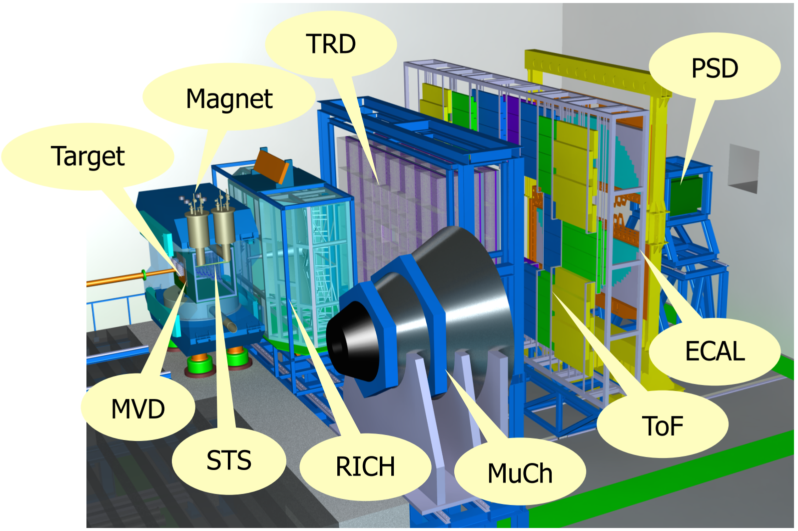
On-line Event Reconstruction in the CBM Experiment

V. Akishina[1,2,3,4], I. Kulakov[3,1,2], I. Kisel[1,2,3], I. Vassiliev[3] and M. Zyzak[3,1,2]
for the CBM Collaboration

1 – Goethe-Universität Frankfurt, Frankfurt am Main, Germany
2 – Frankfurt Institute for Advanced Studies, Frankfurt am Main, Germany
3 – GSI Helmholtzzentrum für Schwerionenforschung GmbH, Darmstadt, Germany
4 – Joint Institute for Nuclear Research, Dubna, Russia

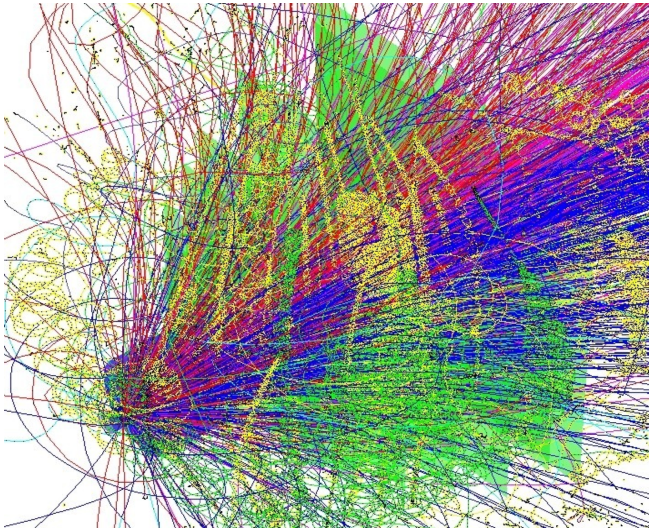
International Conference on Science and Technology for FAIR in Europe 2014
Worms, Germany
16.10.2014

The CBM Experiment



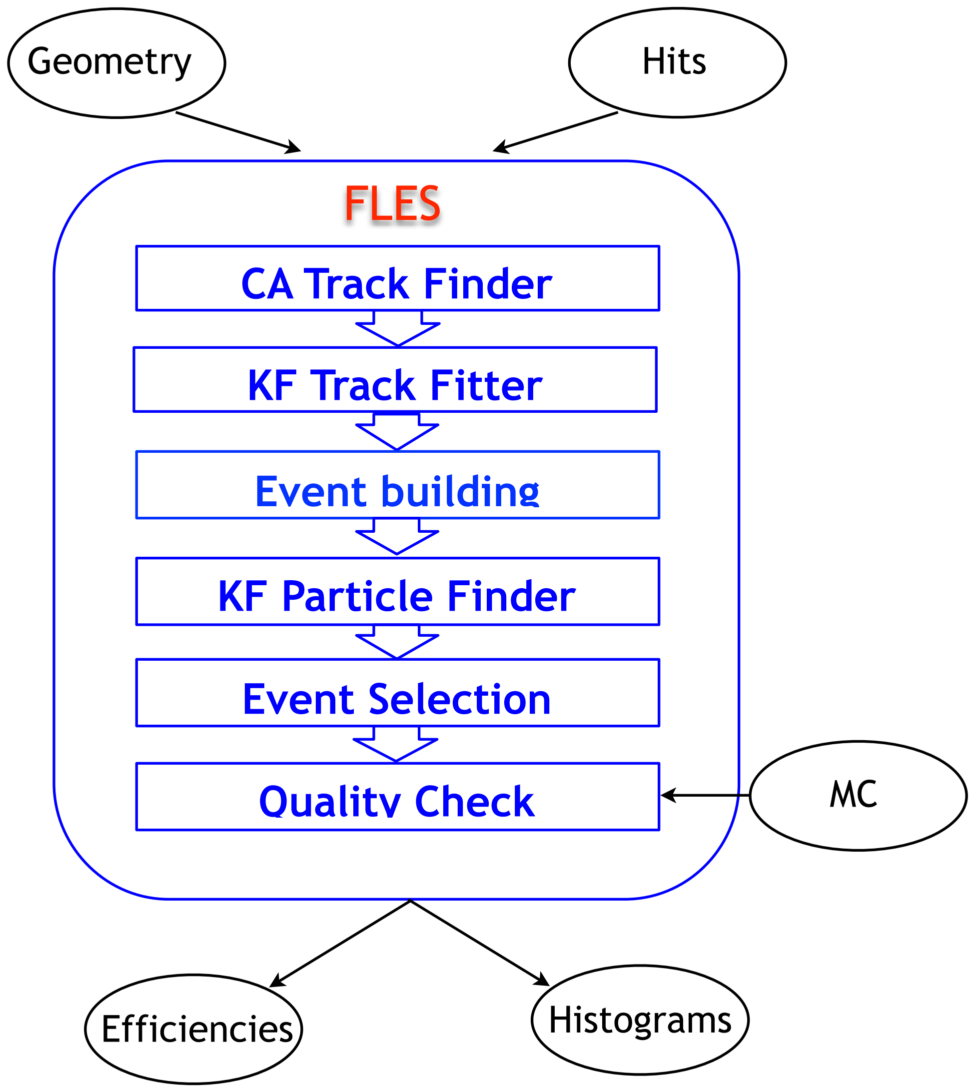
- CBM - future fixed-target heavy-ion experiment at FAIR, Darmstadt, Germany.
- 10^5-10^7 collisions per second.
- Up to 1000 charged particles/collision.
- Free streaming data.
- No hardware triggers.
- On-line event reconstruction and selection is required in the first trigger level.

