EPJ Web of Conferences **108**, 01006 (2016)
DOI: 10.1051/epjconf/201610801006
© Owned by the authors, published by EDP Sciences, 2016

New Approaches for Data Reconstruction and Analysis in the CBM Experiment

Ivan Kisel[a]

[1] *Goethe University, Theodor-W.-Adorno-Platz 1, 60323 Frankfurt am Main, Germany*
[2] *FIAS Frankfurt Institute for Advanced Studies, Ruth-Moufang-Str. 1, 60438 Frankfurt am Main, Germany*
[3] *GSI Helmholtz Center for Heavy Ion Research, Planckstr. 1, 64291 Darmstadt, Germany*

Abstract. The future heavy-ion experiment CBM (FAIR/GSI, Darmstadt, Germany) will focus on measurement of very rare probes at interaction rates up to 10 MHz with data flow of up to 1 TB/s. The beam will provide free stream of beam particles without bunch structure. That requires full online event reconstruction and selection not only in space, but also in time, so-called 4D event building and selection.

The FLES (First-Level Event Selection) reconstruction and selection package consists of several modules: track finding, track fitting, short-lived particles finding, event building and event selection. A time-slice is reconstructed in parallel between cores within a same CPU, thus minimizing the communication between CPUs. After all tracks are found and fitted in 4D, they are collected into clusters of tracks originated from common primary vertices, which then are fitted, thus identifying 4D interaction points registered within the time-slice. Secondary tracks are associated with primary vertices according to their estimated production time. After that, short-lived particles are found and the full event building process is finished. The last stage of the FLES package is the selection of events according to the requested trigger signatures.

1 Introduction

The CBM (Compressed Baryonic Matter) experiment [1] is an experiment being prepared to operate at the future Facility for Anti-Proton and Ion Research (FAIR, Darmstadt, Germany). Its main focus is the measurement of very rare probes, that requires interaction rates of up to 10 MHz. Together with high multiplicity of charged particles produced in heavy-ion collisions, this leads to huge data rates of up to 1 TB/s. Most trigger signatures are complex (short-lived particles, e.g. open charm decays) and require information from several detector sub-systems.

The First Level Event Selection (FLES) package [2–4] of the CBM experiment is intended to reconstruct the full collision (event) topology including trajectories (tracks) of charged particles and short-lived particles. The FLES package consists of several modules: track finder, track fitter, particle finder, and physics selection. As an input the FLES package receives a simplified geometry of the tracking detectors and the measurements (hits), which are created by the charged particles crossing the detectors. Tracks of the charged particles are reconstructed by the Cellular Automaton (CA) track

[a] e-mail: I.Kisel@compeng.uni-frankfurt.de

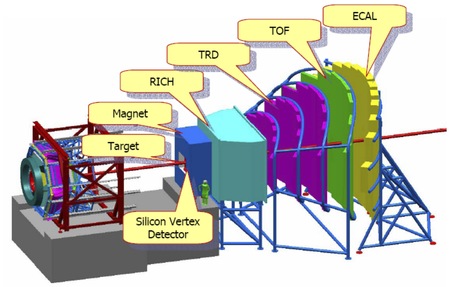


Figure 1. CBM — a future heavy-ion experiment at FAIR

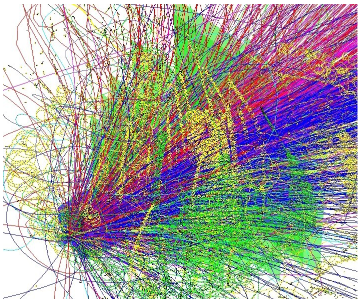


Figure 2. Tracks in a central Au-Au event at 25 *A*GeV collision energy in the CBM experiment. On average there are about 1000 tracks of charged particles.

finder [2, 5, 6] using to the registered hits. The Kalman filter (KF) based track fit [7] is used for precise estimation (fitting) of the track parameters. The short-lived particles, which decay before the tracking detectors, can be reconstructed via their decay products only. The KF particle finder, which is based on the KF Particle package is used in order to find and reconstruct the parameters of short-lived particles by combining the already found tracks of long-lived charged particles. The KF particle finder also selects particle-candidates from a large number of random combinations. In addition, a module for quality assurance is implemented, that allows to control the quality of the reconstruction at all stages. It produces an output in a simple ASCII format, that can be interpreted later as efficiencies and histograms using the ROOT framework. The FLES package is platform and operating system independent.

The FLES package in the CBM experiment will be performed on-line on a dedicated many-core cluster. The FLES algorithms have to be therefore intrinsically local and parallel and thus require a fundamental redesign of the traditional approaches to event data processing in order to use the full potential of modern and future many-core computer architectures. Massive hardware parallelization has to be adequately reflected in mathematical and computational optimization of the algorithms.

