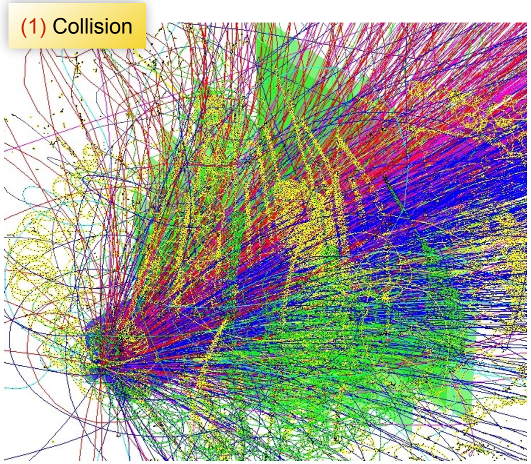
Heavy-Ion Physics
on Many-Core Computer Architectures
(CBM Experiment at FAIR)

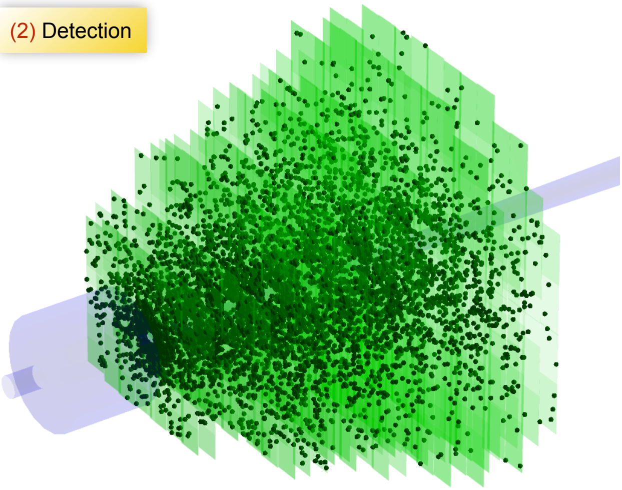
Perspectives of GPU Computing in Science

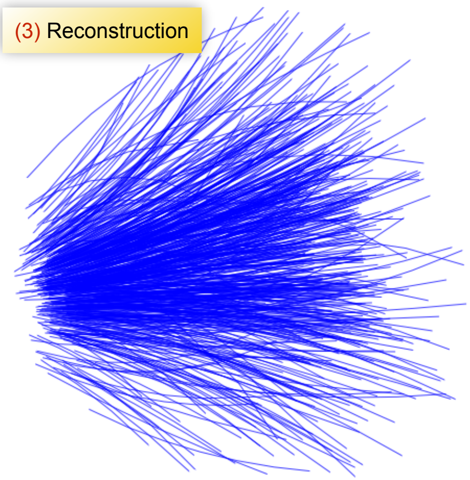
Ivan Kisel

Goethe-University Frankfurt am Main
FIAS Frankfurt Institute for Advanced Studies
GSI Helmholtz Center for Heavy Ion Research

Reconstruction Challenge in CBM at FAIR/GSI





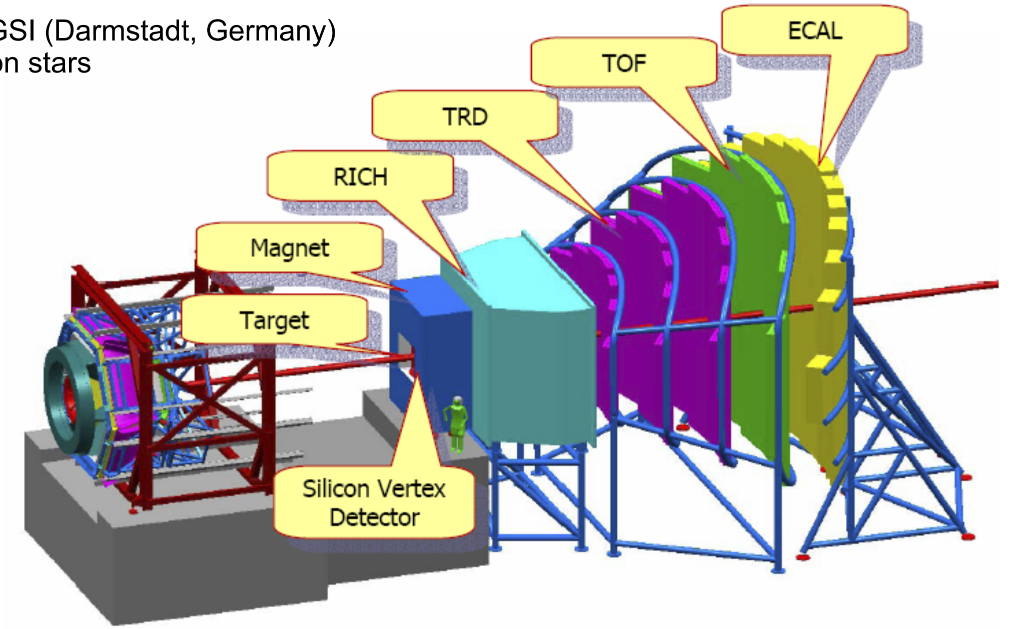


- Future Compressed Baryonic Matter experiment at FAIR/GSI (Darmstadt, Germany)
- Produce matter conditions similar to that in cores of neutron stars
- Fixed-target heavy-ion experiment
- 10^7 Au+Au collisions/sec
- ~ 1000 charged particles/collision
- Non-homogeneous magnetic field
- Double-sided strip detectors (85% fake space-points)

Full event reconstruction will be done on-line at the First-Level Event Selection (FLES) and off-line using the same FLES reconstruction package.

Cellular Automaton (CA) Track Finder
Kalman Filter (KF) Track Fitter
KF short-lived Particle Finder

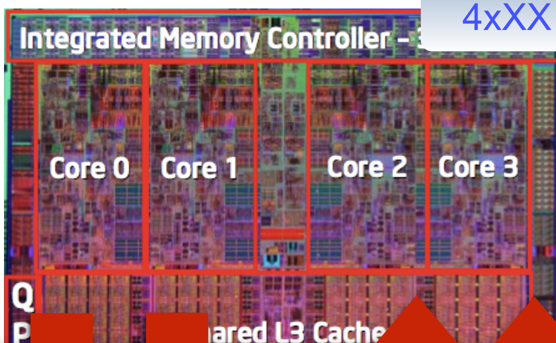
All reconstruction algorithms are vectorized and parallelized.



Many-Core CPU/GPU Architectures

Intel/AMD CPU

4xXX cores



- Optimiz r lo tency access ache ata sets
- Control for -of-order and cula execution

Math

Memory

Parallelism

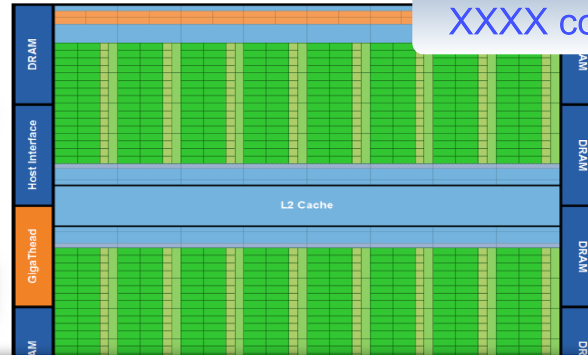
Math

Memory

#Cores

Nvidia/ATI GPU

XXXX cores



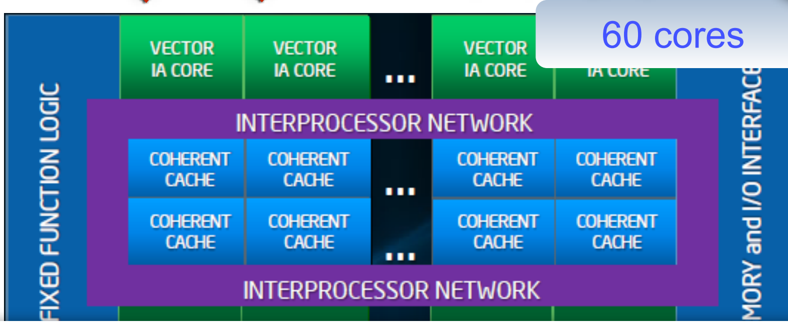
- ptimized for data-parallel, throughput computation
- lore transistors dedicated to computation

Stability

Memory

Intel Phi

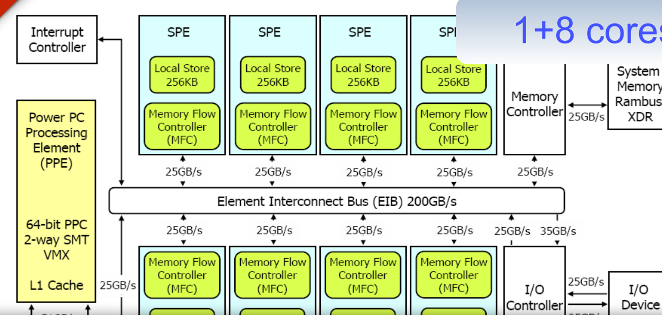
60 cores



- Many Integrated Cores architecture announced at ISC10 (June 2010)
- Based on the x86 architecture
- Many-cores + 4-way multithreaded + 512-bit wide vector unit