- On-line reconstruction on the 60000 CPU equivalent cores farm.
- High speed and efficiency of the reconstruction algorithms are required.
- The algorithms have to be highly parallelised and scalable.
- CBM event reconstruction: Kalman Filter and Cellular Automaton.



Simulated central Au-Au collision at 25 AGeV

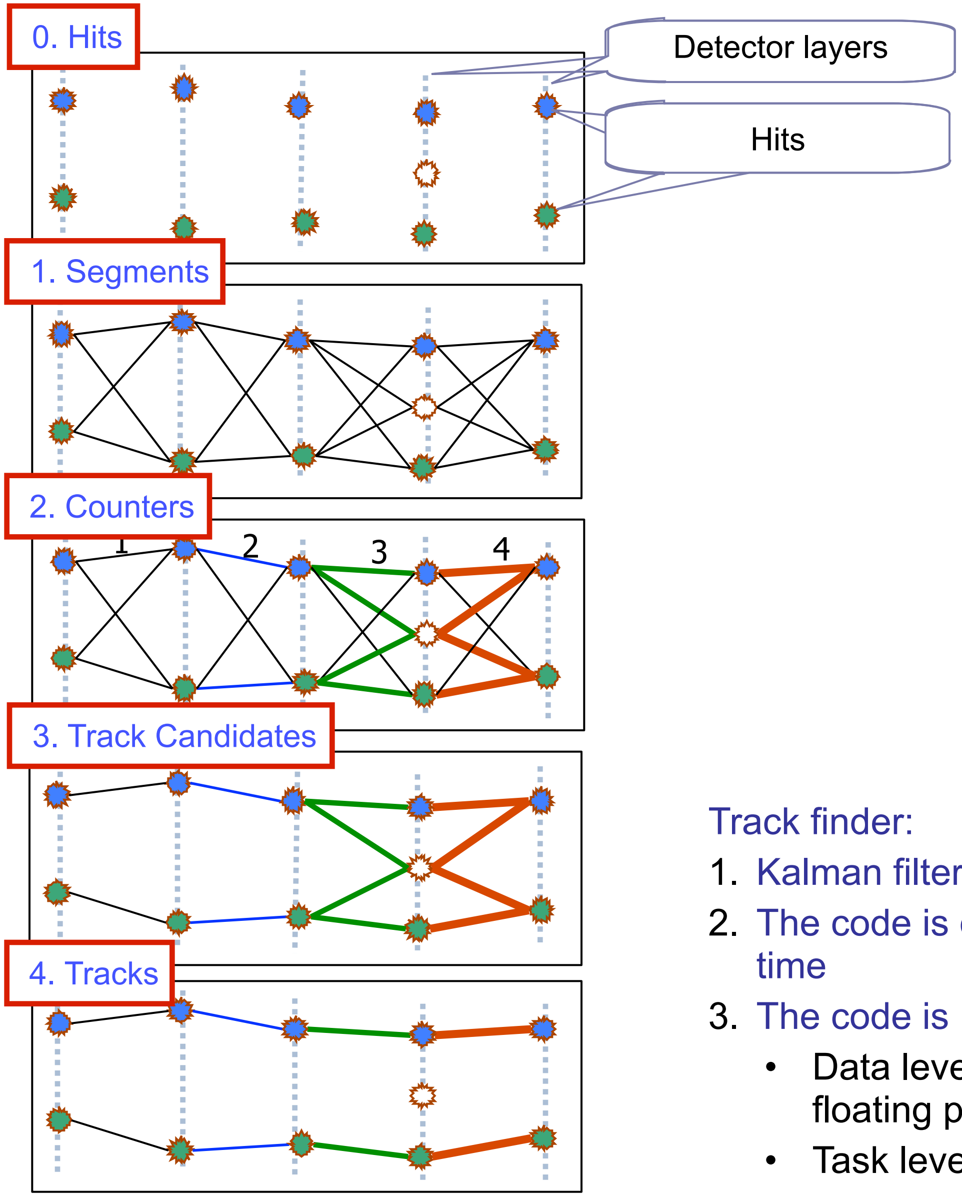
First Level Event Selection Package for the CBM Experiment



The First Level Event Selection (FLES) package for the on-line reconstruction consist of:

- Cellular Automaton based track finder for track search in the Silicon Tracking System;
- Kalman filter based track fitter for track parameters reconstruction;
- event building based on the obtained set of tracks;
- KF Particle Finder for short-lived particles reconstruction and physics analysis;
- the module for a quality check.

Cellular Automaton 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

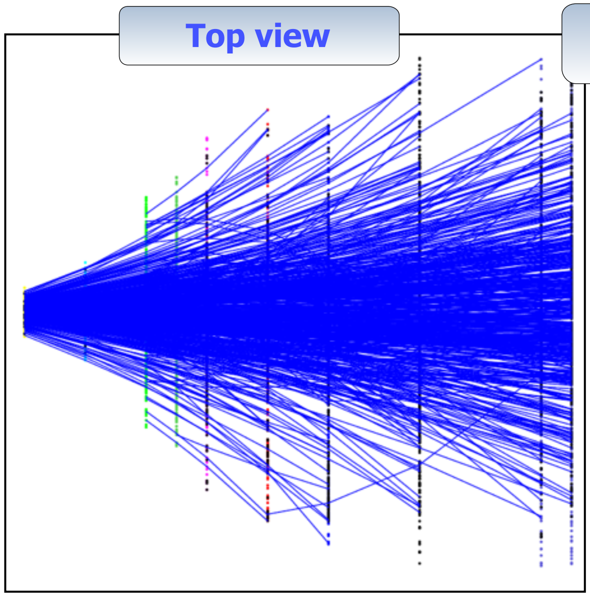
Perfect for many-core CPU/GPU !

