# Cellular Automaton based Track Finder in the STAR Experiment (BNL, USA)

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Informatik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

von
Grigory Kozlov
aus Severomorsk, Russland

Frankfurt am Main 2022
(D 30)

accepted by the Faculty of Computer Science and Mathematics of the Johann Wolfgang Goethe University as a dissertation.

| | |
|---|---|
| Dean: | Prof. Dr. Martin Möller |
| Expert assessor: | Prof. Dr. Ivan Kisel<br>Prof. Dr. Volker Lindenstruth |
| Date of disputation: | 24 May 2022 |

# Abstract

The relevant field of interest in High Energy Physics experiments is shifting to searching and studying extremely rare particles and phenomena. The search for rare probes requires an increase in the number of available statistics by increasing the particle interaction rate. The structure of the events also becomes more complicated, the multiplicity of particles in each event increases, and a pileup appears. Due to technical limitations, such data flow becomes impossible to store fully on available storage devices. The solution to the problem is the correct triggering of events and real-time data processing.

In this work, the issue of accelerating and improving the algorithms for reconstruction of the charged particles' trajectories based on the Cellular Automaton in the STAR experiment is considered to implement them for track reconstruction in real-time within the High-Level Trigger. This is an important step in the preparation of the CBM experiment as part of the FAIR Phase-0 program. The study of online data processing methods in real conditions at similar interaction energies allows us to study this process and determine the possible weaknesses of the approach.

Two versions of the Cellular Automaton based track reconstruction are discussed, which are used, depending on the detecting systems' features. HFT CA Track Finder, similar to the tracking algorithm of the CBM experiment, has been accelerated by several hundred times, using both algorithm optimization and data-level parallelism. TPC CA Track Finder has been upgraded to improve the reconstruction quality while maintaining high calculation speed. The algorithm was tuned to work with the new iTPC geometry and provided an additional module for very low momentum track reconstruction.

The improved track reconstruction algorithm for the TPC detector in the STAR experiment was included in the HLT reconstruction chain and successfully tested in the express production for the online real data analysis. This made

it possible to obtain important physical results during the experiment runtime without the full offline data processing. The tracker is also being prepared for integration into a standard offline data processing chain, after which it will become the basic track search algorithm in the STAR experiment.

# Kurzfassung

Das Interessensgebiet der Hochenergiephysik-Experimente verschiebt sich hin zur Suche und Untersuchung extrem seltener Teilchen und Phänomene. Zur Erhöhung der Anzahl verfügbarer Statistiken über seltene Ereignisse erfordert es einen Anstieg der Kollisionsrate von Teilchen. Die Struktur der Ereignisse wird komplizierter, die Teilchenvielfalt in jedem Event nimmt zu und es kommt zur Überlagerung von Ereignissen. Aufgrund technischer Einschränkungen kann ein solcher Datenfluss nicht vollständig auf verfügbaren Speichergeräten zur späteren Analyse gesichert werden. Die Lösung des Problems liegt in der korrekten Selektion von Ereignissen und der Datenverarbeitung in Echtzeit.

In dieser Arbeit wird der Aspekt der Beschleunigung und Verbesserung der Algorithmen zur Rekonstruktion der Flugbahnen geladener Teilchen basierend auf einem zellulären Automaten in dem STAR-Experiment betrachtet, um die Algorithmen für die Spurrekonstruktion in Echtzeit innerhalb des High-Level-Triggers zu implementieren. Dies ist ein wichtiger Schritt in der Vorbereitung des CBM-Experiments im Rahmen des Phase-0-Programms der FAIR. Die Untersuchung von Online-Datenverarbeitungsmethoden unter realen Bedingungen bei ähnlichen Kollisionsenergien ermöglicht es uns, diesen Prozess zu untersuchen und die möglichen Schwächen des jeweiligen Ansatzes zu bestimmen.

Es werden zwei Varianten der auf einem zellulären Automaten basierenden Spurrekonstruktion analysiert, die je nach den Merkmalen der Erkennungssysteme verwendet werden. Der HFT CA Track Finder, ähnlich dem Tracking-Algorithmus des CBM-Experiments, wurde um das mehrere Hundertfache beschleunigt, indem sowohl Algorithmusoptimierungen durchgeführt als auch Datenparallelität ausgenutzt wurden. Der TPC CA Track Finder wurde aktualisiert, um die Rekonstruktionsqualität zu verbessern und gleichzeitig eine hohe Berechnungsgeschwindigkeit beizubehalten. Dieser Algorithmus wurde auf die Arbeit mit der neuen iTPC-Geometrie abgestimmt und stellt ein zusätzliches

Modul für die Streckenrekonstruktion von Teilchen mit sehr geringem Momentum bereit.

Der verbesserte Spurrekonstruktionsalgorithmus für den TPC-Detektor im STAR-Experiment wurde in die HLT-Rekonstruktionskette aufgenommen und erfolgreich in der „STAR-Express-Production“ für die Online-Echtdatenanalyse getestet. Dadurch war es möglich, wichtige physikalische Ergebnisse während der Versuchslaufzeit, ohne die vollständige Offline-Datenverarbeitung, zu erhalten. Der Tracker wird auch für die Integration in eine Standard-Offline-Datenverarbeitungskette vorbereitet, wonach er zum grundlegenden Spur-Suchalgorithmus im STAR-Experiment wird.

# Contents

# Chapter 1

# Introduction

Nowadays, humanity has made significant progress in studying the world around. Scientific and technological progress leads to a deeper understanding of the laws of nature and paves the way for new discoveries. To better understand global processes and principles, one should start with studying the basic structure of a matter — elementary particles and the rules of its interaction. But even the most modern technical solutions do not allow us to observe these processes directly. On the other hand, humanity collected enough knowledge and prepared a suitable mathematical apparatus to make assumptions and simulate processes as close to reality as possible.

Talking about the structure and interactions of elementary particles, we cannot see and confirm the processes with our own eyes. Therefore, at the moment, such studies are carried out by constructing hypotheses based on known facts and physical laws. Further, in the course of experiments, scientists create conditions under which such hypotheses can be confirmed or disproved. A full-fledged model of the structure and interactions of elementary particles comes from the accumulation of significant amounts of knowledge. Today, the generally accepted theory that describes the microworld as fully as possible is the Standard Model [1].

According to the Standard Model, the fundamental particles that form all matter in the known world are 12 fermions: 6 leptons and 6 quarks. Each of the 6 types of quarks can be in three color states: red, green, or blue. Color, or color charge, has nothing to do with actual colors but is a kind of similarity to an electric charge in the interactions of elementary particles. In addition, there are 12 antifermions that correspond to 12 fundamental fermions, for which anti-colors replace three basic colors. The interactions of fundamental fermions

are carried out due to the exchange of interaction carriers — gauge bosons. The model assumes the existence of three types of interactions of elementary particles in addition to a gravitational interaction, which almost has no effect at the micro-level. Strong interaction occurs due to the exchange of gluons — electrically neutral massless carriers of strong interaction [2]. The carriers of the weak interaction are massive W and Z bosons [3]. The electromagnetic interaction is carried out using a special massless boson — a photon.

Under normal conditions, nuclear matter exists in the form of neutrons and protons, bound together through strong interaction. These and other elementary particles represent bound states of either a quark-antiquark pair (mesons) or three (baryons), four (tetraquarks), and five (pentaquarks) quarks. The interaction between quarks is described by the theory of Quantum Chromodynamics (QCD) [4]. Following this theory, quarks interact with each other by exchanging special particles — gluons. Quarks are in a state of confinement when each of them cannot be separated from the others, being a part of an elementary particle. Modern theoretical models of strong interaction predict the possibility of violation of confinement at rather high temperatures and densities [5]. There is a phase transition of normal nuclear matter into the so-called Quark-Gluon Plasma (QGP) [6] when quarks and gluons are not bounded and can freely move in a certain region exceeding the size of hadrons. It is assumed that it was in this state that matter existed in the early Universe. Currently, the QGP can exist inside neutron stars and other space objects, where extreme temperature and pressure values can be reached.

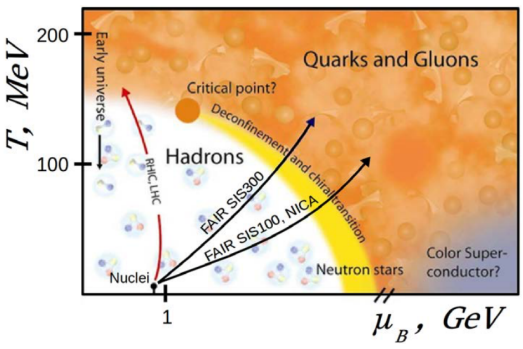


Figure 1.1: Phase diagram of baryonic matter [7].

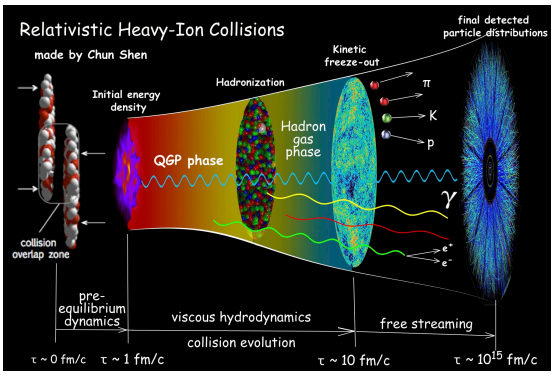


Figure 1.2: Stages of collisions of relativistic heavy ions and the evolution of a fireball [9].

The dependence of the state of nuclear matter on temperature and density in the QCD is often represented in the form of a phase diagram (Fig. 1.1), which is built on the basis of theoretical research and experimentally confirmed data. At the low temperature and pressure level, nuclear matter exists as a set of protons and neutrons, which are well separated and do not overlap. In other words, the substance is in gaseous form. An increase in the temperature and density of matter brings particles into an excited state, transferring them to a class of resonances — extremely short-lived particles ($10^{-22}$ – $10^{-24}$ sec.). Because of such a short lifetime, the trajectories of resonances cannot be observed directly, and the observation method is to analyze the energies and momenta of their decay products. At the same time, resonances have quite definite quantum characteristics — mass, electric charge, spin, etc. The formation and decay of baryon resonances lead to the appearance of hadronic matter — a mixture of nucleons, baryon resonances, and mesons. In this state, the particles are clearly separable from each other, and their constituent quarks are inextricably bounded. With an increase in density to critical values, nucleons merge into a uniform substance, where each quark can no longer belong to a separate nucleon. Thus, a first-order transition occurs with the QGP formation under conditions similar to the neutron stars. Another way to obtain the QGP as a result of a second-order transition (or crossover) is to increase the temperature to a level where the frequency of collisions between nucleons no longer allows to assign a quark to a particular group. It is assumed that the crossover and the first-order transition are combined at the critical point [8], where strong fluctuations of physical parameters are expected.

Today the QCD is still far from completion and needs additional research and expansion of the empirical base. Since we cannot get direct access to the neural star or the conditions of the early Universe, we have to look for other ways to create the QGP in the laboratory. Under terrestrial conditions, a clot of nuclear matter with an increased temperature and density can be obtained by collisions of heavy ions with high energies. The result of such a collision is a fireball — a strongly interacting hadrons system in the zone of their overlap. Figure 1.2 demonstrates a scheme of the fireball's evolution after the collision of two heavy ions occurs. Suppose the energy and the overlap zone of the particles are large enough. In that case, the QGP appears inside the fireball, and emitted particles could provide information about the matter and its transformation. The expansion of the fireball leads to a decrease in its density and temperature and, as a consequence, hadronization — the formation of stable hadrons from free

quarks and gluons. This stage is called a kinetic freezeout and is accompanied by the production of particles, the individual and collective parameters of which are indicators of the state of nuclear matter at different times after the collision of primary ions.

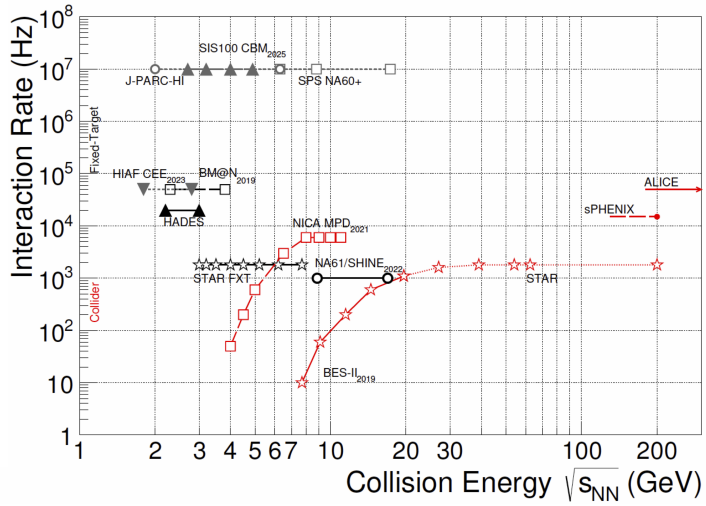


Figure 1.3: The energies and frequencies of interaction of the main active and planned experiments of the High Energy Physics, operating with a relatively high baryon density [7].

Today, scientists from different countries are studying the processes occurring during the collision of nuclei of heavy ions at the already operating and planned experiments of High Energy Physics (HEP). The parameters of the most important modern experiments are schematically presented in Figure 1.3. Different energies of the particle interactions allow researchers to study various regions of the phase diagram. At the same time, an increase in the frequency of interaction makes it possible to collect large statistics for the study of rare phenomena. Still, it puts more serious demands on the detecting equipment and computer systems for processing the results.

The largest particle accelerator to date, the Large Hadron Collider (LHC), is located at CERN [10], Switzerland. The LHC makes it possible to accelerate particles to an all-time high-speed level, which increases the likelihood of processes with high energy transfer and makes it possible to test the perturbative QCD models. The study of the interaction of heavy ions at the LHC is carried out as part of the ALICE experiment (A Large Ion Collider Experiment) [13]. Collisions of beams of lead nuclei with energies up to $\sqrt{s_{NN}}$=5.02 TeV at the interaction rate of about 10 kHz make it possible to register $Z^0$ and $W^{\pm}$ bosons [11, 12] that born surrounded by hot nuclear matter. The relative lifetime of the QGP increases. The fireball expansion is determined mostly by the dynamics of the

partons that carry information about the thermodynamic conditions at the early hot phase of the collision.

Another important modern HEP experiment that aimed to study nuclear matter at high temperatures and relatively low values of the baryon potential is the STAR experiment (Solenoidal Tracking At RHIC) [14] which is being conducted at the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory (BNL), New York, USA. The main task of the STAR experiment is to study the formation and characteristics of the QGP. The RHIC was initially designed to achieve high collision energy $\sqrt{s_{NN}}$=200 GeV. Further, starting in 2010, it was decided to adapt the collider to study a wider area of the phase diagram. Within two phases of the Beam Energy Scan (BES) [15] project, the accelerator has been tuned to work with energies 7.7–200 GeV/nucleon for the colliding beams. Also, as part of the BES-II [16] program, the STAR experiment was supplemented with the ability to operate in the FiXed Target (FXT) mode at energies from 7 to 3.5 GeV/nucleon. The energy range considered within the BES corresponds to the region of the phase diagram in which a first-order phase transition and critical point are expected. The BES program made it possible to use the RHIC accelerator to study the region of the phase diagram with high baryon density, which is of great physical interest.

The STAR experiment makes it possible to cover a significant part of the considered energy range, but its weak side is the low collision frequency. An experiment with a fixed target CBM (Compressed Baryonic Matter) [7] should play an important role in studies of heavy-ion collisions at the energies of $2A$ – $11A$ GeV [17]. It will be held at the FAIR (Facility for Antiproton and Ion Research) accelerator under construction in Darmstadt, Germany. The planned frequency of nuclear collisions is $10^7$ Hz, which will allow the accumulation of a huge amount of data for research.

In such conditions, fast and high-quality processing and analysis of the incoming experimental data plays a critical role in the success of the experiment. The best option is the most complete reconstruction and classification of particle trajectories in real-time immediately upon receipt of the information from the detectors. It is this approach that is planned to be applied in the CBM experiment. The extremely high frequency of particle collisions places particularly stringent requirements on the algorithms used to reconstruct events. It should be noted that modern trends in the development of microprocessor architectures tend to improve parallel computing technologies instead of increasing the quickness of

individual computational elements. In such conditions, easily parallelized algorithms are in a more favorable state in relation to purely sequential ones since they allow maximum use of the resources of large multiprocessor computer farms available for data processing.

The most time-consuming and challenging part of event reconstruction is the reconstruction of particle trajectories — tracks. This is a process of combining hits — measurements left on the detecting planes by passing particles — into segments that represent the path of the particle. Reconstruction of tracks is usually denoted as — tracking, and the software used for tracking is called a track finder or just a tracker. One of the most successful solutions in this area is a family of tracking algorithms based on the Cellular Automaton (CA). CA has tremendous possibilities for parallelizing computations both at the data level and at the task level. A number of actions for organizing hits into tracks can be performed independently, which opens up enormous opportunities for speeding up calculations on modern microprocessors and graphics cards. In turn, the determination of track parameters and selection of correctly reconstructed trajectories can be carried out using the Kalman Filter (KF), which also has a massive potential for parallelization. The track finder based on a combination of the CA and the KF is currently the primary and most promising candidate for use in the CBM experiment since only it can meet the speed requirements while maintaining high efficiency of the track reconstruction. Another version of the CA-based tracking algorithm has been successfully used in the STAR experiment for several years, consistently demonstrating a combination of high efficiency and speed.

This thesis will present a study of the possibilities for accelerating and optimizing CA-based tracking algorithms in the STAR experiments.

# Chapter 2

# The STAR experiment

## 2.1 The RHIC accelerator complex

Among the particle accelerators existing today, only a few provide the ability to work with heavy ions. One of them is Relativistic Heavy Ion Collider (RHIC), located at Brookhaven National Laboratory (BNL), New York State, USA. RHIC is capable of operating with a relatively wide range of heavy ions (Cu, Zr, Ru, Au, U) and can also accelerate and collide polarized protons, which is not possible at any other accelerator in the world.

The accelerating complex consists of a set of sequential accelerators, each of them performs specific functions and ensures the operation of the facility as a whole (Fig. 2.1). The source of heavy ions and protons is the Electron Beam Ion Source (EBIS), which replaced the Tandem Van de Graaff accelerator in 2012. The primary acceleration of the proton beam is provided by the Linear Accelerator (Linac). After that, protons from Linac or ions directly from EBIS are fed to the Booster Synchrotron, where they are further accelerated. The next part of the accelerating complex, which the particles pass through, is the Alternating Gradient Synchrotron (AGS). And finally, the beam accelerated to maximum speed goes directly to the RHIC through a special AGS to RHIC (AtR) channel. In the finite part of the AtR, special magnets that separate the beam and guide it into the RHIC rings clockwise and counterclockwise are installed. Such an organization of the accelerating complex makes it possible to accelerate Au ions to energies of 100 GeV/nucleon and protons to 250 GeV/nucleon.

The RHIC accelerator consists of two rings with the same diameter and characteristics. The beams move along these rings in opposite directions and collide

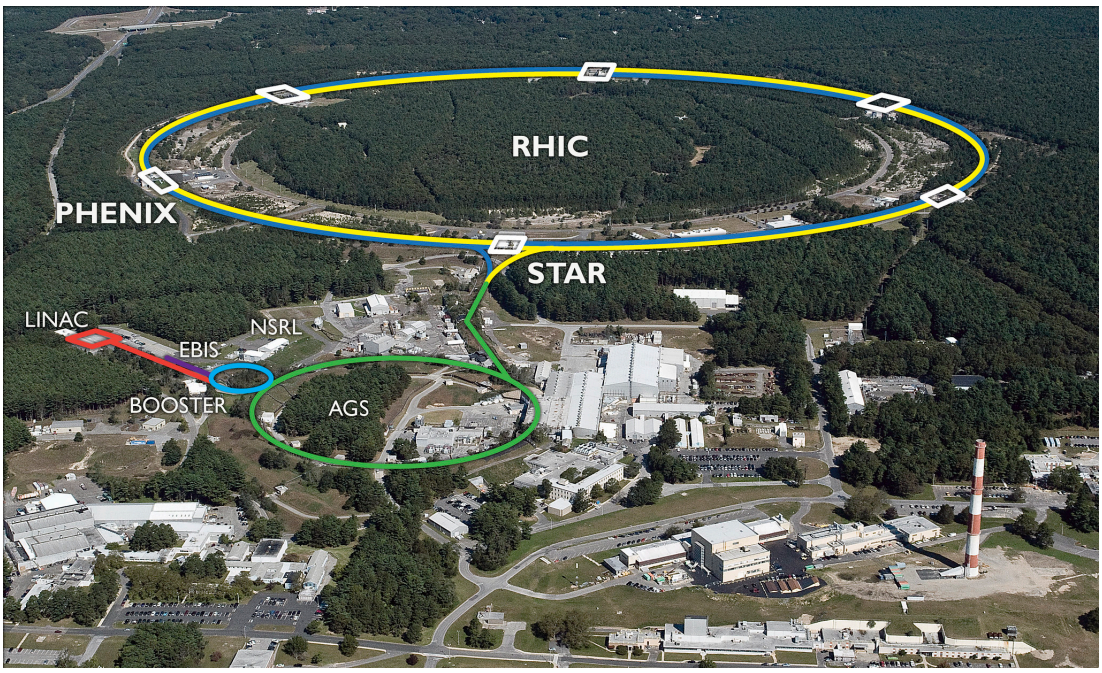


Figure 2.1: Scheme of the RHIC complex [18].

at 6 points provided for this, traditionally indicated by values in accordance with the watch dial.

The RHIC accelerator began operations in 2000. Since then, several experiments have been carried out, are currently being carried out, and are being prepared for conducting, namely: STAR, PHOBOS, BRAHMS, PHENIX, and sPHENIX.

## 2.2 The Solenoid Tracker At RHIC

The STAR experiment was developed and implemented to study the formation process and the characteristics of the quark-gluon plasma (QGP) in detail. This area of physics was extremely poorly studied at the time of preparation of the experiment. For the formation of a state of matter close to that in the first seconds after the Big Bang, it was decided to carry out studies of collisions of heavy Au+Au ions at collision energy $\sqrt{s_{NN}}$=200 GeV. According to theoretical research, such collisions should lead to the formation of a complex system, the study of which will make it possible to draw convincing conclusions about QGP. The need to carry out complex research required developing a detector system

consisting of detectors of various types, each of which specializes in detecting certain particles or parameters of their motion.

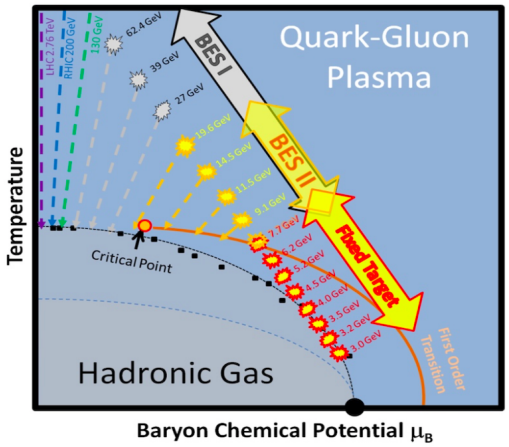


Figure 2.2: QCD phase diagram with shown coverage of the RHIC BES program [23].

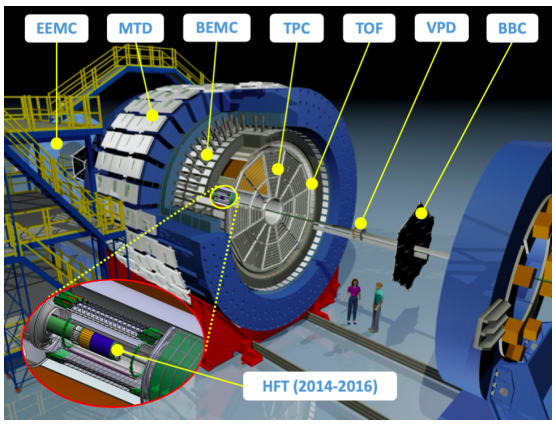


Figure 2.3: Schematic representation of the STAR detectors facility.

During its work, the STAR experiment has shown actual results and fully confirmed its significance and effectiveness [19, 20]. In 2005, a quark-gluon plasma was obtained at the RHIC accelerator. The experiment made a significant contribution to the study of the phase diagram of QCD, collective dynamics, physics of the initial state, and other paramount issues of high-energy physics [21].

Considering the results obtained, further operation of the STAR experiment was organized within the Beam Energy Scan (BES) program to draw up a QCD phase diagram map. The list of the main tasks declared for this phase of research includes the following main positions [22, 23]:

- search for the beginning of confinement and restoration of chiral symmetry;
- search for the critical point of the QCD phase diagram;
- search for first-order phase transition signatures.

In the course of the research, a large number of exciting and important results were obtained [24]. For example, scanning the phase diagram with variable collision energy showed that QGP features persist at least up to 27 GeV. Multiple observables have been found in the energy range from 20 to 40 GeV, indicating

a possible softening of the equation of state and a first-order phase transition. It was impossible to draw exhaustive conclusions about the critical point due to insufficient statistical data at the lowest BES energies.

To continue studying the structure of QCD within the STAR experiment, it was decided to expand the BES program (Fig. 2.2). The new program was named Beam Energy Scan Phase II (BES-II) [25]. BES-II aims to study the phase transition type and the critical point localization. The energy range of interest was shifted to the lower sector of the energies available to the RHIC facility in the range from 7.7 to 19.6 GeV. In order to reduce systematic errors, especially those that decrease the value of the results with small measurement statistics, the detector systems have been updated and improved in cooperation with other high-energy physics experiments. The installation of forward detector systems as part of the FiXed Target (FXT) program has expanded the available range of $\mu_B$ and $\sqrt{s_{NN}}$ by limiting the collision energy to the energy of a single beam. Upgrading the RHIC equipment has increased the luminosity of the beams, thereby increasing the statistics collected by the experiment.

## 2.3 STAR detectors setup

As a collider experiment, STAR involves the collision of two oppositely directed beams of particles with the scattering of fragments around the point of interaction. With the aim of the most complete reconstruction of events, the detecting facility is designed as a cylindrical system with a wide acceptance for particles flying at an angle to the direction of the beam. The set of detectors used in the STAR experiment is illustrated in Figure 2.3 and includes the following devices: Time Projection Chamber (TPC), Vertex Position Detector (VPD), Event Plane Detector (EPD), Barrel Electromagnetic Calorimeter (BEMC), Time of Flight (TOF), Muon Telescope Detector (MTD), Heavy Flavor Tracker (HFT);

Next, let's consider the detectors and systems of the STAR experiment in more detail.

### 2.3.1 STAR Solenoid Magnet

The central detecting systems of the STAR experiment are located inside a large cylindrical magnet designed to deflect the trajectories of charged particles and measure their impulses [26]. The magnet measures 7.32 m in diameter, 7.25 m

in length, and weighs 1100 tons. Coils of three different sizes that are cooled to room temperature by a water cooling system are used to generate the magnetic field. The diameter of the inner chamber of the magnet is 5.26 m, and the length is 6.2 m. Inside the chamber, the magnet can generate a stable and uniform field of 0.25 to 0.5 Tesla with magnetic lines parallel to the direction of the beam. The magnet frame also acts as a support system for the experiment's detectors.

### 2.3.2 Vertex Position Detector

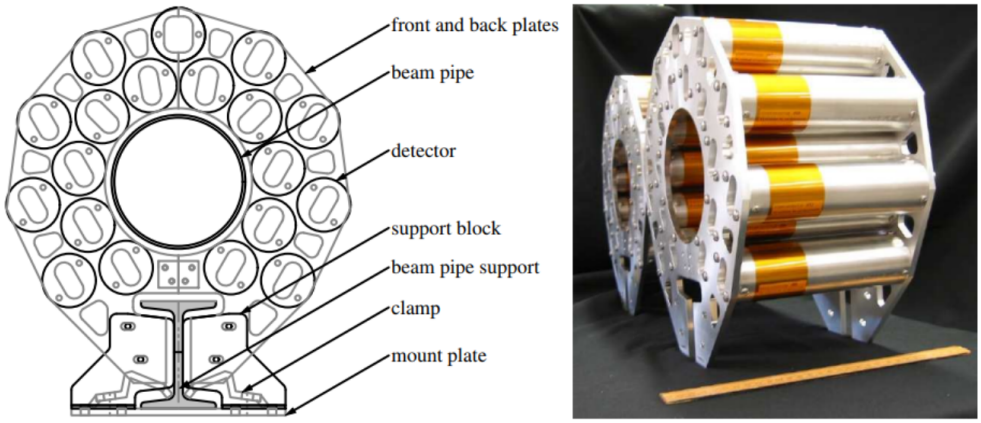


Figure 2.4: The VPD setup diagram (left) and photo of two units (right) [27].

The VPD detector in the STAR experiment executes an event trigger, noting the time of the collision relative to the autonomous master clock used by the experiment detectors [27]. In addition to directly dividing the entire mass of particle collisions into separate logical events, the VPD detector is able to determine the coordinates of the collision on the Z axis along the ray tube. VPD entered service in 2014, replacing the less sensitive pVPD (pseudo-VPD) [28], which could not cope with the detection of p+p and Au+Au collisions at energies below 200 GeV.

The detecting principle is based on the fact that the collision is accompanied by a massive ejection of photons from the $\pi^0$ decay propagating from the point of collision along the ray tube at the speed of light. The detector consists of a Pb converter, a fast plastic scintillator, and a photomultiplier tube for reading the signal. Separate detectors are arranged in assemblies as shown in Figure 2.4. In total, two VPD assemblies are used, located at a distance of 5.7 m from the center of the facility on opposite sides. At the maximum beam energies, the detector

can provide a time resolution of about 20–30 ps for Au+Au collisions and about 80 ps for p+p collisions, which fully meets the requirements of time-dependent TOF and MTD detectors. The accuracy of determining the coordinates of the primary vector in the cases of Au+Au (200 GeV) and p+p (510 GeV) events is ∼1 cm and ∼2.4 cm, respectively.

### 2.3.3 Time Projection Chamber

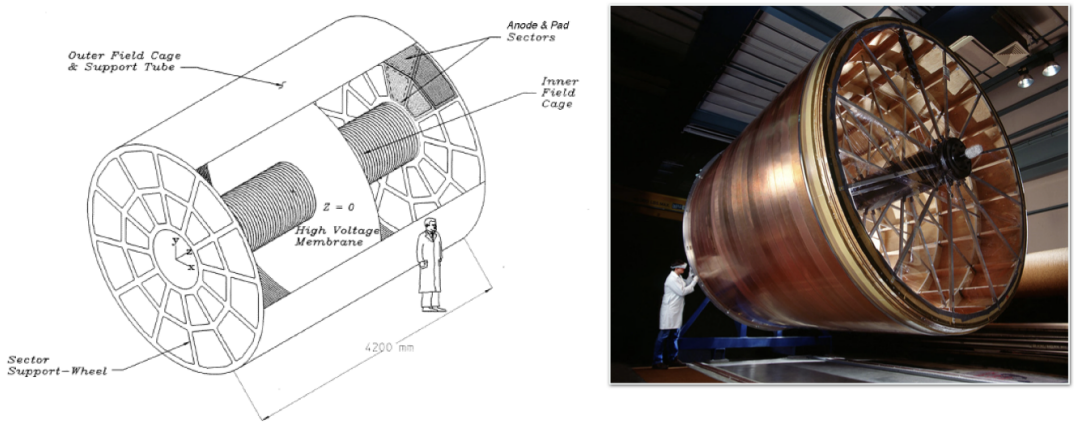


Figure 2.5: Schematic view of the TPC detector (left) [29] and a photograph taken during the installation (right) [18] of the TPC detector in the STAR experiment.