IBM Cell

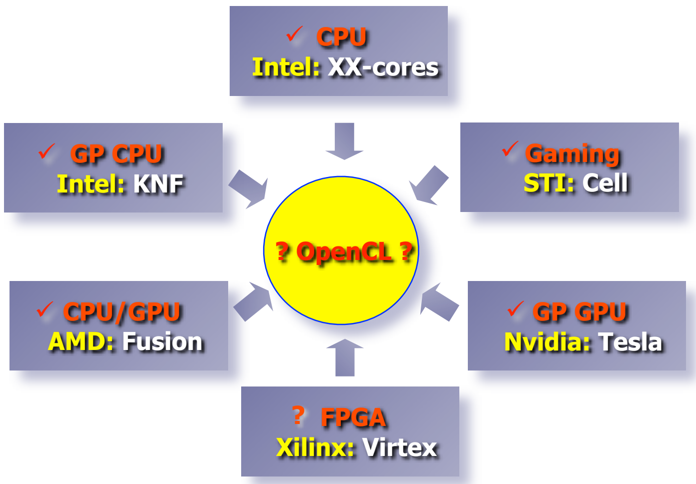
1+8 cores



- General purpose RISC processor (PowerPC)
- 8 co-processors (SPE, Synergistic Processor Elements)
- 128-bit wide SIMD units

Future systems are heterogeneous, but using the same code

CPU/GPU Programming Frameworks

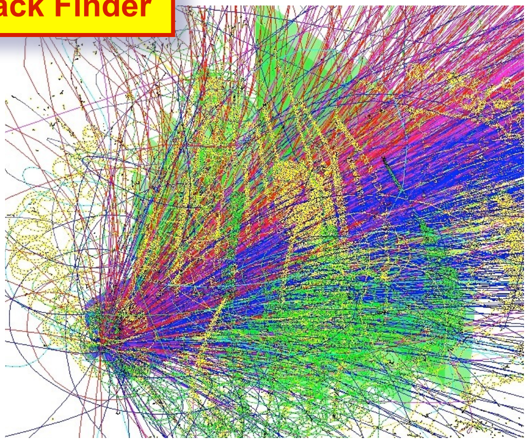


- Headers and Vector classes (Vc)
  - Overload of C operators with SIMD/SIMT instructions
  - Uniform approach to all CPU/GPU families
  - Uni-Frankfurt/FIAS/GSI
- OpenMP (Open Multi-Processing)
  - API (application programming interface) for multi-processing programming
  - Supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran
  - Most of the constructs in OpenMP are compiler directives
  - Programming model – Fork-Join parallelism
- OpenCL (Open Computing Language)
  - Open standard for generic programming
  - Extension to the C language
  - Supposed to work on any hardware
  - Usage of specific hardware capabilities by extensions
- NVIDIA CUDA (Compute Unified Device Architecture)
  - Defines hardware platform
  - Generic programming
  - Extension to the C language
  - Explicit memory management
  - Programming on thread level

Choice of CPU/GPU/Programming is a practical question

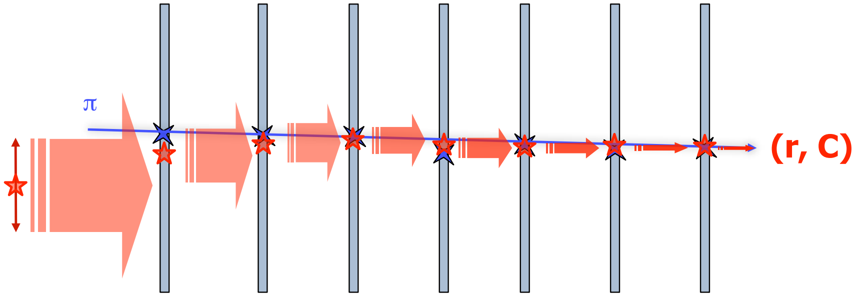
Stages of Event Reconstruction

1 Track Finder



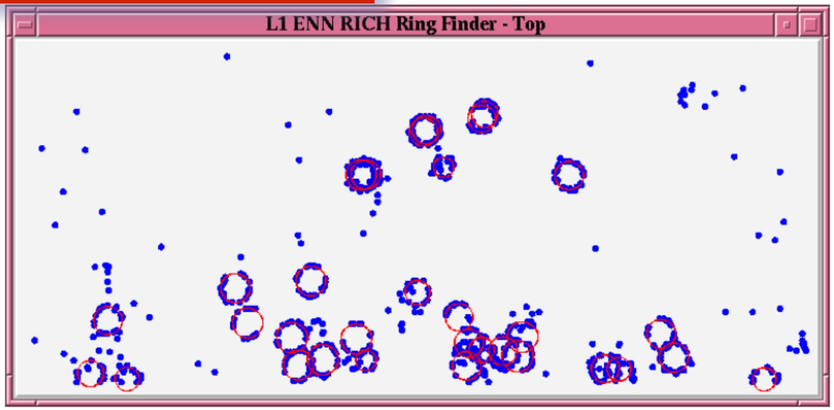
- Cellular Automaton
- Track Following

2 Track Fitter



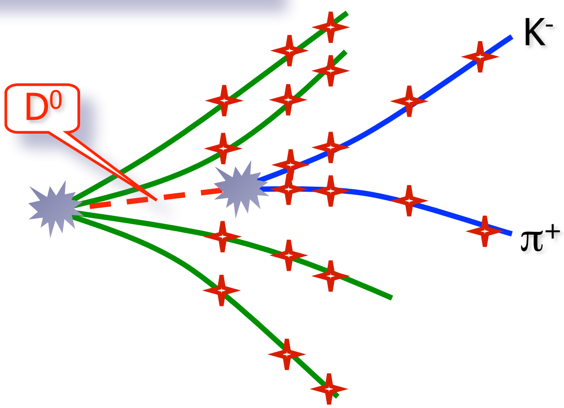
- Kalman Filter

3 Ring Finder (Particle ID)



- Hough Transformation
- Elastic Neural Net

4 Short-Lived Particles Finder



- Kalman Filter

Kalman Filter (KF) Track Fit Library

Kalman Filter Methods

Kalman Filter Tools:
- KF Track Fitter
- KF Track Smoother
- Deterministic Annealing Filter

Kalman Filter Approaches:
- Conventional DP KF
- Conventional SP KF
- Square-Root SP KF
- UD-Filter SP
- Gaussian Sum Filter

Track Propagation:
- Runge-Kutta
- Analytic Formula

Implementations

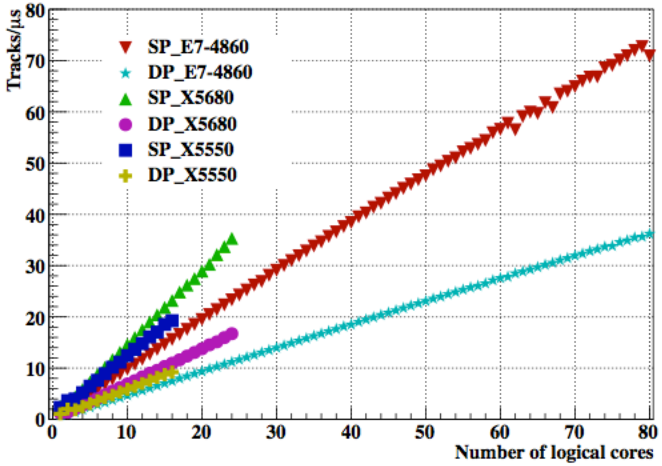
Vectorization (SIMD):
- Header Files
- Vc Vector Classes
- ArBB Array Building Blocks
- OpenCL

Parallelization (many-cores):
- Open MP
- ITBB
- ArBB
- OpenCL

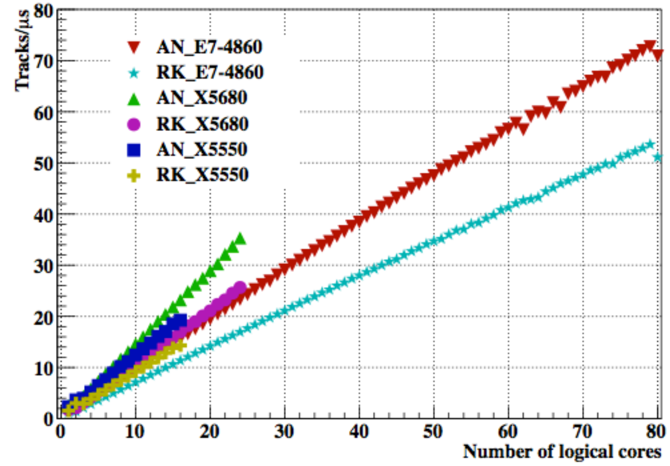
Precision:
- single precision SP
- double precision DP