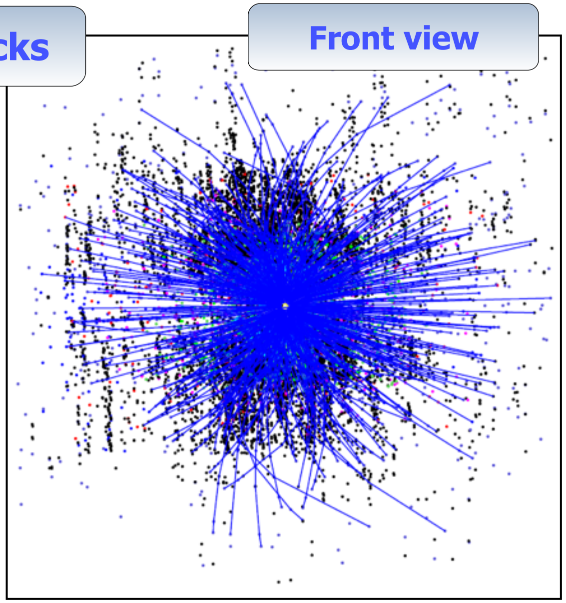
Track finder:

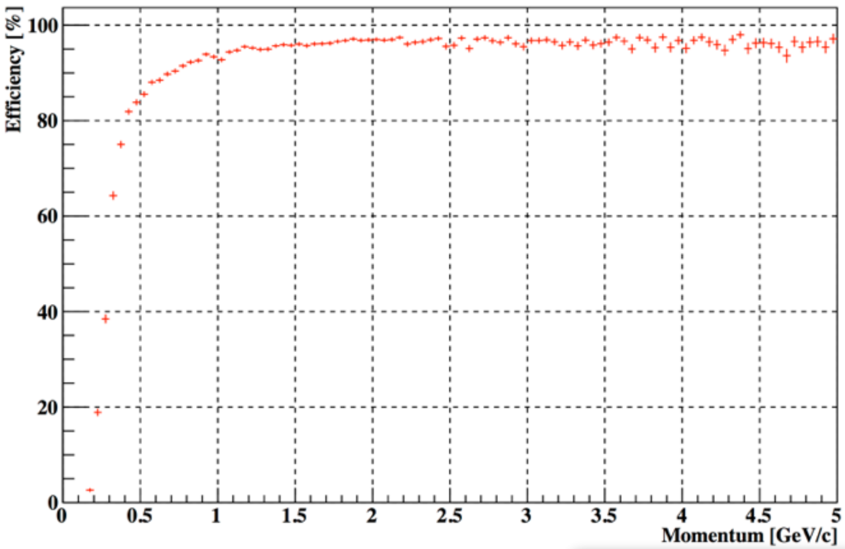
1. Kalman filter for track segments fit
2. The code is optimised with respect to both efficiency and time
3. The code is parallelised
   - Data level (SIMD instructions, 4 single-precision floating point calculations in parallel)
   - Task level (ITBB, parallelisation between cores)

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

Cellular Automaton Track Finder: Efficiency







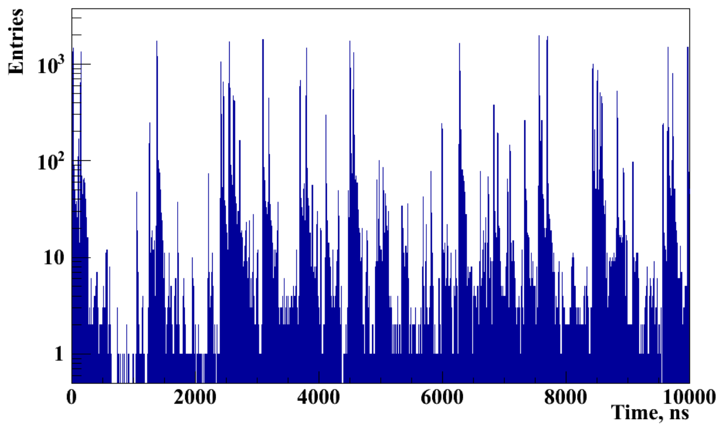
	Efficiency, %	
	mbias	central
Primary high-p tracks	97.1	96.2
Primary low-p tracks	90.4	90.7
Secondary high-p tracks	81.2	81.4
Secondary low-p tracks	51.1	50.6
All tracks	88.5	88.3
Clone level	0.2	0.2
Ghost level	0.7	1.5
Reconstructed tracks/event	120	591
Time/event/core	8.2 ms	57 ms