The Time Projection Chamber (TPC) is the main detector of the STAR experiment (Fig. 2.5). It does the bulk of the tracking work, characterizing and identifying charged particles. The detector makes it possible to reconstruct particle tracks reliably over the entire azimuthal angle $0 < \phi < 2\pi$ within the pseudorapidity $|\eta| < 1.8$ in the center-of-mass coordinate system, to determine their charge and momentum from 100 MeV/$c$ to 30 GeV/$c$. Tracking ionization energy losses is an important tool for particle identification, which is also made possible by the TPC detector. [29].

The detector is a hollow cylinder 4.2 m long along the beam and 4 m in diameter. In the center, the cylinder has a hole 1 m in diameter for conducting the beam and the possible installation of additional equipment. The internal volume of the TPC is filled with a special gas P10 consisting of argon (90%) and methane (10%) at a pressure of 2 mbar more than atmospheric. The detector is divided into two equal parts by a High Voltage Membrane with a voltage of about

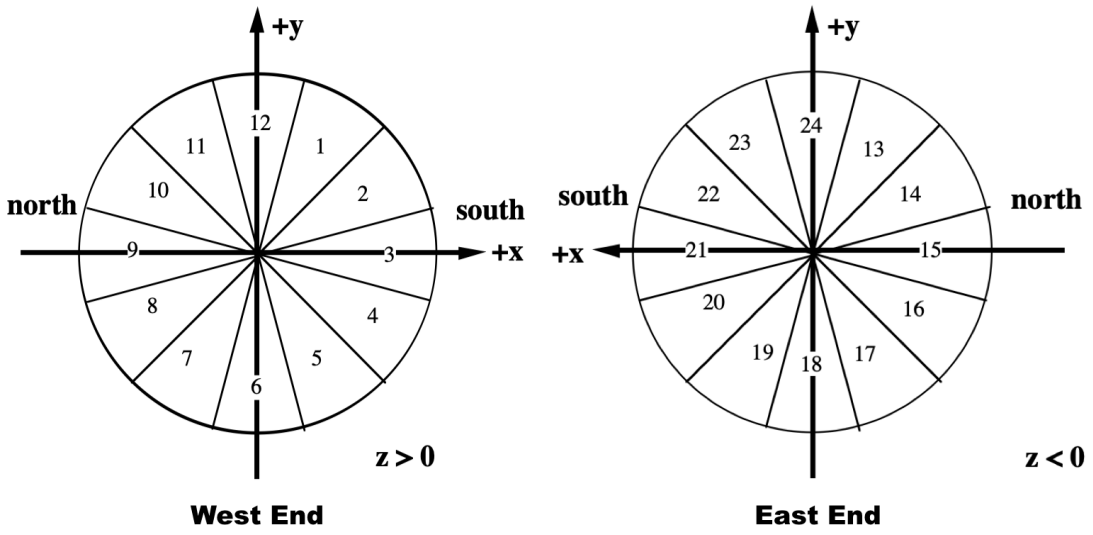


Figure 2.6: Sector numbering scheme for the West and East sides of the TPC detector, superimposed on the global coordinate system [30].

28 kV, perpendicular to the direction of the beam. These parts are traditionally referred to as Western (Z $>$ 0) and Eastern (Z $<$ 0). The sides of the cylinder are the anodes, the outer surface is occupied by the Outer Field Cell (OFC), and the surface of the central hole is the Inner Field Cell (IFC). All of these components work together to form a field cage that provides an ideal uniform electric field inside the TPC. Any distortion of the electric field can lead to errors in the measurements of the particle trajectories. Moreover, the field cage helps prevent the TPC gas from being contaminated by the outside air. In addition to the anodes, contact pads and a supporting structure are installed on the lateral edges of the detector. The pads on each side are geometrically organized into 12 slices or sectors. The numbering of slices and their correspondence to the global coordinate system are shown in Figure 2.6.

The operation of the TPC detector is based on the principle of a drift chamber [31]. When the charged particle passes through the gas in the TPC, it ionizes atoms of that gas at a distance of up to several centimeters, leaving behind clusters of free electrons. Under the influence of the constant electric field inside the field cage, electron clusters drift at a constant speed to the contact pads located at the edges of the detector. While maintaining maximum uniformity and stability of the field, the electron drift time allows you to determine how far from the anode ionization has occurred. That is, to get the coordinate of the particle's

flight along the Z axis. The X and Y coordinates of the particle are determined by the geometric position of the triggered contact pads.

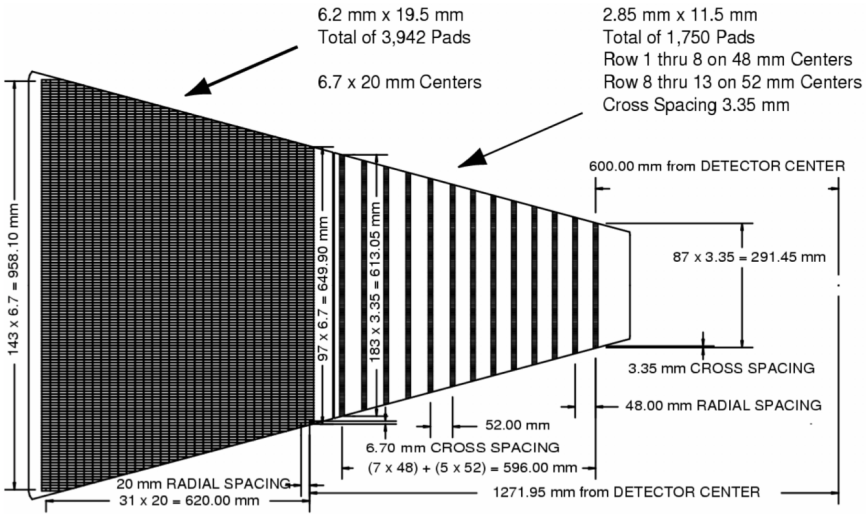


Figure 2.7: The layout of the pad rows of one slice of the TPC detector, intended for reading drifting electrons: 32 pad rows on the outer part and 13 pad rows on the inner part of the detector [29].

The TPC detector includes 24 slices, 12 slices on each side of the detector. One slice contains two sets of sensors — external and internal. The sensory parts are equipped with contact pads installed in rows perpendicular to the centerline of the slice. Initially, the inner part of the TPC was fitted with larger platforms with a larger row spacing than the outer part, forming a different number of rows (Fig. 2.7). The inner part of the TPC consisted of 13 rows, while the outer part included 32 rows with similar radial dimensions of these parts.

Extension of the physical program of the STAR experiment as part of the Phase-II of Beam Energy Scan (BES-II) required an upgrade of the detector to increase acceptance and reduce the minimal momentum of the recovered tracks [32]. To this end, in 2018, internal (iTPC) pad planes were replaced in all 24 slices. The 13-row sparse solution was replaced with 40-rows sensors, bringing the total number of pad rows in a sector to 72. The updated detector has been used in the experiment since Run 19 (2019).

### 2.3.4 Time Of Flight

Time Of Flight (TOF) [28] in the STAR experiment is a single-layer barrel structure surrounding the TPC detector. TOF consists of 120 oblong trays, each of which includes 32 Multi-gap Resistive Plate Chamber (MRPC) [33] modules. The western and eastern parts of the detector contain 60 trays each with 6°coating in azimuth angle and one unit in pseudorapidity, thus providing full acceptance in azimuthal angle and pseudo velocity range $|\eta| < 1$.

The TOF detector is a powerful particle identification tool that complements the capabilities of the TPC. Acting in conjunction with a VPD detector, the barrel TOF can measure the time of flight of each particle as the difference between the time cut of the collision and the recorded time of flight of the particle. Further, the hits left by the tracks in the TOF detector are combined with the tracks reconstructed in the TPC. This allows making the most of the capabilities of both detectors for particle identification. The use of TOF information is crucial for high-energy tracks, for which TPC data alone may not be sufficient for robust identification. Knowing the time of flight of the particle and the length of the trajectory, it is possible to calculate its velocity ($\beta$). Together with the momentum information obtained from the TPC detector, particle mass can be calculated and used for reliable identification.

### 2.3.5 Barrel Electromagnetic Calorimeter

Barrel Electromagnetic Calorimeter (BEMC) [34] surrounds the entire outer surface of the TOF detector, being inside the magnetic coils. The logic and acceptance of the BEMC are similar to the TOF detector. Like TOF, BEMC covers the full azimuth angle and pseudo velocity range $|\eta| < 1$. The calorimeter includes 60 modules each on the West and East sides of the detector. Each module covers one unit in pseudorapidity and 6degree in angle. The module comprises 40 separate elements (towers) — 2 in azimuth angle $\times$ 20 in $\eta$. They consist of 21 active 5 mm thick oscillator layers alternating with 20 5 mm thick lead absorber layers. BEMC modules are inserted from the ends of the detecting system using special rails attached to the magnet iron between the coils. BEMC electronics units are located outside the magnet on its outer surface.

The BEMC has a very fast response speed allowing it to track RHIC collisions up to 9.35 MHz in real-time. Other detectors are not able to maintain such a

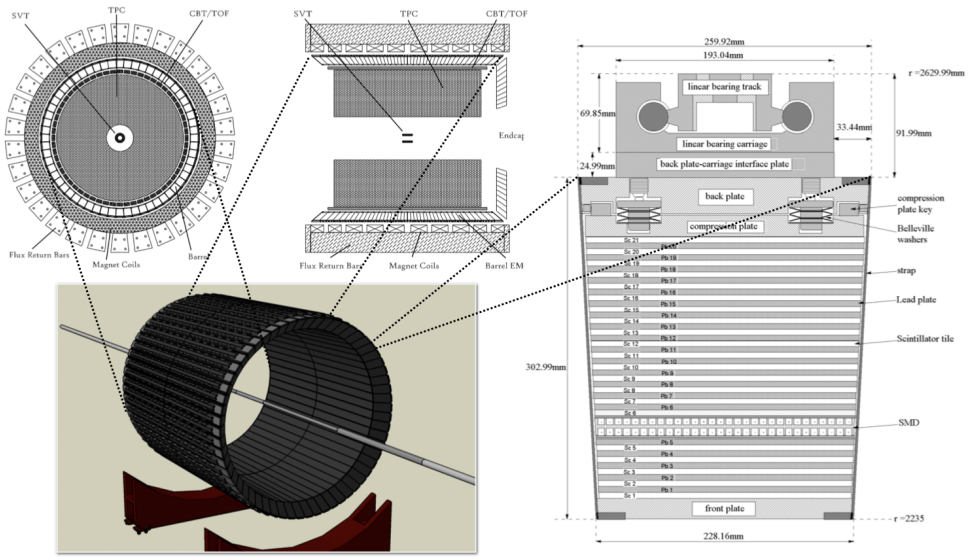


Figure 2.8: 3D view of the Barrel Electromagnetic Calorimeter (bottom left), cross-sectional view of the STAR detector showing the location of the BEMC between the TPC detector and the magnet rings (top left), side view of the STAR EMC module showing the mechanical assembly including compression components and rail mounting system (right).

rate of information reading. Therefore, it is necessary to pre-select the events of interest. The BEMC is able to read information both from separate towers individually and in total. Different physical cases have different BEMC responses. For example, electrons leave behind a much larger part of their total energy than hadrons. Thus, using transverse energy as a threshold, BEMC acts as a trigger to prompt events of a certain class during online data acquisition.

### 2.3.6 Muon Telescope Detector

Muon Telescope Detector (MTD) [35] is located outside the magnet, on top of the BEMC electronics. The presence of a significant amount of material between the point of collision of the beams and the detector makes it possible to filter out most of the formed particles, except for muons, which have the higher penetrating ability. MTD consists of 30 lines of 5 modules each. The modules are arranged as shown in Figure 2.9. This arrangement allows for overlapping 45% of the azimuth angle and the pseudo velocity range $|\eta| < 0.5$. Installation of the detector began in 2012 and ended in 2014. Like the TOF detector, the MTD uses MRPC technology

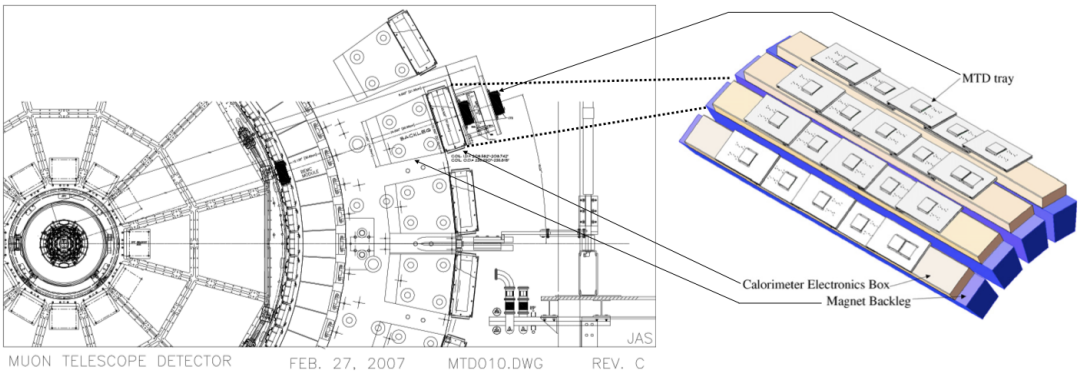


Figure 2.9: Layout of the MTD detector outside the magnet on top of the BEMC electronic blocks (left) [35] and a 3D model including MTD detector elements, BEMC calorimeter electronics blocks and magnet supporting structure (right) [36].

to register particles. MTD is a fast detector with a temporal resolution of 108 ps. The spatial resolution of the detector is 1.4 cm in Y (perpendicular to the beam) and 1.0 cm in Z (parallel to the beam).

Being a fast enough detector capable of tracking muons, MTD can trigger various types of events. So, the MTD together with the TOF detector can be used to trigger cosmic-ray events. The MTD with the VPD can trigger minimum bias events. Together with the electromagnetic calorimeter, it is possible to start $e$–$\mu$ events. And finally, if there are at least two hits in the MTD, a $\mu$–$\mu$ event can be triggered.

### 2.3.7 Heavy Flavor Tracker

Heavy Flavor Tracker (HFT) [37] was installed inside the TPC detector close to the collision of the beams (Fig. 2.10). This position was justified by the main goals of this detector, namely, to improve the resolution of the primary vertex and secondary vertices formed by decays of short-lived particles. The detector included four cylindrical stations with different characteristics and operating principles according to the project.

The two innermost HFT layers are based on monolithic active CMOS pixels (MAPS) and are located at a distance of 2.8 and 8 cm from the beam line [38]. Mechanically, the PiXeL (PXL) [39] detector is divided into 10 sectors, each of

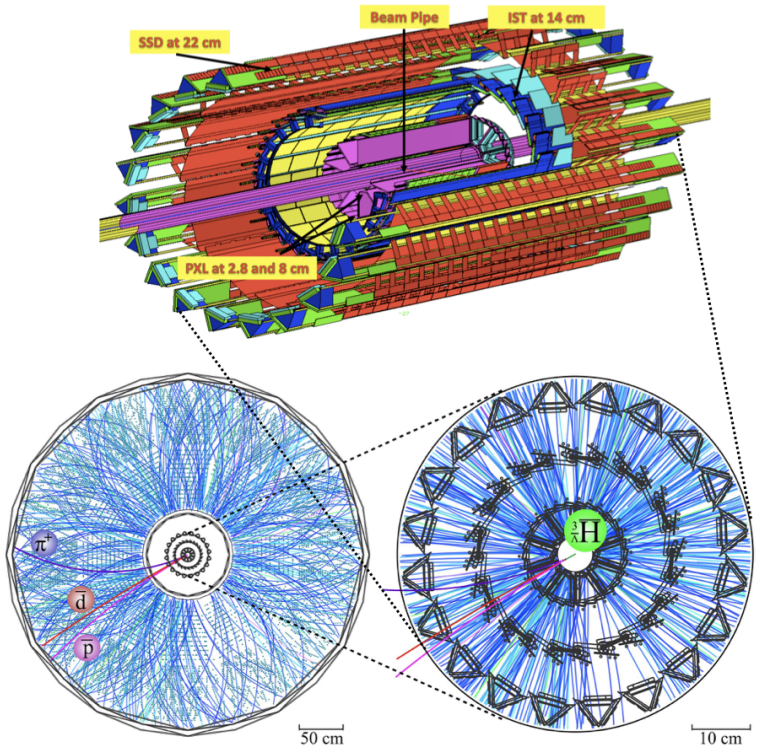


Figure 2.10: 3D model of the HFT detector, including the Beam Pipe, two layers of PXL, IST and SSD detectors (top) and the HFT detector integration scheme in the central part of the TPC detector in the plane perpendicular to the beam direction, explaining the effect of HFT on the positioning accuracy of the primary and secondary vertices (bottom).

which is a thin trapezoidal carbon fiber sector tube. On the inner surface of the sector facing the beam, a single touch panel per sector is installed. On the outer surface of each sector, there are 3 touch panels in the form of a ladder. The detector's sensors are based on pixels with a step of 20.7 $\mu$m and 185.6 $\mu$m. The sensors do not require special liquid cooling systems; an airflow is sufficient to remove heat, which minimizes the material budget.

The third detector station is the Intermediate Silicone Tracker (IST), located 14 cm from the beamline. The IST is based on single-sided silicon pad sensors with a pad size of 600 $\mu$m $\times$ 6 mm. The Silicon Strip Detector (SSD) [40] represents the outer layer of the HFT. The SSD has a radius of 22 cm and is built based on double-sided silicon strip sensors. The detector consists of 20 steps (ladders), each of which has 16 sensors (wafers) with a spatial resolution

of 20 $\mu$m in R–*phi* and 740 $\mu$m in Z. The HFT detector made it possible to significantly improve the resolution of the distance of closest approach (DCA) from ∼1 mm (by TPC) to ∼30 $\mu$m (by PXL).

## 2.4 High Level Trigger

The actual frequency of collisions in the RHIC accelerator significantly exceeds the temporal capabilities of some STAR detectors and the DAQ system. On the other hand, many of the collisions are of no physical interest.

For the purpose of the primary selection of events in the STAR experiment, a system of triggers was developed that initiates the collection of information by slow detectors and allows to filter out the least interesting events. [41]. The original architecture assumed the presence of three main and one additional trigger firing sequentially [42, 43]. With the help of the information obtained from fast and some slow detectors, each trigger level is able to ensure that events are screened out according to specific criteria, thus reducing the volume and rate of data receipt for each subsequent level. Therefore, Level-0 is designed to operate synchronously with the RHIC's beam-crossing counter at around 9.37 MHz and can identify potentially interesting events. Depending on the studied physical processes, the trigger can operate with the information from different fast detectors (VPD, MTD, calorimeters) and use different screening conditions. For example, the MTD detector registration of at least two muons during muon decay. Level-1 can analyze the general characteristics of the event and interrupt data collection during the drift of secondary charges in the TPC. Level-2 can analyze the calorimeter data more deeply and interrupt data collection while the TPC detector reads responses, thus reducing the data rate to 100 Hz. Finally, after the DAQ system has collected all the data, the Level-3 trigger can perform a simplified reconstruction of the tracks in the TPC and decide to accept or discard each specific event. The final output of events with a frequency of about 1 Hz was consistent with the capabilities of the reading equipment with an average TPC event size of about 15 MB.

In the course of the experiment, many systems were improved, which led to a change in the functionality and roles of the triggering systems. The Level-0 trigger received additional opportunities for analyzing events in real-time, removing some of the tasks from later levels. The overall read-out speed of the TPC has also

increased significantly, resulting in ineffective interrupts in Level-2. The role of Level-1, in many respects, was reduced to issuing data read interrupts to protect against pileup in TPC.

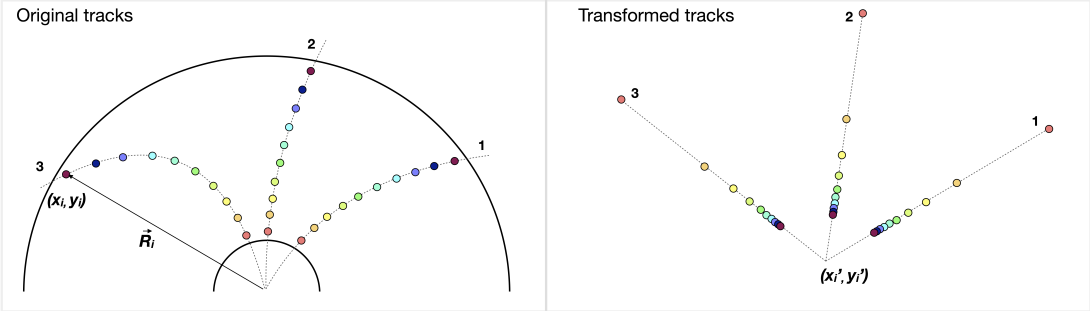


Figure 2.11: Particle trajectories in TPC before conformal mapping (left) and after conformal mapping (right).

The Level-3 trigger was developed and first launched in 2002. TPC track reconstruction was based on conformal mapping algorithm [44]. The input data for the algorithm is a set of hits that carry information about the TPC pads activated by the flying particles: the number of the sector where the response was received, the number of the pad row, the spatial coordinates $(x, y, z)$ of the hit. The algorithm for reconstructing tracks using conformal transformation is based on the fact that the trajectories of particles moving in a constant magnetic field are bent and form circles of different radii. Using the equation of the circle in the plane, the coordinates $(x, y)$ of the hits are converted from the standard $(XY)$ to the modified $(UV)$ coordinate system. As a result, the initially curved lines become straight if the particle trajectory starts in the immediate vicinity of the primary vertex (Fig. 2.11).

$$R_i^2 = (x_i - x_t)^2 + (y_i - y_t)^2 \tag{2.1}$$

$$u_i = (x_i - x_t)/R_i^2 \tag{2.2}$$

$$v_i = (y_i - y_t)/R_i^2 \tag{2.3}$$

In 2009, in the STAR experiment, the DAQ1000 upgrade was carried out, which increased the data acquisition capabilities up to a frequency of 1 KHz for Au+Au collisions at $\sqrt{s_{NN}}$=200 GeV. To ensure fast reconstruction and selection of events in real-time, the High-Level Trigger (HLT) was developed based on the STAR

Level-3 trigger system [45]. HLT 1.0 (2010–2012) did not have its own computer farm and ran on the CPU of TPX machines. Information from each of the 24 TPC sectors was processed independently, ensuring parallel execution of tasks. At the same time, further scalability of calculations was not possible.

HLT 2.0 (2013–2016) received significant improvements in both speed and efficiency of track reconstruction. The trigger was provided with its own independent computer farm. Lack of flexibility and the impossibility of efficient use of available computing resources required replacing the old Level-3 algorithm with a new tracker based on the Cellular Automaton (CA). The peculiarities of the operation of this algorithm will be discussed in detail in the subsequent chapters of this work. The new tracking algorithm, being initially fast, offers significant scalability, making it ideal for fast, real-time track reconstruction. In addition, the CA track finder has demonstrated extremely high track reconstruction efficiency, exceeding by 10% the algorithm based on the track following procedure used in the STAR offline reconstruction. Unlike the old Level-3 tracker, the CA tracker is capable of reconstructing both primary and secondary tracks, regardless of how far from the primary vertex the particle was born. In fact, the use of the CA tracker made it possible to carry out a full-fledged reconstruction of events already at the selection stage. On this basis, a powerful and fast Kalman Filter based particle identification package, KF Particle Finder [46], was experimentally included in the HLT chain. In the course of work under the BES program, KFP article Finder has demonstrated its high efficiency in conjunction with the CA tracker, confidently identifying particles and conducting physical analysis at the HLT level online.

By 2017, the number of available CPU HLT threads in the farm exceeded 1000, and more than 10000 Intel Phi threads were added. The new version of HLT 3.0, operating since 2017, includes the CA tracker and the KFParticle on a full-fledged basis, which allows full reconstruction of events and physical analysis in real-time, despite the growing luminosity of the RHIC accelerator. Thus, the High-Level Trigger today is a unique platform for researching the possibility and features of online event reconstruction and analysis.

# Chapter 3

# High Performance Computing in High Energy Physics

Complex and expensive accelerating and detecting systems are the essential components of modern high-energy physics experiments. They are designed to carry out collisions of particles under given conditions to produce and collect results of such interactions. But the purpose of such investigations is not the accumulation of data, but, first of all, its analysis in order to confirm or refute the ideas laid down in the experiment. The modern HEP, notably the STAR experiment considered in this work, aims to study extremely rare events. The expected detection rate for some of them may be about one in tens of millions. In order to create the necessary statistics, modern accelerator complexes are built, taking into account the need to generate millions of collisions per second. This spawns enormous amounts of data that need to be processed and analyzed.

Data volumes in HEP are so large that manual processing and evaluation of the results are impossible. The critical condition for successful experiments has long been the widespread use of mathematical algorithms and information technologies for maximum data processing automation. There are several important directions related to acquiring and using the information in the HEP experiments.

- Reading data from detectors and transferring it for further analysis. The problem is associated with constructing a fast and reliable electronics system capable of receiving responses from detectors, transforming them into compact packets convenient for further work, and sending them to computational clusters. The DAQ elements of the STAR experiments were briefly discussed in the previous chapter.

- Storing and providing access to the data. A large HEP experiment can generate many terabytes of data that must be processed and stored for additional research, reanalysis, etc.

- Development of the mathematical algorithms for the reconstruction and analysis of events. Such operations as the reconstruction of particle trajectories from detectors' responses turn out to be nontrivial for a computer and require the development of special mathematical algorithms. The problem of track finding using a Cellular Automaton will be considered in more detail in subsequent chapters of this work.

- Make the most computing power to achieve optimal performance for reconstruction algorithms and event analysis. Modern computer systems offer significant opportunities for speeding up computations. Programs optimized for these capabilities can run at speeds tens of times faster than the initial results. This is very important when processing vast amounts of physical data, especially in real-time.

This chapter will touch on the last item of the above list and consider the possibilities of modern computer architectures to speed up computations.

## 3.1 Classification of parallel computing systems

The use of parallel computing in computer systems has a long history dating back to the second half of the 20th century. So, in 1966 at the University of Illinois, the ILLAC-III (Illinois Pattern Recognition Computer) computer was developed and built [47], which used parallel computing to detect particles in bubble chamber experiments. For a long time, parallel computing was the prerogative of supercomputers and specialized devices and was practically not represented among the mass market solutions. The situation began to change after 1999, with the release of the Intel Pentium II [48] processor, which includes a special set of instructions for performing parallel computations at the data layer — Streaming SIMD Extensions (SSE). A few years later, AMD also added support for SSE instructions starting with the Athlon line of processors. The first multicore processor, POWER4, manufactured by IBM, appeared in 2001. And finally, 2005 can be considered the start of the multicore era on the CPU market, when AMD and Intel presented 2-core solutions.

The transition to the multicore solutions in the CPU structure has become a logical step for personal computers due to the impossibility of increasing the frequency of processors at the same rate as before. The reason for that is that the operating frequency of modern processors is limited primarily by the duration of the instructions' execution stages. The two main ways to shorten the CPU instruction stages are to move to a more advanced technical process or increase the chip's voltage. A more advanced technical process involves reducing the distance that the electrical impulse has to overcome, the switching time of transistors, etc. The reduction of the technological process is ongoing, but it is a slow and laborious process that cannot meet the demand for an increase in computing power. On the other hand, growing the voltage on the chip leads to a rise in heat generation in proportion to the cube of the voltage. Using extremely powerful cooling systems makes this method practically inapplicable for the mass user. In such conditions, it is the increase in the number of cores on a chip that becomes the optimal way to ensure a permanent increase in the computing power of processors, provided that tasks are divided into threads with their parallel execution.

Thus, modern hardware solutions provide various possibilities to optimize and accelerate computations. The use of some technologies can be optimized, while others require taking into account their peculiarities in the algorithm and the use of specialized tools. Therefore, when engaging in high-performance computing, always need to be clear about what technologies and capabilities are available and how it is possible to use together with a particular algorithm to make the most efficient hardware utilization. A logical solution for this is the systematization and classification of computer architectures in terms of the ability to carry out parallel computations.

Perhaps the most famous and frequently used classification system for computer architectures today is Flynn's Taxonomy [49], proposed by Michael Flynn (Fig. 3.1). This classification is based on how many streams of commands and data the device can process simultaneously. Flynn's Taxonomy divides computer architectures into four main types.

**SISD (Single Instruction stream, Single Data stream)** — the most straightforward scheme of work, in which only one command is executed at a time on a single data element. Considering that superscalar and pipelined processors (the features of which will be described later) are also referred to as SISD architecture, we can say that this approach was dominant in the personal computer market until the beginning of the 21st century. Many of the classic

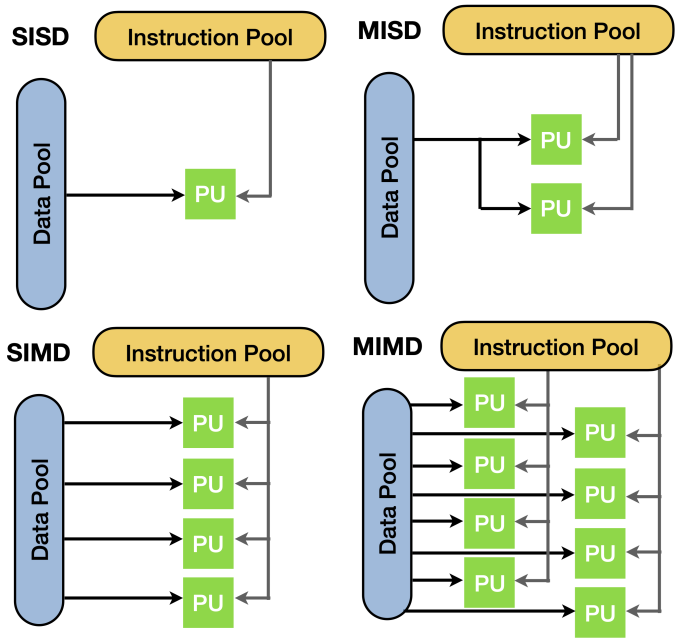


Figure 3.1: Flynn's Taxonomy.

algorithms and data structures used by programmers to this day were explicitly designed to work with the SISD architecture. Despite the twenty-year history of the development of parallel technologies on the PC market, it is scalar data types (such as int, float, char in C++) and single-threaded calculations that remain in demand and popular.

**SIMD (Single Instruction stream, Multiple Data stream)** — a computer architecture involving the execution of the same instructions on several data items at the same time. Initially, SIMD assumes using both one and several processor cores to ensure this process. SIMD calculations with multiple cores are performed using matrix processors, which are an array of simple processors with their own memory under the reign of a single control processor. But the modern understanding of this architecture usually implies using a single core, equipped with enlarged SIMD registers and a special set of instructions. SIMD instructions operate on datasets of specific lengths instead of single elements. Such datasets are called vectors, the approach itself is vectorization, and CPU devices that support this technology are named vector processors. Almost all modern mainstream processors are vector and support multiple SIMD instruction sets. The very first set of such instructions can be considered MMX (MultiMedia eXten-

sions) from Intel [50]. MMX technology was an extension for encoding video and audio streams. MMX is not a full-fledged SIMD architecture since it does not use its own registers but 64-bit general-purpose registers intended for scalar floating-point calculations. Perhaps the most popular is the SSE (Streaming SIMD Extensions) instruction set, based on the 128-bit registers. The original SSE received several additions, making this technology extremely flexible and functional. 128-bit registers allow you to operate with 4 elements of integer (int32) or floating point type, and with 2 elements of long integer (int64) or double precision, with 8 elements of short integer (int16), work with strings, perform horizontal vector operations with numbers stored in one register, logical operations and more. AVX (Advanced Vector Extensions), proposed by Intel in 2008, became the next step in developing the SIMD technologies. The new standard offered an updated instruction coding system, introduced non-destructive operations with three operands (VEX coding scheme), and expanded the registers' capacity to 256 or even 512 (AVX-512) bits. Another set of SIMD instructions specific to the first generation of Intel Xeon Phi cards is IMIC (Intel Many Integrated Core), which supports 512-bit registers and allows you to operate with 16 floating point elements at a time. The AVX-512 took its place in the second generation of the Xeon Phi cards. It is also worth noting that for video cards, a special class of SIMD is allocated — Single Instruction Multiple Threads.