Comp. Phys. Comm. 178 (2008) 374-383

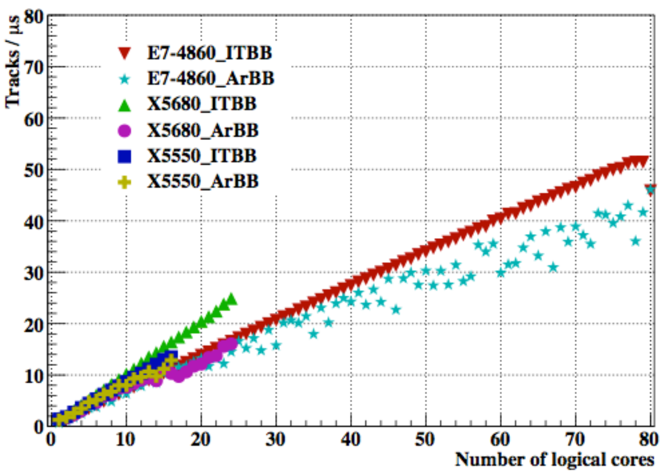
Conventional KF DP vs. SP



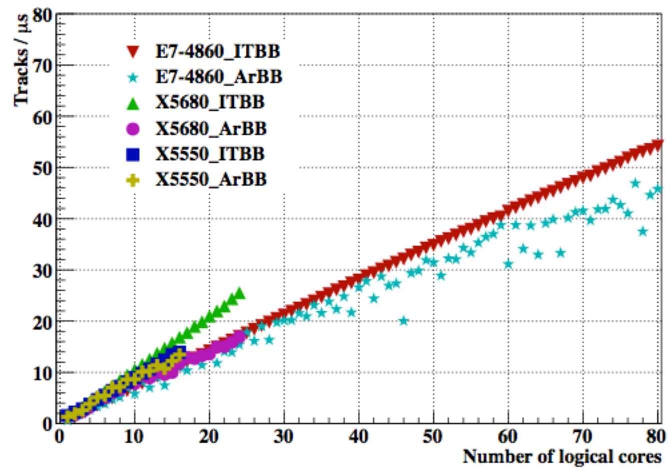
Conventional KF RK4 vs. Analytical



Square-Root KF



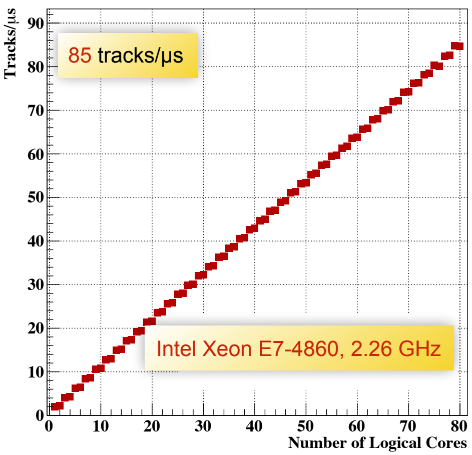
UD KF

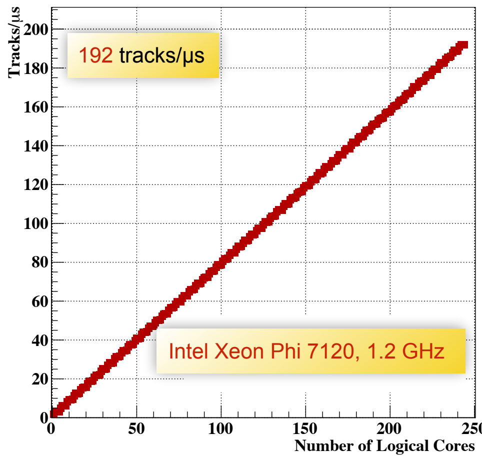


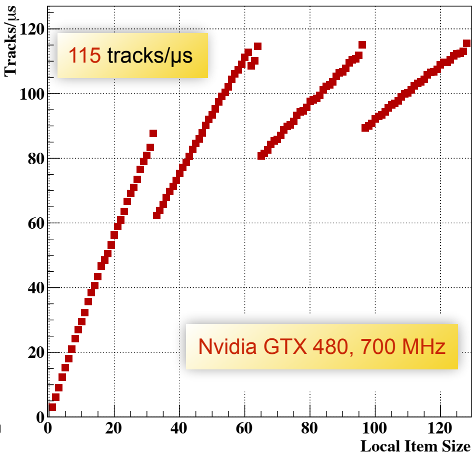
Strong many-core scalability of the Kalman filter library

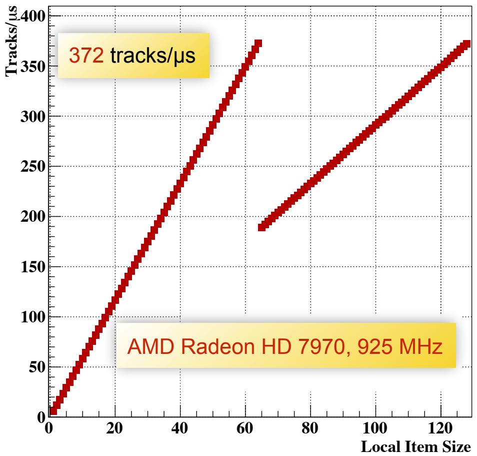
with I. Kulakov, H. Pabst* and M. Zyzak (*Intel)