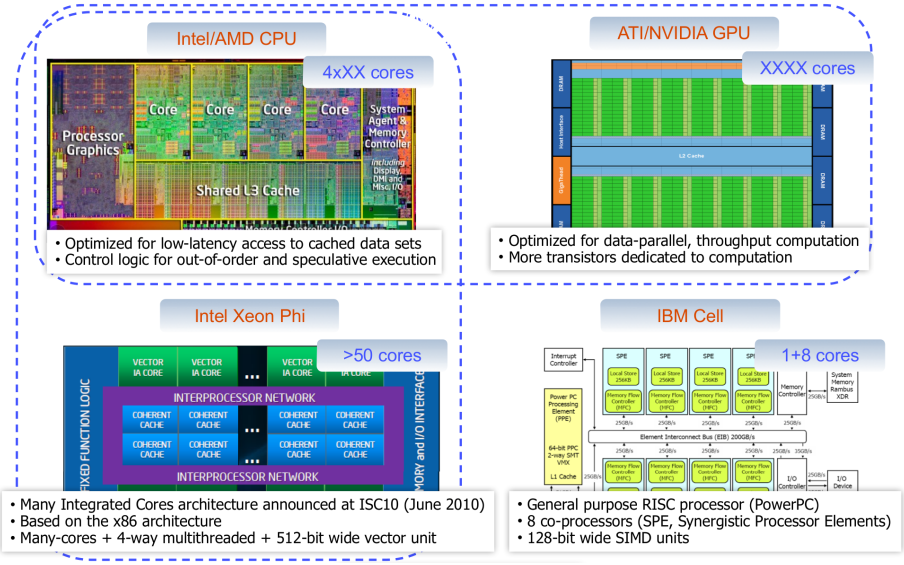
One of the efficient features supported by almost all modern processors is the SIMD (Single Instruction, Multiple Data, vector operations) instruction set. It allows to pack several data values into a vector register and to work with them simultaneously obtaining a factor more calculations per clock cycle. Therefore the reconstruction routines have been revised in order to use SIMD.

In addition, the reconstruction algorithms have been parallelized between cores using the Intel Threading Building Blocks package (ITBB), that provides a scalable event-level parallelism with respect to the number of hardware threads and cores.

2 Many-core computer architectures: cores, threads and vectors

Modern high-performance computing (HPC) nodes are equipped with central processing units (CPU) with dozens of cores and graphics processing units (GPU) with thousands of arithmetic units (Fig. 3).

To illustrate the complexity of the HPC hardware, let us consider a single work-node of a High-Level Trigger (HLT) computer farm, a server equipped with CPUs only. Typically it has 2 to 4 sockets with 8 cores each. In the case of Intel CPUs, each core can run in parallel 2 hardware threads (processes), that increases the calculation speed by about 30%. The arithmetic units of CPUs operate



- Optimized for low-latency access to cached data sets
- Control logic for out-of-order and speculative execution

- Optimized for data-parallel, throughput computation
- More transistors dedicated to computation

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

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

Figure 3. Future high-performance computing systems are heterogeneous many-core CPU/GPU compute nodes

with vector registers, which contain 4 (SSE), 8 (AVX) or 16 (MIC) data elements. Vectors realize the SIMD paradigm, that means they apply an operation to a vector as a whole, giving a speed-up factor of 4/8/16 with respect to the same operation, but with a scalar. In total, the pure hardware potential speed-up factor of a host is:

$$\mathrm{f} = 4 \text{ sockets} \times 8 \text{ cores} \times 1.3 \text{ threads} \times 8 \text{ SIMD} \approx 300,$$

which is already equivalent to a moderate computer farm with scalar single-core CPUs.

Table 1. List of some heterogeneous HPC nodes, used in our investigations

Location		Architecture	(Nodes·)sockets·cores·threads·SIMD	Data streams
CERN	Switzerland	AMD 6164HE	4·12·1·4	192
GSI	Germany	Intel E7-4860	4·10·2·4	320
JINR	Russia	Intel E5-2650+AMD HD 7970	2·8·2·8+2·32·16·4	256+4960
BNL	USA	Intel E5-2680+Intel Phi 5120D	2·8·2·8+2·60·4·16	256+7680
FIAS	Germany	Intel E5-2600+Intel Phi 7120	2·8·2·8+2·61·4·16	256+7808
ITEP	Russia	AMD 6272	(100·)2·16·1·4	12800

In order to investigate the HPC hardware and to develop efficient algorithms we use different nodes and clusters in several high-energy physics centers over the world (see Table 1) ranging from dozens to thousands of cores with up to 12 800 parallel data streams.

3 Parallel programming

The hardware provides us two levels of parallelization: a task level parallelism working with cores and threads, and a data level parallelism working with SIMD vectors. Both levels are implemented in the reconstruction algorithms. The parts of the algorithms with parallel streams of data, like fit of several tracks, are SIMDized and run on vectors providing a speed-up factor up to 4/8/16.

For SIMDization we have developed special header files, which overload SIMD instructions inlining the basic arithmetic and logic functions. An illustrative example of a simple code for the calculation of a polynomial function of the first order, which is written using SSE instructions, is:

```
__m128 y = _mm_add_ps(_mm_mul_ps(a,x),b);
```

The same function, but implemented using the header file, recovers the scalar-like form:

```
fvec y = a*x + b;
```

with overloading in the SIMD header file:

```
friend fvec operator+( const fvec &a,
                       const fvec &b ) {
       return _mm_add_ps(a,b); }
friend fvec operator*( const fvec &a,
                       const fvec &b ) {
       return _mm_mul_ps(a,b); }
```

As a further evolution of the header files, the Vc library implements in addition to vertical operations with full vectors also horizontal operations with elements of a single SIMD vector in order to manipulate with data within the vector. Random access to array elements is implemented with the gather and scatter functionality. All functions and operators of the vector classes are able to take a mask argument optionally. The Vc library automatically determines the platform and chooses the corresponding instruction set during the compilation.