Efficient and stable event reconstruction

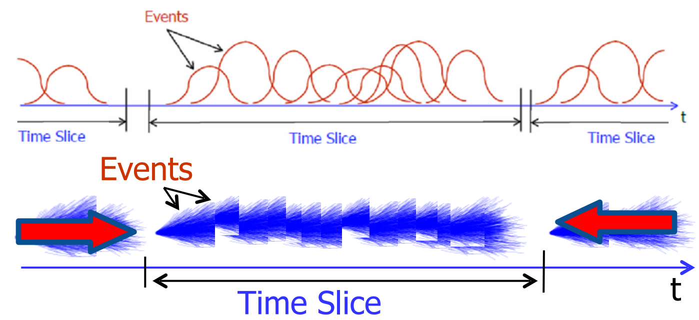
4D Event Building

100 AuAu mbias events at 25 AGeV at 10^7 Hz

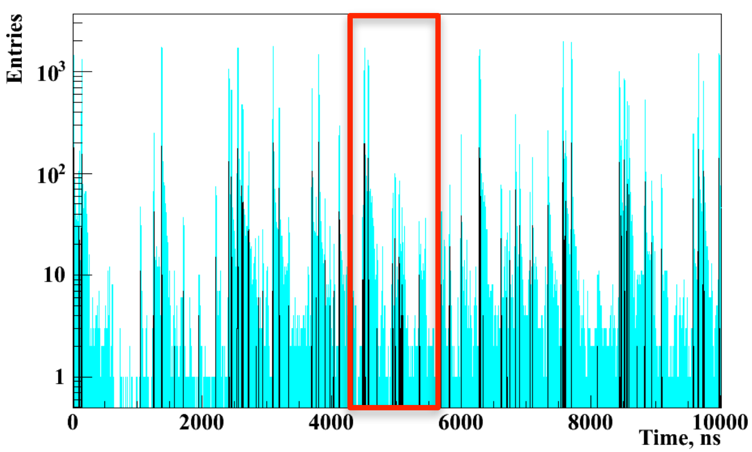
Input hits



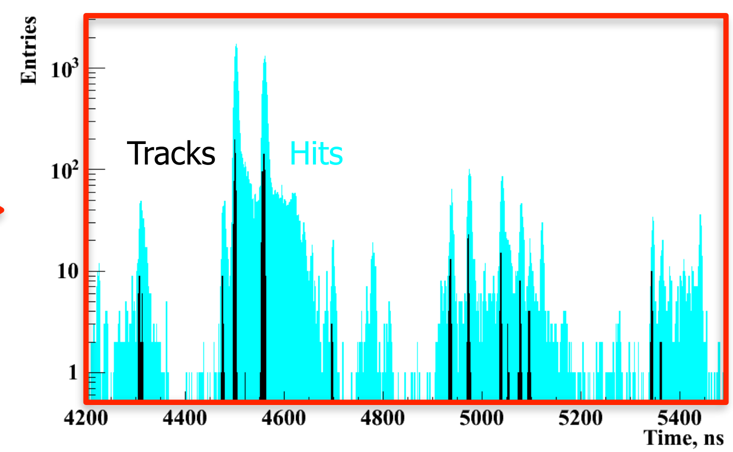
- The SIS100/300 beam will have no bunch structure, but continuous.
- Free streaming data.
- Measurements in this case will be 4D (x,y,z,t).
- Reconstruction of time slices rather than events will be needed.



Reconstructed tracks



Reconstructed tracks - zoom

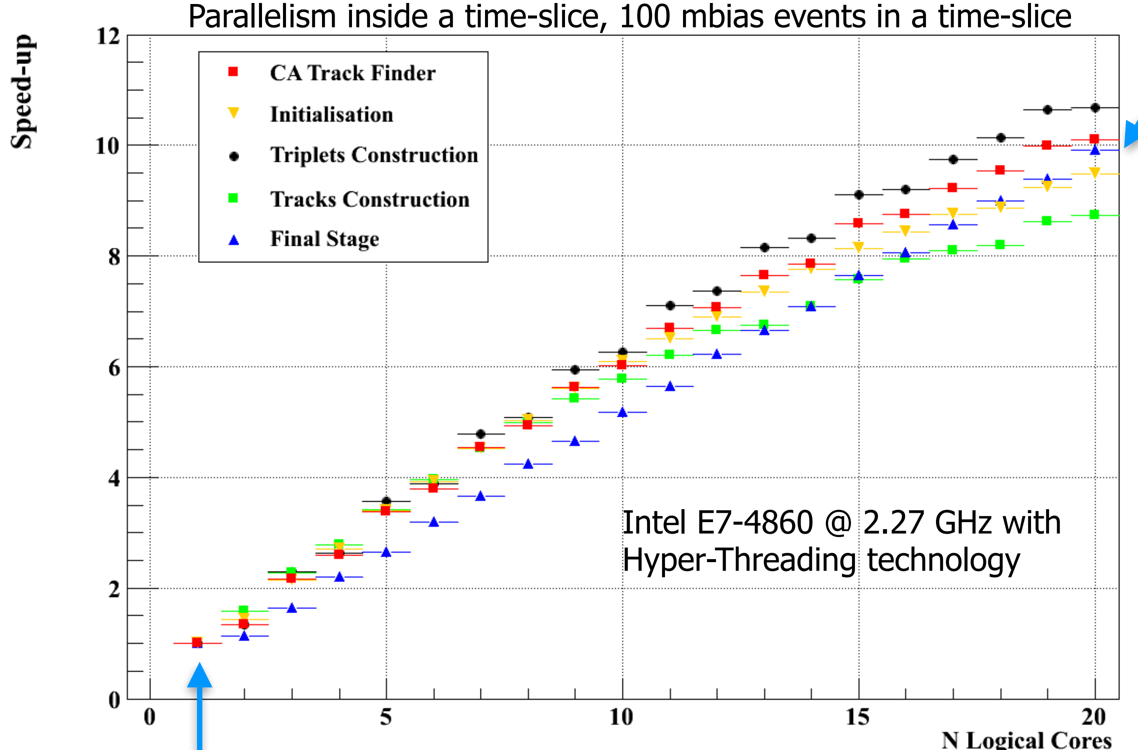


Reconstructed tracks clearly represent groups, which correspond to the original events

4D Track Reconstruction with the CA Track Finder

Algorithm Step	% of total execution time
Initialisation	8 %
Triplets construction	64 %
Tracks construction	15 %
Final stage	13 %

