruct the image
- Test the global navigation
- Stress vector API + basket transport tracing multiple identical tracks through the same grid
- Test parallelism producing multiple identical images



GeantV version 3: A generic vector flow approach

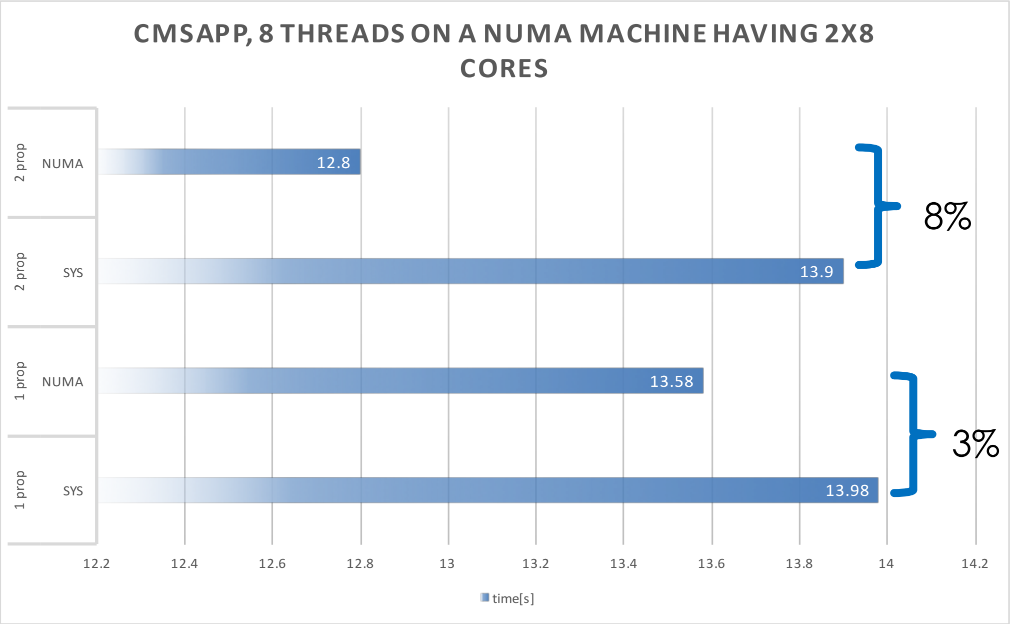


Processing flow per propagator/NUMA node



NUMA awareness

- Implemented using hwloc > 1.8
  - Enumerating NUMA nodes, cores, CPU's
  - Threads are bound to CPU's
- Compact thread policy within single node, scatter for different nodes
- Thread local data



We expect larger improvement on Intel Xeon Phi