The Vc library is now a part of the CERN ROOT framework, that makes it available for physics analysis by default.

At the task level parallelism we localize independent parts of the algorithms and run them in parallel on different cores or threads with or without synchronization between the processes. Parallelization between cores is done using the Intel Threading Building Blocks (ITBB) and the Open Multi-Processing (OpenMP) techniques.

The OpenCL standard provides a higher abstraction level for the parallel programing. It allows to write a universal code, which can be run on different types of CPU and GPU processing units, thus providing a portable and efficient access to heterogeneous computer platforms. The OpenCL standard supports both vectorization and parallelization between cores of CPUs and GPUs. The vectorized code in OpenCL looks similar to the previous tools:

```
float4 y = a*x + b;
```

In order to be flexible and efficient with respect to the modern many-core computer architectures we develop the algorithms in a portable form and using advantages of the languages and frameworks mentioned above. Within the KF track fit library we have reached 72.2% efficiency of hardware utilization.

4 Kalman Filter track fit library

Searching for rare interesting physics events, most of modern high-energy physics experiments have to work under conditions of still growing input rates and regularly increasing track multiplicities and densities. High precision of the track parameters and their covariance matrices is a prerequisite for finding rare signal events among hundreds of thousands of background events. Such high precision is usually obtained by using the estimation algorithms based on the Kalman filter (KF) method. In our particular case, the KF method is a linear recursive method for finding the optimum estimation of the track parameters, grouped as components into the so-called state vector, and their covariance matrix according to the detector measurements.

The Kalman filter based library for track fitting includes the following tracking algorithms:

- track fit based on the conventional Kalman filter;
- track fit based on the square root Kalman filter;
- track fit based on the UD Kalman filter;
- track smoother based on the listed above approaches and
- deterministic annealing filter based on the listed above track smoothers.

High speed of the reconstruction algorithms on modern many-core computer architectures can be accomplished by:

- optimizing with respect to the computer memory, in particular declaring all variables in single precision,
- vectorizing in order to use the SIMD instruction set and
- parallelizing between cores within a compute node.

Several formulations of the Kalman filter method, such as the square root KF and the UD KF, increase its numerical stability in single precision. All algorithms, therefore, can be used either in double or in single precision.

The vectorization and parallelization of the algorithms are done by using of: header files, Vc vector classes, Intel TBB, OpenMP and OpenCL.

The KF library has been developed and tested within the simulation and reconstruction framework of the CBM experiment, where the precision and speed of the reconstruction algorithms are extremely important.

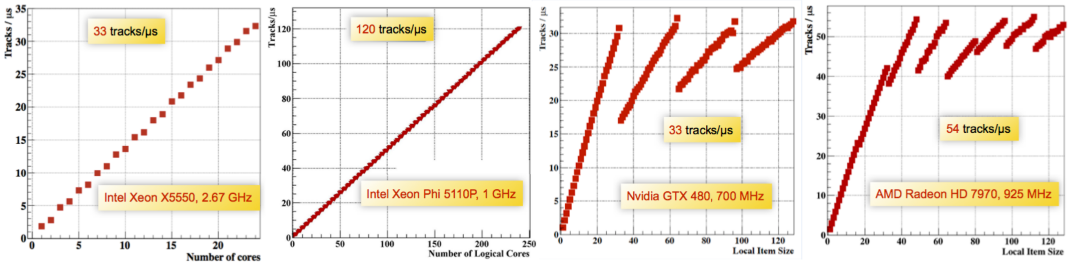


Figure 4. Portability of the Kalman filter track fit library on different many-core CPU/Phi/GPU architectures

When running on CPU, the scalability with respect to the number of cores is one of the most important parameters of the algorithm. Figure 4 shows scalability of the vectorized KF algorithm. The strong linear behavior shows, that with further increase of the number of cores on newer CPUs the performance of the algorithm will not degrade and the maximum speed will be reached. The stair-like dependence appears because of the Intel Hyper-Threading technology, which allows to run two threads per core and gives about 30% of performance advantage. The scalability on the Intel Xeon Phi coprocessor is similar to CPU with four threads per core running simultaneously.

In the case of the graphic cards, a set of tasks is divided into working groups and distributed among compute units (or streaming multiprocessors) by OpenCL and the load of each compute unit is of particular importance. Each working group is assigned to one compute unit and should scale within it with respect to the number of tasks in the group. Figure 4 shows that the algorithm scales linearly on the graphic cards up to the number of cores in one compute unit (for Nvidia GTX480 – 32, for AMD Radeon HD 7970 – 16). Then a drop appears, because when first 32 (for Nvidia) or 16 (for AMD) tasks are processed, only one task is left and all other cores of the compute unit are idle. Increasing the number of tasks in the group further the speed reaches the maximum with the number of tasks dividable by the number of cores in the compute unit. Due to the overhead in tasks distribution the maximum performance is reached when the number of tasks in the group is two-three times larger than the number of cores.

5 Cellular Automaton track finder

Every track finder must handle a very specific and complicated combinatorial optimization process (see figure 2 with a simulated Au-Au collision), grouping together one- or two-dimensional measurements into five-dimensional tracks.