**MISD (Multiple Instruction stream, Single Data stream)** — a rather rare in-use class of computer architectures, which involves the performance of various operations using different functional devices on the same data set. MISD is mainly used in cases where the absence of errors and failures in the execution of calculations is critical. This method of masking errors is also called task replication. The most striking example of the use of this architecture was the control computers of the Space Shuttle program. Systolic arrays are another example of MISD architecture. But this issue is still controversial being subject to debate among researchers.

**MIMD (Multiple Instruction stream, Multiple Data stream)** — the most versatile class of Flynn's Taxonomy, which implies the execution of different instructions on different data at the same time. MIMD refers to task-level parallelism and requires more than one core or processor. In this case, parallelization is based on dividing the program execution process into several threads, each of which can be executed separately. The number of threads may exceed the number of available cores, and some threads may be idle. This approach enables

more efficient use of hardware resources and more benefits from technologies such as Intel Hyper-Threading. The negative side of this technology is the danger of mutual blocking of threads and the emergence of a race to possess resources. Therefore, parallel computing at the task level often requires careful utilization of special control structures (semaphores) and thoughtful access to the memory and various functions from different program threads.

Talking about parallel computing, we can also distinguish several different kinds or levels of parallelism.

### 3.1.1 Instruction Level Parallelism

The deepest level of parallelism in computations can be considered Instruction Level Parallelism. In basic terms, the task of the CPU is to execute the micro commands after the other in sequence. In reality, many instructions can be executed independently using different function modules and registers. Thus, scalar execution of instructions by the processor leads to downtime and inefficient use of the available processor power. Over the decades of computer technologys' development, many methods have been proposed to optimize the CPUs' operation. Some of them are successfully utilized and developed in modern processors.

**Conveyor.** A long command to the microprocessor can be split into individual instructions that are executed in sequence. The code read from memory is not executed immediately but is placed in a queue. Proper planning allows distributing the execution of instructions in such a way as to ensure the maximum load of the functional modules of the processor (simultaneous execution of several commands) while at the same time maintaining the order of execution of instructions within each command. Depending on the processor model, the number of commands executed simultaneously (pipeline levels) can reach several tens. Despite its efficiency and the ability to increase the number of instructions issued per clock cycle (IPC), the pipeline is not without drawbacks compared to the pipelineless systems. A severe problem with such processors can be considered a significant drop in the efficiency of the pipeline operation with an abundance of conditional statements in the code. In this case, the command queue has to be constantly changed or completely rebuilt, which leads to additional overhead costs and the inability to use the technology to the fullest.

**Superscalarity.** Another way to increase the number of instructions that the processor can treat simultaneously while remaining within the scalar paradigm

of Flynn's Taxonomy. Being ideologically related to the pipeline, superscalarity has a fundamental difference. The technology involves creating and utilizing additional functional modules of the processor, duplicating the main ones. At the same time, superscalarity is perfectly combined and complements the pipeline. Thus, the kernel can simultaneously execute several instructions processed by the pipeline if these instructions do not depend on each other. In earlier versions, the kernel made the concurrency decision, increasing the time and resources involved.

**VLIW (Very Long Instruction Word).** A special architecture for superscalar processors in which a single processor instruction contains several operations that must be executed in parallel. Thus, an essential disadvantage of superscalar systems is solved — the decision on simultaneous operations is passed from the kernel to the compiler. The price for this is the low density of the code, which contains many empty instructions in situations where concurrent execution is not required.

### 3.1.2 Data Level Parallelism

Data level parallelism follows the SIMD architecture of Flynn's Taxonomy. It is based on performing operations not on one operand but on a group of variables simultaneously. As mentioned earlier, special instructions and processor registers that are longer than necessary for normal operations perform parallel operations on data. The modern set of instructions available for SIMD calculations is quite diverse. It allows performing both arithmetic and logical operations between operands stored in different registers and horizontal operations within the same register. The use of the vector (SIMD) registers is now ubiquitous. All modern processors that have entered the market in the last two decades are invariably equipped with them in addition to the classic registers.

Thus, vector computing is a cheap way to significantly improve program performance without using more powerful hardware. The theoretically achievable speedup from the utilization of SIMD instructions instead of scalar calculations is determined by the number of operands that can be placed in one SIMD register, provided that there are instructions for working with variables of this type. In real life, the situation looks a little more complicated. Any vector computation has an overhead that can significantly reduce actual speedup relative to expected. The first logical reason for the overhead is memory access operations when forming a vector from individual elements. This problem is partially offset by utilizing

certain data structures to store the variables involved in parallel computations. Minimizing random access to the memory and timely loading the data into the CPU cache significantly reduces overhead and allows to approach the theoretical maximum speed. Another technique for increasing the efficiency of SIMD calculations is to ensure the maximum possible number of productive processing cycles of data loaded into registers. Thus, the negative effect of memory operations is leveled, and the benefit from vector computing becomes maximum.

Another negative factor concerns the process of instructions operating with very long registers, such as AVX-512 and, to a lesser extent, AVX2. Working with larger registers also requires longer instructions. Such instructions, in turn, increase the power consumption of the CPU during execution. Intel has developed and implemented CPU Throttling technology into its processors to maintain control over power consumption. It lowers the base frequency of the processor when using the AVX2 or AVX-512 instructions. If the code uses only SIMD computations with high vector occupancy, the advantage is achieved due to the simultaneous processing of a larger number of elements. But with incomplete vectorization, there may be cases when the AVX code is executed slower than the SSE analog.

Based on the above facts, it is easy to conclude that utilization of data parallelism requires additional efforts on the programmer's part to ensure the most complete and efficient use of vector registers. Most of the existing traditional algorithms were created for scalar computing. Unfortunately, some of them may be difficult to vectorize or not possible at all. Almost any algorithm with a high level of combinatorics can be cited as an example of a complex vectorized problem since vector computations, in this case, are usually associated with significant overheads when accessing memory to iterate over variables.

In recent years, the task of vectorization has slowly begun to be simplified. Many compilers provide the ability to vectorize increasingly complex code automatically. Classical algorithms such as sorting and searching are being replaced by their SIMD counterparts. But despite this, the full use of data parallelism remains a serious challenge for the programmer.

### 3.1.3 Task Level Parallelism

As mentioned earlier, at a certain level, the development of microprocessors was faced with the impossibility of increasing the processor frequency at the same rate.

The solution to the problem of the rising power of CPUs was found in increasing the number of processor cores. Several independent cores with their own register sets and high-level memory caches (L1, L2) allowed devices to perform several different operations simultaneously. Parallel execution of tasks is based on the concept of a thread — an ordered set of instructions entering the processor input.

The operating system provides utilization of command streams at the logical level. More precisely, at the OS level, this property is usually called multitasking. It can provide 'simultaneous' execution of several tasks (processes). Multitasking appeared long before the market for multiprocessor systems. Still, it was implemented by switching a single processor core to execute the thread with the highest priority while the rest of the threads wait their turn.

At the physical level, attempts to optimize the simultaneous execution of threads were made even before the advent of multicore systems with the implementation of the Simultaneous Multithreading (SMT) technology, which is essentially a development of the ideas of instruction-level parallelism. In this direction, the most famous technology was Intel Hyper-Threading, published in 2002 and is invariably embedded in Intel processors until now. The idea of the technology is to optimize the utilization of processor resources without creating additional elements by distributing the tasks of one thread into functional modules to those not involved by another executable thread. This technology effectively separates read/write (first thread) and processing (second thread) operations. Possible acceleration differs from task to task and can reach 30% compared to the operation of the same processor in standard single-threaded mode.

True parallelism at the task (or thread) level is ensured by the multiple cores in the system available for computation. Each core is a somewhat independent device with its own memory and the ability to execute its own separate command stream. If necessary, processes running on different cores can be synchronized, exchange data, or use the same data by a common low-level cache (L3, L4), expanding the possibilities of utilizing parallel computations. Thus, the thread-level parallelism using multicore computer systems most fully implements the MIMD concept in Flynn's Taxonomy.

Compared to data parallelism, thread parallelism is more flexible since splitting a program into several independent instructions threads is often more accessible than organizing data arrays for vector computations. At the same time, the process of program parallelization is also not devoid of several difficulties and possible problems. Even if we discard non-parallelizable algorithms, the execu-

tion of which must be performed strictly sequentially, the process of dividing computations into simultaneously executing instruction streams is often a non-trivial task. In ideal conditions, the streams should be completely independent of each other. In reality, concurrently running threads can interact with each other accidentally or as needed. Incorrect design of parallel code can lead to hard-to-detect 'floating' errors, also called race conditions [51]. The reasons for the race condition are most often the dependence of the execution of one thread on variables changed by another thread or the uncontrolled change of the same data by several threads. Cache synchronization protocols allow excluding situations when different versions of the same data appear in different caches but still do not protect data from errors in the logic and order of the algorithm execution. To ensure the stability of parallel code in cases of the risk of the race condition, programmers use a system of semaphores and locks, which defines the rules and order of memory access and execution of certain functions for different threads. This approach can negatively affect the performance of individual threads waiting for their turn to use resources, but it can make the program stable if properly planned.

There is no doubt that thread-level parallelism is an extremely powerful mechanic for improving the performance of computer programs. This cannot be ignored by programmers who care about the speed of their code execution. At the same time, do not forget about vector registers, an independent set of which is available in each core of a modern processor. A successful combination of data-level and thread-level parallelism can give acceleration up to several tens of times even for a home PC. In contrast, this figure can rise up to several hundred times on servers used for scientific computing.

## 3.2 Architecture of modern computing devices

### 3.2.1 CPU architecture

The Central Processing Unit is the heart of most modern computing systems. The history of processors' evolution goes back several decades, during which the devices underwent significant changes developed new or lost outdated technical solutions. But the basic principles of building the CPU remain largely unchanged.

Almost any modern processor is based on the architecture proposed by John

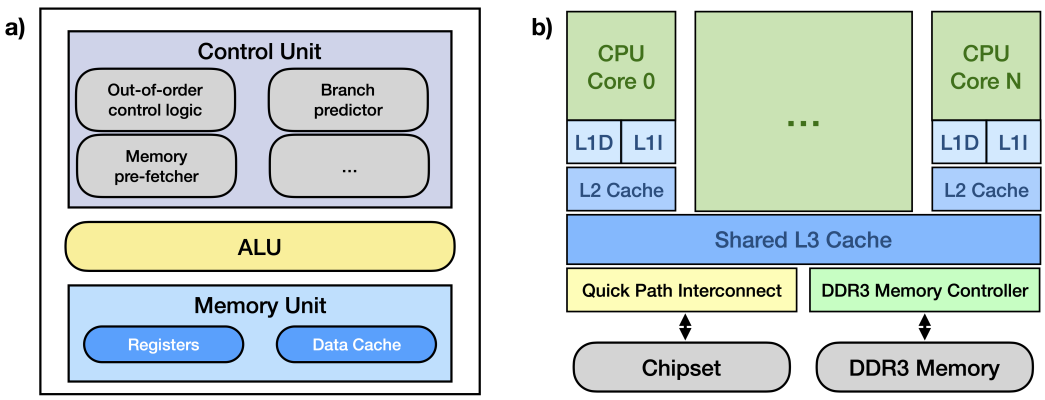


Figure 3.2: The structure of modern CPUs: a) Processor logic; b) Scheme of memory organization in modern processors of the Intel Core i7 family.

von Neumann over 70 years ago [52]. The cornerstone of von Neumann's architecture is the principle of storing data and instructions in the same memory. And also the very structuring of memory in the form of a set of numbered cells. The processor can access the memory cells by reading the instructions that it must execute and the data it must operate. The maximally simplified diagram of the implementation of the von Neumann architecture for modern processors is shown in Figure 3.2 a. The Arithmetic Logic Unit (ALU) is the very heart of the processor, responsible for the direct execution of microinstructions and basic operations. It has processor registers adjacent to it — a small number of the fastest memory elements of limited length. The size of the registers can vary from a few bytes to several hundred bytes. Registers contain data that the ALU directly operates on and is not intended for long-term information storage. The Control Unit is the command center for the processor. It controls the flow of data within the CPU and determines the utilization consistency of other components.

The processor's work is associated with constant interaction with the computer memory to receive data and commands and save the results of calculations. The increase in the frequency and overall performance of the CPU has long surpassed those of the system bus and RAM, not to mention the significantly slower energy-independent storage devices such as HDD. For the computer's memory not to slow down the processor, negating the advantages of new devices, the CPU is supplied with the fast memory, which is called the cache (Fig. 3.2 b). The purpose of the cache is to duplicate some of the data from the main memory, which the processor

accesses most often. The cache is divided into levels, which correspond to their characteristics: the size of the available memory and the access time. Directly, the CPU registers make it possible to operate with individual variables (or groups of the same type of variables for the SIMD registers) and have an access time of about a 1 processor cycle. In contrast, for the main RAM, this time is about 200 cycles. For the cache memory, these characteristics change to decrease the operating speed but increase the available size. L1 cache is accessible for 4 cycles and allows to operate with several tens of KB. They are usually divided into the data cache and the instruction cache containing instructions ready to be split into micro-ops for execution on the ALU. L2 cache access time is about ten cycles, but available memory grows to several hundred KB. L1 and L2 caches are unique to each core of modern multi-core CPUs. In contrast, the next L3 cache is common for the entire chip and is intended to store information and fast data exchange between cores. Access times range from 40 to 70 cycles, which is significantly slower than higher-level caches but still faster than accessing RAM.

As mentioned earlier, the same data contained in different caches are always synchronized, and this does not require additional control from the programmer. This cache property is called coherence and is supported using special algorithms. But knowledge of caches is extremely important for a programmer who cares about the performance of their code. It is easy to see that the less the program will access the slow RAM, the faster it will run. Therefore, to ensure high-performance computing, it is strongly recommended to make the most of the data loaded into the cache before processing a new, not yet cached, data array. Proper use of the properties of the cache memory can increase the speed of an application tenfold since it is memory operations that most often turn out to be the bottleneck that slows down computations.

Another critical factor affecting performance for various tasks is the organization of the main memory structure in relation to the processor. Uniform- and Non-Uniform Memory Access (UMA and NUMA) are most common in modern systems. UMA is a system architecture with shared memory for all CPUs. Memory access is provided over a single system bus with equal read and write capabilities for each processor. It is also possible that a switch connects several data buses. Systems based on the UMA architecture are called symmetric multiprocessor (SMP) systems because they provide balanced data access. UMA is suitable for classic user applications. It is used in most PCs but much less often as a server solution.

The NUMA architecture is a distributed memory system, where each processor is associated with its own memory, located closest to it physically and logically. The interaction between the processors is carried out using special memory controllers. Together with its associated memory, each processor is represented as a separate node, which has its own memory controller. The bandwidth of such a system is superior to UMA, but the memory access time becomes dependent on the distance between the memory and the processor. The competent design of parallel code allows extracting significant advantages from the utilization of this architecture. The NUMA is the standard for high-performance server solutions.

### 3.2.2 GPU architecture

Another type of device suitable for high-performance computing is Graphics Processing Units (GPU). As the name suggests, the GPU's first and primary task is video stream processing, 2D and 3D graphics rendering, and other graphics processing tasks. At the same time, the availability of computing power and software developer tools allows using the graphics cards for other kinds of tasks. It is incorrect to compare GPUs to CPUs in terms of clock speed, number of cores, or available memory. The most indicative metric, in this case, is the number of floating point operations per second that a device can perform (FLOPS). According to this indicator, modern GPUs have long surpassed classical processors, and this gap is only growing. Thus, in theory, the use of graphics cards for high-performance computing looks like a logical and promising solution.

A GPU has components similar to a CPU, but its structure is fundamentally different. For example, graphics cards lack several logical blocks, such as a branch predictor, that facilitate the execution of complex algorithms. On the other hand, the GPU has a significantly larger number of ALUs dedicated directly for computations. This is due to the original task of graphics cards, which is to stream large amounts of data according to a preliminary known scenario. Besides the lack of some blocks, GPUs have more primitive instruction sets than CPUs. For this reason, the execution of algorithms with an abundance of branches and complex logic can be difficult or even impossible on graphic cards. Thus, graphics cards cannot fully replace the CPU in computing, despite the significant superiority in the basic data processing speed. Despite this, graphics cards have proven to be extremely useful in high-performance systems, as long as the peculiarities of their architecture are considered.

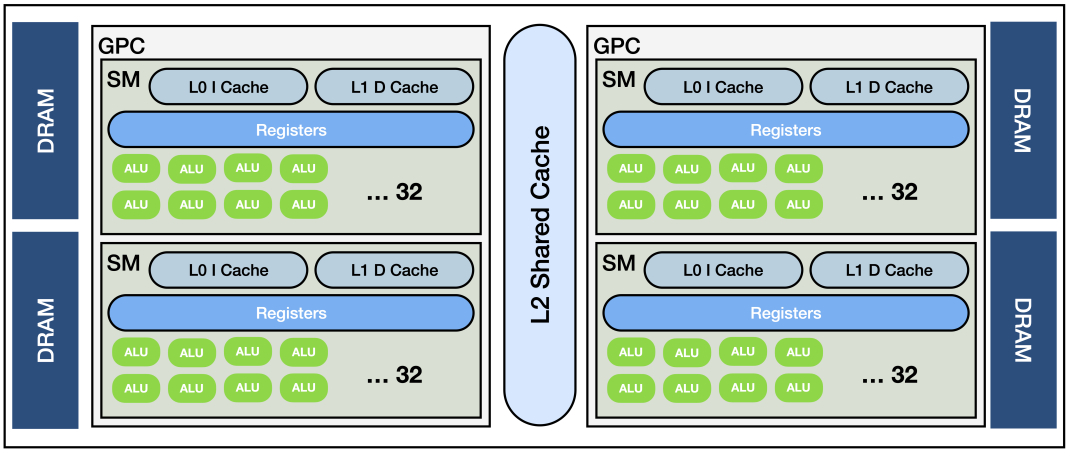


Figure 3.3: Generalized GPU block diagram.

The strength of a GPU is an extremely large number of computational cores, which reaches several thousand for modern systems. This number of nodes fully compensates for the fact that the frequency of graphical cores is 1.5–2 times lower than those of the similar generation CPUs. Graphics cards' ALUs are not completely independent elements. They are combined into blocks of several dozen elements — Streaming multiprocessors (SM). For NVIDIA cards, the block size is traditionally 32 elements. All cores of the block execute the same instruction but with different data, thus realizing the SIMD paradigm from Flynn's Taxonomy. Several SMs form a Graphics Processing Cluster (GPC). Therefore, the application of graphics cards is computing that allows massive parallelization at the data level.

It is important to note that the GPU does not share a memory with the central processing unit. In addition to registers, graphics cards contain significant amounts of RAM, usually faster than the speed of the computer's RAM. But loading and unloading data for this memory are separate procedures and can become a bottleneck that slows down the GPU. Considering these features of graphic cards, working with them has several features that seriously distinguish this task from interacting directly with the CPU at the programmer's side. The central processor acts only as a host that controls the process, performs basic logical operations, and provides the GPU's task setting and data loading. The graphics card itself, designated Compute Device, works as an executor capable of quickly performing a specific set of instructions over a large amount of data. A

Compute Device contains a set of Compute Units, which are collections of grouped threading cores that execute the same instruction on different data items.

All of this makes the GPU a powerful yet limited computing device that can provide significant benefits when used correctly. In addition to simple calculations, graphics cards are capable of executing several more complex algorithms tailored to the specific architecture. For example, the sorting process using SIMD algorithms such as Bitonic-sort [53] or Radix-sort [54] is carried out on graphic cards more than 20 times faster than sorting a similar amount of data using the CPU. At the same time, the weak point of the GPU is algorithms that involve a small number of operations and a large amount of output data. In this case, the overhead of transferring the data in/out of the graphics card can offset the speed gain. Thus, the utilization of graphics cards requires a special approach on the developer's side, specific algorithms, and the selection of suitable tasks. But, if used successfully, graphics cards can provide computing speed gains that are not available to other devices.

### 3.2.3 Neural processor architecture

In recent years, Artificial Neural Networks (ANN) has become increasingly popular in solving a wide range of problems. While the classical algorithm is a well-defined set of conditions and operations unique for a particular task, the neural network is a unified tool for solving various tasks. Based on principles according to which the human brain works, neural networks allow performing even difficult or impossible by algorithmic methods actions. At the same time, many ready-made software solutions hide the technical implementation of the ANN from the user, which makes it possible to successfully utilize this tool even for people who do not have deep knowledge of programming.

The computing logic in ANN differs from classical algorithms. As a result, many traditional computing devices cannot provide maximum efficiency when working with ANN, being originally designed for other kinds of tasks. This problem has not gone unnoticed by large hardware manufacturers, who have offered the user several devices explicitly designed to accelerate the utilization of ANN. Let's consider three main directions in which the modern market of neural network accelerators is developing.

**Machine vision processors**

Image recognition and machine vision tasks usually involve processing a video

stream or images to detect and track specific objects. At the same time, the functions used are fundamentally different from graphics cards, despite the similarity of the input data. Modern computer vision algorithms are based on the Convolutional Neural Networks (CNN) that are used to highlight special points and their descriptors on the image.

Focusing on the use of CNN for video stream processing, it is possible to point out the main characteristics that will make machine vision processors most efficient:

- SIMD computations. Working with convolutions involves applying the same operations to arrays of data. Vector calculations allow to speed up the process as much as possible. Low demands on the precision of floating point calculations make it possible to utilize vector registers maximally efficiently.

- Multithreading. Simultaneous processing of multiple convolutions is the basis for the successful functioning of CNN. Consequently, numerous computing cores are an important feature of the machine vision processor.

- Memory. To store the intermediate results of the CNN calculations, the neuro processor needs a sufficient amount of fast memory — an increased cache size.

- Interfaces. A machine vision neuro processor must have broad capabilities for reading data directly from a source — a video camera. An insufficient number and bandwidth of such interfaces can limit the device's functionality, even with high computing power.

A striking representative of machine vision processors are devices of the Intel Movidus [55] family. The form factor of the neuro processor is a small microchip, which is usually completed in the form of a USB stick. The popular Myriad X model includes 16 SHAVE [56] computational cores with VLIW architecture and 128-bit SIMD registers, 2.5 MB of the fast cache memory, and allows to connect up to 8 RGB cameras with HD resolution. Thus, a small specialized device can process a data stream of about 700 megapixels per second without loading the main CPU.

**Neuromorphic processors**

ANN applications are not limited to machine vision. The advantages of neural networks in problems of classification and decision-making based on learning, rather than rigorous algorithms, inevitably leads to development of specialized computing devices that are not limited to one type of problems.

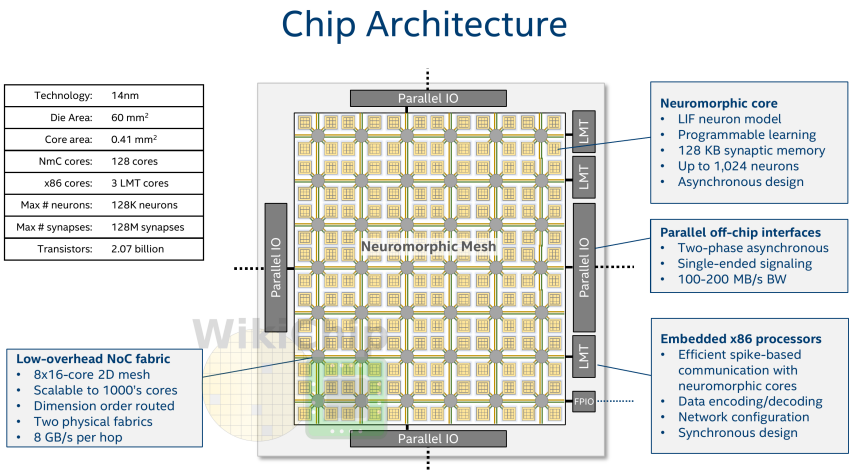


Figure 3.4: Block diagram of the Intel Loihi neuromorphic processor [58].

The utilization of ANN software emulation leads to overhead costs in the transition to calculations with classical computing systems. The solution to this problem was found in the emulation of the neural network logic already at the hardware level. The class of devices implementing this paradigm is called neuromorphic processors. A distinctive feature of such devices is a large number of computing cores. So, the research project of a neuromorphic processor from IBM — TrueNorth [57], developed in 2014, had 4096 cores, completed in the form of a two-dimensional 64×64 array. However, the architecture of the device is not a von Neumann architecture. Each core is an independent computational node and has its own:

- scheduler;
- request control module;
- memory for storage of states;
- router for communication with the neighbors;
- logical implementation of the neurons.

Each core of the TrueNorth allowed 256 neurons to be emulated. Thus, the total capabilities of the device made it possible to emulate more than a 1 million neurons and 256 million synapses.

Another well-known neuromorphic chip is Loihi [58] from Intel (Fig. 3.4). The chip has more modest characteristics relative to the IBM product, but it can be scaled to meet the required system power. The device, released in 2017, has 128 cores, which powers 128 thousand neurons and 128 million synapses. Additionally, several x86 cores are built into the chip, performing general system control. In the second half of 2021, Intel introduced the second generation of this neuro processor, which allows emulating up to 1 million neurons.

Neuromorphic processors are seriously different from classical solutions and are still at the research stage. But the available results carried out in the framework of open research programs of Intel and other companies demonstrate the superiority of such neuromorphic systems in solving classification problems both in response time and in energy efficiency [59].

**Tensor processors**

The utilization of deep learning neural networks is associated with a large number of calculations of the states of neurons depending on the states and weights of the neurons from the previous level and the recalculation of the weights depending on the gradients obtained from the next level. Without going into the details of the mathematical mechanism of ANN, we note that it is convenient to describe changes in the state of objects (neurons in this case) using a tensor — a linear multicomponent algebraic object defined on a vector space. In reality, this means that using neural networks on processors with classical architecture requires numerous calculations of vectors and matrices.

SIMD architectures themselves are excellent accelerators for vector computing. Specialized CPU registers and GPU streaming multiprocessors can easily deliver massive performance gains. But, moving on to operations with matrices in the deep learning framework, we are faced with the need to save intermediate calculations even during such simple operations as matrix multiplication. As the amount of computation increases, the overhead also grows up. In order to achieve the maximum speed of performing standard operations with matrices, it was decided to use cores, the architecture of which is explicitly designed for matrix calculations.

In addition to ANN, fast matrix handling is essential when working with video and graphics. Therefore, it is not surprising that NVIDIA has become a pio-

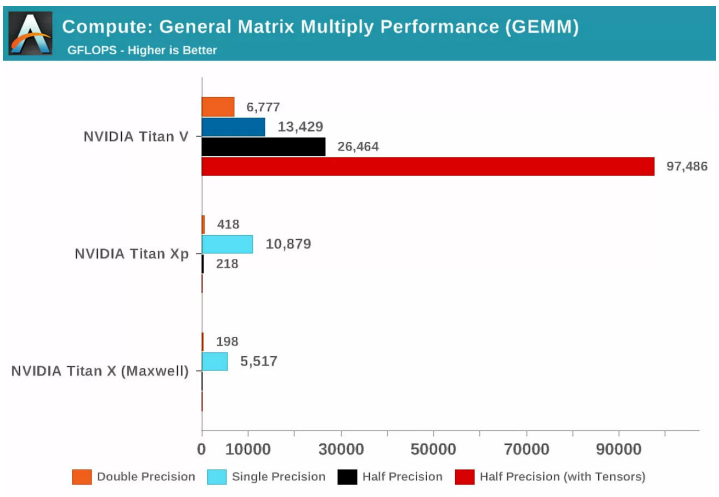


Figure 3.5: General Matrix Multiplicity Performance: NVIDIA Titan V (with tensor cores) vs NVIDIA Titan Xp vs NVIDIA Titan X.

neer in developing tensor calculations. In late 2017, the NVIDIA Titan V line of graphics cards based on the Volta [60] architecture was released, with a set of cores that are intended only for matrix computing. When the Volta came out, Anandtech web site [61] ran math tests on three NVIDIA cards: the new Volta, the most powerful of the Pascal lineup, and the older Maxwell card. The results of the research shown in Figure 3.5 demonstrate a significant performance gain for matrices containing half-precision floating-point elements when using tensor cores. In the next generations of graphics cards, NVIDIA has integrated the well-proven Tensor Cores into the specialized Titan series and the more affordable GeForce gaming solutions. Today, the flagship computing processor from NVIDIA is the A100 Tensor Core [62], based on the Amper [63] architecture. Thus, NVIDIA completely occupies the niche of tensor computing from gaming solutions with GeForce series GPUs to large computing nodes for AI calculations with ready-made computing systems GDX A100 [64], that include up to 8 A100 based cards and 640 GB of video memory.

NVIDIA's main competitor in the graphics accelerator market, AMD, began the transition to tensor cores in the second half of 2020 with the introduction of the HPC-oriented architecture CDNA [65]. CDNA assumes using its own implementation of tensor cores, called the Matrix Core Engine. Along with the new architecture, AMD announced the new generation AMD Instinct MI100 [66] graphics accelerator, which should demonstrate the advantages of CDNA.

Table 3.1: Major neuroprocessors and AI accelerators developed by various companies.

| | |
|---|---|
| Machine vision (VPU) | **Intel**: Movidus Miriad 2/X [55, 56];<br>**Mobileye**: EyeQ [72];<br>**Microsoft**: HoloLens Holographic processing unit (HPU) [73]; |
| Neuromorphic | **IBM**: TrueNorth [57];<br>**Intel**: Loihi [58];<br>**University of Manchester**: SpiNNaker 1/2 [74];<br>**Heidelberg University**: HICANN [75];<br>**Stanford University**: Neurogrid [76]; |
| Tensor (NPU) | **Intel**: Nervana Network Processor (NNP-T/I) (closed) [68], Habana Goya [70], Habana Gaudi [71];<br>**NVIDIA**: Volta V100 [60], Ampere A100 [62];<br>**AMD**: CDNA MI100 [65];<br>**Google**: Tensor Processing Unit (TPU) [67];<br>**Amazon/AWS**: Inferentia [77];<br>**ARM**: Machine Learning Processor (MLP) [78];<br>**Alibaba/T-Head**: Hanguang 800 [79];<br>**Qualcomm**: AI Engine [81];<br>**Apple**: Neural Engine (AI-coprocessior for A11-A14 Bionic and M1 ARM processors) [80];<br>**Samsung**: Neural Processing Unit (NPU) (AI-coprocessor for mobile devices) [82];<br>**Huawei**: Ascend [83];<br>**Tesla**: Full Self-Driving Chip (FSD Chip, Autopilot Hardware 3.0) [84];<br>And many others... |

In addition to graphics cards, tensor cores and coprocessors are being developed for mobile devices. Deep learning technologies allow modern smartphones to recognize speech and predict user actions, and device cameras can quickly process images, increasing detail and equalizing color balance. Such microchip developers as Qualcomm, Huawei, Apple have achieved the greatest successes to date. Since 2016, Google has offered its own version of a separate tensor processor — Google TPU [67], which is designed to accelerate calculations of 8-bit numbers and is capable of providing up to 92 trillion operations per second.

Intel also introduced the tensor processor in 2018. Intel NNP-T (Nervana Neural Network Processor for Training) [68] is a large machine learning processor with 24 1.1 GHz Tensor Cores, 60 MB of distributed memory, and 32 GB of flash memory. Intel subsequently closed the project, focusing on funding Israeli startup Habana. [69]. Proprietary chips Habana Goya [70] (2018) and Gaudi [71] (2019) have shown significant superiority over the Nervana architecture.

The widespread use of Artificial Neural Networks inevitably entails the development of hardware that should speed up and simplify the operation of systems with deep learning. Many hardware solutions are still in the testing and research phase. But the general vector of development of the NN systems suggests that the use of the AI accelerators will become, if not ubiquitous, then quite frequent in the coming years. Table 3.1 lists the most notable modern mass and research AI processors.