Total time - 84 ms

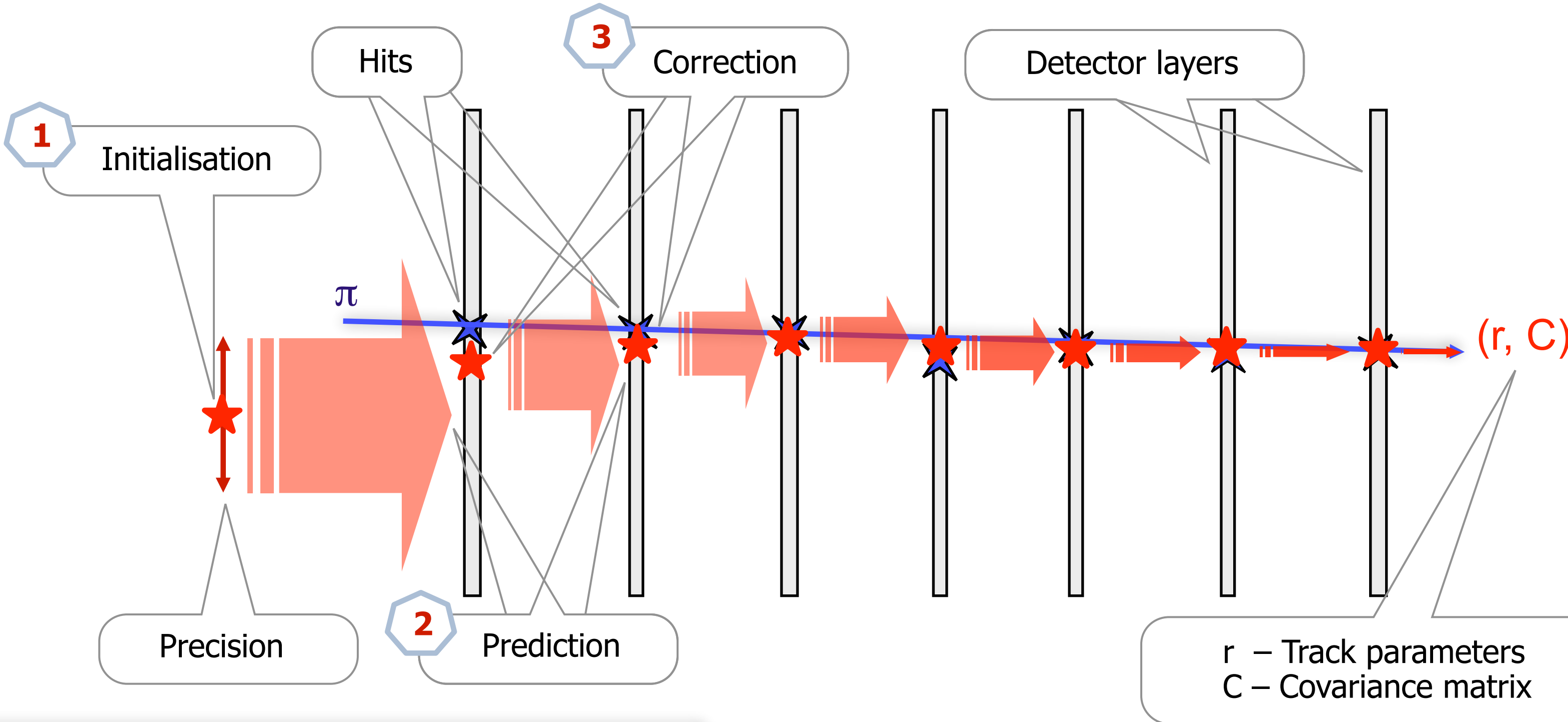


Total time - 849 ms

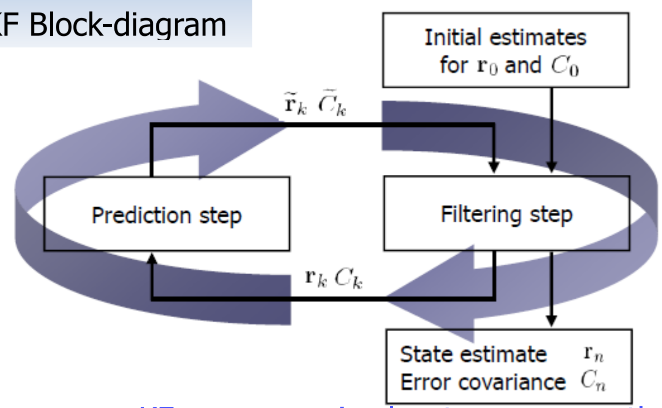
4D track reconstruction is scalable with the speed-up factor of 10.1 (out of 13)

Kalman Filter for Track Fitting

Track fit: Estimation of the track parameters at one or more hits along the track – Kalman Filter (KF)



KF Block-diagram



KF as a recursive least squares method

State vector

Position, direction and momentum

$r = \{ x, y, t_x, t_y, q/p \}$

Kalman filter:

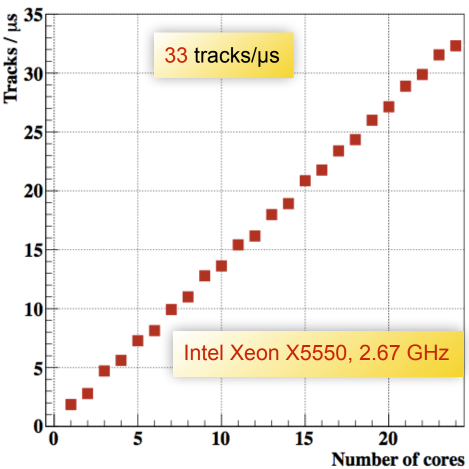
1. Start with an arbitrary initialisation.
2. Add one hit after another.
3. Improve the state vector.
4. Get the optimal parameters after the last hit.