In the Cellular Automaton (CA) method, first, short track segments, so-called cells, are created. After that, the method does not work with the hits any more but instead with the created track segments. It puts neighbor relations between the segments according to the track model here and then one estimates for each segment its possible position on a track, introducing in such a way position counters for all segments. After this process a set of tree connections of possible track candidates appears. Then one starts with the segments with the largest position counters and follows the continuous connection tree of neighbors to collect the track segments into track candidates. In the last step one sorts the track candidates according to their length and χ^2-values and then selects among them the best tracks.

Table 2. Track reconstruction efficiency for minimum bias and central collisions

	Efficiency, %	
	mbias	central
All tracks	88.5	88.3
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
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

The efficiency of the track reconstruction for minimum bias Au-Au UrQMD (Ultra relativistic Quantum Molecular Dynamics) simulated collisions at 25 *A*GeV for different sets of tracks and ratios of clones (double found) and ghost (wrong) tracks are shown in table 2. The tests have been performed on a server with Intel Xeon E7-4860 CPUs.

The majority of signal tracks (decay products of D-mesons, charmonium, light vector mesons) are particles with momentum higher than 1 GeV/c originating from the region very close to the collision point. Their reconstruction efficiency is, therefore, similar to the efficiency of high-momentum primary tracks that is equal to 97.1%. High-momentum secondary particles, e.g. in decays of K_s^0 and Λ particles and cascade decays of Ξ and Ω, are created far from the collision point (primary vertex), therefore their reconstruction efficiency is lower – 81.2%. Significant multiple scattering of low-momentum tracks in the material of the detector system and large curvature of their trajectories lead to lower reconstruction efficiencies of 90.4% for primary tracks and of 51.1% for secondary low-momentum tracks. The total efficiency for all tracks is 88.5% with a large fraction of low-momentum secondary tracks. The levels of clones (double found tracks) and of ghost (wrong) tracks are 0.2% and 0.7% respectively. The reconstruction efficiency for central events is also given in the Table in order to show the stable behavior of the CA track finder with respect to the track multiplicity.

The high track finding efficiency and the track fit quality are crucial, especially for the reconstruction of the short-lived particles, which are of particular interest for the CBM experiment. The reconstruction efficiency of the short-lived particles depends quadratically on the daughter track reconstruction efficiency in the case of two-particle decays. The situation becomes more sensitive for decays with three daughters and for decay chains. The level of the combinatorial background for short-lived particles depends strongly on the track fit quality. The correct estimation of the errors on the track parameters improves the differentiation between the signal and the background particle candidates, and thus suppresses the background. Ghost (wrong) tracks usually have large errors on the track parameters and therefore are easily combined with other tracks into short-lived particle candidates, thus a low level of ghost tracks is also important to keep the combinatorial background low. As a result, the high track reconstruction efficiency and the low level of the combinatorial background improve significantly the event reconstruction and selection by the FLES package.

6 Track finding at high track multiplicities

Since the CBM experiment will operate at extremely high interaction rates, different collisions may overlap in time. Thus, the need to analyze so-called time-slices, which contain information from a number of collisions, rather than isolated events, arises. The need to work with time-slices instead of events is triggered not only by physical circumstances, but also is encouraged by computing hardware reasons. Not only minimum bias events, but even central events were proved to be not large enough in order to be processed in parallel on modern many-core computer architectures. For implementing in-event level parallelism these events do not have enough sources of parallelism in order to be reconstructed on 20 or more CPU cores simultaneously.

As a first step on the way toward the time-slice reconstruction we introduce a container of packed minimum bias events with no time information taken into account. To create such a group we combine space coordinates of hits from a number (from 1 up to 100) AuAu minimum bias events at 25 AGeV ignoring such information as event number or time measurements (Fig. 5). The group was treated by the CA track finder as a regular event and the reconstruction procedure was performed with no changes. Varying the number of minimum bias events in a group we have studied the track reconstruction efficiency dependence with respect to track multiplicity. As one can see in Fig. 6, high momentum primary tracks (RefPrim), that have particular physical importance, are reconstructed with excellent efficiency of about 96%, which varies within less than 2% up to a hundred events grouped. If we include secondary tracks (RefSet) the efficiency is a bit lower – 93.7%, since some secondary tracks originate far from the target. This value varies within 3% for the extreme case of 100 minimum bias events grouped. The efficiency for low momentum tracks is 79.8% (ExtraPrim) due to multiple scattering in the detector material. It changes within 6% window in case of the largest track multiplicities.

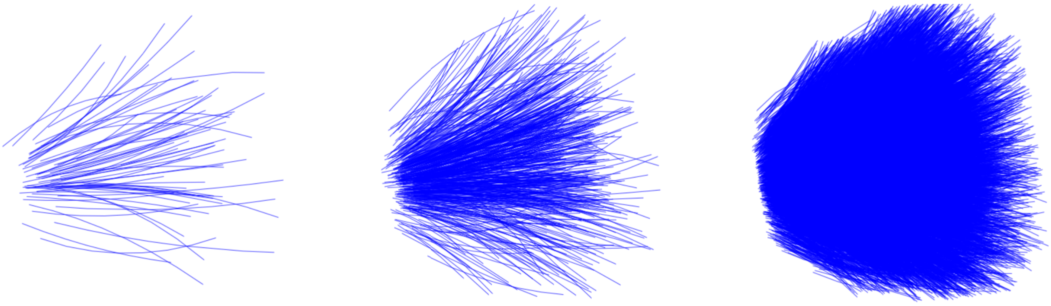


Figure 5. Reconstructed tracks in a minimum bias event (left) and in packed groups of 5 (middle) and 100 (right) minimum bias events with 109, 572 and 10 340 tracks on average respectively

The ghost fraction remains at acceptable level (less than 10%) up to the highest track multiplicities. Thus, the CA track finder is proved to be stable with respect to the high track multiplicities.