## 3.3 Parallel programming toolkit

Effective use of the advantages of modern hardware is practically impossible without the right tools. To date, a large number of specialized packages, APIs, and software solutions have been developed to facilitate parallelization processes both at the data level and at the thread level. A detailed review of existing solutions within the framework of this work is inappropriate due to the huge amount of information. Instead, we will consider several significant software tools often used in HEP to parallelize computations and are suitable for working with programs written in C++.

### 3.3.1 Vectorization tools

Parallelization of computations at the data level is carried out by using vector registers and special instructions. Intel's set of SIMD instructions is currently standardized and is also used by their competitors, AMD, in their solutions. This provides good code portability as long as the CPUs support the same SIMD version. On the other hand, rewriting the vector code to instruct other versions or sets can be challenging.

The main problem, in this case, is the rather complex command format, which makes even logically primitive constructions difficult to read for the developer. So, instead of the usual operations such as addition and multiplication, SIMD instructions assume the use of functions of such format as __mm_add_ps(a, b) and __mm_mul_ps(a, b). When forming mathematical constructions, functions have to be nested into another, seriously cluttering the code. Data type names also differ from the usual ones: __m128 (float vector), __m128i (integer vector), __m128d (double vector).

In order to increase the convenience of operating with SIMD code, additional tools are often used. Let's consider some of them that have found application in scientific high-performance computing.

**Header files**

The code readability problem can be solved by simple operator overloading. At the same time, the built-in functionality and the speed of program execution are not subject to change since the actual assembler code generated by the compiler will not differ. Thus, for SIMDization of the program, we need to create special header files containing overriding data types, overloading the required operators, and declaring important constants, such as the number of elements in a vector.

At its core, such a header file allows the vector data type to mimic the base types, hiding the component from the SIMD developer. It is important to remember to align data in memory and correctly fill SIMD vectors when using header files. Otherwise, the code becomes almost indistinguishable from the usual, significantly simplifying the development process.

Another major benefit of this approach is its flexibility. Header files can be easily replaced if needed to switch to a CPU with a different architecture or use a different version of SIMD intrinsics. Using templates in the main program code will automate this process and make it possible to switch between different implementations, including the scalar one, quickly.

**Vector classes (Vc) library**

Vc library [85] is an implementation of the idea of header files, taken to a qualitatively new level. At the same time, the idea of interchangeability of vector data types is built directly into Vc, which allows writing code for all supported versions of SIMD instructions: SSE, AVX, IMIC. The library introduces standardized data types denoting vector variables by adding the "_v" postfix after the type name (int_v, float_v, etc.).

The goal of developing Vc was not only to create another version of the header files but also to achieve maximum functionality, the use of which does not require additional labor from the programmer. Thus, horizontal operations within one vector and several simple vector algorithms that short-circuit the code development were introduced.

Among the important innovations, we should note the convenient masking, which is a vector analog of the conditional operator. For each vector data type, there is an additional type with a "_m" postfix instead of "_v", called a mask. The mask is a vector of elements of a type equivalent to bool and can be used to perform logical operations in vector form without switching to scalar calculations.

The Vc library is compiled as a separate project and has specific software requirements. Therefore, header files are still more flexible to use. On the other hand, there is more functionality on the Vc side, which justifies the difficulties associated with it for some projects.

### 3.3.2 Parallelization tools

Computing parallelism is the most popular method for increasing the speed of the software by utilizing multiple processing cores on the CPU. Today there are many tools for parallelizing computations in both manual and automatic modes. Let's take a closer look at some of them.

**Open Multi-Processing**

OpenMP [86] is an API (Application Programming Interface) that successfully combines a wide range of parallel computing capabilities with a simple and intuitive syntax. It allows you to parallelize code written in C, C++, or Fortran by embedding its functions into a finished program. At the same time, the original code usually does not require significant changes. This makes OpenMP powerful and extremely flexible as a tool for creating multi-threaded applications.

The OpenMP programming model is based on fork-join parallelism (Fig. 3.6).

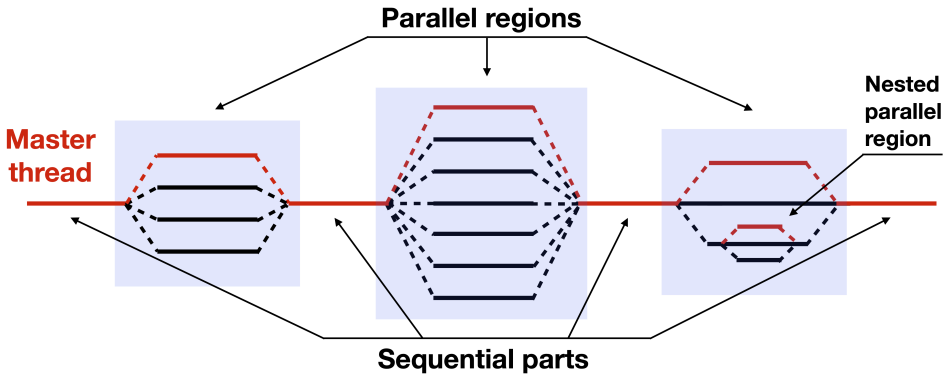


Figure 3.6: An illustration of the fork-join concurrency model underlying OpenMP.

This approach assumes running the program in single-threaded mode. The thread is called master and follows the overall flow of the program. Where the algorithm can run in parallel (parallel region), the master thread is split into multiple parallel threads. After completing the necessary calculations, the threads are combined into one again. This variant of parallelism is especially convenient since completely parallel programs practically do not occur in reality. In addition, OpenMP supports nested parallelism, where one of the parallel streams is re-split into several more. The developer has the necessary tools to control the number of concurrently existing threads, exceeding the maximum available number of cores, executing partially sequentially on the available hardware.

Most modern operating systems and compilers support OpenMP without installing additional libraries or software. API can be utilized by simply including its header file — omp.h. The creation and management of streams are carried out using preprocessor directives, which have a simple and understandable form:

```
#pragma omp <construct> [clause [clause] ...]
```

OpenMP can create parallel regions or set them with several features and settings depending on the commands used. Thus, variables of the same name in parallel threads can be private or public, allowing for interactions between processes. There is a high likelihood of race conditions occurring when using shared memory. OpenMP provides significant control over thread synchronization. This control can be carried out both at a high level using directives that restrict certain actions (omp critical, omp atomic) or removing restrictions (omp nowait)

and at a low level using locks and semaphores.

In addition, OpenMP has several built-in automation algorithms that can make a developer's work much easier. The most popular is the automatic parallelization of for loops, triggered by the **omp parallel for** construct.

**POSIX threads library**

While OpenMP is an extremely powerful parallelization tool, its capabilities are not always sufficient for precise flow control. When performing calculations on servers with NUMA architecture, the distribution of threads across cores can be important due to unequal memory access times. Changing a thread from one processor to another at runtime can result in significant overhead because of the need to read from memory belonging to another processor.

The solution to the problem of incorrect thread behavior can be the use of the Pthreads library [87], which belongs to the thread-oriented part of the Portable Operating System Interface (POSIX) standards. Pthreads provides access to a low-level API for manually creating and manipulating threads. Using the Pthreads toolkit with other parallel computing APIs can maximize computational efficiency.

# Chapter 4

# Reconstruction of the particle trajectories in the HEP experiments

## 4.1 Event reconstruction

Modern HEP experiments are becoming more complex and expensive technical projects. New research aimed at studying the state of matter under more difficult attainable conditions or extremely rare interactions requires the construction of huge accelerator complexes and ultra-precise fast detection systems. But ensuring the collision of particles under the given conditions, as well as the ability to trace the result, is not sufficient to carry out the experiment and obtain the expected results. Collection, processing, and interpretation of results is a critical research stage.

The results of the first experiments in particle physics could be processed directly by humans due to the relatively small amount of incoming data. Over time, the accumulation of physical knowledge and construction of the new theoretical models increase the complexity and rarity of the studied processes and objects' manifestation. On the other hand, technological progress provides ever-increasing opportunities for intensifying experiments. So, the tasks of the CBM and STAR experiments include the search for particles, the creation probability of which is estimated at 1 in several million collisions. Considering the fact that the total number of particles formed as a result of each collision can reach 1000 (CBM) or even exceed 2000 (STAR), handling each event becomes a non-trivial process (Fig. 4.1). Thus, the reconstruction and analysis of the results of particle collisions must occur not only extremely quickly but also have a maximally high

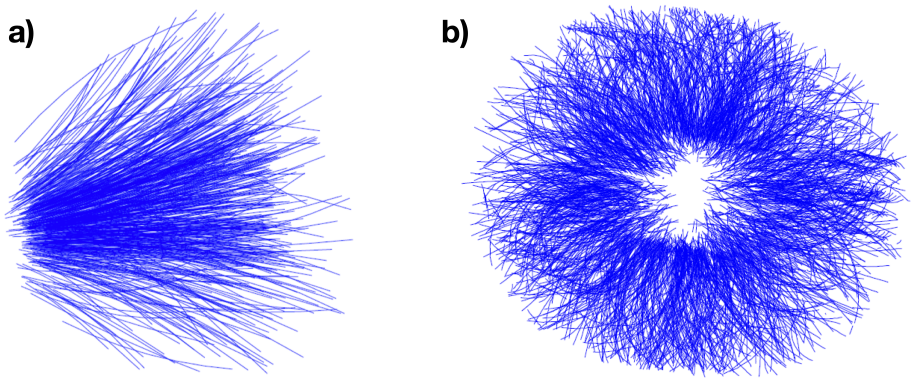


Figure 4.1: Examples of reconstructed tracks: a) CBM STS Au+Au event with 587 tracks; b) STAR TPC Au+Au event with 1446 tracks.

accuracy since the loss of rare particles is unacceptable.

Event reconstruction itself is not a single process. At a basic generalized level, it is possible to distinguish several important steps, divided into three logical groups:

- data preprocessing (digitization, search for clusters, search for hits);
- reconstruction of charged particle trajectories (track search, track fitting);
- physical analysis (track fitting, particle identification, restoration of the decayed short-lived particles);

Modern detectors use different physical principles to register charged particles crossing active elements, planes, or volumes of detecting systems. Detector responses can be presented in various formats, depending on the type and characteristics of the device. But, more often, the 'raw' data is not suitable for utilization directly in tracking algorithms. The task of data preprocessing is to transform arbitrary responses of the detectors into a set of spatial points — hits that most closely correspond to the particle trajectories. Digitization is the process of obtaining digital equivalents for analog detector responses. Usually, it is part of the information reading process and is carried out by the detector's electronics. As a result, the 'imprints' of the particles are presented as groups of activated detector elements, where its intensity corresponds to the energy imparted by the particle, i.e., the proximity of the trajectory. Imprints may vary in

shape and size and maybe split or partially overlapped. The task of clustering is to find and separate individual particles' imprints — clusters. Finally, the search for hits determines each cluster's geometric or weighted center. The search for clusters and hits can be embedded in the detector's hardware or carried out using independent algorithms. The data preprocessing results — a set of hits and the necessary additional information — are used as input data for track finding algorithms and further analysis.

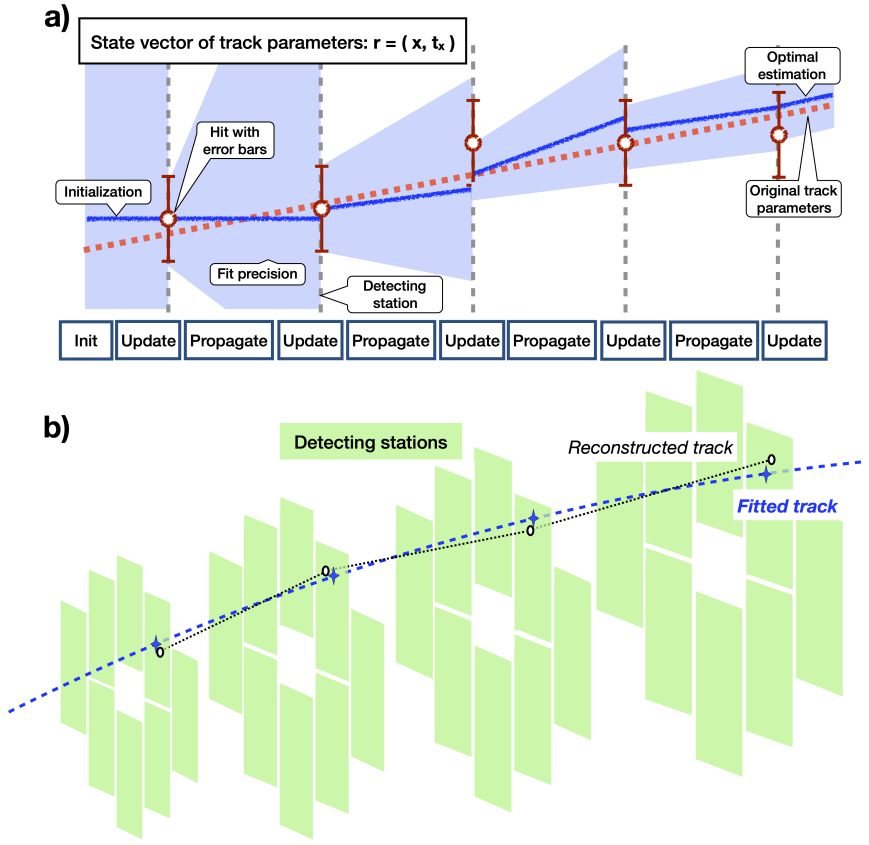


Figure 4.2: a) An example of fitting a straight track in 2D coordinates. b) An example of fitting a curved track in 3D coordinates.

Reconstruction of charged particle trajectories (track reconstruction, tracking) is the process of combining hits produced by particles into ordered groups that illustrate the movement of each of the particles inside the detector. The high multiplicity and curvature of tracks, which arises from the deflection of charged particles in a magnetic field, make this step of the event reconstruction the most challenging and time-consuming. The presence of noise in the measurements

creates additional problems.

The reconstructed track is not a smooth straight, or rounded line. The reason for this is the imperfection of the detecting equipment and the inaccuracies of the clustering algorithms. As a result, the hits turn out to be displaced relative to the actual trajectory with some error, setting the direction but not having a complete match. But the physical analysis does not require a geometric representation of the trajectory in the form of a broken line or a set of points, but the parameters of the track, which make it possible to conclude the most important characteristics of the particle.

The process of parameterizing the particle's trajectory over a set of discrete measurements is called fitting. Depending on the specific task and the track model used, various algorithms can carry fitting from the simplest (approximation by a line, parabola, circle) to complex multistage algorithms, such as the Kalman Filter (KF). At present, the KF is the most popular and frequently used algorithm for estimating track parameters in High Energy Physics. This recursive algorithm makes it possible to efficiently calculate the track state vector even in the case of a small number of erroneously attached hits, as well as to make a decision about the correctness of finding the tracks as a whole (Fig. 4.2). The composition of the vector of track parameters does not have a common standard. It is set by the track model, which, in turn, is selected based on the geometry of the detector, magnetic field, features of the tracking algorithm, or other conditions. In addition to evaluating and approximating the resulting tracks, the fitter often becomes part of the track search algorithms or even serves as the basis for them.

Successful implementation of the previous steps allows to reconstruct the picture of the collision but does not draw clear conclusions about the physical component of the process. In order to correlate the obtained results and theoretical calculations, it is necessary to determine and analyze the list of names and the number of registered particles. Special algorithms or even specialized software packages, such as KFParticle Finder [**?**], make it possible not only to recognize the found particles using the parameters of their trajectories but also to restore decayed short-lived fragments from their decay products. The exact calculation of the state vector of each track is extremely important for physical analysis. Therefore, before the stage of particle recognition, the tracks can be additionally re-fitted.

With the interaction speed of up to several million collisions per second, the opportunity to filter out unimportant information becomes critical due to the

inability to transmit, process, and save the generated data stream in non-volatile storage. It is easy to see that sequential data transformation significantly reduces its volume at each subsequent reconstruction step. In order to achieve an acceptable load on the equipment without the loss of important information, data collection in HEP experiments is controlled using a set of triggers of various levels, from low-level triggers built into the detection systems to high-level triggers based on the inspection of partial or complete reconstruction results and physical analysis of events.

The use of high-level triggers is a highly productive and promising solution that allows bringing the selection of experimental data to a level of efficiency otherwise unavailable. It can be attained due to the ability to evaluate the usefulness of the events in real-time. The main problem in implementing such complex selection mechanisms is the speed of the event reconstruction algorithms and, in particular, the search for tracks. A successful solution to this problem is possible only using very fast and efficient algorithms developed and implemented following modern principles of high-performance computing. It is such a trigger that is planned to be used in the CBM experiment within the FLES complex, which will make it possible to operate with the declared interaction rate up to 10 MHz. A striking proof of the possibility of using this approach is the High-Level Trigger of the STAR experiment, which has been successfully operating in the running experiment for more than five years.

## 4.2 Track reconstruction

Reconstruction of charged particle trajectories based on the responses of track detectors is an algorithmically complex and time-consuming task. Even in the simplest cases, when a human eye can easily distinguish separate chains of hits, the automation of the process using a computer turns out to be non-trivial. Equipment imperfection increases the complexion of the task. This leads to both the loss of some hits and the occurrence of the noise during measurements. At the same time, it is the search for tracks that is usually the slowest part of the event reconstruction. Increasing the speed of this step has the most significant effect on the overall acceleration of computations.

Most of the registered particles are born directly at the collision point, which is usually called a primary vertex. Therefore, using this point as the starting

measurement is a typical tracking solution. On the other hand, the search for short-lived tracks is often possible only by reconstructing their decay products outside the primary vertex. Thus, depending on the features of the experiment, minimizing the dependence of the track reconstruction process on the primary vertex can be important when choosing a tracking algorithm.

Modern methods of tracks searching are divided into two categories depending on the approach to reconstruction: global and local.

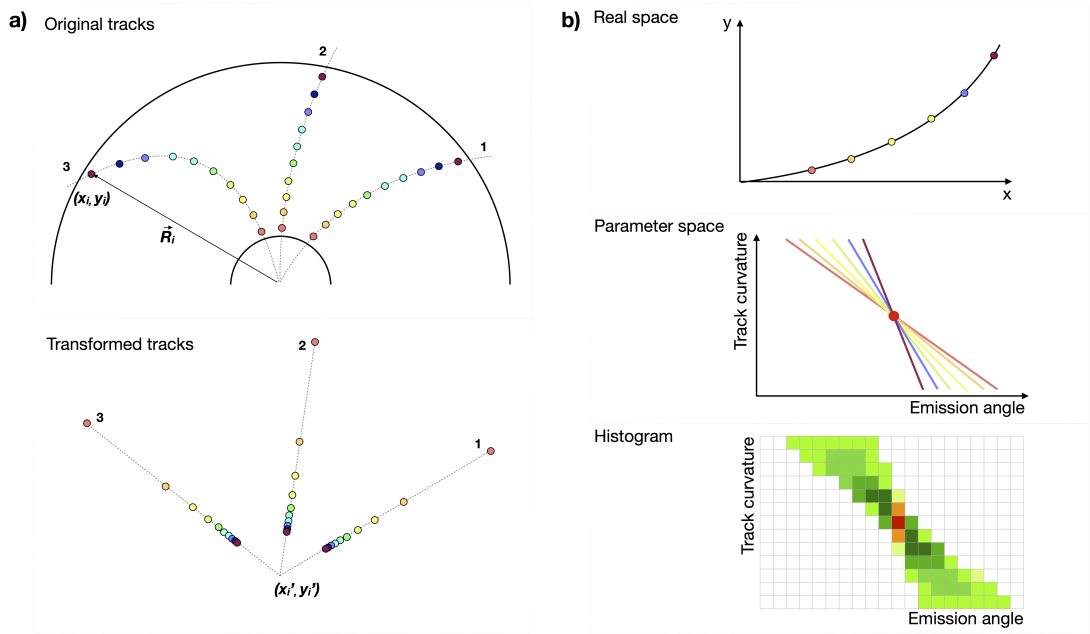


Figure 4.3: Global track reconstruction algorithms: a) Conformal mapping; b) Hough transform.

Global methods assume sequential processing of all measurements within the events to group them based on a given track model. The operation of such algorithms is usually clearly divided into two steps: data transformation and classification of results. This approach is characterized by mathematical simplicity, a strictly ordered sequence of actions, and minimal combinatorics. But this simplicity leads to the appearance of several disadvantages that limit the acceptability of global tracking methods.

A striking example of global reconstruction is conformal mapping [88], considered earlier in the framework of STAR High-Level Trigger. The method allows transforming the curved particle trajectories into straight lines with simple math-

ematical operations, taking into account the availability of information about the vertex (Fig. 4.3 a). In turn, tracking straight-line tracks in a modified coordinate system becomes a trivial task. An obvious disadvantage of this approach is the impossibility of finding secondary tracks that do not cross the primary vertex.

Hough transform [89] is another global method that has found its application in HEP. The coordinates of hits are transferred from the space of real coordinates $(x, y)$ to the space of parameters $(a, b)$, which is possible only if there is a sufficiently accurate track model. As parameters, it is convenient to consider the curvature of the trajectory and the emission angle, taking into account the specifics of the experiments. Then, a line is drawn in the parameter space corresponding to each dimension. The intersection of several lines indicates that the hits belong to the same track (Fig. 4.3 b). The process of finding intersections and distributing hits into tracks is carried out using a simple histogram. Each line contributes to the cells of the histogram that it crosses. In addition to the low flexibility of the algorithm due to the rigidly fixed track model, the Hough transform has other significant disadvantages. The histogramming process becomes slow and memory expensive with the high multiplicity of particles, typical of modern experiments.

Local methods of track reconstruction involve deciding to join a track separately for each considered hit. This approach makes it possible to produce the tracks reconstruction a much more flexible process, reduce the dependence on the primary vertex and the track model, more accurately take into account the physical processes in the detecting chamber — multiple scattering on the material and the effect of a magnetic field, and also control the errors of the track parameters at all stages of the reconstruction. The price for this is the increasing algorithmic complexity, combinatorics, and lower stability of the algorithms.

The most known representative of this class of tracking algorithms is a track following [90]. It is based on predicting the position of the next hit in a track based on the previous hit's position. The track following is based on principles similar to the process of tracks simulating with software packages such as GEANT [91]. Monte Carlo simulation [92] plays a significant role in the preparation of any HEP experiment. The processes of collision, birth, scattering of particles, and their interaction with detectors are emulated as accurately as possible, taking physical models and interactions into account. This allows to programmatically obtain results close to a real experiment and use it to debug and optimize both software and hardware. Using a full-fledged 'pulling' of tracks through the detec-

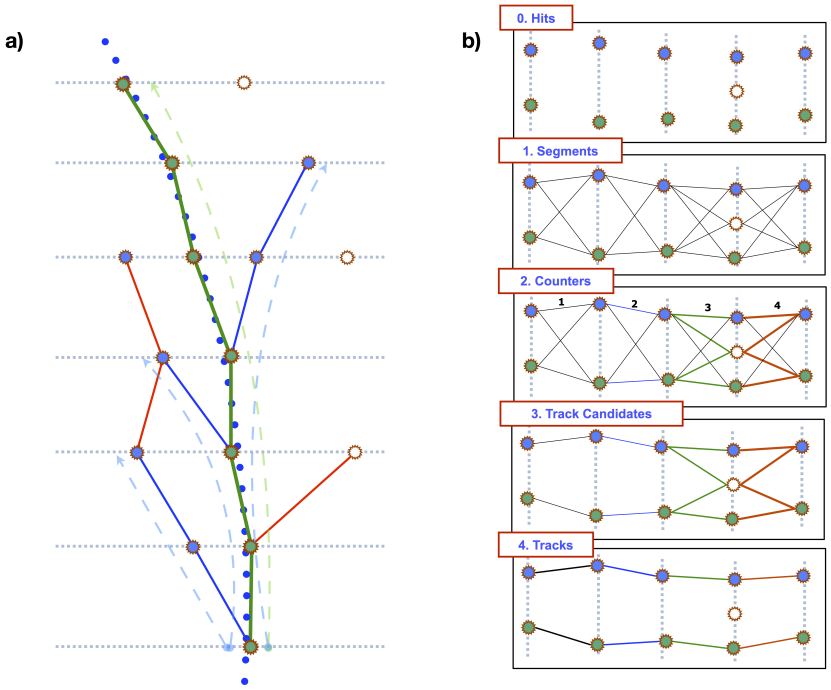


Figure 4.4: Local track reconstruction algorithms: a) Track following; b) Cellular Automaton.

tor is impossible within the framework of an actual reconstruction. Besides the fact that this is an extremely time-consuming process, some of the most crucial particle characteristics are simply not known in real conditions. But a significant simplification of particle trajectory calculation can give such extrapolation the necessary speed and flexibility.

Tracks finding using the track following method usually consists of two stages. The first step is to find short segments of several hits long — seeds that can form the basis of tracks. Then, the algorithm extrapolates each of these seeds to the next station and looks for hits in some small area. If a suitable hit is found, it joins the track, and the algorithm moves to the next station. Multiple measurements in a region of interest can lead to forks, followed by the selection of the best chain (Fig. 4.4 a).

The track following acquires the greatest efficiency when using the Kalman filter, which provides good prediction accuracy and extrapolation errors that can be utilized to determine the region of interest.

The weakness of this method lies in the significant drop in operating speed

with increasing particle multiplicity. The high density of hits leads to frequent branching and growing combinatorial computations during their processing. The software implementation of the algorithm requires random access to the memory, not allowing efficient utilization of the computational resources of the CPU or GPU. In addition, this algorithm is difficult to parallelize both at the data level and the thread level since this can lead to collisions when branching nearest tracks.

Track finding based on the Cellular Automaton represents a different approach to the principle of locality. In general, the Cellular Automaton [93] is a discrete model that includes a regular lattice of cells, each of which can be in one of a finite set of states, and a set of evolution rules, according to which cells can change their states depending from neighboring cells in a certain neighborhood. The most famous example of the Cellular Automaton is the CA-based model proposed by John Conway — Game of Life [93]. An essential feature of CA is the evolutionary transformation of the global system based on local changes in individual nodes. The principle of locality makes it possible to suppress the combinatorial level of the sequential algorithms. Also, it is ideal for parallelizing computations. The concept of cells can be interpreted differently, making the method applicable to a wide range of problems.

Considering the issue of track reconstruction in HEP experiments, it should be noted that the implementation of the Cellular Automaton can differ significantly depending on the features of the detecting setup and the experimental conditions. This section considers a generalized approach to searching for tracks using a spacecraft for detectors with a clear division into stations and pronounced point hits.

Since the main 'building element' of CA is the cell, and the ultimate goal of the work is to build correct tracks, each cell should carry enough information about the track to which it should be assigned to evolve. A single hit without additional binding does not allow to obtain the minimum sufficient information. A couple of hits at neighboring stations give the direction of movement and allow to initialize some trajectory parameters, depending on the used track model. A bunch of two measurements that form the CA cell will be denoted as a doublet. A combination of three hits — a triplet — allows defining the curvature of the segment and expands the set of track parameters available for consideration. Depending on the task, increasing the number of hits in the segment allows determining additional track characteristics and improves the accuracy of the

existing ones. But, there is no need to consider longer segments as the CA cells for the general case.

Returning to the concept of the Game of Life, it is easy to see that the entire evolutionary process is built around a change in the state of cells from 'living' to 'dead' and vice versa. The state of the surrounding cells is used as a condition for changes. In our case, a cell segment's state is determined by the presence or absence of a connection between the hits belonging to it. The conditions for evolution are complemented by an analysis of the track parameters corresponding to each segment. The initial number of cells in the algorithm is determined by the number of possible combinations of hits that form segments. At the same time, the operation of the algorithm is unidirectional. That is, the segments that meet the conditions 'die off' without the possibility of 'reviving' later. This saves us from cyclical and endless transformations that are characteristic of the Game of Life and allows us to solve the problem in a finite time frame.

Let us consider in more detail the generalized algorithm of the spacecraft-based track search, illustrated in Figure 4.4 b. A pair of hits located at the neighboring stations of the detector was selected as a cell of the algorithm without taking into account the hit's position inside the station. Vertical dashed lines indicate stations. Hits are represented by blue and green dots, depending on the track they belong to, as well as a white color denotes the noise hits.

At the first step of the algorithm, the initial conditions for evolution are set. All algorithm cells are marked as 'alive' by connecting pairs of hits 'all to all'. In real situations, the high multiplicity of tracks and constructed segments is compensated by restrictions that cut off the possibility of forming obviously impossible pairs of hits. The data preparation is considered complete, and the algorithm proceeds directly to evolution.

The second step of the algorithm is to eliminate non-viable cells by breaking the links between hits. The main conditions for saving the segment are a neighbor with a common hit and a similar direction of these segments within a certain error determined by the tracking settings. Additional specific options may be imposed depending on the features of the track models and detector geometry. Destroyed segments are lost irretrievably. At each stage, the algorithm works only with the existing state of the system. In addition, at this stage, the track finder evaluates the resulting chains of segments. It assigns additional information about the position and neighbors to each of them, optimizing further calculations.

Further, successively connected segments are assembled into full-fledged track

candidates using the previously obtained information characterizing the position of each segment. If no additional limitation was imposed on the number of links for one hit during the destruction of doublets, the resulting track candidates are grouped into bundles with diverging branches.

Completing the track search procedure selects the best track candidate in each individual bunch. The criteria are usually the longest chain length, the smallest standard deviation error $\chi^2$, and the prohibition of the simultaneous use of the same hits by different tracks.

The presented model of the algorithm has several significant advantages that make it optimal for solving the problem of the track searching:

- Scalability — the change in the configuration of the detector setup can be carried out automatically without significant interference in the program structure due to the uniformity of the approach.

- Flexibility — the search for particles with trajectory features (primary and secondary, high-momentum and low-momentum) can be carried out by a single algorithm with some differences in the evolution criteria.

- Locality — at each stage, the algorithm works with geometrically or logically closely spaced measurements, which allows building compact and fast data structures for software implementation.

- Consistency — the algorithm sequentially converts data from massive and scattered to more consolidated and informative: hits — segments — chains — track candidates — tracks. At the same time, storage of intermediate information is generally not required, the number of read/write operations is minimized.

- Independence — a direct result of locality, the algorithm is easily split into independent threads, contributing to vectorization and parallelization of computations.

Thus, track finding based on the Cellular Automaton is a convenient and highly efficient method for reconstructing charged particle trajectories, which is excellent for executing on the modern high-performance computing systems. Currently, the method is used in many existing and upcoming experiments of the HEP, being the most popular of the presented ones.

Achieving the best result in the reconstruction of particle trajectories is not possible without considering the features of the detecting system. Different variants of the input data require a special approach when processing them. Among the most frequently used tracking detectors in HEP experiments, the two most significant classes can be distinguished, which have their own characteristics that must be taken into account when reconstructing tracks.

Detectors with clearly defined stations, usually pixel or strip, are characterized by large distances between stations and high measurement accuracy. The high cost of creating and maintaining each station leads to the fact that the number of measurements in the track is small and the correct connection of each of the hits becomes a crucial task. In such cases, it is important to focus on the quality of the track's micro-segments — doublets and triplets — even if it would take additional time.

Another option is gas detectors, similar to the previously considered TPC of the STAR experiment. The peculiarities of tracking the movement of elementary particles in the electro-magnetic field lead to a decrease in the accuracy of hit recovery in comparison, for example, with pixel detectors, but allow to place significantly more pad rows, which are a logical analog of stations. Thus, more measurements per track reduce the value of each one. At the same time, the smaller distance between adjacent measurements makes the process of merging them easier.

As an example of the reconstruction of tracks based on the Cellular Automaton in detectors of each of the considered types, let us dwell in more detail on two existing versions of track finders: CBM CA Track Finder — for reconstruction in pixel and strip detectors of the CBM experiment and STAR TPC CA Track Finder — for the TPC detector of the STAR experiment.