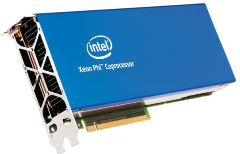
Kalman Filter (KF) Track Fit Library







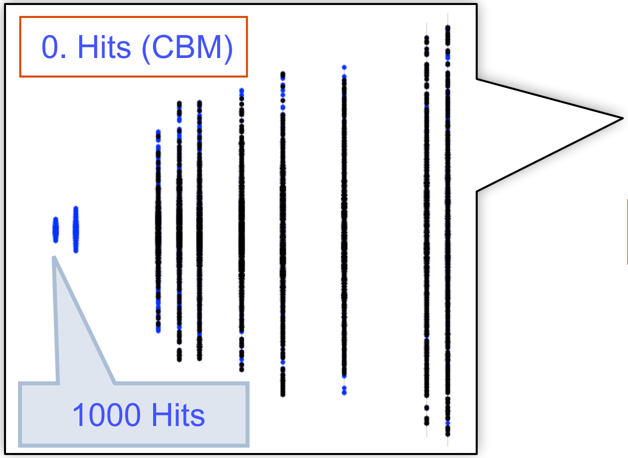


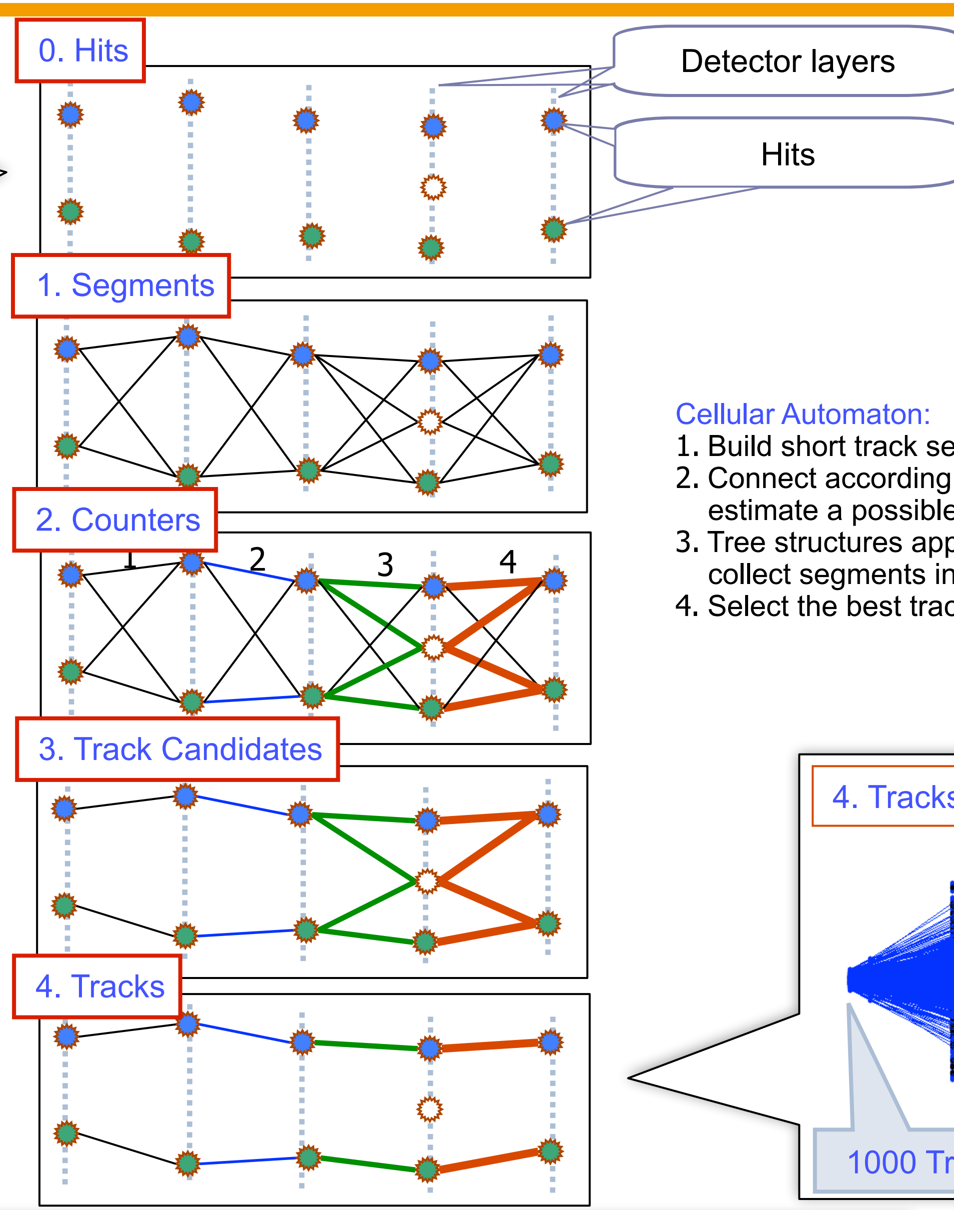


- Scalability with respect to the number of logical cores in a CPU is one of the most important parameters of the algorithm.
- The scalability on the Intel Xeon Phi coprocessor is similar to the CPU, but running four threads per core instead of two.
- In case of the graphic cards the set of tasks is divided into working groups of size *local item size* and distributed among compute units (or streaming multiprocessors) and the load of each compute unit is of the particular importance.

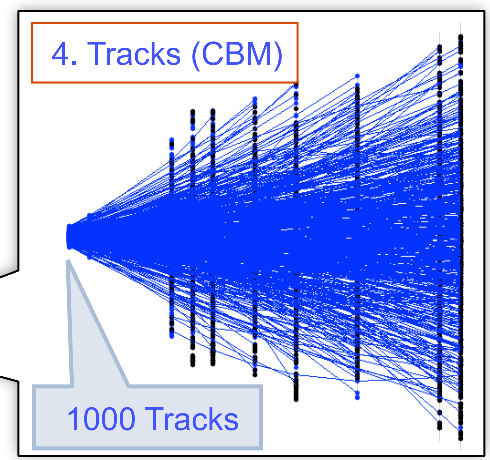
Full portability of the Kalman filter library

Cellular Automaton (CA) Track Finder





Cellular Automaton:
1. Build short track segments.
2. Connect according to the track model, estimate a possible position on a track.
3. Tree structures appear, collect segments into track candidates.
4. Select the best track candidates.



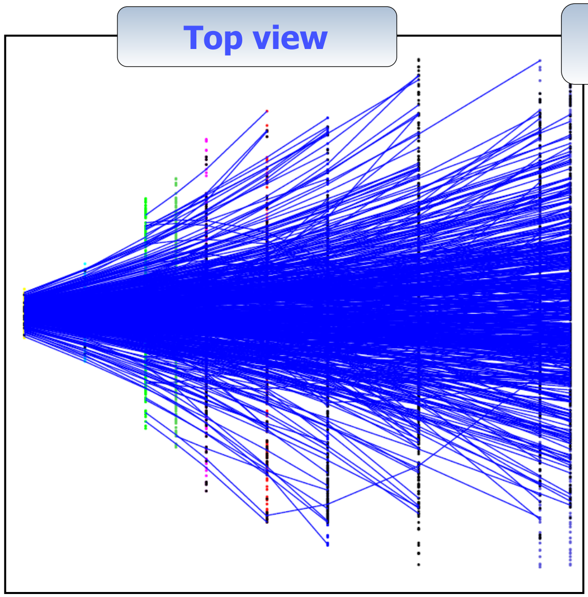
Cellular Automaton:

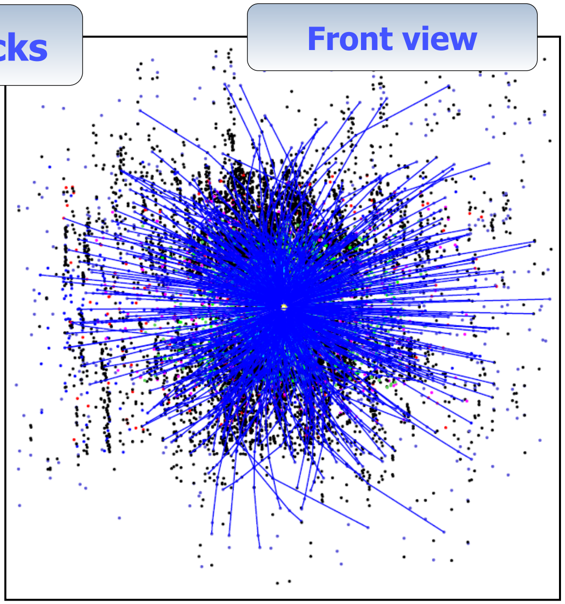
- local w.r.t. data
- intrinsically parallel
- extremely simple
- very fast

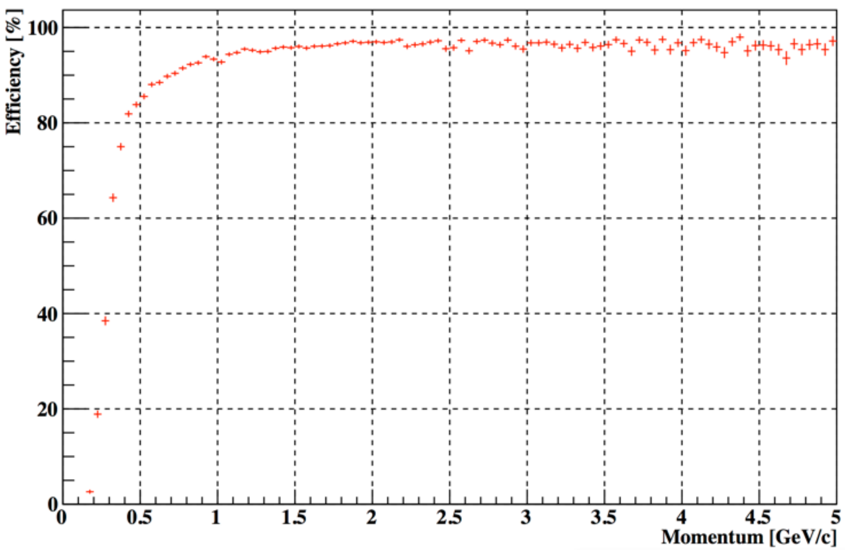
Perfect for many-core CPU/GPU !

Useful for complicated event topologies with large combinatorics and for parallel hardware

CA Track Finder: Efficiency





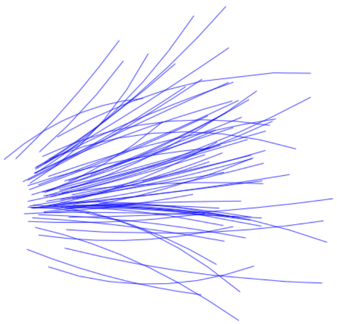


	Efficiency, % mbias	Efficiency, % central
Primary high-p tracks	97.1	96.2
Primary low-p tracks	90.4	90.7
Secondary high-p tracks	81.2	81.4
Secondary low-p tracks	51.1	50.6
All tracks	88.5	88.3
Clone level	0.2	0.2
Ghost level	0.7	1.5
Reconstructed tracks/event	120	591
Time/event/core	8.2 ms	57 ms

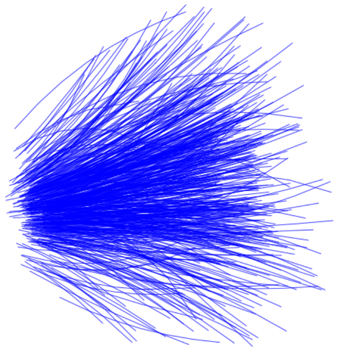
Efficient and clean event reconstruction

CA Track Finder at High Track Multiplicity

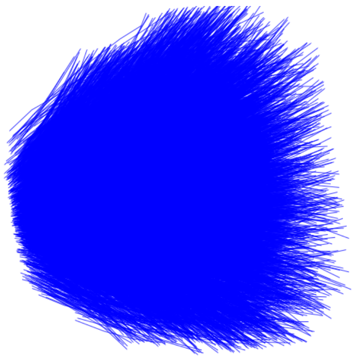
A number of minimum bias events is gathered into a group (super-event), which is then treated by the CA track finder as a single event