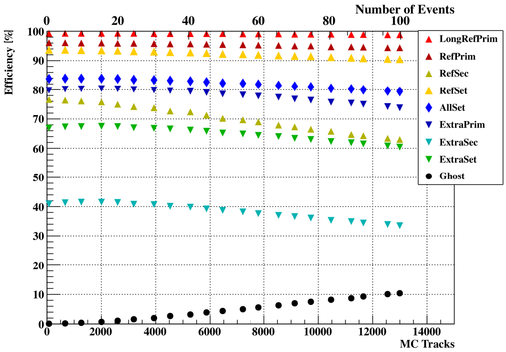


Figure 6. Track reconstruction efficiencies and ghost rate for different sets of tracks versus track multiplicity

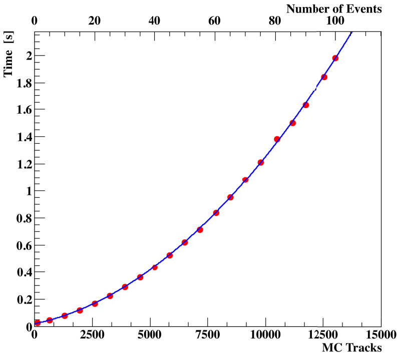


Figure 7. The CA track finder time needed to reconstruct groups of minimum bias events without time information with respect to the track multiplicity. The dependence is fitted with a second order polynomial.

However, not only efficiency, but also the speed of the reconstruction algorithm is crucial for successful performance in case of CBM. We have studied the time that the CA track finder needs to reconstruct a grouped event as a function of the number of Monte-Carlo tracks in a group (figure 7). The results show that the dependence is perfectly described with a second order polynomial. This is a remarkable result, if one keeps in mind the exponential growth of combinatorics with the track multiplicity. This dependence can be improved further and turn into a linear one, which corresponds to the case of event-based analysis, after introducing time measurements into the reconstruction algorithm.

7 Reconstruction of time-slices

After the CA track finder proved to be fast and stable with respect to the track multiplicity, the next step towards the time-slice based reconstruction is an implementation of time measurements.

In order to introduce time measurements into the reconstruction procedure the event start time was assigned to each minimum bias event i n a 100 events group during the simulation phase. The start time was obtained with the Poisson distribution, assuming an interaction rate of 10^7 Hz. A time stamp, that we assign to a certain hit, consists of the event start time plus a time shift due to the time of flight from the collision point to the detector station. This time of flight differs for each hit. In order to obtain the time measurement for a hit we then smear the time stamp according to the Gaussian distribution with a sigma value of the detector resolution of 5 ns. The initial distributions of hits measurements representing the complexity of determining event borders in a time-slice at different interaction rates of 10^5–10^7 Hz are shown in figure 8.

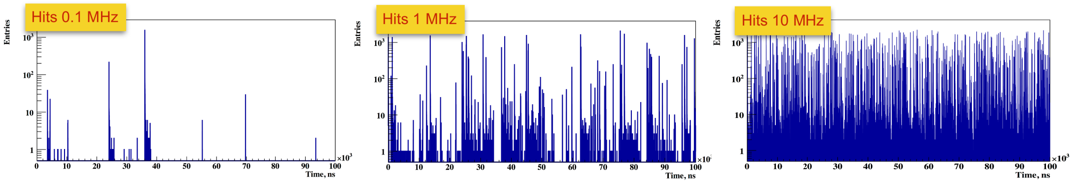


Figure 8. Distributions of hit time measurements in a time-slice at different interaction rates are shown

After introducing time measurements we can use the time information in the CA track finder (figure 9). Here we do not allow to build short track segments (cells) out of hits with time differences larger than 3.5σ of the detector time resolution. It is a justified assumption, since the time of flight between the detector planes is negligible in comparison to the detection precision. Apart from that, we perform the reconstruction procedure in the regular way described above. After the reconstruction we assign to each track a time measurement, which is calculated as the average of the hit time measurements.

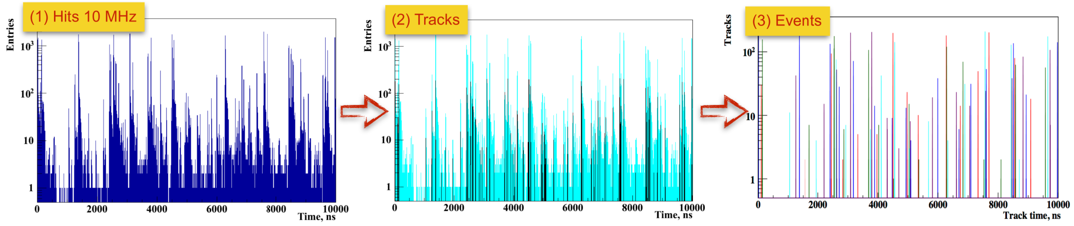


Figure 9. Part of a time-slice with 100 minimum bias events. The left picture: the distribution of hit time measurements in the time-slice is shown. The picture in the middle: the reconstructed tracks are shown over the initial distribution of hit measurements. The right picture presents the reconstructed tracks clustered into event groups.