## 4.3 CA Tack Finder in the CBM experiment

The experimental setup of the CBM experiment is a complex of different detectors, each of which has its own structural features and functions. The core of this system is the STS detector, which should provide stable reconstruction and determination of the most important characteristics of particles produced in collisions. In conditions of a high multiplicity of tracks per event and the need for the fastest possible data processing, the tracking algorithm based on CA is

the best possible solution. Consider the features of the implementation of the algorithm, taking into account the specifics of the problem.

CBM is a fixed target experiment. Since the high-momentum projectile collides with a stationary target, the particles generated during the collision continue to move in the same direction while simultaneously creating an expanding spherical front in the moving coordinate system. Thus, in a stationary coordinate system associated with the detector, the flying fragments form a cone and overcome a small sector of the sphere with a solid angle of about 30°practically in full force. This results in an extremely high particle density and an increased likelihood of false alarms for strip sensors, which complicates the task of tracking and determining the correctness of the found tracks.

The optimal solution for achieving high accuracy of calculation and control of track parameters was found in using the Kalman filter to fit individual segments and tracks as a whole. The track state vector, which KF operates, also acts as the key parameter of the cell in the CA algorithm that is used in the evolution process. The following basic components can fully describe the position and characteristics of the track in the magnetic field: coordinates of the intersection of the track and the detector station $(x, y)$ in the Z plane, track slope with respect to the OZ axis $(t_x, t_y)$, charge $(q)$ and momentum $(p)$ of the particle. Considering that the charge can only take on the values $\pm 1$, the track state vector includes 5 components and looks like this $(x_i, y_i, t_{xi}, t_{yi}, q/p)$, where $i$ is the index of the detector station for which the calculations were performed.

A single hit sets the coordinates and allows determining the first two parameters of the state vector. Two hits determine the direction and, therefore, the slope. In order to obtain the curvature of the track, which makes it possible to estimate the momentum knowing the magnetic field, at least three measurements are required. Thus, a reasonable choice for the role of a cell of the CA algorithm, in this case, is a triplet — a combination of three consecutive hits.

Using all possible combinations of hits as a basis for triplets looks impractical and will lead to an unacceptable decrease in the speed of the algorithm. The formation of triplets is divided into three stages, which make it possible not only to qualitatively filter out obviously incorrect objects, but also to prepare a vector of parameters for further use.

The first stage is initializing the initial parameters of a possible future triplet. The algorithm sequentially considers hits at all stations, except for the last two, creating for each of them a vector of track parameters and setting its initial

values. A pair of a hit and a vector of parameters was named a singlet by analogy with larger similar formations. The experimental conditions, confirmed by Monte Carlo simulations, assume that most of the particles originate from the region of the primary vertex and have a momentum of more than 0.1 GeV. This makes it possible to use the primary vertex as an additional dimension when forming a singlet, specifying the starting direction of the search without joining a second hit.

The singlet, equipped with a state vector and a covariance matrix, is extrapolated towards the next station using simplified KF functionality, considering the track model and physical interactions. Thus, we get the approximate position of the next hit and the errors within which the acceptable hit should be located. The larger the errors used to initialize the vertex, the wider the size of the window of interest. By varying this indicator, we get the opportunity to provide a more flexible and less time-consuming process of searching for tracks. The algorithm extends segments by attaching hits to singlets using the KF and filters out combinations that do not meet the $\chi^2$ proximity criterion. The singlet array is transformed into a new group of two-hit objects — doublets. Since no fundamentally new objects are created, only connections are determined, and parameters are updated, the number of memory accesses is minimized. The computational speed remains high despite the amount of work.

In the same way, doublets are extended to form triplets. The presence of a larger number of measurements reduces the area of interest when catching a hit and makes filtering by $\chi^2$ more efficient.

After receiving the complete set of triplets-cells, the algorithm goes directly to the CA part of the reconstruction. The evolutionary stage must weed out the weak elements, preserving those of them that can form tracks. Considering the selection criteria, it is worth noting that the track retains momentum with minimal losses throughout its entire trajectory in a uniform magnetic field, and the slope cannot change abruptly. Therefore, the main signs of the triplets proximity are two common hits and similar momentum values. Using KF to fit each triplet, among other things, allows estimating errors optimizing the selection process.

The evolution process is a sequential enumeration of all triplets (except for those located at the last three stations). The algorithm selects possible neighbors for each of them and compiles a list for further processing. Station-by-station processing from the inner to the outer part of the detector also makes it possible to estimate the position of the triplet in the chain and its entire length. For

each 'surviving' triplet, the algorithm stores its sequence number in the longest possible chain, defined as the number of the previous triplet (or the largest of those, if there are several) plus one.

Further combining the previously prepared chains of triplets and creating track candidates becomes a trivial task. The final part of the tracking algorithm is to select the best combinations of hits in the branches and evaluate the found tracks. The preference is given to the longest tracks since the probability of wrong random combinations of triplets for them is much less than shorter competitors. Therefore, the procedure for constructing tracks begins with the longest of them. If the track is recognized as correct according to the fitting results, all hits that compose it are marked as used and are excluded from further consideration. Further, the algorithm continues the procedure for constructing tracks, decreasing the reviewed length step by step. If there is a competition between tracks of equal length during branching, the winner is determined by the minimum value of $\chi^2$ as the most reliable.

After the reconstruction of tracks with a certain length is completed, the found candidates are ranked by $\chi^2$ in ascending order and are sequentially checked for the presence of used hits. If used hits are found, the candidate is discarded. Otherwise, the candidate is saved. All its hits are marked as used and are excluded from further consideration in subsequent stages of the algorithm.

At last, the final stage of reconstruction is an attempt to extend the received tracks and pick up additional hits that belong to the track but, for some reason, are not included.

The given scheme of the algorithm is well developed and demonstrates a stable result. However, it still has several weak points, leading both to a loss of efficiency and a serious slowdown in computations.

An obvious drawback of the above scheme for forming triplets is the need for a complete enumeration of candidates for assignment to a singlet or doublet among hits at a certain station. Without the possibility of additional geometric differentiation of hits within the station, the procedure for selecting a neighbor by checking all of them with each other can slow down the algorithm hundreds of times. This problem becomes especially significant, considering that about 90% of the entire time of the algorithm operation is spent on the triplets finding.

The solution to this problem is storing hits in the computer memory in a special structured form and using special functions for accessing the data. This concept is called a grid structure. A rectangular grid consisting of cells of equal

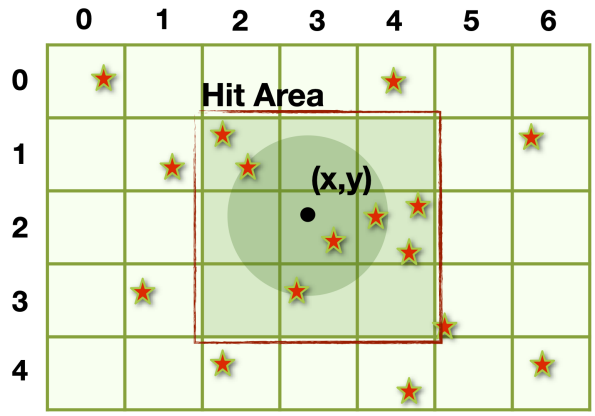


Figure 4.5: An illustration of how the grid structure works to optimize storage and access to the hits in the computer memory.

sizes is superimposed on the coordinate plane of the detector station. The cells are numbered sequentially, starting from the upper left corner of the coordinate plane and ending with the lower-right one. The hits on the station are pre-sorted so that their order matches the grid numbering. Navigation inside the structure is carried out using a special array, the number of elements corresponding to the number of the grid cells. For each non-empty cell, the ordinal number of the first hit that fell into it is determined and entered into the navigation array. The number of the array element is equal to the number of the considered cell. Knowing the geometric dimensions of the cells allows unambiguously matching the coordinate reference with the cell number and knowing the cell number to quickly access the hits inside it.

The utilization of the grid structure in the algorithm is as follows. Instead of looking for a hit directly, after extrapolating to the station and determining the area of interest, a helper object named HitArea is created. HitArea is a rectangular area of one or more grid cells that completely covers the hit search box. Boundary cells of the considered area are determined based on the minimum and maximum coordinates of the window of interest using the internal mechanisms of the grid. Further, knowing the numbers of the required cells and using the navigation array, the algorithm gets the opportunity to quickly access the necessary hits without exhaustive search.

Another problem that arises during the formation of triplets is associated with

the possible loss of hits due to the detector's inefficiency. This can lead to both the appearance of gaps and the complete loss of theoretically reconstructable tracks. The solution to this problem is carried out by introducing an additional step in constructing doublets and triplets. If, after extrapolating the singlet or doublet to the next station, there is no suitable hit in the region of interest, the algorithm attempts to step over this station and extend the segment further. Thus, depending on the stage at which this procedure is carried out, we can get triplets with a missed hit after the first or second position, covering four stations at once instead of three. Attempts to extend the track through more than one station were considered inappropriate due to the estimated probability of losing two hits in a row at less than 0.05%.

The initial conditions and imposed restrictions at all stages of the track search significantly impact both the result and the speed of the algorithm. Minor errors at the initialization of the singlet through the vertex and narrow corridors of interest during the selection of hits make it possible to speed up the algorithm a lot but lead to the loss of low-energy (more curved) and secondary (born outside of the primary vertex) tracks. In order to achieve maximum efficiency while maintaining a high computation speed, the algorithm is divided into several stages. Approximately the same operations are performed during each of them, but with different settings and restrictions. The following is a quick overview of each step.

**1. Search for primary tracks with momentum** $p > 1GeV$**.** Fast tracks with high momentum have straight or slightly curved by the magnetic field paths. The vertex and the first hit fairly accurately fix the direction of motion of particles born at the vertex. It is possible to introduce maximum restrictions without losing efficiency. At the end of the stage, all hits of the found tracks are marked as used. As a result, the number of hits considered at the next stages is significantly reduced. According to the research carried out, the number of tracks being reconstructed at this stage reaches 70% of the total.

**2. Search for primary tracks with momentum**$0.1 < p < 1GeV$**.** Slow tracks are more strongly affected by the magnetic field, which bends their trajectories. For a successful reconstruction, it is necessary to expand the range of search and used cuts. The required effect is achieved by increasing the error in determining $q/p$ in the covariance matrix by a factor of ten when initializing the singlets. The reduced number of hits decreases the volume of erroneous connections and shortens the algorithm's running time.

**3. Search for the secondary tracks.** The decay of short-lived particles

forms such tracks at some distance from the primary vertex. In order to include such tracks in the search area, the initialization of singlets is carried out with the coordinate errors increased by ten times. Using such starting conditions at the first stage will lead to a huge number of incorrect connections and will significantly reduce the algorithm's speed. But after completing the first two steps, 80% of the hits are excluded from consideration, allowing to search for secondary particles in a reasonable time.

**4. Search for tracks with missed hits.** The triplets formation algorithm is supplemented by the previously described possibility to carry out extrapolation through the station to restore tracks, the reconstruction of which is impossible in any other way due to the inefficiency of the detector.

The absence of hits can lead to a complete loss of tracks and the formation of separated segments belonging to the same particle but not connected. Such repeated segments are called clones. The number of clones rarely exceeds the level of 1% of the total number of reconstructed tracks and can be further reduced. For this, in the final part of the tracking algorithm, a special clone suppression procedure has been added, which allows finding and combining separate segments belonging to the same tracks.

The considered algorithm is planned to be used for online event reconstruction in the CBM experiment as an essential part of the FLES (First Level Event Selection) software package. At the same time, the experiment does not imply special triggering of events. The decision to keep or discard an event should be made in real-time based on fast reconstruction and analysis. This is the only possible approach given the tremendous interaction rate of up to 10 MHz and the inability of the equipment to handle such a large data stream. The rejection of dedicated event triggering is an innovative solution that has not been tested before in real experiments. The FAIR Phase-0 program involves the study of technical and software solutions of the CBM experiment under the conditions of existing HEP experiments. This allows to evaluate existing developments, identify and correct existing shortcomings in advance. The STAR experiment provides extensive opportunities for testing both hardware solutions and detection systems, as well as algorithms. The High-Level Trigger, which includes the CA-based reconstruction algorithm, is a unique platform for exploring the possibilities of online reconstruction and data analysis. In addition, the energy range considered within the BES-II program is similar to that in the CBM experiment. Such a study allows evaluating the quality of the results obtained compared to offline processing and

the reliability of the FLES concept in the CBM experiment.

## 4.4 CA Track Finder in the STAR experiment

The applicability of Cellular Automata for reconstructing the trajectories of charged particles is not limited to forward detecting systems with the fixed target. The STAR experiment is one of the striking examples of utilization of such approach both for the offline and online reconstruction of events with the barrel structure.

A feature of collider experiments, which class STAR belongs to, is that the creation of particles occurs due to the collision of two oppositely directed beams moving with the same speed and energy. As a result, scattering fragments occurs in a wide front in all directions from the collision point. The TPC was chosen as the main detector of the experiment for tracing trajectories and particle characterization.

The readout system of the detector is divided into 24 independent sectors, 12 on each side of the cylinder. Each sector, or slice, has, in fact, its local coordinate system, rotated relative to the global system at an angle multiple of 30°. An analog of the detecting stations is the pad rows, which per slice significantly exceeds the number of STS stations in the CBM experiment. In the original version of the TPC detector, there were 45 pad rows. In the modern version, this number has been increased to 72. The number of particles generated in the collision also significantly exceeds the expected CBM values and can reach over 2000 per event. At the same time, particles twisted in a magnetic field can not only cross several slices, creating separate sections of the trajectory, but do it several times, forming spring-like loopers.

Under such conditions, the previously considered approach to the implementation of the CA mechanism turns out to be ineffective due to the increasing amount of computations and the need to additionally take into account the peculiarities of the detector geometry. Instead, to solve this problem, a different CA-based track search approach was taken, which was previously used in the ALICE [95] experiment at CERN. The main paradigms of this approach are separate search for tracks in slices, minimization of computations when constructing spacecraft cells, and additional processing of the track segments. Separate tracking becomes possible due to the independence of measurements in different sectors and

the ability to designate different coordinate systems in each of them. At the same time, work in the local coordinate system turns out to be even more convenient than the global one because of the possibility of a clear binding of hits to the coordinates of 'stations' — pad rows perpendicular to the radial guideline. The need to simplify the CA cells creating procedure is associated with extremely high volumes of data. The fulfillment of this condition becomes possible due to the close arrangement of adjacent pad rows, which significantly simplifies the selection of the closest candidates for joining. On the other hand, this simplification increases the likelihood of track discontinuities. The high efficiency of combining track segments within and between slices corrects this defect and provides the necessary tracking quality.

Since the main phase of the reconstruction takes place in local coordinates, the vector of track parameters must be defined in the same coordinate system. The coordinate system itself looks like this. The X axis corresponds to the radial line of the cylindrical detector. The X coordinates of the hits correspond to or are close to the coordinates of the pad rows. The Z axis is the cylinder axis. The Y axis is perpendicular to X and Z. Under such conditions, we need the following characteristics to describe the track position: track coordinates in the plane of the pad row $(y, z)$, track slope in the XY plane $(sin(\phi))$, track slope in the XZ plane $(d_z/d_s)$, charge and momentum $(q/p_T)$, where $p_T$ is a transverse momentum — the momentum component perpendicular to the direction of the beam, which is more indicative than the total momentum in collider experiments. Thus, the full vector of parameters looks like this: $(y_i, z_i, sin(\phi_i), d_{zi}/d_{si}, q/p_T)$, where $i$ is the pad row number that defines $x$-coordinate.

The basis for determining the neighborhood of cells in the evolution process is the directly adjacent position of the hits and a similar curvature. Thus, the optimal cell of the Cellular Automaton, as in the case of CBM, is the triplet. Determining the neighborhood by two adjacent hits is a simple and understandable solution, but the full use of curvature becomes too resource-intensive with a critically high number of elements. Therefore, the approach to organizing triplets has been changed, taking into account the peculiarities of the data. In such conditions, the risk of incorrect triplet chains formation increases, and it becomes necessary to check and evaluate them in more detail.

Thus, the CA based track reconstruction in the TPC detector includes the following basic steps:

- triplets finding;
- connecting triplets into chains;
- sector tracks (tracklets) finding;
- merging of the sector tracks.

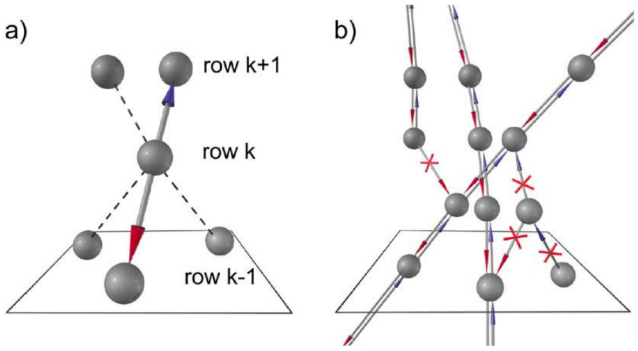


Figure 4.6: Cellular Automaton track finding procedure in the TPC detector [96]. a) Combination hits into triplet based on the minimal difference in the slope; b) CA based combination of triplets into chains.

Triplets finding in the TPC detector occurs directly without intermediate creation of singlets and doublets. The CA part of the algorithm does not use the track parameters, so the initialization of the state vector and fitting are not performed at this stage. This allows processing a huge number of hit combinations in a reasonable amount of time. In addition, a distinctive feature of this implementation of the algorithm is the competitive selection of triplets directly during the construction process, rather than within the further evolution.

The method for constructing triplets is as follows. Three consecutive pad rows are examined at once. The hit at the middle station of the triplet becomes a basis for further searching. Each middle hit can form only one triplet, which won due to the competitive selection. The hit coordinates are linearly extrapolated to the previous and next rows using point $(0, 0, 0)$ to specify the direction. A more accurate choice of direction and a better extrapolation at this stage seem impractical due to the nature of the particle trajectories. For the same reason, the window of interest for capturing hits must be wide enough to get hits even for significantly curved tracks. As in the case of the STS detector, access to hits in the

previous and subsequent rows is carried out using the grid structure by creating HitArea objects. With some differences in software implementation, the grid's logic fully corresponds to that discussed earlier. Then, the algorithm sequentially considers all the triplets that can be composed of the average hit and hits that form HitAreas at the neighboring pad rows. Among the many combinations, one is selected with the smallest slope difference between the doublets constituting the triplet, that is, the most straightforward one. If the difference between the slopes of the doublets satisfies the given criterion $\chi^2$, the established connections from the central hit to the edges are preserved. It becomes available for use at the next step of the algorithm.

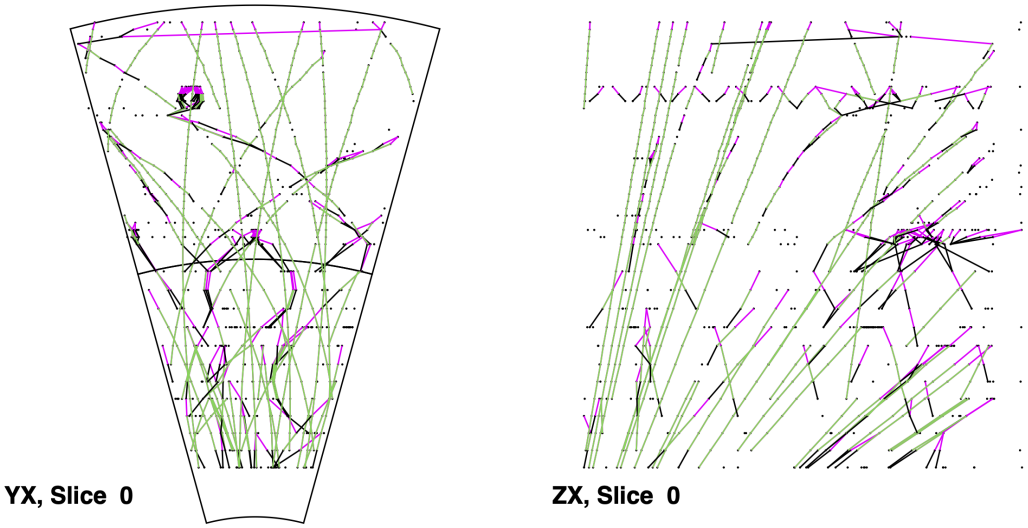


Figure 4.7: An example of the process of building hit chains in the TPC detector. Transverse (YX) and longitudinal (ZX) projections of one slice are shown. The black color indicates the connection of hits towards the previous (inner) pad row. The pink color indicates the connection of hits towards the next (outer) pad row. Bidirectional links are shown in green.

The second step of the algorithm is the CA evolution based on the direct neighborhood. A triplet 'survives' if it has two hits in common with at least one other triplet (Fig. 4.7). Thus, the branching of tracks at this stage turns out to be impossible. Track candidates are fully defined and do not compete with each other. No special objects for storing information about the composition of chains are created. Instead, the starting hit of each chain is flagged and entered into a special list. The chains themselves are determined by the sequence of links

between hits, defined earlier.

As well as in the case of the CBM track finder, optimization of the reconstruction of low-momentum tracks is carried out using an additional iteration of the algorithm, completely repeating the actions of the first one but using a wider $\chi^2$ cut and windows of interest when selecting hits. A distinctive feature of this algorithm implementation is that this operation is performed only with the CA part of the algorithm. At the beginning of the second iteration of the reconstruction, all hits included in chains at the previous step are marked as used and no longer participate in the formation of triplets. The final combined results of both chains reconstruction iterations are used to create and merge sector tracks. Before proceeding to the next step, chains are sorted by the number of hits from highest to lowest. This positively affects the quality and speed of the further algorithm's operation.

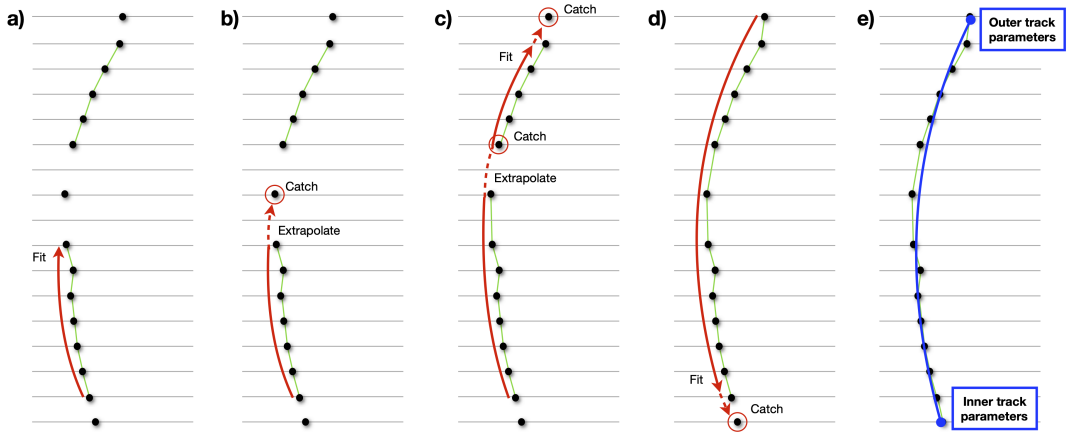


Figure 4.8: The process of constructing of the sector track based on the chain of hits: a) fitting of the chain; b) extrapolation and picking up an additional hit; c) extrapolation, catching and fitting of a suitable chain with additional extension; d) fitting of the track in the opposite direction with extension; e) reconstructed sector track with parameters determined at the first and last point.

The primary function of the tracklet creation stage is to evaluate the correctness of the previously composed chains of hits (Fig. 4.8). The first hit of the chain initializes the track parameters vector. After that, the algorithm will fit the track hit by hit using the KF track fitter. Moving through the hits of the chain is carried out using links that form triplets. Upon reaching the last hit in

the chain, the algorithm tries to pick up additional measurements on subsequent pad rows, if possible. The filtered hit chain allows KF to extrapolate the track accurately enough to make the window of interest for hit matching very small. In the case of finding several hits in the area of interest, the most suitable one is selected with the condition of the minimum change in the slope, identical to the triplets. If a hit is found, the algorithm tries to fit the track with this new measurement and evaluate the $\chi^2$ deviation. If successful, the hit is joined, and the search continues. As a maximum, the algorithm allows to skip up to four hits in a row.

In addition to a single hit, the tracklet extending procedure can stumble upon a whole chain of hits, which can be part of the same or a completely different track. The processing of encountered hits as a full-fledged chain occurs when the following conditions coincide: during the extrapolation — the first hit of the chain was picked up, the chain was not fitted earlier. This prevents the use of the same segments in multiple tracks at the same time. With this, the joint use of single hits by a few tracks at this stage is permissible. This is consistent with the physics of the detector, according to which the hits of closely flying overlapping tracks may be combined into one measurement for the detecting device.

Upon completion of the fitting procedure, if the last pad row of the detector or the tracklet extension limit is reached, the algorithm reverses the direction of the fit and performs all of the above operations, trying to extend the track at the inner side.

The above sequence of operations occurs independently for each slice of the TPC detector and must be executed before starting the sector tracks merging.

The final step of the algorithm is designed to combine disparate segments into full-fledged tracks. It processes all sector tracks in the event simultaneously. The merger's work on combining tracklets is based on analyzing the position of the ends of the tracks and the sufficient coincidence of their parameters. The algorithm handles several situations sequentially, collecting segments into long tracks: combining overlapping tracks within a slice, combining non-overlapping segments within a slice, combining segments between slices. Overlapping tracks can occur due to the tracklet builder extending tracks.

The parameters of the tracks entering the merger are often of poor quality and do not correspond to the first/last hit of the track due to the extension mechanism. Therefore, the first step of the algorithm is to re-fit the tracks with the KF fitter. Then, the track indices are written into special auxiliary arrays

and sorted. The order of the tracks is determined by the pad row number of its first hit. This allows to simplify and optimize the search for possible candidates for attachment. If the algorithm knows the pad row number of the certain tracks' last hit, it can determine the corridor of row numbers, where to look for the first hit of a possible candidate for combining. It specifies the amount of the first matching track and the number of tracks to be checked.

The possibility of combining tracks is examined by comparing the coordinates, slope, and momentum of the inner track at the last point and the outer track at the first point. Couples that have successfully passed the procedure are marked as neighbors for further processing. Particular attention should be paid to the case of combining segments located on adjacent slices. As mentioned earlier, each slice has its own coordinate system. Translating the track parameters and errors into the global coordinate system for the comparison procedure is difficult and may lead to errors. As the solution, the parameters of one of the tracks will be rotated by 30°in the direction opposite to the slice with the segment to attach. Thus, the rotated segment is transferred to the coordinate system of the adjacent slice while keeping the direction and necessary parameters.

At the final stage of merging, the algorithm estimates the position of the tracks relative to each other and attempts to fit the segments as a single track. The mechanism of track parameters rotating is built into the fit procedure and is triggered at the right time of the fitting process in case of intersectoral combinations. If successful, the first one of the combined segments receives the hits of the second and the parameters of the general fit, the second segment is deleted.

The merging procedure is highly efficient and allows to successfully compose sector tracks, reconstructed independently of each other, into global tracks, which are the final result of the track finder operation.

# Chapter 5

# Implementation of the CA Track Finder for the HFT detector in the STAR experiment

Achieving the optimal quality of track reconstruction and ensuring a high data processing speed when developing a tracking algorithm requires taking into account specific features of the detecting system. The Heavy Flavor Tracker (HFT) differs from both the STS and MVD detectors of the CBM experiment and the TPC detector of the STAR experiment. As a barrel detector, like the TPC, it is critically different in its operating principle and the number of available measurements per track. It prevents the use of the TPC-class reconstruction approach. Technically, the HFT detector is similar to the MVD+STS bundle, but it has fewer stations and a fundamentally different shape.

It is the similarity of particle detection technologies and the small number of stations that, in this case, are the most important criteria in developing a tracking algorithm. The maximum available number of measurements in a track is 4, as well as the minimum number of hits required for a segment to be consistently recognized as a particle trajectory and not as a random combination. In such conditions, the CA tracking option used in the CBM experiment is optimal for reconstruction. Sequential search and fitting of doublets and triplets allow to realize the detector's advantage in measurement accuracy and minimize the possible loss of hit combinations. It is especially important since every doublet not found results in losing the entire track to which it belongs.

An essential feature of the HFT detector is also the fact that it includes stations

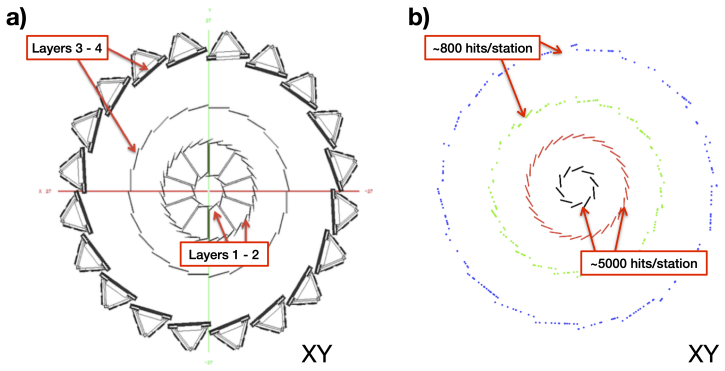


Figure 5.1: XY projection of the HFT detector: a) layout of the HFT detector; b) distribution of hits at the detector stations including the pileup.

built on the basis of different technologies (Fig. 5.1 a). The two inner stations are PiXeL. It provides increased accuracy in finding the hits that are closest to the vertex. Thus, the maximum resolution is achieved when determining the primary and secondary vertices, which is the crucial goal of installing this detector. The negative side of using a pixel detector is its low operating speed. The dead time of sensors exceeds the intervals between the readings of individual events by the TPC detector, which leads to the overlap of events and the formation of pileup — noise hits that do not belong to the given event and complicate the reconstruction process. The next two stations, made up of silicon pad and strip sensors, are characterized by lower hit positioning accuracy, which, however, has a minimal effect on the final vertex resolution, in contrast to the internal stations. At the same time, external stations are much faster and can maintain the speed of the TPC detector.

The multiplicity of tracks in the detector can be up to 800 per event. The number of hits at each of the outer stations roughly corresponds to the number of particles, slightly exceeding it due to slight overlaps of neighboring ladders, on which the sensors are fixed, or due to the noize. At the same time, the typical value of pileup hits at each of the inner stations reaches 5000 (Fig. 5.1 b).

The algorithm for track reconstruction in the HFT detector was initially developed based on the CBM CA Track Finder concept without taking into account several features of the problem. Later, it went through a series of successive improvements, which allowed to track and evaluate the effect of each of them.

Thus, in addition to the final result, the work on the HFT Track Finder became a useful research project that allowed us to study the impact of certain algorithm upgrades on tracking quality and speed.

## 5.1 CBM CA based track reconstruction in the HFT detector

The original version of the HFT CA Track Finder was built without considering the detector's features. It was a reworking of the CBM CA algorithm concept for the barrel geometry of the detector without additional optimization.

The tracking process is carried out in three stages. First, the parameter vector and the covariance matrix are initialized for each hit, forming a singlet. Then, pairs of hits at neighboring stations are combined into short segments — doublets. Doublets with a common hit and suitable parameters form triplets that can be combined into a track candidate, provided there are two common hits and similar momentums. And finally, at the last stage, the best one is selected from each bunch of candidates, which becomes the final 4-hit track.

The algorithm actively uses the Kalman Filter (KF) at almost every stage of its work. Choosing the right track model in such a situation is critical to ensure a robust reconstruction. One of the fundamental factors in this is the choice of the coordinate system. As well as in the case of the TPC detector, the use of the global coordinate system is not optimal for the CA algorithm since this would require a more complex implementation of KF. On the other hand, the HFT detector does not have clearly defined sectors, the number of modules is different for each station, and the modules themselves are installed at a slight angle to the tangent to the circle. That is, they are not perpendicular to the radius. It makes no sense to associate the local coordinate system with any of the stations or modules in such conditions. Instead, hits are translated into a polar coordinate system. Their position is expressed in terms of the polar angle $\alpha$, the radius $X$, and the $Z$-coordinate corresponding to that in the global system. A segment or track is fitted in the local coordinate system associated with its first hit, that is, in the coordinate system rotated by an angle $\alpha$ relative to the global one. Thus, the process of fitting a track in the HFT detector becomes similar to the TPC but concerning the coordinate system of a specific hit instead of a sector. The vector of track parameters is also similar to that in the TPC CA

— $(y_i, z_i, sin(\phi_i), dz_i/ds_i, q/p_T)$, where $i$ is a station number.