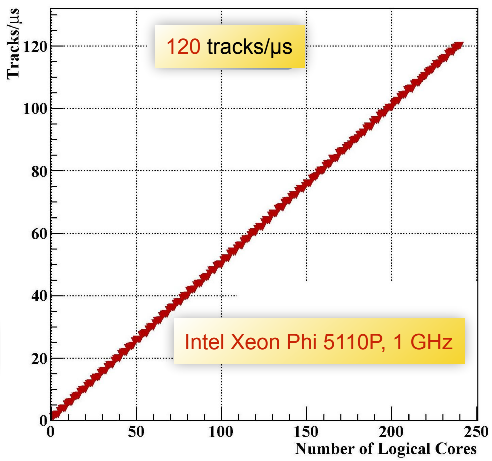
Nowadays the Kalman Filter is used in almost all HEP experiments

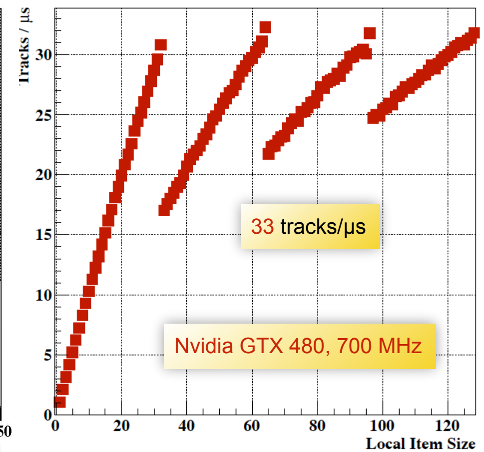
Portability of the KF Track Fitter

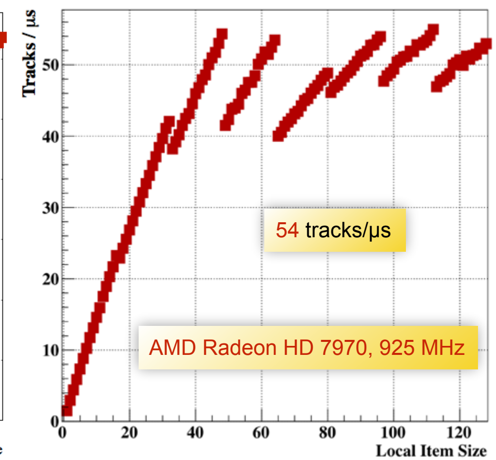
- CPU-like approach.
- Steps due to the hyper-threading.

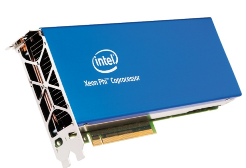
- GPU-like approach.
- Jumps because some of the streaming multiprocessors are loaded partially.





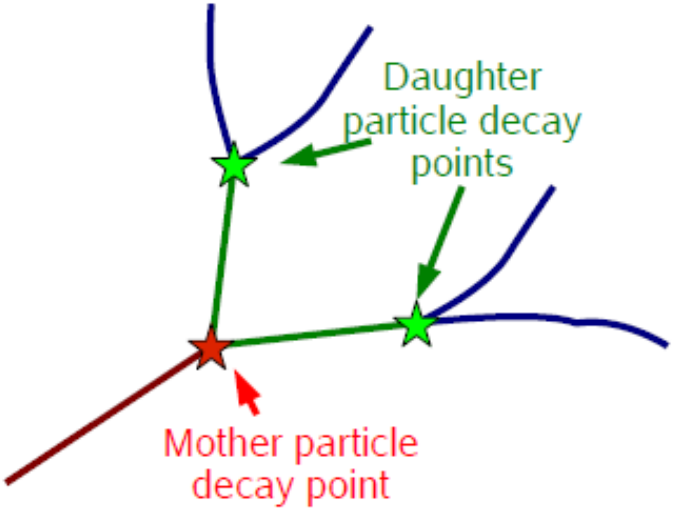






Full portability of the Kalman filter fitter on
Intel/AMD CPUs, Intel Xeon Phi, Nvidia GPUs, AMD GPUs.

Concept of KF Particle



State vector — Position, momentum and energy

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

Concept:

- Mother and daughter particles have the same state vector and are treated in the same way
- Geometry independent
- Kalman filter based

Functionality of the package:

- Construction of the particles from tracks or another particles
- Decay chains reconstruction
- Transport of the particles
- Simple access to the particle parameters and their errors
- Calculation of the distance to point
- KF Particle Finder for short-lived particles reconstruction

KF Particle Finder for On-line Event Selection

Tracks: $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$, $K^{\pm}$, $p^{\pm}$
secondary and primary

Open-charm:

$D^0 \rightarrow \pi^+ K^-$
$D^0 \rightarrow \pi^+ \pi^+ \pi^- K^-$
$\bar{D}^0 \rightarrow \pi^- K^+$
$\bar{D}^0 \rightarrow \pi^- \pi^- \pi^+ K^+$
$D^+ \rightarrow \pi^+ \pi^+ K^-$
$D^- \rightarrow \pi^- \pi^- K^+$
$D_s^+ \rightarrow \pi^+ K^+ K^-$
$D_s^- \rightarrow \pi^- K^+ K^-$
$\Lambda_c \rightarrow \pi^+ K^- p$

Strange particles:

$K^0_s \rightarrow \pi^+ \pi^-$
$\Lambda \rightarrow p\, \pi^-$
$\bar{\Lambda} \rightarrow \pi^+ p^-$

Charmonium:

$J/\Psi \rightarrow e^- e^+$
$J/\Psi \rightarrow \mu^- \mu^+$

Light vector mesons:

$\rho \rightarrow e^- e^+$
$\rho \rightarrow \mu^- \mu^+$
$\omega \rightarrow e^- e^+$
$\omega \rightarrow \mu^- \mu^+$
$\varphi \rightarrow e^- e^+$
$\varphi \rightarrow \mu^- \mu^+$
$\varphi \rightarrow K^- K^+$

Strange resonances

$K^{*0} \rightarrow K^+ \pi^-$
$\bar{K}^{*0} \rightarrow \pi^+ K^-$
$\Lambda^* \rightarrow p\, K^-$
$\bar{\Lambda}^* \rightarrow p^- K^+$

Gamma

$\gamma \rightarrow e^- e^+$

Multi-strange hyperons:

$\Xi^- \rightarrow \Lambda\, \pi^-$
$\Xi^+ \rightarrow \bar{\Lambda}\, \pi^+$
$\Omega^- \rightarrow \Lambda\, K^-$
$\Omega^+ \rightarrow \bar{\Lambda}\, K^+$

Strange and multi-strange resonances:

$\Sigma^{*+} \rightarrow \Lambda\, \pi^+$
$\bar{\Sigma}^{*+} \rightarrow \bar{\Lambda}\, \pi^-$
$\Sigma^{*-} \rightarrow \Lambda\, \pi^-$
$\bar{\Sigma}^{*-} \rightarrow \bar{\Lambda}\, \pi^+$
$K^{*-} \rightarrow K^0_s\, \pi^-$
$K^{*+} \rightarrow K^0_s\, \pi^+$
$\Xi^{*-} \rightarrow \Lambda\, K^-$
$\Xi^{*+} \rightarrow \bar{\Lambda}\, K^+$

Open-charm resonances:

$D^{*0} \rightarrow D^+ \pi^-$
$\bar{D}^{*0} \rightarrow D^- \pi^+$
$D^{*+} \rightarrow D^0 \pi^+$
$D^{*-} \rightarrow \bar{D}^0 \pi^-$

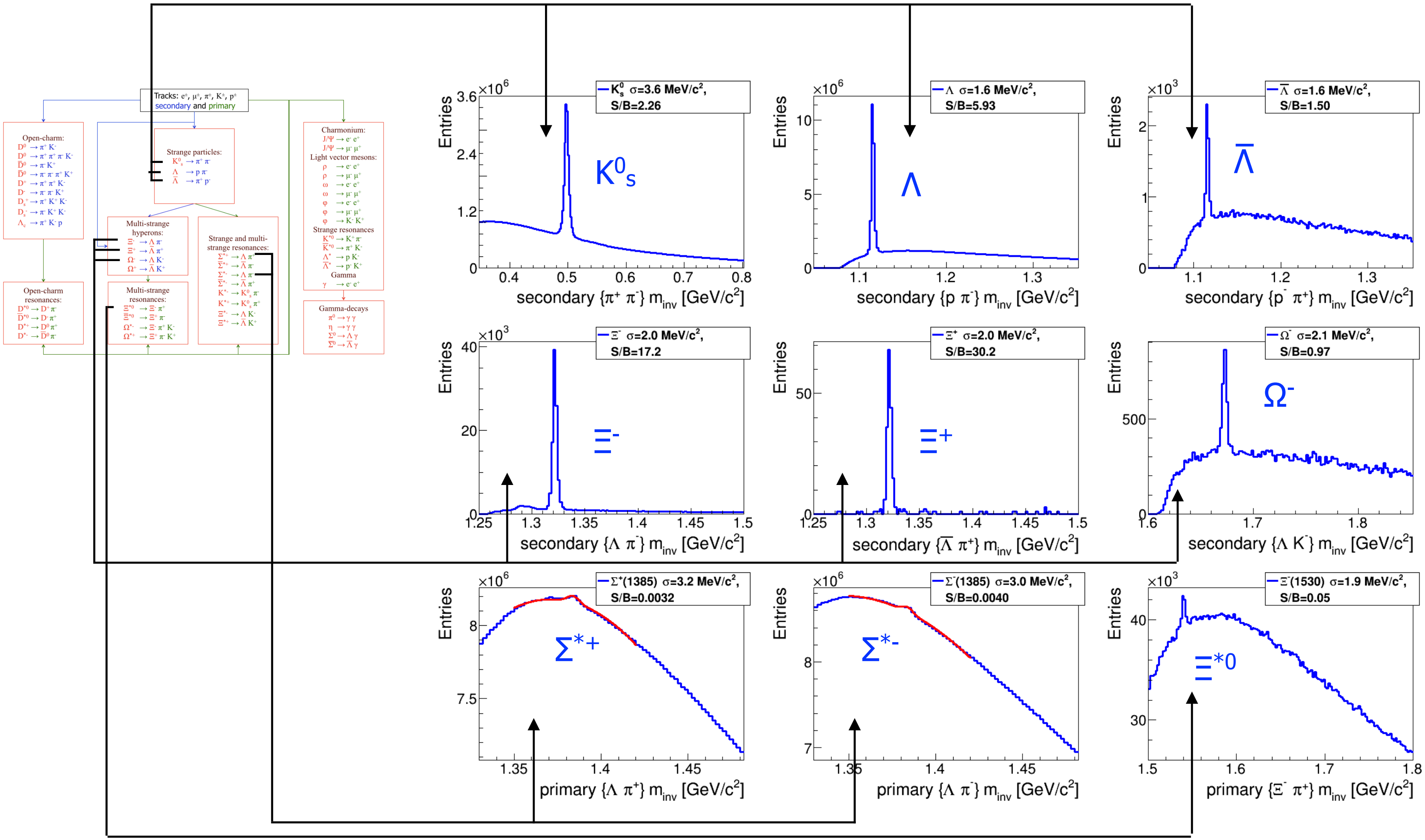
Multi-strange resonances:

$\Xi^{*0} \rightarrow \Xi^- \pi^+$
$\bar{\Xi}^{*0} \rightarrow \Xi^+ \pi^-$
$\Omega^{*-} \rightarrow \Xi^- \pi^+ K^-$
$\Omega^{*+} \rightarrow \Xi^+ \pi^- K^+$

Gamma-decays

$\pi^0 \rightarrow \gamma\, \gamma$
$\eta \rightarrow \gamma\, \gamma$
$\Sigma^0 \rightarrow \Lambda\, \gamma$
$\bar{\Sigma}^0 \rightarrow \bar{\Lambda}\, \gamma$