The reconstructed tracks clearly represent groups, corresponding to events, which they originate from. Even in the area of the most severe overlap the time-based CA track finder allows to resolve tracks from different events in time.

8 KF Particle Finder – a package for reconstruction of short-lived particles

Today the most interesting physics is hidden in the properties of short-lived particles, which are not registered, but can be reconstructed only from their decay products. A fast and efficient KF Particle Finder package, based on the Kalman filter (hence KF) method, for reconstruction and selection of short-lived particles is developed to solve this task. A search for more than 70 decay channels has been currently implemented. The package doesn't require any specific information about the geometry of an experiment, therefore it is implemented as a common package for and tested in the CBM, PANDA, ALICE and STAR experiments.

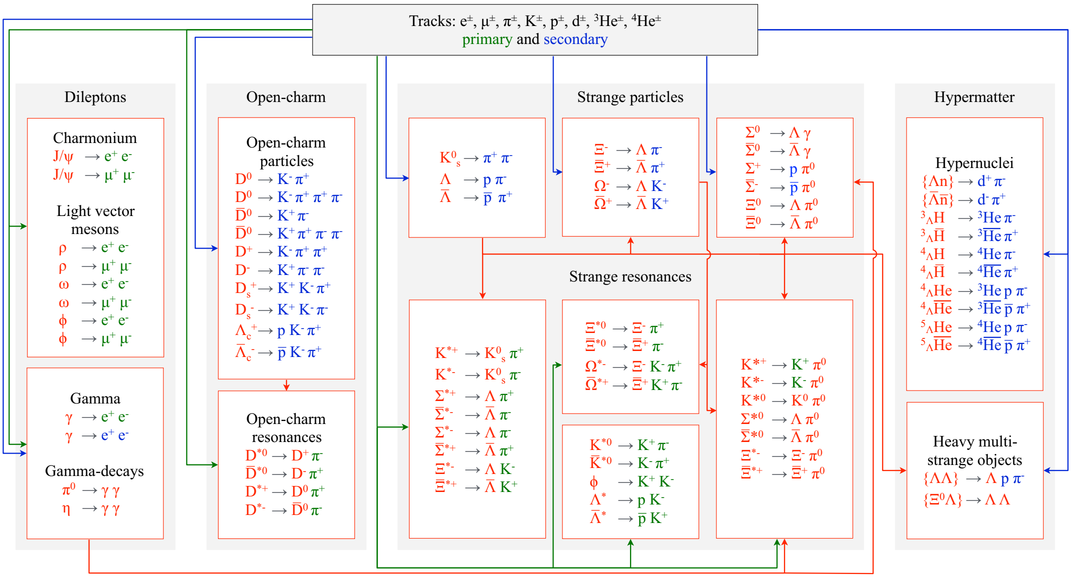

Tracks: $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$, $K^{\pm}$, $p^{\pm}$, $d^{\pm}$, $^3He^{\pm}$, $^4He^{\pm}$
primary and secondary
Dileptons
Charmonium
$J/\psi \rightarrow e^+ e^-$
$J/\psi \rightarrow \mu^+ \mu^-$
Light vector mesons
$\rho \rightarrow e^+ e^-$
$\rho \rightarrow \mu^+ \mu^-$
$\omega \rightarrow e^+ e^-$
$\omega \rightarrow \mu^+ \mu^-$
$\phi \rightarrow e^+ e^-$
$\phi \rightarrow \mu^+ \mu^-$
Gamma
$\gamma \rightarrow e^+ e^-$
$\gamma \rightarrow e^+ e^-$
Gamma-decays
$\pi^0 \rightarrow \gamma\,\gamma$
$\eta \rightarrow \gamma\,\gamma$
Open-charm
Open-charm particles
$D^0 \rightarrow K^- \pi^+$
$D^0 \rightarrow K^- \pi^+ \pi^+ \pi^-$
$\bar{D}^0 \rightarrow K^+ \pi^-$
$\bar{D}^0 \rightarrow K^+ \pi^+ \pi^- \pi^-$
$D^+ \rightarrow K^- \pi^+ \pi^+$
$D^- \rightarrow K^+ \pi^- \pi^-$
$D_s^+ \rightarrow K^+ K^- \pi^+$
$D_s^- \rightarrow K^+ K^- \pi^-$
$\Lambda_c^+ \rightarrow p\, K^- \pi^+$
$\bar{\Lambda}_c^- \rightarrow \bar{p}\, K^- \pi^+$
Open-charm resonances
$D^{*0} \rightarrow D^+ \pi^-$
$\bar{D}^{*0} \rightarrow D^- \pi^+$
$D^{*+} \rightarrow D^0 \pi^+$
$D^{*-} \rightarrow \bar{D}^0 \pi^-$
Strange particles
$K^0_s \rightarrow \pi^+ \pi^-$
$\Lambda \rightarrow p\, \pi^-$
$\bar{\Lambda} \rightarrow \bar{p}\, \pi^+$
$\Xi^- \rightarrow \Lambda\, \pi^-$
$\bar{\Xi}^+ \rightarrow \bar{\Lambda}\, \pi^+$
$\Omega^- \rightarrow \Lambda\, K^-$
$\bar{\Omega}^+ \rightarrow \bar{\Lambda}\, K^+$
$\Sigma^0 \rightarrow \Lambda\, \gamma$
$\bar{\Sigma}^0 \rightarrow \bar{\Lambda}\, \gamma$
$\Sigma^+ \rightarrow p\, \pi^0$
$\bar{\Sigma}^- \rightarrow \bar{p}\, \pi^0$
$\Xi^0 \rightarrow \Lambda\, \pi^0$
$\bar{\Xi}^0 \rightarrow \bar{\Lambda}\, \pi^0$
Strange resonances
$K^{*+} \rightarrow K^0_s\, \pi^+$
$K^{*-} \rightarrow K^0_s\, \pi^-$
$\Sigma^{*+} \rightarrow \Lambda\, \pi^+$
$\bar{\Sigma}^{*-} \rightarrow \bar{\Lambda}\, \pi^-$
$\Sigma^{*-} \rightarrow \Lambda\, \pi^-$
$\bar{\Sigma}^{*+} \rightarrow \bar{\Lambda}\, \pi^+$
$\Xi^{*-} \rightarrow \Lambda\, K^-$
$\bar{\Xi}^{*+} \rightarrow \bar{\Lambda}\, K^+$
$\Xi^{*0} \rightarrow \Xi^- \pi^+$
$\bar{\Xi}^{*0} \rightarrow \bar{\Xi}^+ \pi^-$
$\Omega^{*-} \rightarrow \Xi^- K^- \pi^+$