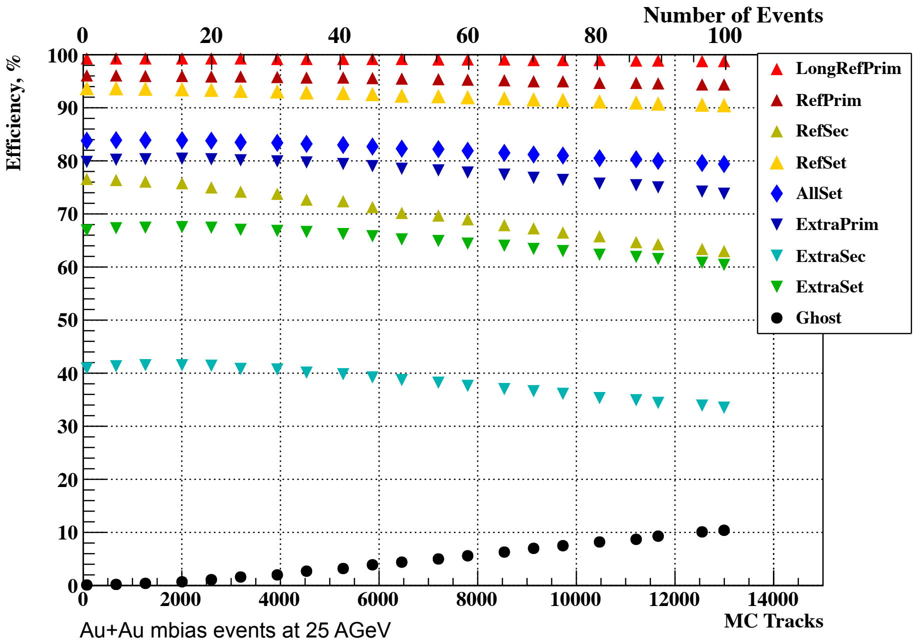
1 mbias event, $\langle N_{reco} \rangle$ = 109



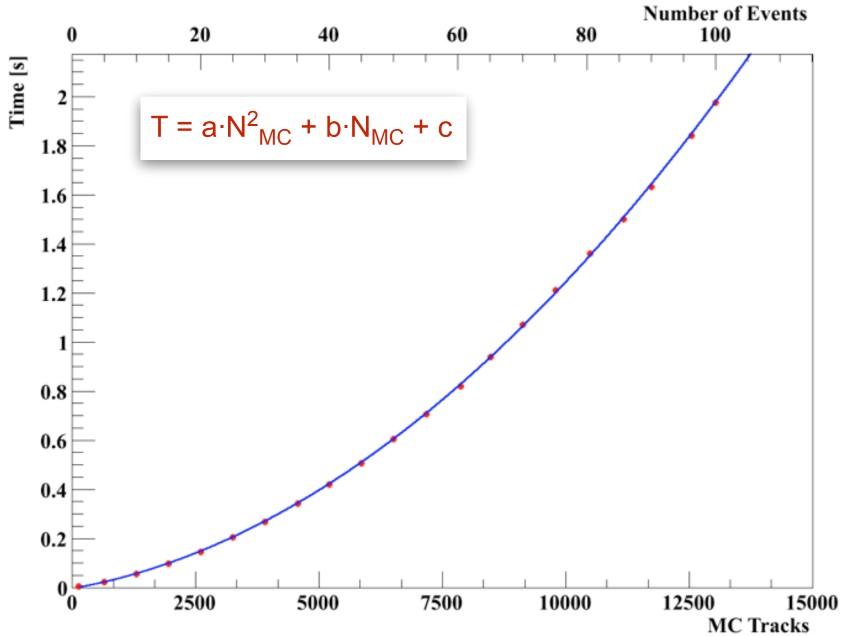
5 mbias events, $\langle N_{reco} \rangle$ = 572



100 mbias events, $\langle N_{reco} \rangle$ = 10340

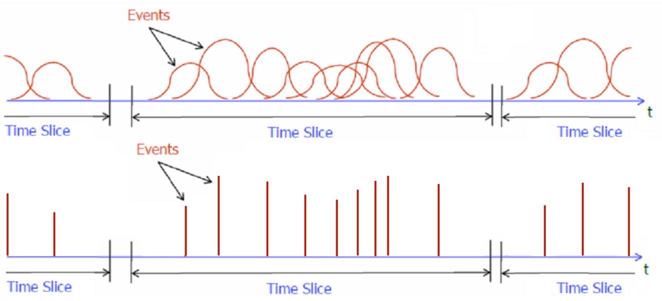


Au+Au mbias events at 25 AGeV



Stable reconstruction efficiency and time as a second order polynomial w.r.t. to track multiplicity

Time-based (4D) Track Reconstruction with CA Track Finder



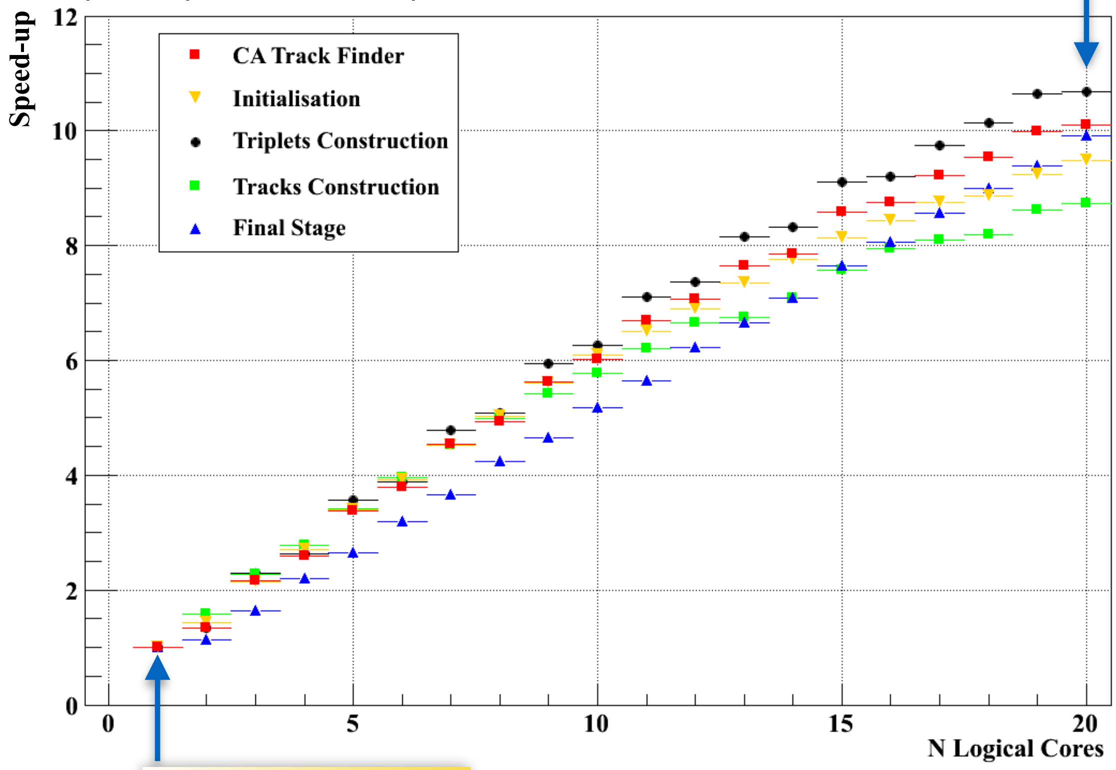
- The beam in the CBM will have no bunch structure, but continuous.
- Measurements in this case will be 4D (x, y, z, t).
- Significant overlapping of events in the detector system.
- Reconstruction of time slices rather than events is needed.

Stage of the algorithm	% of total execution time
Initialisation	8
Triplets construction	64
Tracks construction	15
Final cleaning	13

Efficiency, %	3D	3+1 D	4D
All tracks	83.8	80.4	83.0
Primary high-p	96.1	94.3	92.8
Primary low-p	79.8	76.2	83.1
Secondary high-p	76.6	65.1	73.2
Secondary low-p	40.9	34.9	36.8
Clone level	0.4	2.5	1.7
Ghost level	0.1	8.2	0.3
Time/event/core, ms	8.2	31.5	8.5