The CA procedure in the HFT tracker is similar to the CBM experiment, while the track and fit model is similar to those in the TPC detector. The fixed target in the CBM experiment allows one to determine the coordinates of the primary vertex with high accuracy and use them when initializing the singlet parameters during the reconstruction process. The STAR experiment is the collider one, and the primary vertex formed by the collision of two oppositely directed beams of particles can be significantly displaced relative to the geometric center of the detector. The collision's coordinates search is carried out by a special algorithm based on the TPC detector responses. Further, the coordinates of the reconstructed vertex are transferred to the HFT CA algorithm and used when searching for tracks.

The tracking process itself includes three iterations with different settings to ensure maximum efficiency while maintaining high computational speed. The process is organized by analogy with the previously considered tracking algorithms. At the first iteration, the admissible errors during the initialization of the vertex are minimal. The algorithm reconstructs straight primary tracks with a momentum of more than 1 GeV. Then, the used hits are excluded from consideration, the search limits are expanded, and low-momentum tracks are reconstructed in the range from 50 MeV. At the last stage, secondary tracks are reconstructed, born at a small distance from the primary vertex.

Initially, the development and testing of the algorithm were carried out using simulated Monte Carlo data without considering the presence of the pileup. In such conditions, the algorithm did not require additional improvements, providing efficiency of 80–90% depending on the characteristics and track multiplicity and an acceptable operating time for use both in the offline and online reconstruction of the STAR experiment. The transition to more realistic data, characterized by a very high hit multiplicity at the first two stations, revealed significant disadvantages of the algorithm. Thus, the reconstruction time for an individual event exceeded 1 minute even with a small number of tracks. In order to solve the problem, several improvements were implemented that are traditional in solving such problems, as well as new approaches to vectorization and optimization of the calculations, were developed and investigated. The most effective improvements and the effect of their use are discussed below.

## 5.2 The HFT tracker improvements

### 5.2.1 Direction of tracking

The standard approach to track reconstruction using the Cellular Automaton involves creating connections between hits at neighboring stations in all theoretically possible combinations. This approach makes it possible to simplify and generalize the algorithm scheme, making the calculations as fast as possible. Despite the fact that some of the established connections, doublets, and triplets, do not correspond to real tracks and are destroyed during the evolution of the CA or due to some other conditions, this is justified. But the tracking conditions in the HFT are different from the standard ones. The pileup, densely sowing the surfaces of the first two stations, leads to the formation of a huge number of segments that cannot be connected to any track. In this case, the share of useful hits falls below 1%, and the calculations become excessively expensive. At the same time, pileup hits are not excluded from consideration when moving from iteration to iteration of the algorithm, not allowing to take advantage of the separate search for primary and secondary tracks. An effective solution was to

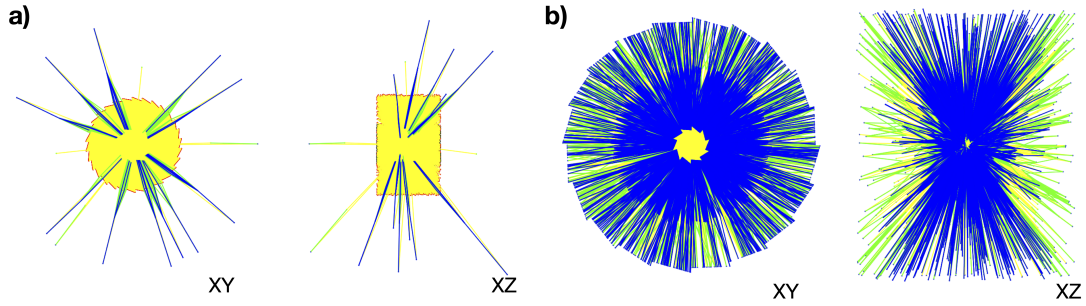


Figure 5.2: Visualization of the track finding procedure starting from the outer station in the HFT detector. Singlets are drawn in yellow, doublets in green, and triplets in blue. a) Low track multiplicity. b) High track multiplicity.

exclude from consideration some of the pileup hits. The impossibility of joining them to the tracks is obvious. For this reason, the direction of the reconstruction was changed. The search for doublets begins not from the inner but the outer station and is carried out sequentially station by station. For each subsequent station, only those hits that are included in the doublets on the previous station can take part in the doublets construction on this station. (Fig. 5.2). Thus, as

Table 5.1: Comparison of the calculations speed when reconstructing tracks from the inner to the outer stations and from the outer to the inner stations.

| N tracks | Inner to outer (ms) | | Outer to inner (ms) | |
|---|---|---|---|---|
| | Doublets | Triplets | Doublets | Triplets |
| 10 | 32500 | 130 | 2100 | 5 |
| 100 | 37000 | 1250 | 2900 | 780 |
| 350 | 39500 | 11500 | 3400 | 2650 |

well as in the case of the standard reconstruction in the CBM experiment, the segment search results resemble tree structures with one base and branching at the end. But these structures are directed not from the inside out but from the outside to the inside.

The effectiveness of this approach depends on the number of tracks in the event since a sufficiently high track density can lead to the formation of a large number of doublets at the inner stations of the detector. In the Table 5.1 events with 10, 100 and 350 tracks are considered. A server, based on two eight-core Xeon X5550 processors with a clock frequency of 2.7 GHz, was used as a hardware platform for testing. In the normal tracking direction, the drop in the doublet search speed is relatively small since most of the operations are related to the pileup hits. Tracking from the last station allows you to minimize calculations tenfold, thus speeding up the program. At the same time, hits at the 3rd and 4th stations can be excluded from consideration if the tracks are successfully reconstructed, thus increasing the effect of separate reconstruction of the primary and secondary tracks.

### 5.2.2 Grid structure

The inclusion of a hit in a doublet or triplet occurs in two stages. A quick check allows determining the theoretical possibility that the hit can be included in the segment based on the geometric position. If the hit matches, the segment parameters have to be fitted together with that hit. If the standard deviation $\chi^2$ does not exceed the allowable value, the segment is saved; otherwise, it is discarded. A quick check of hits does not take much time, but it involves examining objects at

neighboring stations, each with each other. The HFT detector does not imply an explicit division into sectors. Therefore, the number of possible hit combinations between the first and second stations exceeds 25 million with a pileup of about 5000 hits per station. In such circumstances, using a grid structure becomes a prerequisite for fast reconstruction.

The grid structure is based on the idea of imposing a grid with numbered rectangular cells on the surface of the station and sorting hits depending on which cell they fall into. The HFT detector stations are cylindrical. Even if they consist of separate flat modules, the use of modules when positioning a hit is not an optimal solution since it requires additional determining which module the measurement belongs to. Given the location of the modules relative to each other and the possibility of partial overlap, this can cause difficulties and lead to errors when processing hits.

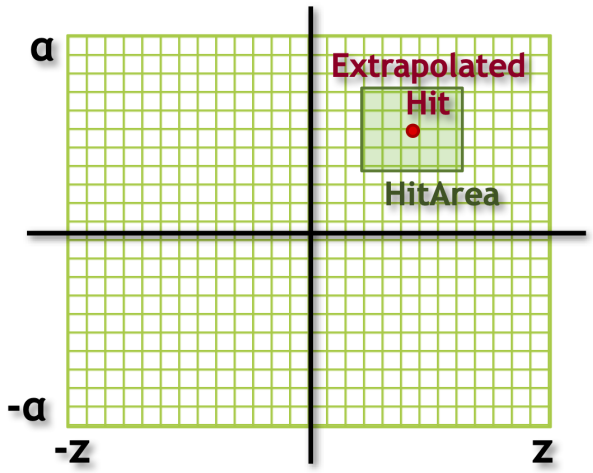


Figure 5.3: Grid srtucture in the HFT detector based on the $Z$-coordinate and angle.

Instead of working with individual flat elements, a single grid structure was applied to the entire surface of the station. Thus, the $Z$ axis is used as the horizontal axis of the grid. The vertical axis corresponds to the hit angle $\alpha$ in the $XY$ plane of the global coordinate system, measured in the range from $-\pi$ to $\pi$ in radians (Fig. 5.3).

Usually, grid cells sizes are selected, taking into account the hits density at the station. The optimal situation is when individual cells contain no more than one or two hits. Dynamically determining the number of cells in the grid structure

Table 5.2: Calculation speed of the doublets and triplets reconstruction steps using grid structure with different HitArea sizes.

| HitArea | Efficiency (%) | | | Doublets | | Triplets | |
|---|---|---|---|---|---|---|---|
| $(dz; d\alpha)$ | High $p$ | Low $p$ | All | Time (ms/ev) | Speed up factor | Time (ms/ev) | Speed up factor |
| No grid | 96.1 | 87.5 | 88.1 | 14000 | - | 790 | - |
| (1.5; 0.1) | 96.1 | 87.1 | 87.8 | 165 | 85 | 660 | 1.2 |
| (1.5; 0.05) | 98.0 | 85.6 | 87.0 | 86 | 160 | 133 | 5.9 |
| (1.0; 0.05) | 98.0 | 85.3 | 86.8 | 64 | 220 | 131 | 6.0 |
| (0.5; 0.05) | 90.2 | 82.5 | 83.1 | 37 | 380 | 105 | 7.5 |

allows optimizing it both for events with a high multiplicity of tracks and for simple cases with a low particle density. The presence of a huge number of pileup hits at the first stations of the detector makes the grid structure fine-grained, increasing the dependence of the successful hit selection on the exact size of the window of interest — HitArea — when reconstructing doublets and triplets. The smaller the HitArea, the more likely you will lose track, but the less time it takes to review all measurements inside the window.

Considering the peculiarities of the collision, the magnetic field, and the detector's geometry, the optimal size of the HitArea on the inner stations was determined as 1.5 cm in $Z$ and 0.1 radians in $\alpha$. Increasing the window of interest size has little effect on performance, but it slows down the algorithm. The possibility of decreasing HitArea and the effect on the efficiency and reconstruction speed was additionally investigated. The results of comparing the work of the grid structure with different areas of interest for catching hits are shown in Table 5.2. As input data, 200 simulated Monte Carlo Au+Au collisions at the energy of 200 GeV were used.

The main influence of the grids is related to the reconstruction of doublets since this stage accounts for the largest part of the combinatorial enumeration associated with the pileup. The search for triplets is based on combining previously obtained doublets, so the influence of the grid, in this case, is indirect. The purity of the reconstructed tracks is 75%; that is, one of the four hits may not belong to the corresponding MC track. Such a criterion is justified given the

abundance of pileup hits.

The use of the optimal region of interest $(1.5; 0.1)$ allows one to fully preserve the efficiency of high-energy tracks, losing no more than 0.5% of the most curved tracks with the smallest momentum. At the same time, a more than an eighty-fold reduction in the calculation time for doublets is achieved. The effect on the reconstruction of triplets is minimal. It is explained by the loss of some doublets outside of the HitArea. Narrowing the area of interest leads to a gradual decrease in the efficiency of the low-momentum tracks reconstruction. Still, it makes it possible to find more accurately straight trajectories that could otherwise be defined as ghost — false tracks that don't match any of the Monte Carlo originals. The time to reconstruct the doublets decreases in proportion to the changes in the HitArea size. It is due to the inner stations' more or less uniform filling with pileup hits. At the same time, the triplet reconstruction speed does not increase so quickly since it is primarily determined by the number of available combinations of doublets and not by the number of hits.

Thus, in addition to the original HitArea $(1.5; 0.1)$, which provides the maximum overall efficiency, $(1.0; 0.05)$ is also an acceptable option, providing a significant acceleration of the program with a loss of about 1% efficiency.

### 5.2.3 General vectorization of the computations

The found segments correctness estimation at each step of the algorithm is based on the parameters obtained by fitting using the Kalman Filter track fitter. The widespread use of KF requires significantly more time to build each doublet and triplet than the simplified reconstruction procedure typical of the TPC detector. Thus, even with only four stations and a lower average number of particles per event, track reconstruction takes much longer than in the TPC detector.

On the other hand, KF is an ideal algorithm for exploiting data-level parallelism. SIMDization of all calculations using KF allows to significantly speed up the algorithm without losing efficiency. Due to the general logical simplicity of the CA, direct vectorization of the reconstruction does not require changing the algorithm. It can be achieved by simply replacing scalar data structures with vector ones.

The vectorization of complex functions is based on the joint use of vector data structures and functions that can operate with them. The simplest way to vectorize the search for doublets is a primitive transition from processing single

pairs of hits to simultaneous calculations for 4 or 8 pairs of hits at once, depending on which SIMD intrinsics are used.

The HFT CA Track Finder was vectorized using the Vc library to simplify work with the code and use additional functionality of the package. Thus, SIMD data structures for hits and track parameters can be created by simply replacing scalar data types with vector ones or using templates. The first method seems to be the most convenient in this case since it allows you to change the vector code more flexibly, providing the necessary functionality. Vector analogs of object classes for hits, singlets, doublets, and triplets were implemented. Each SIMD object contains information about several scalar objects in the vector form. The Kalman Filter functional, integrated into the class of track parameters, was SIMDized in a similar way, together with the state vector and the covariance matrix. In addition to the transfer of calculations to the vector basis, a special mask was added to the function that controls which segments are active and which were rejected for one reason or another. It provides the ability to perform calculations only for a part of the segments included in one vector, excluding the possibility of erroneous use of a previously rejected segment.

The search for doublets itself is carried out as follows. The algorithm initializes the track parameters in the vector form using the hit and the primary vertex coordinates to calculate the initial approximation. Hits are considered in portions with a step corresponding to the number of elements in the SIMD vector. The scalar hit set is converted to the vector form. Further, a SIMD vector of track parameters is created and initialized by the primary vertex and the hits vector at the outer station, forming a singlet. The vector of parameters is extrapolated towards the vertex to the nearest station. Then, using the grid structure, a list of neighbors on the inner station is created for each hit in the vector. Hits-neighbors are combined into SIMD vectors according to the singlets they can be extended from. The algorithm sequentially tries to fit pairs of hits from the internal and external stations in vector form, using the parameters of the corresponding singlets as an initial approximation and neighbor hits as an additional dimension. According to the fitting results, combinations, for which the standard deviation $\chi^2$ turns out to be less than the permissible limit, are saved as doublets.

Each hit of the singlet vector, after extrapolation, forms its own independent HitArea. The calculations associated with converting the coordinates of the hit to the numbers of grid cells in which to look for neighbors are also independent of other hits. Hence, these operations can be easily vectorized. The vector grid's

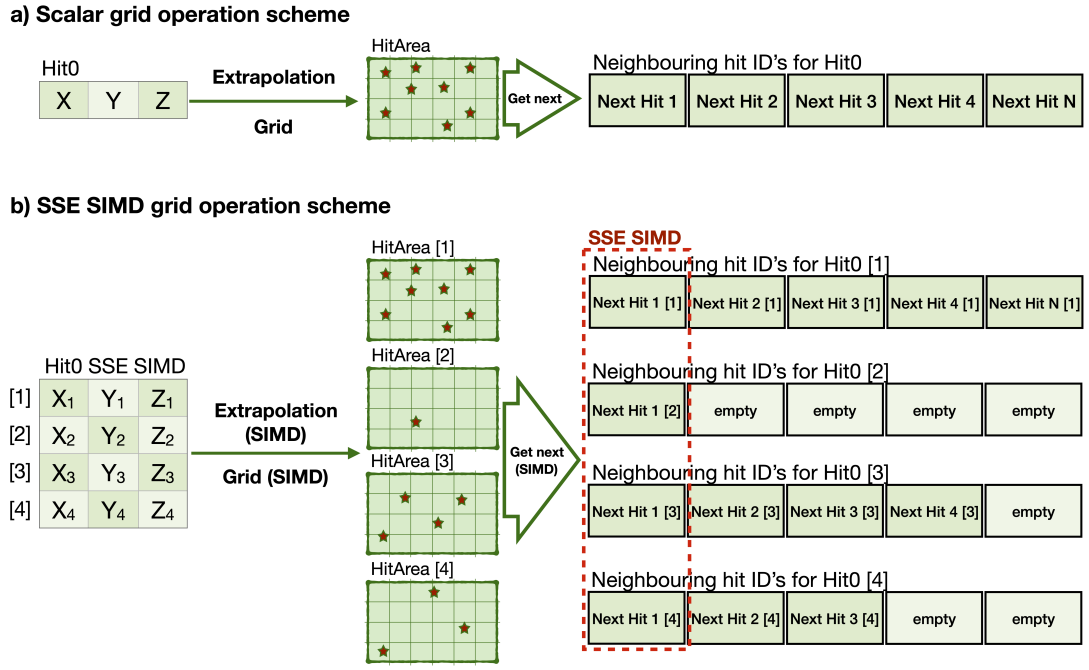


Figure 5.4: Grid structure operation scheme: a) in scalar mode; b) in SIMD mode shown by the example of the SSE intrinsics.

operation scheme is similar to a scalar one but uses SIMD variables instead of scalar ones as the input and output data. At the same time, the data structures themselves and access to the hits in the cells remain scalar, while the construction and operation of the HitArea go into a vector format. It makes it possible to select the indices of neighbor hits at once for the entire vector of the measurements under consideration (Fig. 5.4). But the access to the hits themselves, arranged in an array of structures ordered in accordance with the grid, is still carried out as separate elements. The disadvantage of this approach is that the number of iterations of the SIMD HitArea corresponds to the biggest number of neighbors among the vector elements. Thus, if this value differs significantly from element to element of the vector, an essential part of the elements turns out to be empty, which leads to an abundance of idle operations and a decrease in the vectorization efficiency. In the HFT detector, this problem is mainly offset by a more or less uniform distribution of the pileup hits at the stations. The use of a vectorized grid together with the previously considered SIMD scheme for doublets searching made it possible to speed up this procedure by 2–3 times in comparison with the

Table 5.3: Calculation time and speed up factor received from the doublets and triplets SIMDization using the grid structure with respect to the basic version of the algorithm with searching starting from the outer station.

| | Calculation time (ms) | | Speed up factor |
|---|---|---|---|
| | Before | After | |
| Singlets | 6 | 2.8 | 2.1 |
| Doublets | 14000 | 18.6 | 750 |
| Triplets | 790 | 11.2 | 70 |
| Neighbours | - | 6.5 | - |
| Tracks | 50 | 4.5 | 11.1 |
| Grid | - | 5.9 | - |
| Total | 15000 | 55 | 270 |

data given in Table 5.2.

As in the case of doublets, vectorization of triplets involves the simultaneous fitting of several segments. The presence of a common hit determines the primary match of doublets. For each vector element of outer doublets, neighbors are selected from the number of inner doublets. The hit vector is formed from the inner hits of the attaching doublets. Then, the algorithm tries to fit the resulting combinations by transporting the parameters of the outer doublets to the attached hits and subsequent filtering. Combinations whose parameters satisfy the $\chi^2$ criterion are saved as triplets and are used further during the track reconstruction. This approach is simple enough algorithmically and, together with the utilization of the vector grid, can speed up the reconstruction of triplets by more than 6–7 times in comparison with the values given in Table 5.2. The weakness of this implementation of vector computations is the increased number of memory accesses when repacking the SIMD data structures and idle operations in the case of a different number of neighboring doublets for elements of the same vector.

In addition to the vectorization, many functions of the program have been optimized to eliminate unnecessary computations and data transformations. Comparison of the calculation speed between old and new algorithm versions at each stage is given in Table 5.3. 200 simulated Monte Carlo Au+Au collisions at the energy of 200 GeV were used for the tests. Each contains about 10000 pileup

hits with an average number of tracks per event of about 240. A variant of the tracker implementation with track search from the outer station was used as a basic version. In order to provide vector computations, the SSE instruction set is used, overloaded with the Vc library.

Singlet initialization is accelerated by 2.1 times due to the vectorization. The greater speedup was hampered by the simplicity of the calculations and the relatively large volume of the memory operations while saving the singlets for further calculations.

Finding doublets, which was the most expensive part of the reconstruction, received the most improvement and became approximately 750 times faster. The most significant role in this was played by using the grid structure, which made it possible to reduce the number of computations by hundreds of times.

Reconstruction of triplets has been accelerated by about 70 times due to vectorization and optimization of computations. This stage was significantly influenced by creating a list of neighboring doublets simultaneously with their search at the previous stage of the algorithm.

The tracker's time spent searching for adjacent triplets with two common hits and building track candidates for the initial version of the program was not measured, so it is not possible to evaluate the changes in this part. The very same stage of direct reconstruction of tracks began to work 11 times faster. In this part of the algorithm, no fundamental changes were made. The increase in the computational speed is associated, first of all, with a decrease in the number of track candidates due to more efficient filtering out of incorrect doublets and triplets.

Formation of the grid structure and sorting of hits by the numbering of the cells required an additional investment of time absent in the original version of the program. But the acceleration of the algorithm obtained from using these mechanics compensates for this time many times over.

Thus, for the program as a whole, it was possible to achieve an acceleration of approximately 270 times, reducing the average reconstruction time to 55 ms per event. This result appears to be significant compared to the original tracker performance but is still too slow to reconstruct tracks in the detector with only four stations. Based on the research results, three main weaknesses in the program can be identified, correcting which will increase the performance of the algorithm to the maximum:

- data structures — focus on the SIMD data structures leads to the need for numerous repackaging of vector variables during the execution of the algorithm and additional overhead costs associated with random access to the memory;

- separate stages of calculations — a clear separation of calculations into singlets, doublets, and triplets with saving the results of each of them and reading the data with the next function makes the code simpler and more understandable, but does not allow stream processing, creating the need for additional read/write operations;

- inefficient vectorization — a straightforward approach to packing data in the SIMD vectors leads to empty and redundant computations, the number of which increases with the level of combinatorics inherent in the CA algorithm.

### 5.2.4 Efficient vectorization of the computations

Work on the further development of the HFT track finder was continued to maximize the acceleration of the algorithm by eliminating the discovered weaknesses in its implementation.

**Data Structures** are the backbone of high-performance computing. Choosing the right data structures can optimize memory access and reduce overhead while maximizing the effect of parallelism at both the data and task levels. The efficiency of using certain data structures depends on the operations that are planned to be performed with them. Since the work of a track finder is primarily associated with the merging of hits into segments and tracks, it is the method of storing hits that has the greatest effect on the speed of calculations at all stages of the algorithm.

In order to optimize computations, three different ways of storing objects in memory have been used (Fig. 5.5). The initial preparatory part of the work consists in the fact that the algorithm sequentially reads hits from an external source, initializes the grid structure, and sorts the measurements following the numbers of the grid cells. For such operations, storage of information in the form of an array of structures is most suitable (Fig. 5.5 a). A set of objects, each containing complete information about one hit, is sequentially located in the memory. Also, this type of data structures is suitable for the temporary storage

**a) Array of Structures**

| Objects: | Hit 1 | | | | Hit 2 | | | | Hit 3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | float | float | float | ... | float | float | float | ... | float | float | float | ... |
| Memory: | $X_1$ | $Y_1$ | $Z_1$ | ... | $X_2$ | $Y_2$ | $Z_2$ | ... | $X_3$ | $Y_3$ | $Z_3$ | ... |

**b) Structure of Arrays / Simple Arrays**

| | float | float | float | float | float | float | float | float | float | float | float | float... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Memory: | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $X_7$ | $X_8$ | $X_9$ | $X_{10}$ | $X_{11}$ | ... |
| | $Y_1$ | $Y_2$ | $Y_3$ | $Y_4$ | $Y_5$ | $Y_6$ | $Y_7$ | $Y_8$ | $Y_9$ | $Y_{10}$ | $Y_{11}$ | ... |
| | $Z_1$ | $Z_2$ | $Z_3$ | $Z_4$ | $Z_5$ | $Z_6$ | $Z_7$ | $Z_8$ | $Z_9$ | $Z_{10}$ | $Z_{11}$ | ... |

**c) Array of Structures of Arrays (SIMD vectors)**

| Object: | SSE SIMD Hit | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | float_v (__mm128) | | | | float_v (__mm128) | | | | float_v (__mm128) | | | | ... | | | |
| Memory: | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $Y_1$ | $Y_2$ | $Y_3$ | $Y_4$ | $Z_1$ | $Z_2$ | $Z_3$ | $Z_4$ | ... | ... | ... | ... |

Figure 5.5: Basic data structures in the HFT CA Track Finder illustrated by the example of hits: a) Array os Structures; b) Structure of Arrays; c) Array of Structures of Arrays.

of other objects, such as doublets, triplets, track candidates, etc. It provides the fastest access to each element by its index since all object's characteristics are compactly located in the same memory area.

It is convenient to perform vector calculations with vector objects described earlier and illustrated in Figure 5.5 c. Such objects can either be formed immediately before use or stored as an array, corresponding to a data structure called Array of Structures of Arrays.

In addition, frequently used scalar data that you plan to use in the vector calculations should be stored in such a way as to provide the fastest access to it, even if it requires additional memory costs. In this case, these variables are the coordinates of the hits since most of the CA operations use them in one way or another. It is convenient to store such variables as separate arrays aligned in the memory by the size of the corresponding SIMD data type (Fig. 5.5 b). That is, for an array of floating point objects with a data type size of 4 bytes, you should choose alignment 16 in case of using SSE intrinsics, 32 for AVX, etc. This allows not only to access a specific element of the array quickly but also to convert several variables at once into a SIMD vector without additional data

copying, using the reinterpret_cast functionality to cast the type without data copying and conversion.

**Separation of computations of singlets, doublets, and triplets** is convenient from the developer's point of view since it allows better control over the reconstruction progress and works with separate modules independently of the others observing only the agreed format of the input/output data. But this is not strictly required by the CA algorithm. In order to reduce the number of memory operations and get rid of the use of intermediate data structures, the three basic stages of the algorithm were combined into a single function, and its structure was optimized for the streaming calculations.

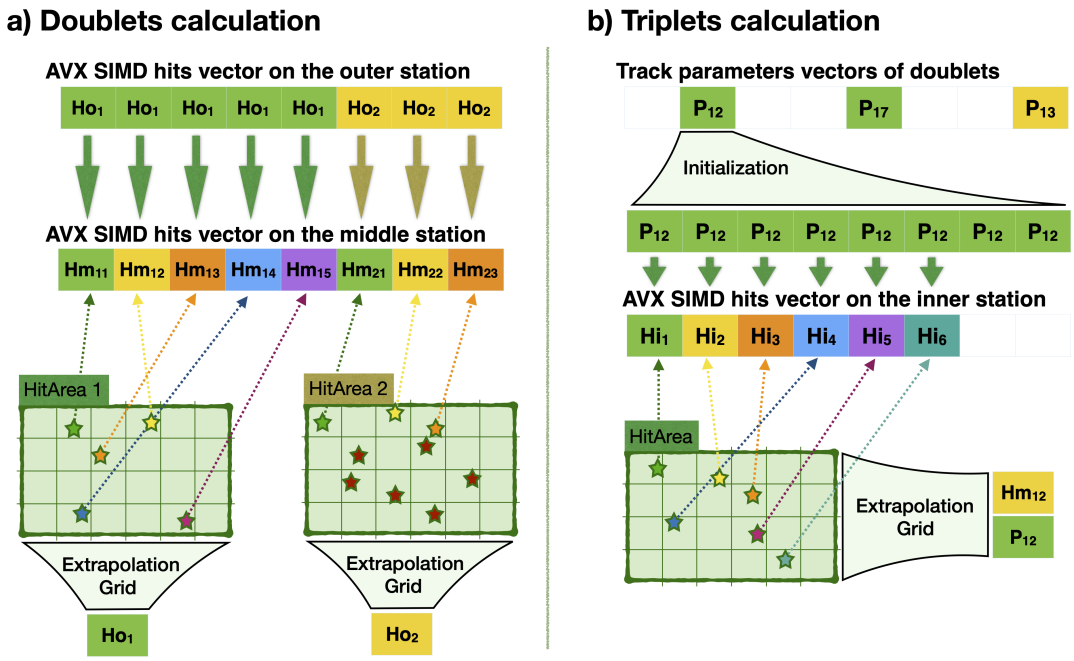


Figure 5.6: Block diagram of the SIMDized triplet search algorithm using the example of AVX vectors, which includes two stages: a) reconstruction of doublets; b) reconstruction of triplets.

The key to the **efficient vectorization** is to minimize memory access while ensuring as many consecutive operations as possible on the same SIMD variables. The search for doublets is traditionally the most time-consuming part of the calculations due to the high level of combinatorics, which does not allow taking full privilege of the advantages of SIMD calculations. In the course of work

on accelerating the algorithm, an approach to vectorization was developed and tested, which reduces the number of idle operations at this stage, making vector calculations much more efficient [97, 98].

Calculations start from the outer station and are carried out in two stages: stations 4–3–2, then stations 3–2–1. Using a unified approach to both tracking steps makes the algorithm scalable, allowing, if necessary, to change the number of detector stations without making significant changes to the program. Moreover, the lists of neighboring triplets with two common hits are compiled directly during their reconstruction without additional calculations.

Significant acceleration when using SIMD calculations cannot be achieved without the maximum filling of the vectors. In order to implement this task, a non-standard scheme of the algorithm was used, designed to ensure the efficient utilization of the SIMD vectors at the cost of some additional logical operations.

First of all, the algorithm sequentially examines the hits at the external station, extrapolates them to the middle station, and creates the HitArea's to find the hits-candidates for joining. All this occurs in the scalar mode since vectorization does not significantly speed up the process but leads to an unjustified complication of the algorithm. The hit-candidate indices from the middle station are stored in a special temporary array for further consideration. After processing each hit from the outer station, the algorithm checks the number of objects in the temporary list. If it is greater than or equal to the size of the SIMD vector, it proceeds to the next stage of triplet reconstruction. Otherwise, the next hit is considered. Thus, a situation is ensured when all pairs of hits in a group represent real candidates for the formation of doublets. This excludes the possibility of idle calculations for all groups, except the last one.

Having received filled arrays of indices of neighboring hits at the outer and middle stations, the sizes of which correspond to those of the SIMD vectors, the algorithm proceeds to the next stage of calculations. Now you need to form SIMD objects for hits participating in calculations in accordance with the obtained indices (Fig. 5.6 a). The positioning of the hit in the local coordinate system is done using six floating point variables: $(X, Y, Z, err_Y, err_Z, \alpha)$. All data elements are stored in advance in the special arrays for faster access. Considering the small number of variables, the overhead costs of generating SIMD vectors of hits are minimal. In addition, the hit vector at the outer station rarely consists of more than 1 to 2 different measurements, further simplifying and speeding up the process.

SIMD vectors of parameters of the future triplets are initialized using the primary vertex and hits at the outer station, similar to the process of creating singlets described earlier. But, unlike the previous case, the results are not saved but are immediately used for further calculations related to the reconstruction of doublets. Hits-candidates for joining have already been identified. So, all that remains is to fit the segments and check their compliance with the $\chi^2$ criterion.

It is impractical to attach the third hit to the segment the same way, ensuring the maximum filling of the SIMD vectors, for three main reasons. First, this may require additional iterations of the doublet reconstruction to replenish the SIMD vectors. This will not only lead to disruption of the data streams but will also require unnecessary complications of the algorithm, which will negatively affect its speed and stability. Secondly, the vector of track parameters will also include a covariance matrix, bringing the number of floating point variables to 31, 20 of which must be initialized when forming the corresponding SIMD object. And third, the pileup in the inner stations ensures a large number of hits-candidates in the region of interest, reducing the effect of vectorization inefficiencies.