$\bar{\Omega}^{*+} \rightarrow \bar{\Xi}^+ K^+ \pi^-$
$K^{*0} \rightarrow K^+ \pi^-$
$\bar{K}^{*0} \rightarrow K^- \pi^+$
$\phi \rightarrow K^+ K^-$
$\Lambda^* \rightarrow p\, K^-$
$\bar{\Lambda}^* \rightarrow \bar{p}\, K^+$
$K^{*+} \rightarrow K^+ \pi^0$
$K^{*-} \rightarrow K^- \pi^0$
$K^{*0} \rightarrow K^0 \pi^0$
$\Sigma^{*0} \rightarrow \Lambda\, \pi^0$
$\bar{\Sigma}^{*0} \rightarrow \bar{\Lambda}\, \pi^0$
$\Xi^{*-} \rightarrow \Xi^- \pi^0$
$\bar{\Xi}^{*+} \rightarrow \bar{\Xi}^+ \pi^0$
Hypermatter
Hypernuclei
$\{\Lambda n\} \rightarrow d^+ \pi^-$
$\{\bar{\Lambda}\bar{n}\} \rightarrow d^- \pi^+$
$^3{}_\Lambda H \rightarrow {}^3He\, \pi^-$
$^3{}_\Lambda \bar{H} \rightarrow {}^3\overline{He}\, \pi^+$
$^4{}_\Lambda H \rightarrow {}^4He\, \pi^-$
$^4{}_\Lambda \bar{H} \rightarrow {}^4\overline{He}\, \pi^+$
$^4{}_\Lambda He \rightarrow {}^3He\, p\, \pi^-$
$^4{}_\Lambda \overline{He} \rightarrow {}^3\overline{He}\, \bar{p}\, \pi^+$
$^5{}_\Lambda He \rightarrow {}^4He\, p\, \pi^-$
$^5{}_\Lambda \overline{He} \rightarrow {}^4\overline{He}\, \bar{p}\, \pi^+$
Heavy multi-strange objects
$\{\Lambda\Lambda\} \rightarrow \Lambda\, p\, \pi^-$
$\{\Xi^0\Lambda\} \rightarrow \Lambda\, \Lambda$

Figure 10. Block-diagram of the KF Particle Finder package. The particle parameters, such as decay point, momentum, energy, mass, decay length and lifetime, together with their errors are estimated using the Kalman filter method.

In the package all registered particle trajectories are divided into groups of secondary and primary tracks for further processing. Primary tracks are those which are produced directly in the collision point. Tracks from decays of resonances (strange, multi-strange and charmed resonances, light vector mesons, charmonium) are also considered as primary, since they are produced directly at the point of the primary collision. Secondary tracks are produced by the short-lived particles, which decay far from the point of the primary collision and can be clearly separated. These particles include[1] strange particles (K_s^0 and Λ), multi-strange hyperons (Ξ and Ω) and charmed particles (D^0, $D^{\pm}$, $D_s^{\pm}$ and Λ_c). After that the appropriate tracks are combined according to the block-diagram in figure 10. The package estimates the particle parameters, such as the decay point, momentum, energy, mass, decay length and lifetime, together with their errors. The package has rich functionality, including particle transport, calculation of the distance to a point or another particle, calculation of the deviation from a point or another particle, constraints on mass, decay length and production point. All particles

[1] Internal structure of some particles, listed in the block-diagram: $\pi^+ = u\bar{d}$, $K^+ = u\bar{s}$, $D^0 = c\bar{u}$, $J/\psi = c\bar{c}$, p = uud, n = udd, Λ = uds, Σ^- = dds, Ξ^0 = uss, Ω^- = sss, Λ_c^+ = udc; d = pn, $\{\Lambda n\} = \Lambda n$, ${}^3_\Lambda H = pn\Lambda$, ^3He = ppn, ^4He = ppnn, ${}^4_\Lambda He = ppn\Lambda$.

produced in the collision are reconstructed at once, that makes the algorithm local with respect to the data and therefore extremely fast.