Total CA time = 84 ms

Speed-up factor due to parallelization within the time-slice



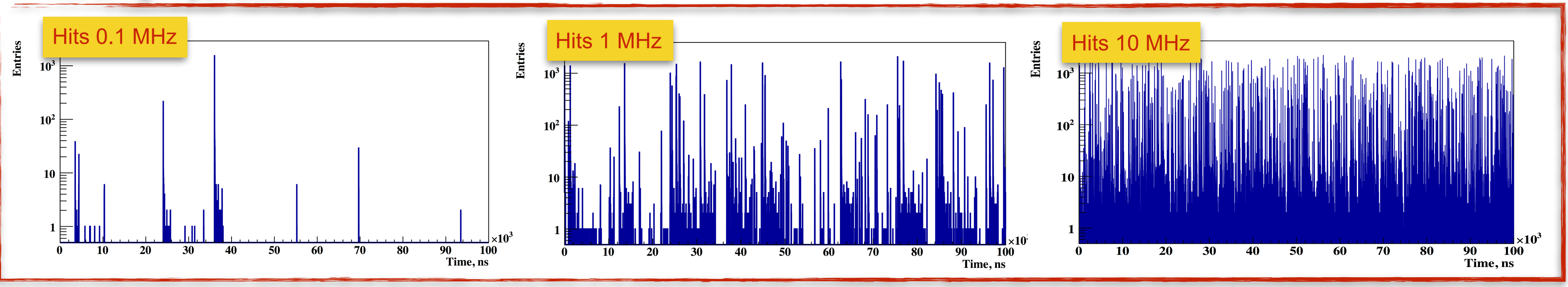
Total CA time = 849 ms

100 mbias events in a time-slice

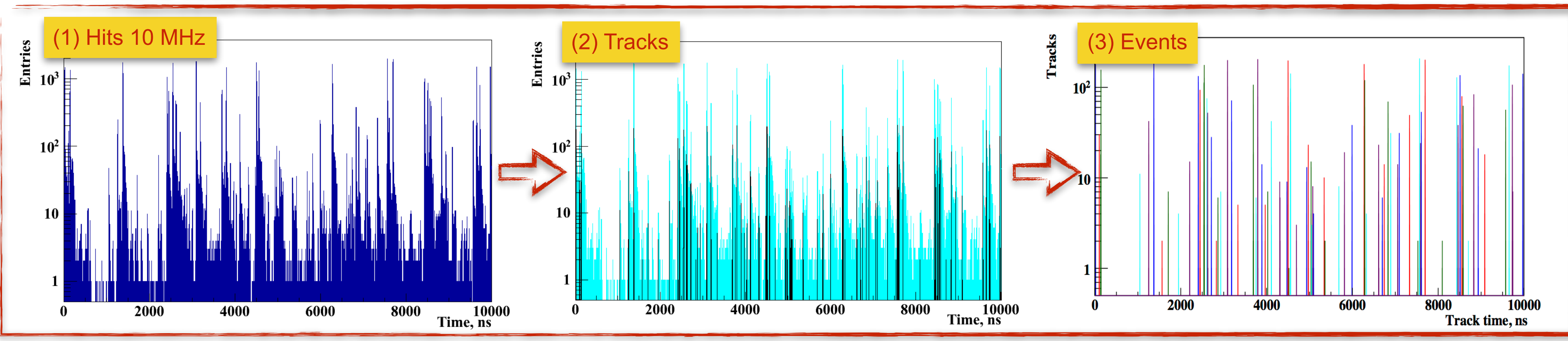
4D event building is scalable with the speed-up factor of 10.1; 3D reconstruction time 8.2 ms/event is recovered in 4D case

4D Event Building at 10 MHz

Hits at high input rates

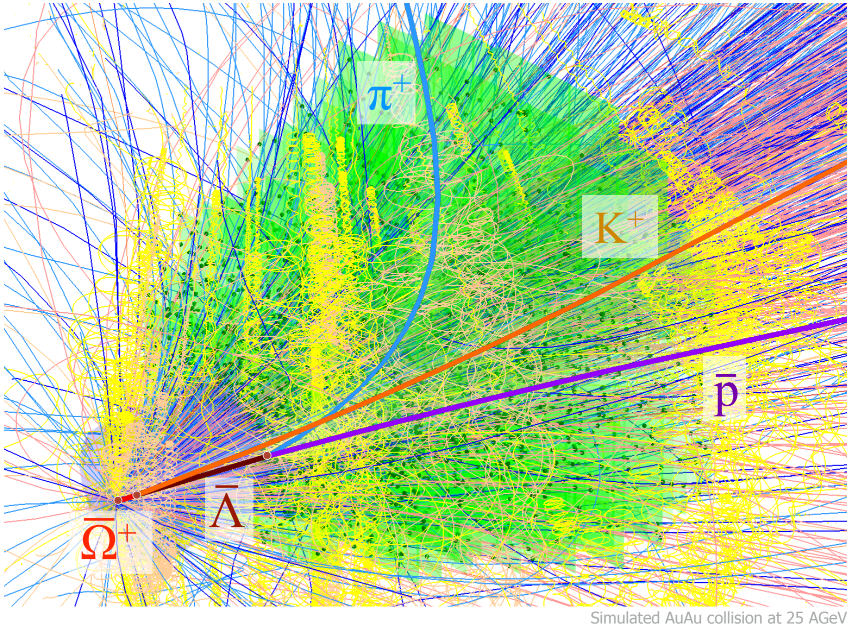


From hits to tracks to events



Reconstructed tracks clearly represent groups, which correspond to the original events
83% of single events, no splitted events, further analysis with TOF information at the vertexing stage

KF Particle: Reconstruction of Decayed Particles



$$\bar{\Omega}^+ \rightarrow \bar{\Lambda}\, K^+$$
$$\hookrightarrow \bar{p}\, \pi^+$$

```
KFParticle Lambda(P, Pi);                  // construct anti Lambda
Lambda.SetMassConstraint(1.1157);          // improve momentum and mass
KFParticle Omega(K, Lambda);               // construct anti Omega
PV -= (P, Pi, K);                          // clean the primary vertex
PV += Omega;                               // add Omega to the primary vertex
Omega.SetProductionVertex(PV);             // Omega is fully fitted
(K, Lambda).SetProductionVertex(Omega);    // K, Lambda are fully fitted
(P, Pi).SetProductionVertex(Lambda);       // p, pi are fully fitted
```

State vector — Position, direction, momentum and energy

$$r = \{ x, y, z, p_x, p_y, p_z, E \}$$

Concept:

- Mother and daughter particles have the same state vector and are treated in the same way
- Reconstruction of decay chains
- Kalman filter based
- Geometry independent
- Vectorized
- Uncomplicated usage

Functionality:

- Construction of short-lived particles
- Addition and subtraction of particles
- Transport
- Calculation of an angle between particles
- Calculation of distances and deviations
- Constraints on mass, production point and decay length
- KF Particle Finder

KFParticle provides uncomplicated approach to physics analysis (used in CBM, ALICE and STAR)

KF Particle Finder Algorithm

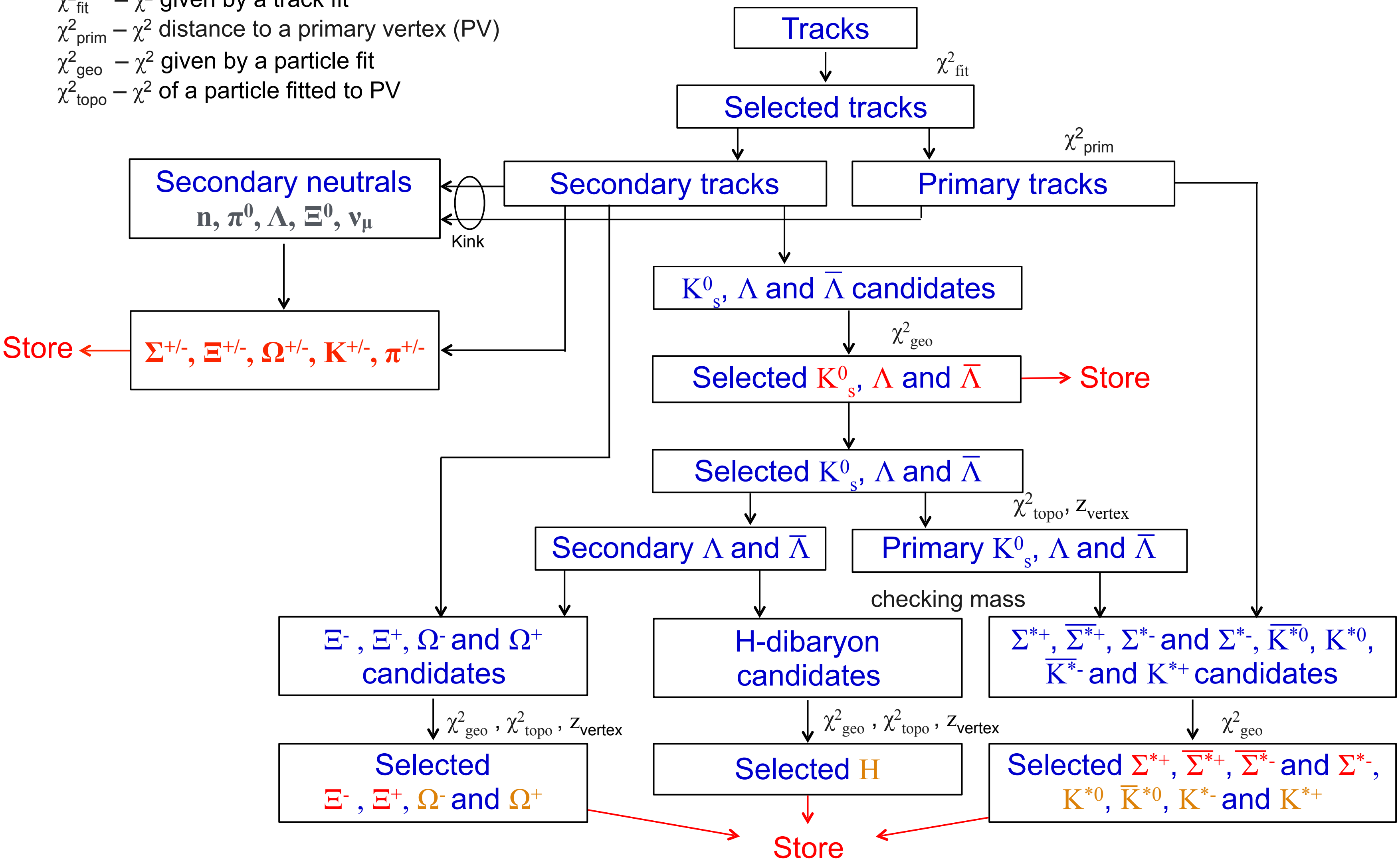
Selection criteria:

χ^2_{fit} – χ^2 given by a track fit

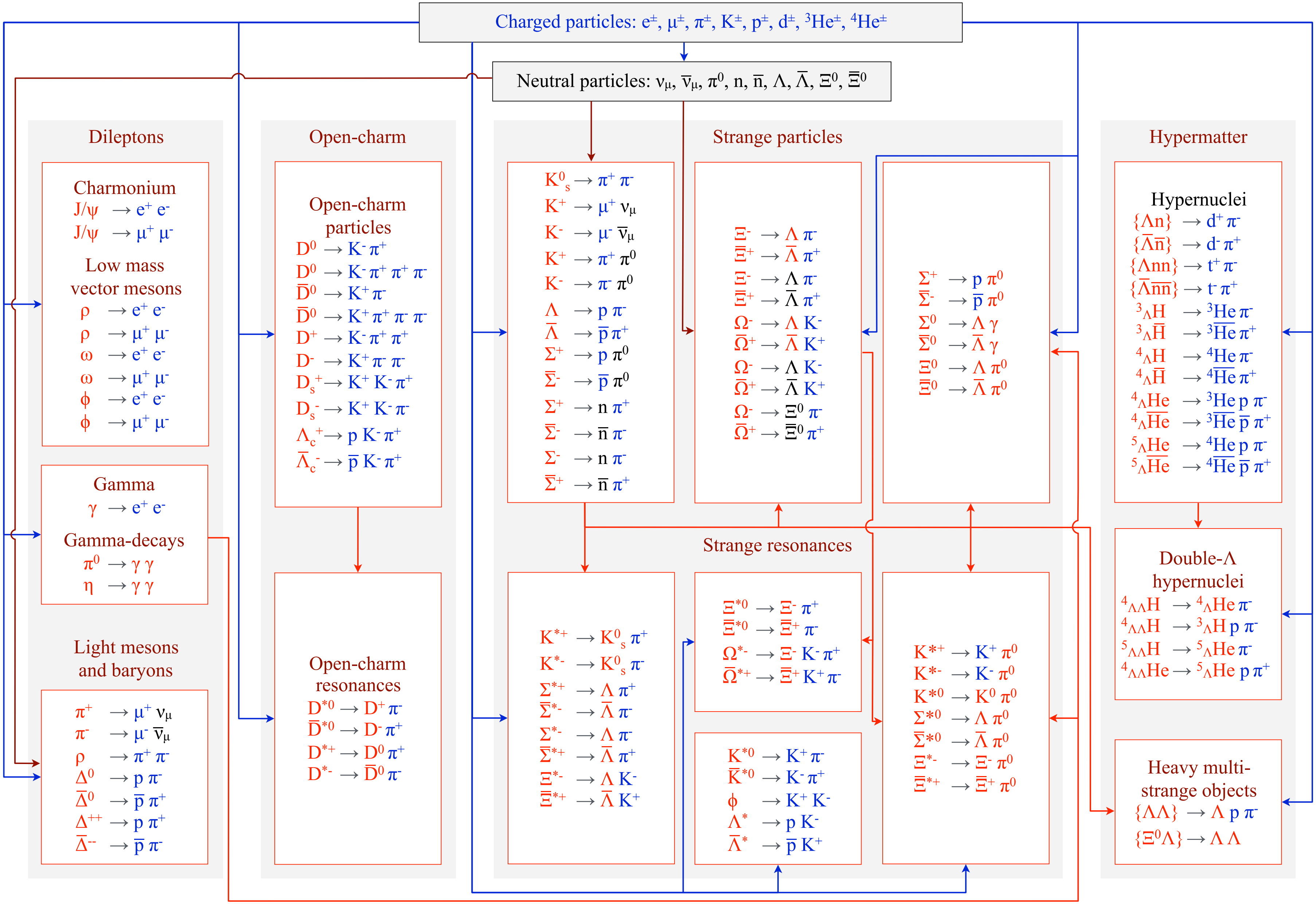
χ^2_{prim} – χ^2 distance to a primary vertex (PV)

χ^2_{geo} – χ^2 given by a particle fit

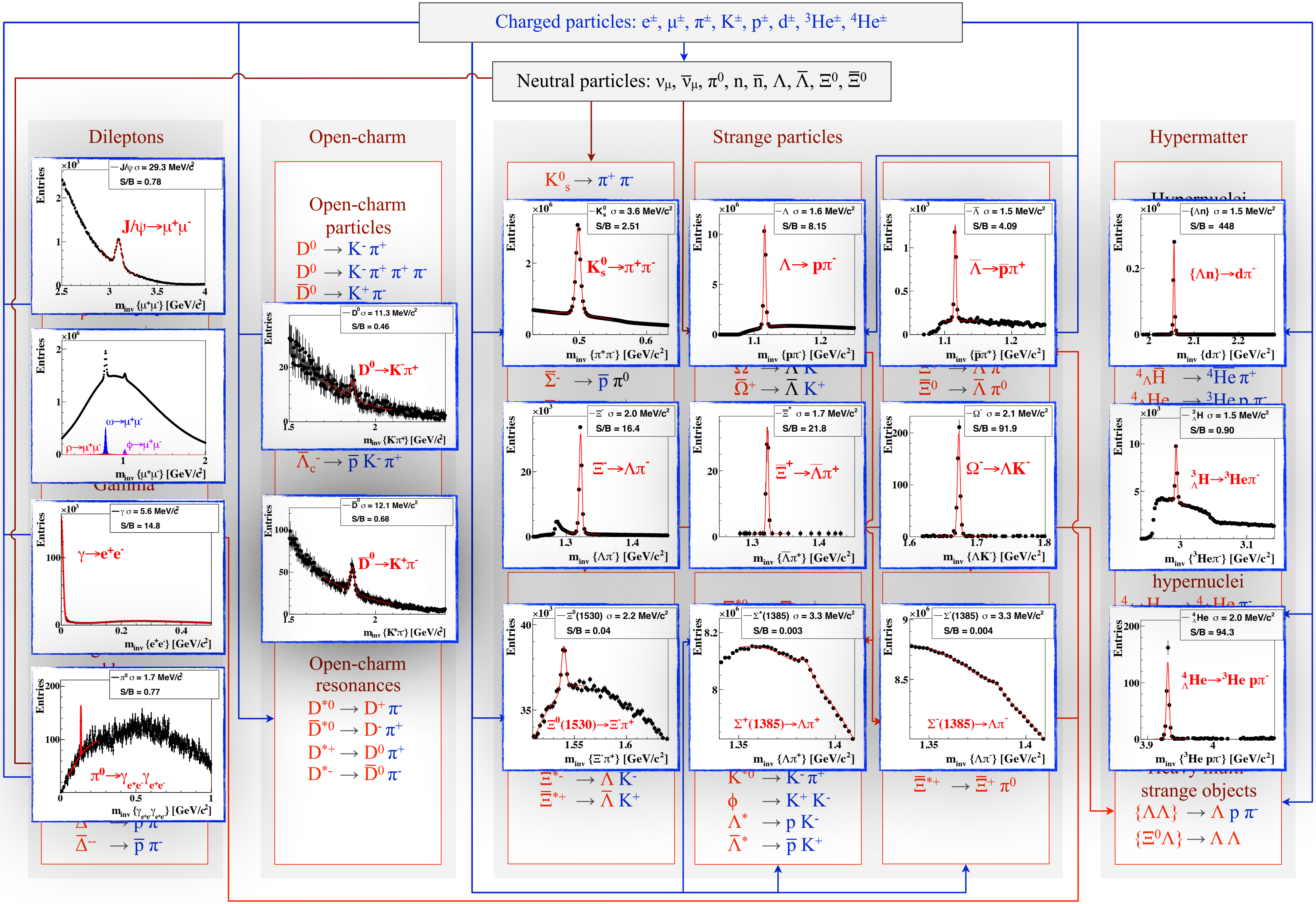
χ^2_{topo} – χ^2 of a particle fitted to PV