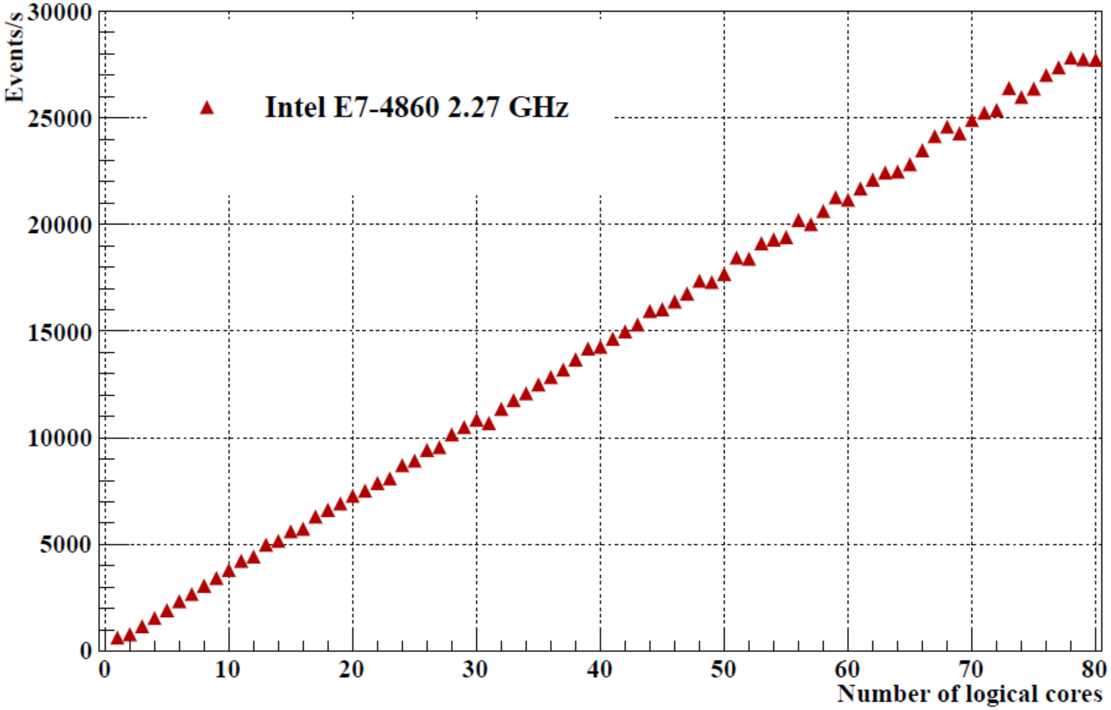
Search for Strange Particles, AuAu, SIS 100



- 5M central AuAu UrQMD events at 10 AGeV with realistic ToF PID with 80 ps resolution.
- All particles - UrQMD output.
- Event by event 3D analysis.

Scalability of KF Particle Finder on Many-core System

- The KF Particle Finder has been vectorised and parallelised.
- The KF Particle Finder shows linear scalability on many-core machines (the scalability on a computer with 40 physical, 80 logical cores is shown).

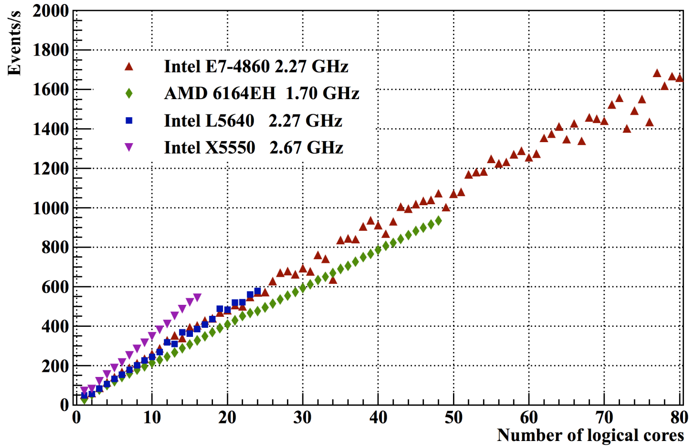


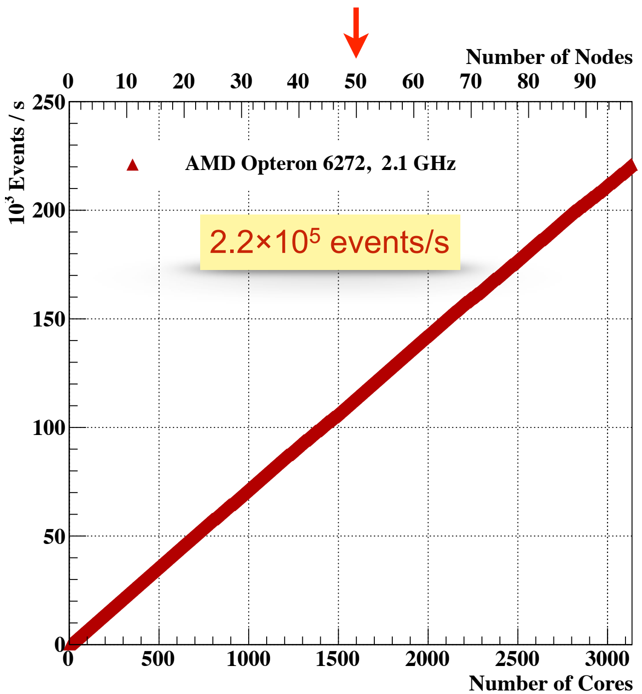
1 thread per 1 core, each filled with 1000 events,
AuAu mbias events at 25 AGeV

The speed of the package:
- mbias AuAu collisions at 25 AGeV – 1.5 ms/event/core
- central AuAu collisions at 25 AGeV – 10.5 ms/event/core

CBM FLES Performance on Many-Core Systems

1 thread per 1 core, each filled with 1000 events,
AuAu mbias events at 25 AGeV





FAIR-Russia HPC Cluster

The first version of the FLES package is

- vectorised and parallelised;
- is portable;
- shows scalable behaviour on nodes with different types and number of CPUs.

The packages currently starts with STS space points. Hits reconstruction is to be added.

Towards CBM FLES farm

Available prototype clusters:

- LOEWE CSC (FIAS, Frankfurt)
- Mini Cube (GSI, Darmstadt)
- FAIR-Russia HPC Cluster (ITEP, Moscow)

Summary and Plans

Summary

- The first version of the FLES package for the on-line reconstruction includes stages for:
  - track reconstruction and fitting;
  - event building;
  - short-lived particles reconstruction and physics selection.
- All methods demonstrates high reconstruction quality.
- The package for on-line reconstruction shows high speed and strong scalability including HEP many-core clusters.

Plans for FLES

- Implement 4D physics selection with KF Particle Finder.
- Add global tracks reconstruction.
- Add reconstruction in PID detectors.
- Add reconstruction of hits.
- Port the whole FLES to graphic cards.