KF Particle Finder shows a high efficiency of particle reconstruction. For example, for the CBM experiment 4π-efficiencies of about 15% for Λ and 5% for Ξ^- in AuAu collisions at 35 AGeV are achieved together with high signal-to-background ratios (1.3 and 5.9 respectively).

9 FLES – a standalone First Level Event Selection package

The First Level Event Selection (FLES) package of the CBM experiment is intended to reconstruct on-line the full event topology including tracks of charged particles and short-lived particles. The FLES package consists of several modules: CA track finder, KF track fitter, KF Particle Finder and physics selection. In addition, a quality check module is implemented that allows to monitor and control the reconstruction process at all stages. The FLES package is platform and operating system independent.

The FLES package is portable to different many-core CPU architectures. The package is vectorized using SIMD instructions and parallelized between CPU cores. All algorithms are optimized with respect to the memory usage and the speed.

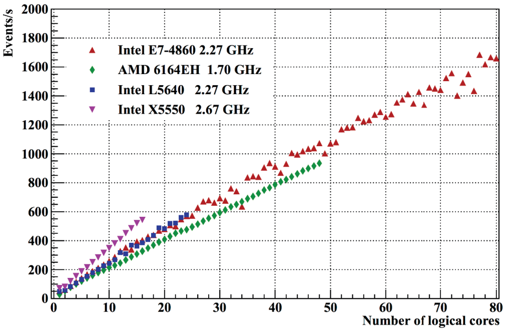


Figure 11. Scalability of the FLES package on many-core servers with 16, 24, 48 and 80 logical cores

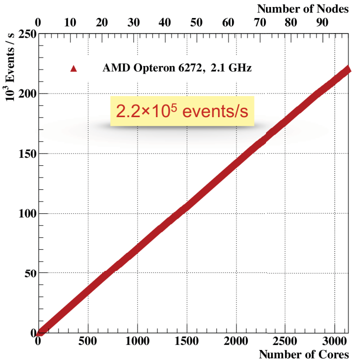


Figure 12. Scalability of the FLES package on 3 200 cores of the FAIR-Russia HPC cluster (ITEP, Moscow)

Four servers with Intel Xeon E7-4860, L5640 and X5550 processors and with AMD 6164EH processor have been used for the scalability tests. The AMD server has 4 processors with 12 physical cores each, in total 48 cores. All Intel processors have the hyper-threading technology, therefore each physical core has two logical cores. The most powerful Intel server has 4 processors with 10 physical cores each, that gives 80 logical cores in total.

The FLES package has been parallelized with ITBB implementing the event-level parallelism by executing one thread per one logical core. Reconstruction of 1000 minimum bias Au-Au UrQMD events at 25 AGeV has been processed per each thread. In order to minimize the effect of the operating system each thread is fixed to a certain core using the pthread functionality provided by the C++ standard library. Fig. 11 shows a strong scalability for all many-core systems achieving the reconstruction speed of 1700 events per second on the 80-cores server.

The FLES package in the CBM experiment will be performed for the on-line selection and the off-line analysis on a dedicated many-core CPU/GPU farm. The farm is currently estimated to have a compute power equivalent to 60 000 modern CPU cores. Fig. 12 shows the scalability of the FLES package on a many-core computer farm with 3 200 cores of the FAIR-Russia HPC cluster (ITEP, Moscow).

10 Summary

The challenges in the data reconstruction and physics analysis of the CBM experiment, discussed in the paper, are typical for modern and future experiments at LHC and other research centers in the world.

Acknowledgements

This work was supported by the Hessian Ministry of Science and Art, Hessian LOEWE Initiative through the Helmholtz International Center for FAIR (HIC for FAIR), HGS-HIRe, GSI F&E, BMBF Verbundforschung and EU-FP7 HadronPhysics2.

References

[1] The CBM Collaboration, *Compressed Baryonic Matter Experiment*, Tech. Stat. Rep., GSI, Darmstadt, 2005; 2006 update

[2] I. Kisel, EPJ Web of Conferences **95**, 01007 (2015)

[3] I. Kisel, I. Kulakov and M. Zyzak, IEEE Trans. Nucl. Sci. **60**, 3703–3708 (2013)

[4] V. Akishina, I. Kisel, I. Kulakov and M. Zyzak, DOI: 10.3204/DESY-PROC-2014-05, 23–29 (2014)

[5] I. Kisel, Nucl. Instr. and Meth. **A566** 85–88 (2006)

[6] V. Akishina and I. Kisel, IEEE Trans. Nucl. Sci. (2015), to be published

[7] S. Gorbunov, U. Kebschull, I. Kisel, V. Lindenstruth, and W.F.J. Müller, Comp. Phys. Comm. **178**, 374–383 (2008)