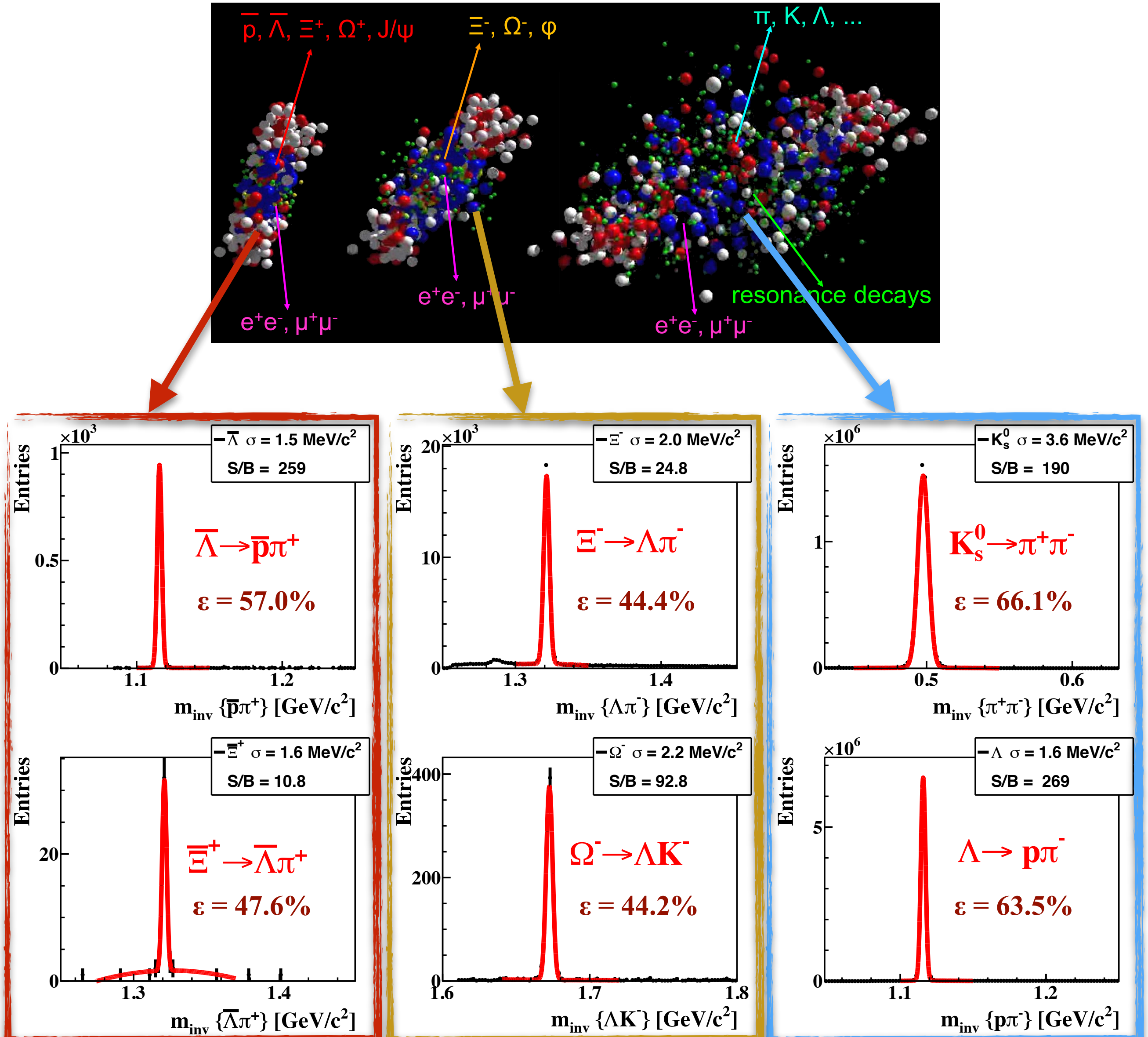
KF Particle Finder for Physics Analysis and Selection



KF Particle Finder for Physics Analysis and Selection



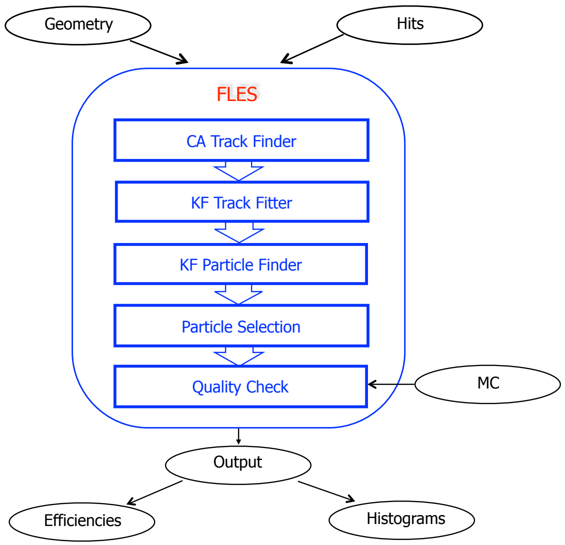
Clean Probes of Collision Stages

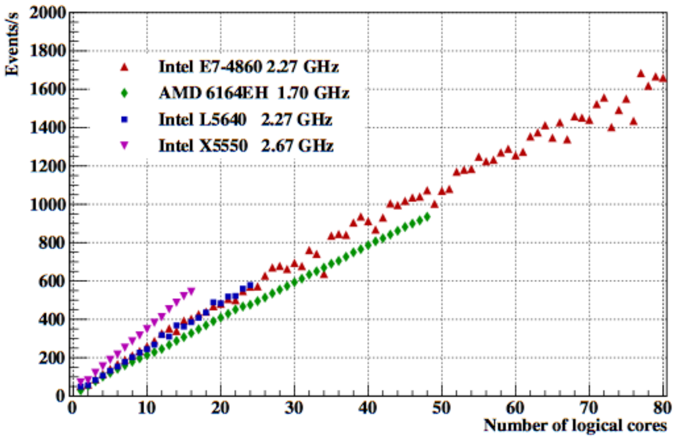


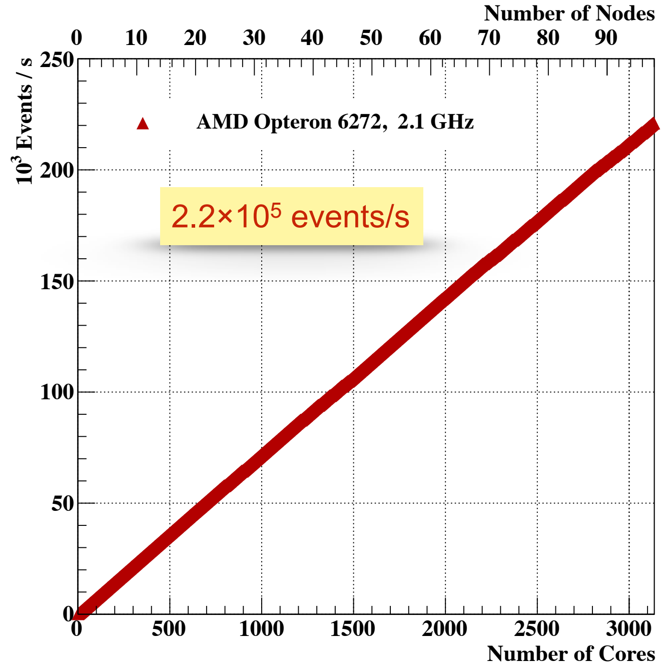
AuAu, 10 AGeV, 3.5M central UrQMD events, MC PID

M. Zyzak

CBM Standalone First Level Event Selection (FLES) Package







The first version of the FLES package is vectorized, parallelized, portable and scalable up to 3 200 CPU cores

Parallelization in the CBM Event Reconstruction

CPU - Full reconstruction

CPU - Tracking

All - Benchmark

GPU/Phi - Selection

Algorithm	SIMD	ITBB, OpenMP	CUDA	OpenCL CPU/GPU	Phi	ArBB
Hit Producers						
STS KF Track Fit	✓	✓	✓	✓/✓	✓	✓
STS CA Track Finder	✓	✓				
MuCh Track Finder	✓	✓	✓			
TRD Track Finder	✓	✓	✓			
RICH Ring Finder	✓	✓		✓/✓		
KF Particle Finder	✓	✓		✓/✓	✓	
Off-line Physics Analysis	✓					
FLES Analysis and Selection	✓	✓				

Andrzej Nowak (OpenLab, CERN) by Hans von der Schmitt (ATLAS) at GPU Workshop, DESY, 15-16 April 2013

	SIMD	Instr. Level Parallelism	HW Threads	Cores	Sockets	Factor	Efficiency
MAX	4	4	1.35	8	4	691.2	100.0%
Typical	2.5	1.43	1.25	8	2	71.5	10.3%
HEP	1	0.80	1	6	2	9.6	1.4%
CBM@FAIR	4	3	1.3	8	4	499.2	72.2%

Parallelization becomes a standard in the CBM experiment

Summary

- The Kalman Filter track fit library is vectorized, parallelized and portable to CPU/Phi/GPU architectures.
- The Cellular Automaton track finder is vectorized and parallelized between CPU cores.
- The KF Particle Finder for reconstruction of short-lived particles is vectorized and portable to CPU/Phi architectures.
- Online physics analysis approaches are under investigation
- Parallelisation is a standard in the CBM experiment

More details:

- V. Akishina, 4D event reconstruction in the CBM experiment, PhD Thesis, Uni-Frankfurt, 2016
- M. Zyzak, Online selection of short-lived particles on many-core computer architectures in the CBM experiment at FAIR, PhD Thesis, Uni-Frankfurt, 2016