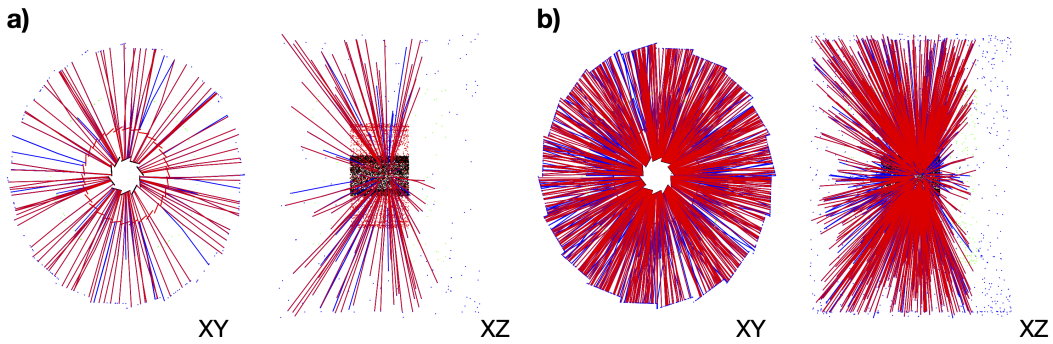


Figure 5.7: Visualization of the track search procedure in the HFT detector using the developed SIMDized reconstruction algorithm. Triplets are shown in blue, tracks — in red. a) Average track multiplicity; b) High track multiplicity.

Thus, ensuring the maximum filling of the vectors during the formation of triplets was considered unpromising. Instead, the stake was made on reducing the overhead costs in the formation of SIMD vectors. The hits vector is still formed from a set of objects, while the vector of parameters is initialized with a single element (Fig. 5.6 b). Upon completion of the resulting triplets' fitting, invalid candidates are screened out, possible neighbors among good triplets are

Table 5.4: Calculation speed of various reconstruction stages of the upgraded HFT CA Track Finder when working in scalar and SIMD mode using SSE and AVX intrinsics.

| | Scalar (ms) | SIMD SSE (ms) | SIMD AVX-1 (ms) |
|---|---|---|---|
| Input (hits) | 0.6 | 0.6 | 0.6 |
| Hits sorting, grid | 4.7 | 2.3 | 2.3 |
| Triplets | 27.1 | 5.5 | 4.6 |
| Tracks | 4.7 | 1.4 | 1.2 |
| Output (tracks) | 0.8 | 0.8 | 0.8 |
| Total | 37.0 | 9.3 | 8.2 |

determined, and the results are saved. Then, the algorithm returns to the first step and considers the next hit from the outer station. As a result, singlets and doublets are not created as separate objects but are used to form triplets directly from which tracks are composed in a similar way to the basic version of the algorithm. The intermediate and final results of the algorithm are shown in Figure 5.7.

The results of comparing the calculation time of each step of the algorithm are shown in Table 5.4. The initial focus on obtaining the best result precisely in vector calculations led to the fact that the scalar version of the program turned out to be more than five times slower than the vector version instead of the expected maximum of 4. Moreover, even with an abundance of overhead, the scalar version turned out to be faster than the previous implementation of the algorithm. The greatest acceleration effect is noticeable in the reconstruction of triplets, where the SSE version of the program is almost seven times faster than the scalar version. It is also quite indicative that launching the program using AVX-1 technology gives an additional, albeit not very large, increase in the calculation speed. This demonstrates that vectorization works efficiently despite the high level of combinatorial calculations and logical operators. The use of AVX intrinsics leads to a significant increase in overhead, both algorithmic and technical, due to the slower processor operation in this mode.

Table 5.5 shows the efficiency of the final version of the algorithm for tracks with the purity of 75 and 100%. The increase in efficiency compared to the

Table 5.5: Reconstruction efficiencies of the upgraded HFT CA Track Finder calculated for tracks with purity of 75 and 100%.

| | 75% correct hits | 100% correct hits | MC tracks/event |
|---|---|---|---|
| All tracks | 90.3% | 81.8% | 237.2 |
| Primary high-p | 98.1% | 93.7% | 27.4 |
| Primary low-p | 92.2% | 83.3% | 198.1 |
| Secondary high-p | 83.6% | 76.0% | 0.7 |
| Secondary low-p | 36.5% | 26.1% | 10.9 |
| Ghost | 17.5% | 36.9% | |
| Correct tracks/ev | 215 | 194 | |

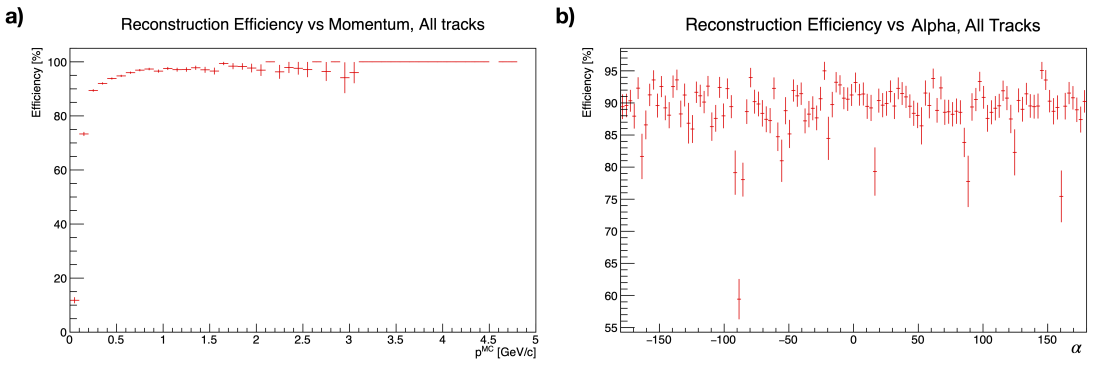


Figure 5.8: Distribution of track reconstruction efficiencies in the HFT detector depending on a) momentum; b) $\alpha$-angle.

previous version of the tracker is due to the extrapolation for the selection of the third hit in the triplet takes into account the position of the first two hits. That is the likelihood of hitting the right hit in the area of interest increases. The efficiency of primary particles reconstruction, which includes all four correct hits, exceeds 80%, which is a good result, given the probability of finding a pileup hit that is close enough to the desired one. A sharp drop in the reconstruction quality when restoring the trajectories of low-momentum secondary particles is associated with the curvature of their trajectories by the magnetic field. This takes hits outside the area of interest, the expansion of which will lead to a significant drop in the computation speed but does not guarantee an increase in

efficiency due to the abundance of the pileup hits. Also noted is the high level of ghost tracks that were not associated with any Monte Carlo particle. The reconstructability condition for simulated trajectories includes hits at all four stations. If a hit on one of them is missing, such a track can still be found using pileup hits, but it will be marked as wrong regardless of its quality.

Figure 5.8 shows distributions of track reconstruction efficiencies with a purity of 75% depending on the momentum (a) and the angle of rotation of the track's local coordinate system relative to the global one (b). Track reconstruction efficiency of less than 90% is typical for low-energy tracks with momentum less than 300 MeV. Tracks with the momentum of 1 GeV and more are correctly found in more than 95% of cases. That corresponds to the data presented in Table 5.5. The change in the efficiency depending on the angle is determined by the position and inclination of the separate stairs-like modules on the detecting stations. The heterogeneity of the detector makes it difficult to find tracks, but the algorithm manages to maintain a fairly high quality of reconstruction.

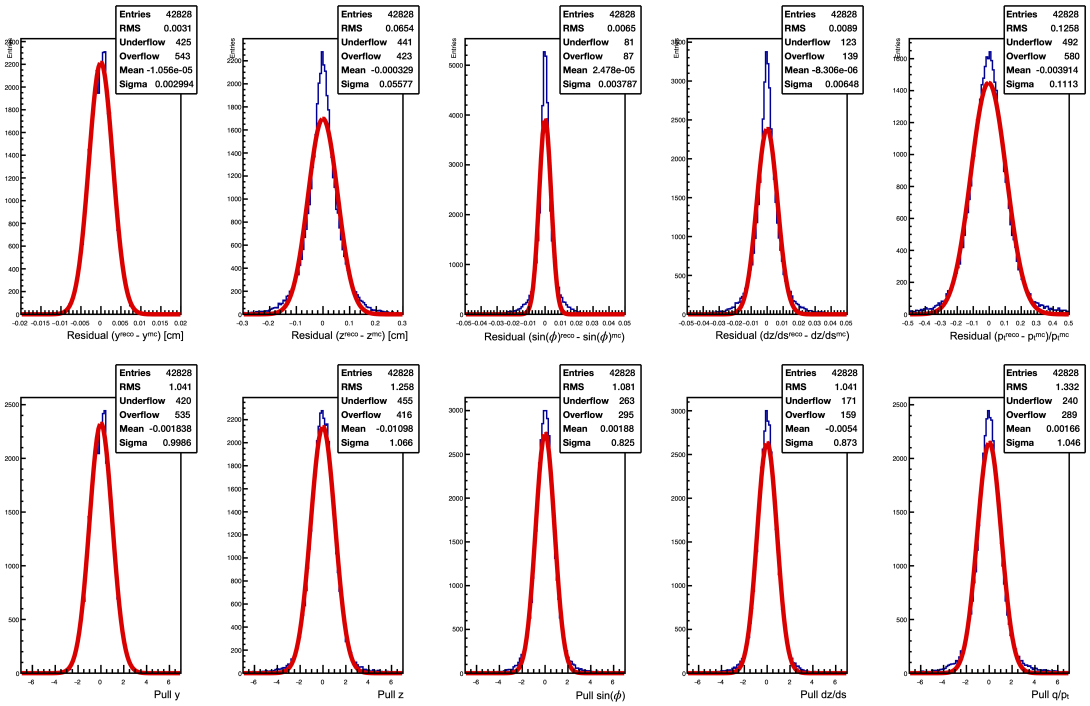


Figure 5.9: Fit quality of the tracks reconstructed by the HFT CA Track Finder measured in the last hit.

The residual and pull distributions shown in Figure 5.9 illustrate the perfor-

mance of the tracker in terms of found tracks parameters correctness. Close-to-Gaussian distributions and near-one Sigma values of the pulls indicate the correct operation of the Kalman Filter during fitting and the high quality of the obtained trajectories. The pulls for $sin(\phi)$ and $dz/ds$ are slightly narrower than expected, which indicates a slight excess of the considered material inside the detector. But Sigma values in the range of 0.8–0.9 are still acceptable for assessing the quality of tracks as high.

Thus, the considered approach to the track reconstruction in the HFT detector demonstrates a high speed and efficiency even in the presence of the pileup and the multiplicity of tracks characteristic of the STAR experiment. At this stage, algorithm improvements were suspended due to the exclusion of the detector from the reconstruction chain and the lack of plans for its further use.

# Chapter 6

# SIMDization and upgrades of the STAR TPC CA Track Finder

The initial challenge for the CA-based track reconstruction algorithm in the STAR experiment was to reconstruct events in real-time within the High-Level Trigger (HLT) operating. The most important requirement for the algorithms that are part of the HLT is a very high computation speed, provided that the tasks of express data processing are stable. The high efficiency of the developed CA implementation made it possible to ensure a full-fledged reconstruction of tracks, but not only an events selection based on several parameters.

The well-proven CA algorithm has been integrated into the offline track reconstruction chain as the seed producer for the main track following based algorithm. This approach not only made it possible to optimize the event reconstruction process but also provided a significant increase in the efficiency of finding particle trajectories. Figure 6.1 shows the comparison of tracking efficiency with and without using the TPC CA. In fact, in this configuration, the CA performs the main work of track reconstruction, while the track following algorithm checks the received chains of hits and extends them, if possible.

The critical factor for offline processing of experimental data is the maximum possible efficiency of the event reconstruction. The most complete reconstruction of the products of particle collisions expands the possibilities for physical analysis and allows realizing the research potential inherent in the data. At the same time, the speed of data processing plays the most crucial role in online reconstruction. The triggering algorithm must be fast enough to provide primary data analysis and decide whether to keep or discard events as they arrive without queuing. The

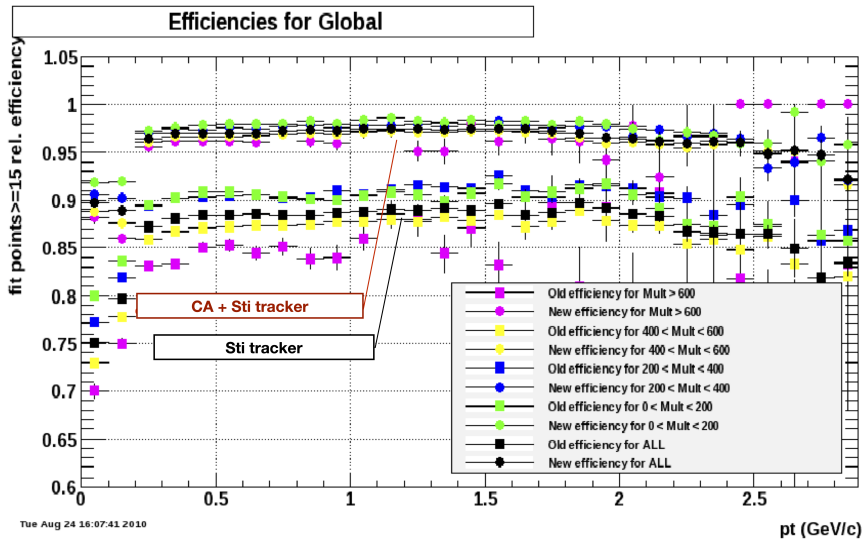


Figure 6.1: Comparison of tracking efficiency with and without the TPC CA Track Finder at different track multiplicity values [99].

initial requirements for the functionality of express reconstruction are usually much lower than for offline systems.

The TPC CA Track Finder replaced a simpler conformal mapping algorithm in the STAR experiment. Unlike its predecessor, the CA tracker is not limited by the ability to stably find only those tracks that were born in the primary vertex. Even though the TPC CA was significantly faster than the implementation of the track following algorithm used in the experiment, this was not enough for use in the HLT, given the hardware available at that time.

In order to achieve the highest possible computational speed without loss of reconstruction quality, the algorithm was vectorized using the SSE SIMD instructions and the specialized Vc library.

## 6.1 Features of the TPC CA Track Finder vectorization

The process of track finding using the Cellular Automaton in the STAR experiment includes several separated steps with various approaches to organizing, managing, and utilizing data. This implementation allows optimization of calculations at each stage following the specifics of the problem but makes it ineffective

to use vector data structures at the global level. Reformatting data structures and random memory access when using SIMD structures are accompanied by additional overhead costs, which makes such data structures sub-optimal in this case. On the other hand, representation of the information in the form of scalar arrays of variables and structures makes it possible to minimize the formation time of the SIMD vectors, provided that the data is pre-sorted following the characteristics of each step of the algorithm.

The features and the execution order of the main algorithm steps are discussed in detail in the previous chapter. The methods for vectorizing each of them and their impact on the program's overall performance are considered below.

**Triplets and hit chains constructor.** It is the simplest, but at the same time, time-consuming part of the reconstruction due to the need to process significant amounts of data. The stage is characterized by a high level of combinatorics and low computational complexity. In such conditions, the decisive role in increasing the computational speed is played by optimizing memory accesses, which is ensured by using the grid structure to sort hits and access them.

The process of forming each triplet does not depend on the others. The middle hit that starts the combination creates a single unique triplet, but the inner and outer hits can belong to several combinations simultaneously. In the SIMD version of the program based on the SSE intrinsics, four consecutive hits are used as the basic element of triplets, located on the middle pad row of the three considered. The hit coordinates form SIMD vectors used in the extrapolation process when choosing neighbors by the grid structure and determining the best combination. SIMD implementation of the grid structure allows generating a separate HitArea for each element of the vector and picking up hits from the considered cells in vector form for four elements simultaneously, returning indexes of candidate hits for combining as a SIMD vector. The selection of the optimal combinations of hits with the smallest difference in the slope of their constituent doublets is also carried out simultaneously for all vector elements by sequential enumeration.

In addition to directly speeding up computations by parallelization at the data level, implying the grid structure also allows optimizing memory utilization. Geometrical ordering of hits suggests that the hits of the outer rows addressed to adjacent hits of the middle pad row will also more likely to be neighbors. This makes information easier to access and allows for more efficient data caching. On the other hand, the number of neighbors for each starting hit can be different.

The number of operations required to check all combinations of hits is determined by the length of the largest list of neighbors among each considered four starting hits. With this, operations with shorter chains turn out to be superfluous. This dramatically reduces the potential efficiency of the vectorization. Considering the general simplicity of calculations, the inefficiency of vectorization when combinatorially enumerating hits significantly affects the overall speed. It does not allow obtaining acceleration more than 1.3 times.

The Cellular Automaton evolution and the hit chains formation are also implemented in a vector form by sequentially applying several masks that are part of the Vc functional instead of conditional operators. This ensures the survival of chains with stable bidirectional links between hits without the possibility of branching. Due to the simplicity of operations and the memory accesses, the actual speed gain is insignificant, despite the vectorization.

**Sector tracks reconstruction.** This stage is based on the fitting of the previously found hit chains with additional functionality that allows to attach measurements to the track and overcome gaps due to the loss of hits on some pad rows. The fitting of each of the tracks is independent of the others, and the algorithm itself includes a large number of sequential calculations. This makes the Kalman Filter a nearly ideal algorithm for vectorization. The initial number of the fitting steps is determined by the number of hits in the longest chain of the vector. To optimize computations, sector track candidates are pre-sorted in descending order of the number of hits, minimizing idle computations.

The tracks extension mechanism works according to the following principle: extrapolation to the next pad row, finding a suitable hit to extend the track, and, if successful, filtering the track parameters and assessing the possibility of joining the hit to the chain. Extrapolation and filtering are performed by KF fitting methods in the vector form. The selection of hits is also carried out simultaneously for the entire vector of track parameters using the grid structure by analogy with the operation of the triplet search algorithm. Extrapolating the track gives the approximate expected hit position, minimizes the size of the HitArea window of interest, and reduces the number of available merge candidates. In addition to individual hits, the algorithm can attach a whole segment to a track, which should be fitted in the same way as the original part.

Despite the SIMDization of almost all parts of the sector track finding algorithm, the calculations speedup is about 1.8 times. Among the negative factors that do not allow achieving the best result, it is worth highlighting the low effi-

ciency of vectorization of the track extension and the growth of idle fit operations due to the attachment of individual hits and complete chains to separate tracks.

Upon completion of the reconstruction, the tracks in each sector of the TPC detector are sorted in length descending order to optimize their further use.

**Sector tracks merging.** A significant part of the merger operation time is occupied by the process of refitting the sector tracks. This step is mandatory since, otherwise, the quality of the track parameters is unsatisfactory for the effective operation of the algorithm. Using the extrapolation for attaching hits in the case of failure leads to the fact that parameters of many tracks do not correspond to the real position of the first or last hits and must be updated.

Storing tracks and hits is carried out in the scalar form as it is more suitable for selecting and combining elements. SIMD vectors of starting tracks' parameters are formed by collecting data from sequential sector tracks, which leads to overhead costs due to random memory access. Still, it is the optimal solution in this case. The hit characteristics required for the Kalman Filter to operate with are also collected in SIMD variables before the fitting. Thus, the KF process itself is vectorized with maximum efficiency and performs idle calculations only in the case of processing tracks of different lengths, which is minimized by preliminary sorting. After completing a one-time fit procedure from the inside to the outside and back from the outside to the inside, the new internal and external parameters are again converted into the scalar form and saved to the corresponding tracks.

The merging candidate selection procedure was not vectorized due to inexpediency. It is a simple form of a list of tracks that theoretically can be connected, considering their geometric position.

Checking the correspondence of tracks from the lists and selecting the best pairs is a more complex computational problem that includes an abundance of logical operators, changing the coordinate system, extrapolating and filtering track parameters. All this makes the vectorization of this part of the program justified despite the expected idle computations.

Candidate lists are composed of track indices. The algorithm operates on two lists, which can be characterized as follows: base tracks — the basis for combining and the tracks to be attached. Base tracks are processed in portions. Each portion has the same number of the outer pad row, concerning which candidates are selected for combining. The tracks to be joined in each portion can start on different pad rows within the acceptable window of interest. To determine the best combinations within the chunk requires making pairs of tracks each with

each other. When switching to SIMD calculations, all elements of SIMD vectors and base track's data structures are filled with one single element. At the same time, this procedure for the attached tracks is carried out by arranging sequential elements from the corresponding list into the SIMD vectors of track parameters by analogy with the stage of refitting sector tracks. This approach allows for complete verification of tracks in vector form without significant algorithm changes, which can affect both the stability and the speed. With a high tracks' multiplicity, possible idle computations due to incomplete filling of the SIMD vectors do not significantly impact the overall performance of the program. With a low multiplicity of tracks, this feature can be neglected due to the overall high speed of the algorithm.

At the final stage of merging, the selected pairs of tracks are fitted together. This allows checking the correctness of the merge and getting the final parameters of the global track. As in the previous cases, this process is fully vectorized. The internal parameters of the base track segments are used as an initial approximation. They are collected into SIMD vectors by analogy with the initial fitting of the sector tracks.

The result of vectorization was the acceleration of the primary initialization and refitting of sector tracks by 1.8 times, which is due to overhead costs of data copying between scalar and vector variables, as well as the time spent on forming data structures necessary for the further operation of the merger. The merger's algorithmic part is characterized by a lower efficiency of vectorization and an abundance of scalar calculations. This part of the program was accelerated by about 1.1 times. That leads to the overall acceleration of the merger by a factor of 1.5 [100].

## 6.2 TPC CA upgrade and optimization for the iTPC geometry

Further development of the TPC CA Track Finder was primarily associated with increasing the algorithm's capabilities without significantly slowing down its operation. Greater acceleration of track reconstruction seems impossible without fundamental changes in the algorithm. Research carried out in this direction has shown that the acceleration of computations, although possible, is associated with a loss of efficiency and quality of the reconstructed tracks.

At the same time, as part of the preparation of the BES-II experiment program, the inner part of the TPC detector was significantly improved by installing new side sensors, which allow recording a larger number of measurements. As a result, the number of pad rows in the inner TPC (iTPC) increased from 13 to 40, for a total of 72 in the entire detector.

The updated iTPC makes it possible to reconstruct short-lived particles, which were extremely difficult to detect earlier due to the small number of measurements that such particles make in the detector before decay. On the other hand, such detector upgrade leads to an increase in the number of hits in the events by more than two times since the inner part of the TPC is characterized by a greater multiplicity of tracks compared to the outer part. Changes in the size and qualitative composition of the data caused the need for additional adjustment and refinement of the track reconstruction procedure.

The TPC CA Track Finder has proven itself both in real-time for express production within the HLT framework and in offline data processing, working in conjunction with the track following algorithm. Therefore, it was decided to keep the basic functions as unchanged as possible, limiting ourselves to fine-tuning to new tracking conditions and adding additional procedures that can improve the reconstruction quality. Improvement of the HLT computer farm and the increase in the available computing power allowed the focus on the track reconstruction efficiency, with less emphasis on maintaining ultra-high computational speed. However, this indicator was not excluded from the list of priorities.

The interior of the updated iTPC is characterized by a high multiplicity of tracks, gradually decreasing with distance from the center. Most of the low-energy tracks with high curvature are concentrated there. Previously, the insufficient number of pad rows and the large distances between them made it impossible to consistently reconstruct many of these tracks due to the lack of available measurements. The stage of constructing triplets for the CA evolution took this circumstance into account, having rather strict conditions for accepting a combination of hits as a triplet based on its curvature. The main part of the work on tuning this stage of the tracking algorithm was associated with reorienting and expanding the conditions for accepting a triplet following the detector's capabilities. This approach inevitably leads to the formation of false combinations that can complicate further reconstruction. In reality, the negative effect was not critical and was successfully intercepted during the evolution of CA and fitting of sector tracks.

More than a twofold increase in the number of hits in an event inevitably leads to an increase in the number of calculations, making the algorithm's execution more time-consuming. On the other hand, reducing the distance between the pad rows by more than 3 times made it possible to decrease the HitArea by a factor of several when searching for candidate hits for combining. This led to a decrease in combinatorial search when looking for optimal triplets since the density of hits on each pad row does not increase compared to the older, more rarefied version of the detector. Thus, despite a significant increase in the number of measurements, a sharp slowdown of the algorithm was avoided.

The sector tracks' reconstruction stage did not require significant changes, being a fairly universal algorithm for working with the various detectors' geometry versions. Work on this part of the Track Finder was aimed at improving the efficiency and quality of tracking, regardless of the version of the TPC detector. In order to do it, the window of interest was increased when searching for a candidate hit to join a track, as well as the process of searching for additional hits itself was extended to all tracks, regardless of their length, but not just the shortest ones, as it was done initially to ensure maximum speed of calculations. In addition, minor technical corrections and extensions were made to the program code to make it work more stable and fix problems that were not noticed before but became obvious when switching to the new geometry.

The minimum size of a chain of hits that can be accepted as a sector track is a combination of two triplets, that is, 4 hits. This approach is fully justified for the original TPC geometry since it allows to separate with a high probability of the real track candidates from random combinations of hits. The ineffectiveness of the detector or fusion of clusters of active pads when tracks come near to each other in rare cases can lead to the impossibility of forming a 4-hit segment. Hits lost due to such situations are usually either joined by longer tracks at the extension stage or, in principle, cannot form a track of sufficient length for a stable reconstruction. The increase in the number and density of pad rows in the iTPC expands the possibilities for reconstructing tracks in the inner part of the detector that would have been lost earlier. This makes single triplets more useful for the reconstruction.

In order to minimize the loss of tracks due to the peculiarities of the detector functioning, the initial reconstruction chain, consisting of two iterations — one for high-momentum particles and one for low-momentum particles — the search for triplets, chains, and sector tracks, was supplemented with one more procedure.

Its essence lies in an attempt to fit and renew not only trusted segments having four or more hits but also individual triplets that were not included in any of the segments. Before being destroyed in the CA evolution, such triplets are stored in a special array for further processing. The operation is performed only at the first iteration of the algorithm for two main reasons. First, it is at the first iteration that the most promising triplets can form, while at the second iteration, the number of obviously incorrect combinations increases significantly due to the less stringent cuts for the formation of a triplet. Secondly, using the information of the second iteration of the algorithm will require complicating the process of storing and utilizing single triplets to avoid collisions, re-reconstruction, and going out of available memory, which will lead to unjustified computational overheads.

If a sector track is successfully reconstructed, its hits are marked as used. They cannot be attached to other track candidates in the form of a sequential set, allowing only individual intersections. After the main stage of the sector tracks finding is finished, the number of hits available for the full-fledged extension of the tracklets becomes minimal. Thus, the processing of the remaining triplets is fast enough and does not significantly contribute to the total running time of the program. The extension itself is carried out using the previously considered algorithm of the sector tracks reconstruction.

The final part of the Track Finder — merging of the sector tracks into the global ones — also underwent several technical changes and fixes to optimize interaction with the new iTPC geometry. The window of interest in the initial selection of candidates for merging has been expanded in order to take into account the closer arrangement of the pad rows. At the same time, the general simplicity and successful vectorization of this step minimized the effect of the changes made on the computational speed. In addition, improvement of the final part of the track merger helped to eliminate the problem of saving tracks in case of unsuccessful merging, which is extremely rare with the old geometry but appeared after the iTPC upgrade. Thus, the track merger performance was not only preserved when switching to the new geometry but also additionally increased.

## 6.3 Reconstruction of tracks with very low momentum

The operation of the TPC detector is based on the effect of charged particle trajectory deflecting in a magnetic field. Changing the trajectories at the known constant magnetic field makes it possible to determine the characteristics of particles and identify them. At the same time, the initial momentum has a significant effect on the nature of the particle's motion. In other words, the faster the object generated by the collision flies out, the less impact the magnetic field will have on it during the flight through the detector camera. Accordingly, fragments with high momentum fly out practically in a straight line with minimal deflection. On the other hand, the smaller the initial momentum, the more curved the trajectory. In a constant magnetic field, typical for TPC, the trajectory of charged objects is a segment of a circle with a gradually decreasing radius due to deceleration of movement. Thus, particles with a sufficiently small momentum — loopers — can make several turns inside the detector. Upgrade of the TPC detector has significantly increased its capabilities for registering such tracks and raised the priority of improving the reconstruction process to restore them as completely as possible.

The track model underlying the Kalman filter of the TPC CA Track Finder assumes the reconstruction of straight and curved trajectories directed from the inside to the outside of the detector. But segments of tracks of similar shape directed from the outside to the inside will also be successfully fitted. The CA part of the algorithm itself has much softer requirements for combining hits, allowing to find backward directed tracks along with the usual ones. At the same time, the algorithm cannot reconstruct track segments that lie parallel to the pad rows. Thus, the looper formed during the passage of a very low momentum particle leads to the formation of a set of similar tracks, the number of which can reach several tens.

Reconstruction of the loopers within the CA part of the algorithm is impossible when using the previously considered TPC CA operation scheme. Combining trajectory segments at the stage of fitting and extending sector tracks requires a revision of the track model and will lead to significant overhead costs and a drop in computational speed, especially in the case of using parallelism at the data level. This approach will be complicated and have an extremely negative effect

on the program's speed.

The most promising possibility of reconstructing loopers is to combine the already found segments at the final stage of the program. This approach does not require changes to the basic functions of the algorithm and allows obtaining both long tracks assembled from parts and their elements at the same time.

The principles used to combine sector tracks in the merger are not suitable for loopers. Therefore, a new algorithm was developed to identify track segments belonging to the same low-momentum track, order them, and save them as a single formation. The Looper Merger was integrated into the algorithm after the merger for three main reasons. First, the merger successfully connects parts of the tracks from different sectors of the detector, including the looper segments. This improves the quality of the tracks and simplifies their further processing. Second, refitting tracks during merging improves track parameters, making the process of finding low-momentum tracks more accurate. Thirdly, such an operating procedure saves from the conflict between the reconstruction of loopers and the merger, which is not intended to work with such combined tracks.

At the initial stage of loopers merging, candidates potentially capable of forming loops are selected. Tracks momentum, determined during the fitting of the tracks at the previous reconstruction stages, is used as a criterion. The boundary after which a particle with a high probability will leave the detector without forming at least two turns in the TPC detector of the STAR experiment is 200 MeV. Tracks with a higher momentum value are excluded from consideration. Several additional parameters, which are necessary for more accurate classification, are calculated for the rest of the tracks.

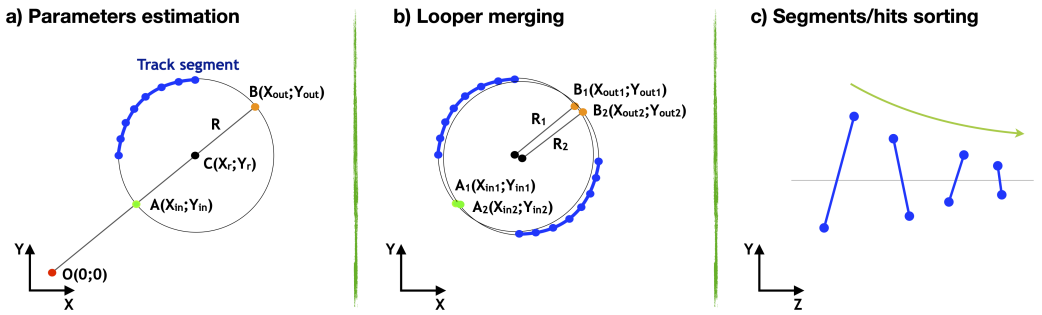


Figure 6.2: Algorithm for combining track segments with very low momentum.

The loopers merging algorithm is based on the fact that the momentum of a particle when passing through the detector decreases rather slowly so that the parameters of the circles built based on the track segments have minimal differences for the consistent loops (Fig. 6.2). A set of characteristics is calculated for each segment-candidate that passed the initial selection. The segment classification is based on the following parameters:

- **Radius.** A pair of segments can belong to the same looper if the radii of the corresponding circles differ by no more than 25%. The circles are determined by three hits of each segment. This criterion is intended to ensure the similarity of the momentum of the combined segments.

- **(X, Y) coordinates.** For each circle, the coordinates of points **A** — closest to the central axis of the detector (0, 0) and **B** — the most distant from the axis of the detector — are determined. The coordinates of points **A** and **B** of a pair of segments of the same looper should not have significant differences. The permissible difference is determined by taking into account the average value of the radii of the circles. The conditions imposed on point **A** are more stringent due to the geometric features of the track arrangement.

- **Z-coordinate.** Knowing the radius of the circle and the Z-coordinates of points **A** and **B** allows determining the approximate step for which the particle makes a half turn. Comparison of the Z-coordinates of the upper, middle, and central points of the circles corresponding to the segments under consideration demonstrates their position relative to each other. If the coordinate difference is significantly less than the step, the segments overlap and cannot be part of the same looper. If the difference is too large, the segments are, at least, not neighbors.

- **DzDs.** The track parameter, defined on fit, indicates the slope of the track along the Z-axis. It is assumed that segments of the same looper should have similar slopes without regard to the sign.

The sequential utilization of the considered conditions makes it possible to determine pairs of segments that are, with a high probability, neighbors within the same looper. Chains of adjacent segments form loopers that need to be formatted correctly before saving.

First of all, the segments of each detected looper are sorted by the $Z$-coordinate, taking into account the particle movement direction. In spite of the fact that the particle's momentum along the course of motion changes rather slowly, the KF allows to determine it with sufficient accuracy in order to define the direction of motion — from larger to smaller value. Then, the algorithm determines in what order — forward or backward — each segment's hits should be perceived, taking into account the direction of the spinning track, and marks the segment with the corresponding flag. The internal and external parameters of the resulting combined track are determined as the parameters of the first/last segment at its extreme points, taking into account the particle's direction of motion.

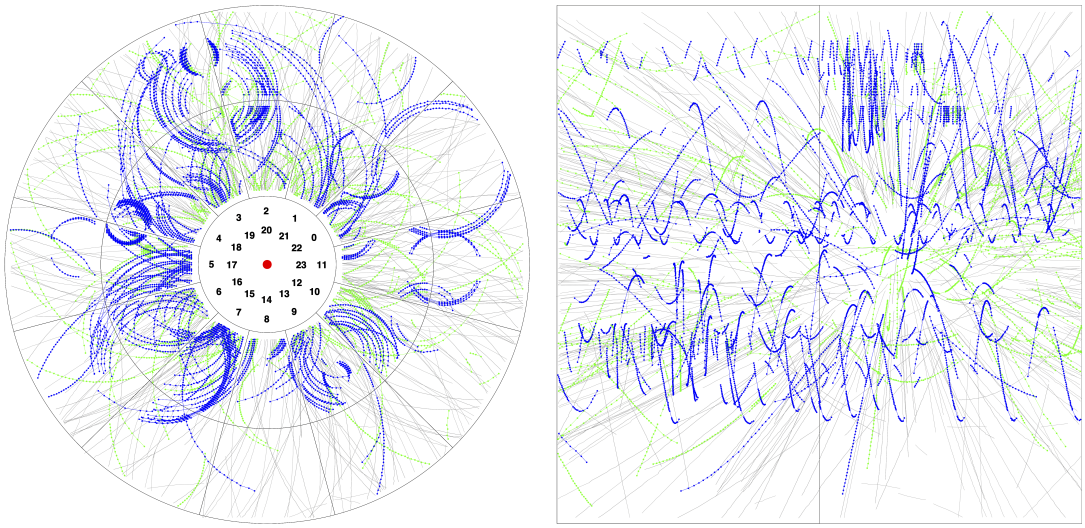

Figure 6.3: An example of a collision containing loopers reconstructed with the Looper Merger in the TPC detector. Tracks with a momentum of more than 200 MeV are shown in gray, tracks with a momentum of less than 200 MeV and not combined into loopers are shown in green, and reconstructed loopers are shown in blue.

Figure 6.3 shows an example of reconstructing loopers for a single event. Tracks with a momentum of more than 200 MeV are marked in gray. Green color — low-energy tracks not included by the algorithm in any looper. Reconstructed loopers are marked in blue. The event under consideration was obtained by simulating the Au+Au collision at the energy of 200 GeV. The efficiency of combining track segments into loopers on such data is 80.7%, with the purity

of the obtained tracks not less than 90%. That is, when calculating the loopers finding efficiency, no more than 10% of hits that do not belong to the main track are allowed, obtained both with the erroneous joining of single segments and as interspersed with incorrect hits in the correct sector tracks. The efficiency of the loopers reconstruction, in this case, is demonstrated by the ability of the Looper Merger to combine previously reconstructed tracks correctly. In order to minimize the influence of possible tracking inefficiency, the looper reconstruction efficiency is defined as the ratio of the number of reconstructed objects to the number of theoretically reconstructed ones, where a Monte Carlo track with a momentum of less than 200 MeV is considered to be reconstructable looper only if the CA tracker has found at least two segments of it. The proportion of loopers in which several torn parts were reconstructed — clones — is 17.6%. This is a good result, given the low quality of many looper segments and the algorithm requirements for observing the rotation step when determining the segment proximity.

## 6.4 Track finding performance and results

The shift in focus from achieving maximum speed to a combination of high speed and efficiency within the TPC CA Track Finder development made it a highly efficient algorithm suitable for full-fledged event reconstruction. Table 6.1 shows a comparison of the effectiveness of stand-alone old and new versions of the tracker when working with the same input data. Efficiency is estimated as the ratio of reconstructed tracks to the number of theoretically reconstructable ones. A track is considered theoretically reconstructable if it contains ten or more hits. A successfully found track must also include at least ten hits, more than 75% of which belong to the same Monte Carlo track. Hit chains with a purity of less than 75% are considered to be found incorrectly and designated as ghosts. MC tracks associated with more than one reconstructed track are additionally marked as clones. Such low requirements for reconstructability and relatively high requirements for the track to be considered as reconstructed lead to a decrease in efficiency indicators in digital form but allow a more detailed evaluation of the tracker operation features.

The track's momentum determines its curvature, hence the difficulty of reconstruction. In order to trace the dependence of the algorithm quality on the characteristics of the tracks, the effectiveness is assessed separately for several

Table 6.1: Comparison of track reconstruction efficiency between old and new stand-alone versions of the TPC CA Track Finder.

| Tracks type | Old efficiency | Old clones | New efficiency | New clones |
|---|---|---|---|---|
| LongRef | 99.7 | 1.0 | 99.7 | 0.7 |
| Refset | 84.8 | 5.4 | 87.9 | 1.6 |
| Allset | 83.5 | 6.9 | 86.7 | 3.6 |
| Extra | 83.2 | 7.4 | 86.3 | 4.1 |
| Ghost | 2.7 | | 4.8 | |

groups. The value of 0.05 GeV is used as the minimum boundary for reconstructability. A smaller momentum will lead to the formation of a looper with a small radius, individual segments of which are extremely difficult to recover. Particles with momenta from 0.05 to 1 GeV are designated as 'Extra'. They have enough hits to combine but are characterized by a curved shape that can make reconstruction difficult. Momentum more than 1 GeV corresponds to 'Refset' particles, with straighter and more easily reconstructable trajectories. Refset tracks that have hits on all pad rows are designated 'LongRef'. 'Allset' is a combination of all the above classes of tracks, which expresses the final efficiency of the algorithm.

As the data for testing, 100 simulated Monte Carlo events corresponding to Au+Au collisions at the energy of 200 GeV were used. The long straight tracks reconstruction efficiency approaches 100% in both cases, which is explained by the simplicity of their finding. The search quality for shorter tracks also correlates with their momentum, but the difference does not exceed 2% for refset and Extra variants. The level of clones also increases with decreasing momentum, reaching a maximum at values less than 300 MeV, characteristic of loopers. At the same time, the number of clones generated by the old version of the TPC CA is also large for Refset particles. This indicates that there are problems in the work of the sector track merger.

The updated tracking algorithm demonstrates an efficiency increase by more than 3% for all types of tracks, except for the longest ones, where the efficiency is already close to 100%. In addition, the changes applied made it possible to significantly reduce the number of clones both by improving the sector track merger

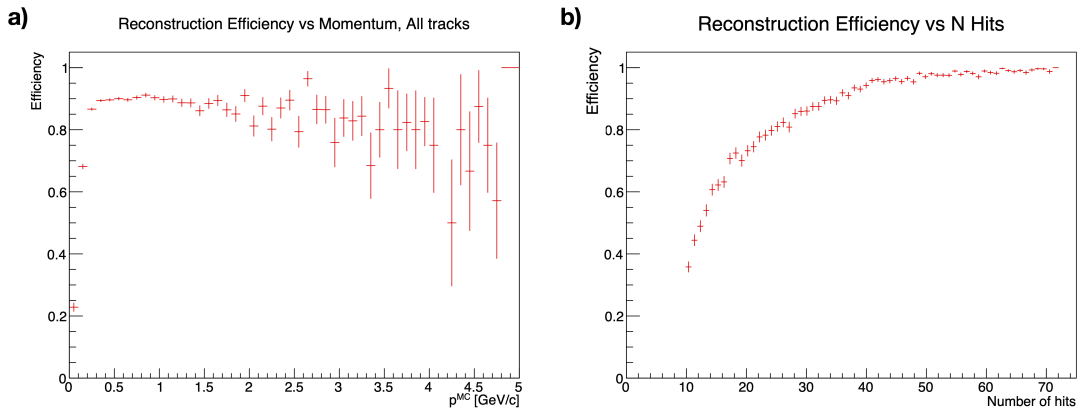


Figure 6.4: Distribution of track reconstruction efficiencies of the upgraded TPC CA Track Finder depending on a) momentum; b) number of hits in the Monte Carlo track.

and by adding the looper merging function. In this case, the effect produced is much lower for tracks with a small momentum. This is due to the greater complexity of combining low-momentum segments and the persistence of clones even among reconstructed loopers. The negative effect of the algorithm upgrade can be called an increase in the ghost level by 2%. But it should be noted that a higher number of the incorrectly found tracks occurs at a slower pace than the increase in the reconstruction efficiency. The calculation of the average track purity indicator was not implemented for the old version of the algorithm, but in the updated version, this number is 98.3%, which testifies the high quality of the reconstruction.

Figure 6.4 shows the dependence of the track reconstruction efficiency in the TPC detector on the momentum and track length. The best efficiency of about 90% is typical for tracks with a momentum in the range of 0.3–1.5 GeV (Fig. 6.4 a). Most tracks with less momentum are loopers. Their trajectories are highly curved, making it difficult to reconstruct as the particle's energy decreases. Tracks with higher momentum values are quite rare in such collisions, making it difficult to accurately assess the quality of the reconstruction within the statistics used for the tests. Considering these indicators in combination with the dependence of efficiency on the track length (Fig. 6.4 b), one can notice the correlation of these parameters. So, long tracks, including more than 40 Monte Carlo hits, are reconstructed with the probability close to one. At the same

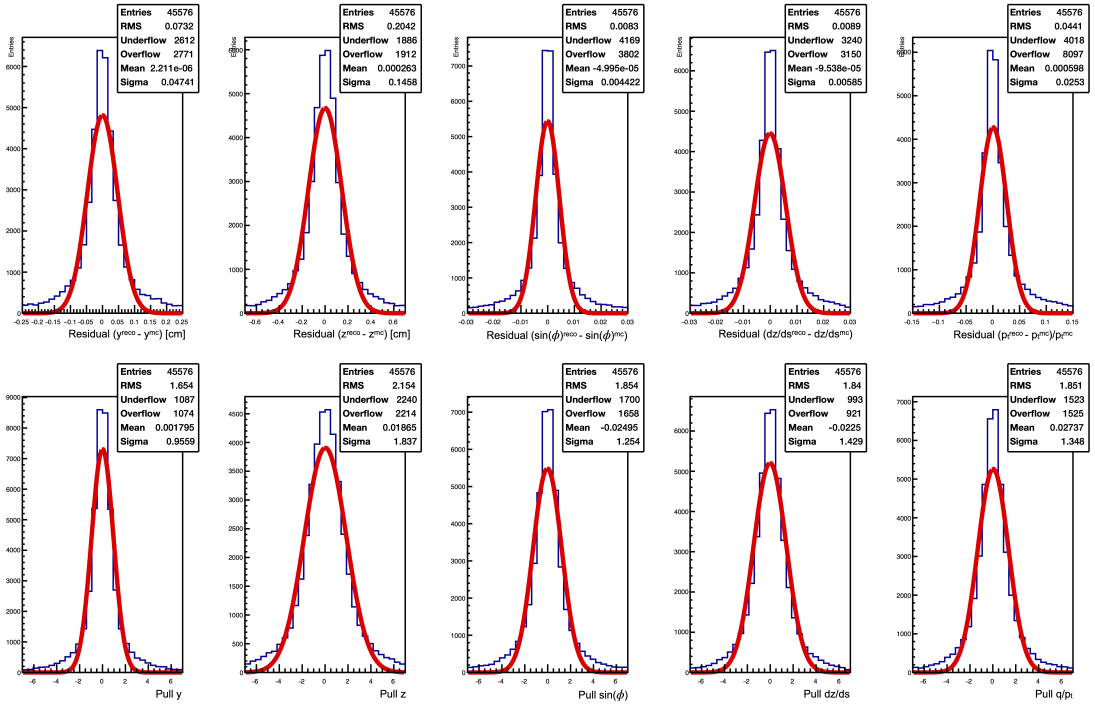


Figure 6.5: Fit quality of the tracks reconstructed by the upgraded TPC CA Track Finder measured in the last hit.

time, as the track length decreases, the reconstruction efficiency also drops down, reaching a value of less than 40% at the minimum allowable number of hits. The high quality of the track reconstruction is confirmed by the distributions of residuals and pulls (Fig. 6.5). Close-to-Gaussian distributions with sigma values close to or slightly greater than one indicate successful fitting of the found trajectories, taking into account the features of the TPC detector and the simulated data under consideration. Wide Z-coordinate pull with the sigma of about 1.8 stands out among others. The reason for this is the abundance of the tracks with a small slope relative to the Z-axis. In this case, the hits are smeared in Z, and the actual coordinate measurement error is higher than expected.

Integrating the updated tracking algorithm into the offline reconstruction of the real experimental data is a complex and responsible process that requires a detailed study and evaluation of the reconstruction quality. The algorithm results should be as reliable as possible since further physical analysis is based on them, and research is carried out. The previous version of the global offline tracking in the STAR experiment, based on one of the early TPC CA Track

Finder variants, was launched in 2013 as a part of StiCA track reconstruction and has been successfully used until now. Improvements made to the algorithm over the past time and an upgrade of the detecting system made it obvious that it was necessary to replace the CA part of the reconstruction with a more modern version. In 2020, studies and comparisons of tracking with the old and new versions of the TPC CA were carried out [101].

The quality of the global tracking within the STAR experiment is evaluated using special tools that differ from those used in developing the stand-alone version of the tracker. These tools are aimed at a deeper study of the results from the physical analysis point of view and not just the logic of the reconstruction algorithm. So, in this case, other criteria are used to select tracks, which leads to numerical results different from those demonstrated earlier. In order to reconstruct a track, there must be 15 or more hits to be combined and fit. The fitting of the tracks obtained in the TPC CA is carried out using the fitting algorithm of the Sti software package, originally intended for the reconstruction using the track following method. Unlike the simplified TPC CA fit, the Sti fit allows to more accurately assess the belonging of each hit to the track and is also capable of fully operating with loopers.

The algorithms were compared with several versions of the StRoot software package used for reconstruction and analysis in the STAR experiment. SL19x and SL20x are official software releases that include the older version of the TPC CA — depicted as the empty square bars in the histograms. TFG20x — releases based on the new TPC CA used in the express production and HLT, shown as the filled circles in the plots. The reconstruction results for particles with a negative charge are shown in red, and for a positive charge — in black.

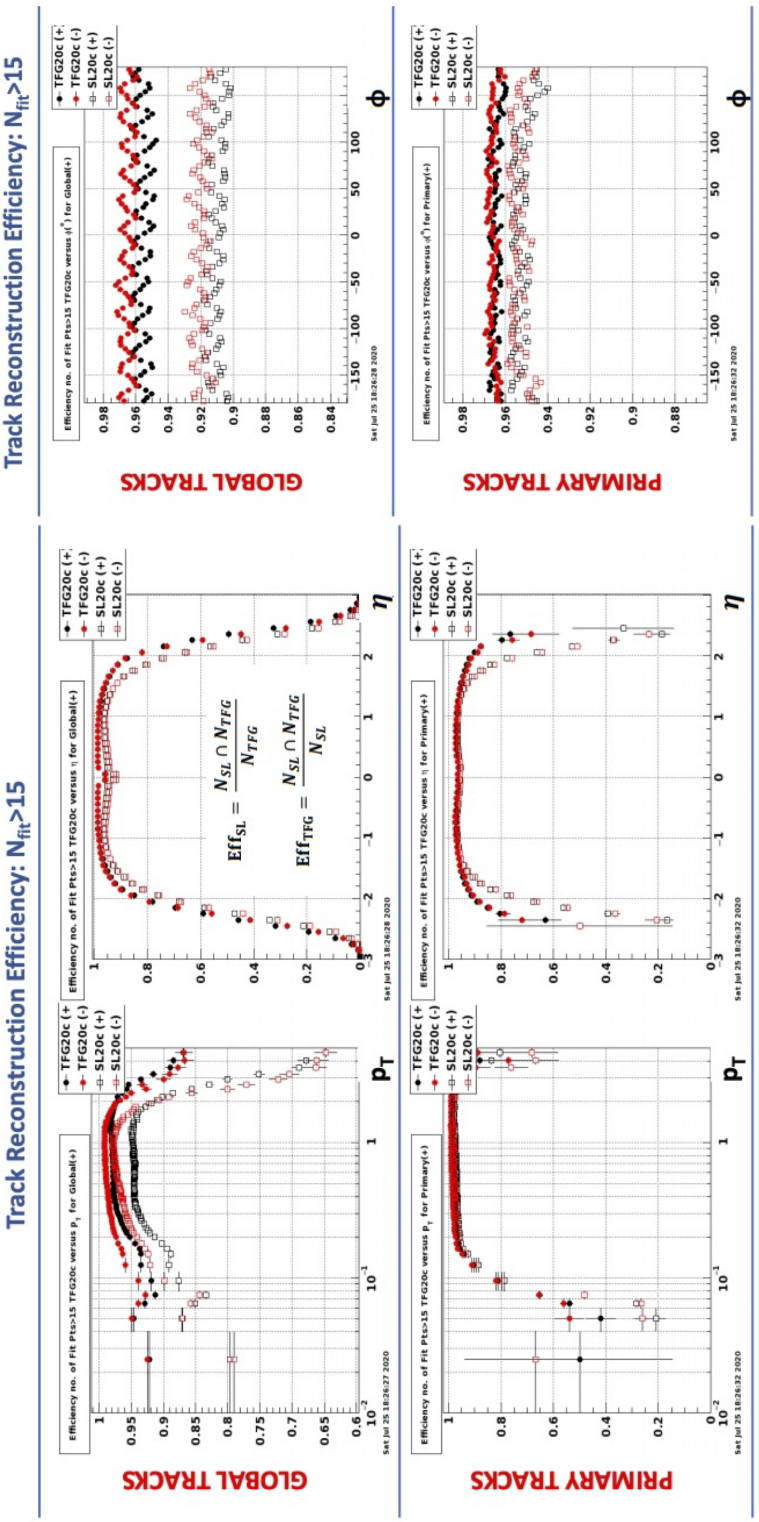


Figure 6.6: Comparison of the efficiency of track reconstruction algorithms, based on the old and new versions of the TPC CA Track Finder, when processing Run20 data obtained in 2020. The relative efficiencies are given depending on the transverse momentum $p_T$, angles $\eta$ and $\phi$ for primary and global tracks [101].

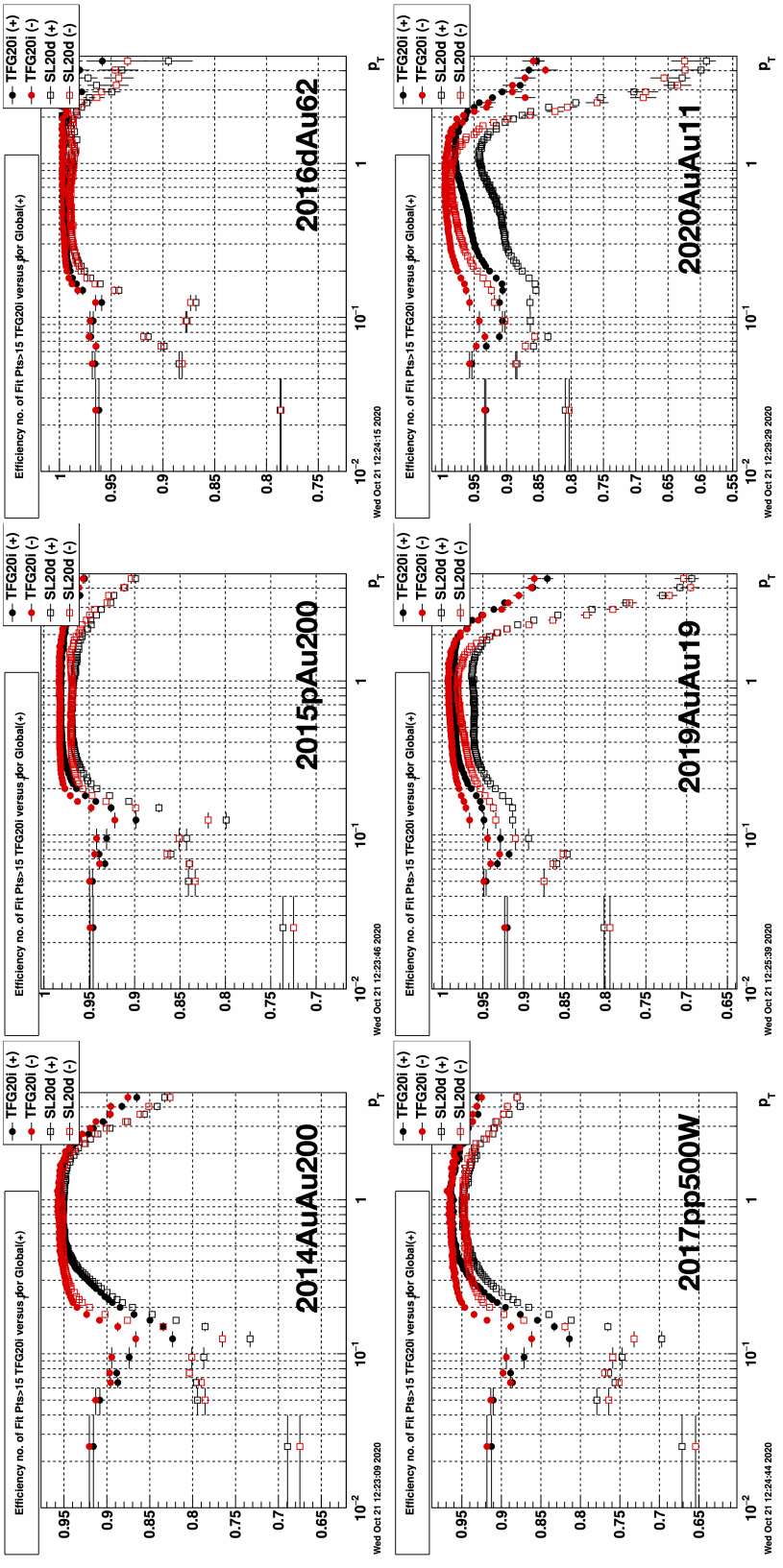


Figure 6.7: Comparison of the efficiency of track reconstruction algorithms that include the old and new versions of the TPC CA Track Finder when processing data from different years, with different energies and different collision systems. The relative efficiencies are given depending on the transverse momentum $p_T$ for global tracks.

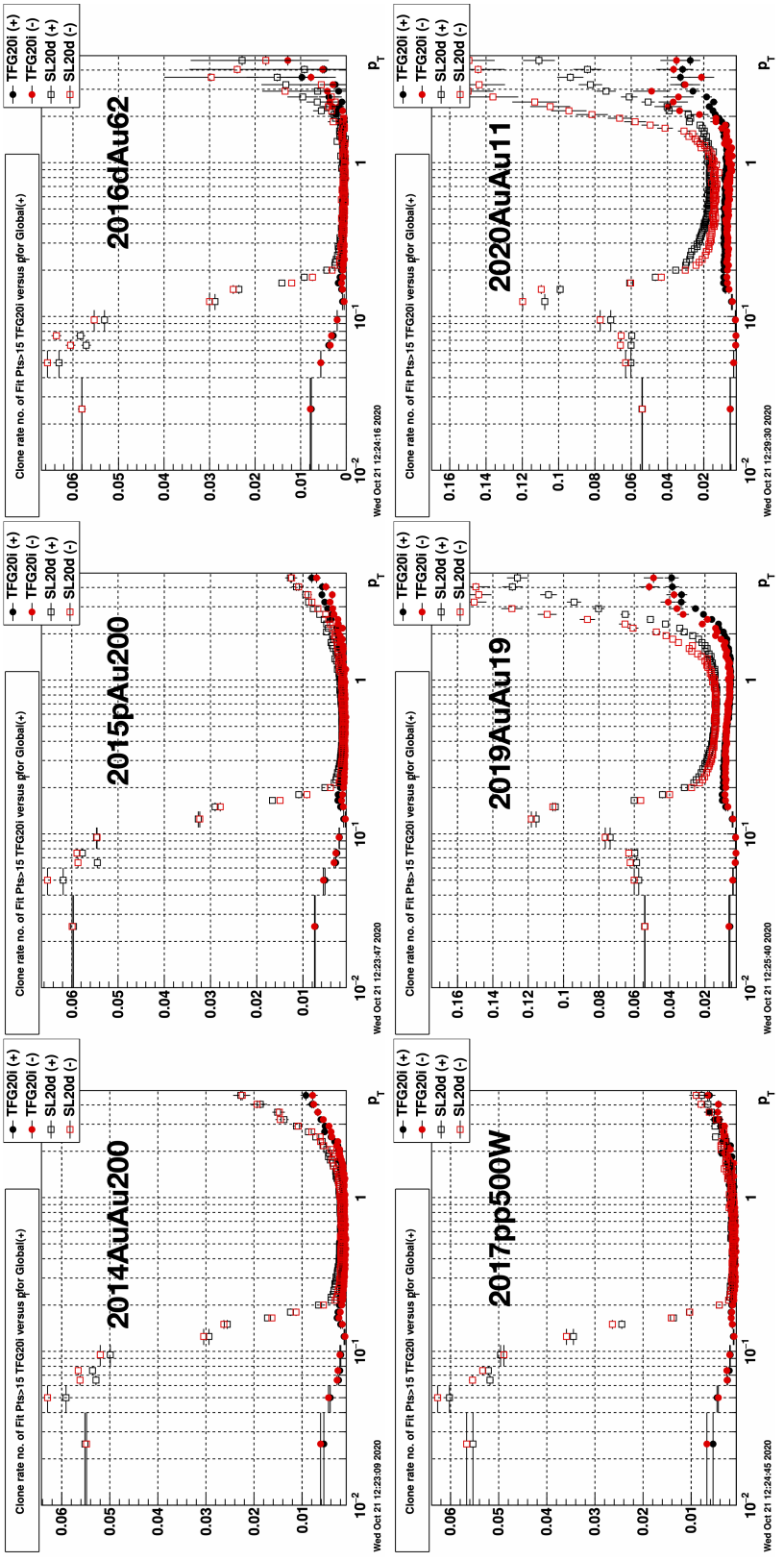


Figure 6.8: Comparison of the clone rate of track reconstruction algorithms that include the old and new versions of the TPC CA Track Finder when processing data from different years, with different energies and different collision systems. The relative clone rates are given depending on the transverse momentum $p_T$ for global tracks.

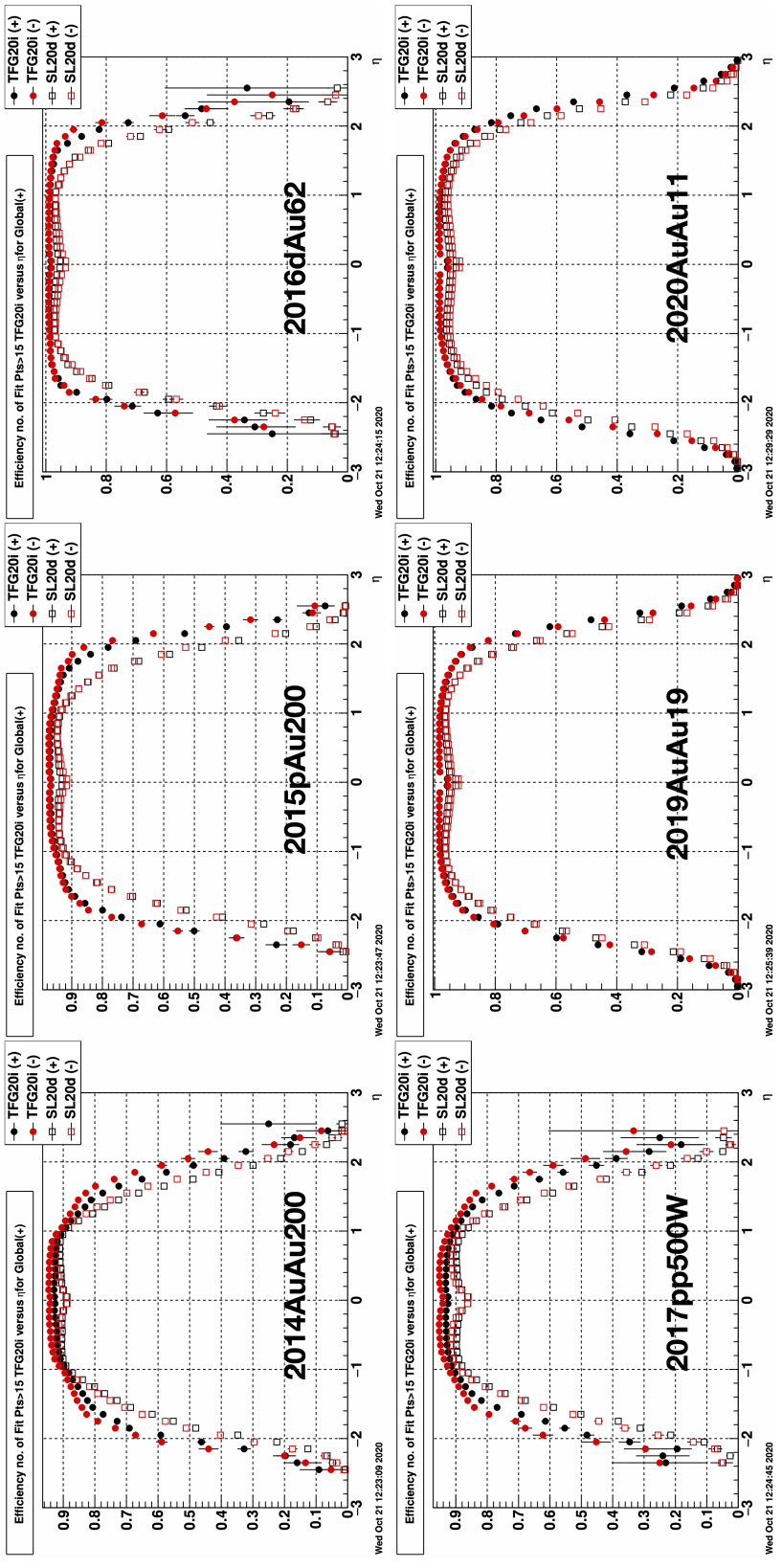


Figure 6.9: Comparison of the efficiency of track reconstruction algorithms that include the old and new versions of the TPC CA Track Finder when processing data from different years, with different energies and different collision systems. The relative efficiencies are given depending on the pseudorapidity $\eta$ for global tracks.

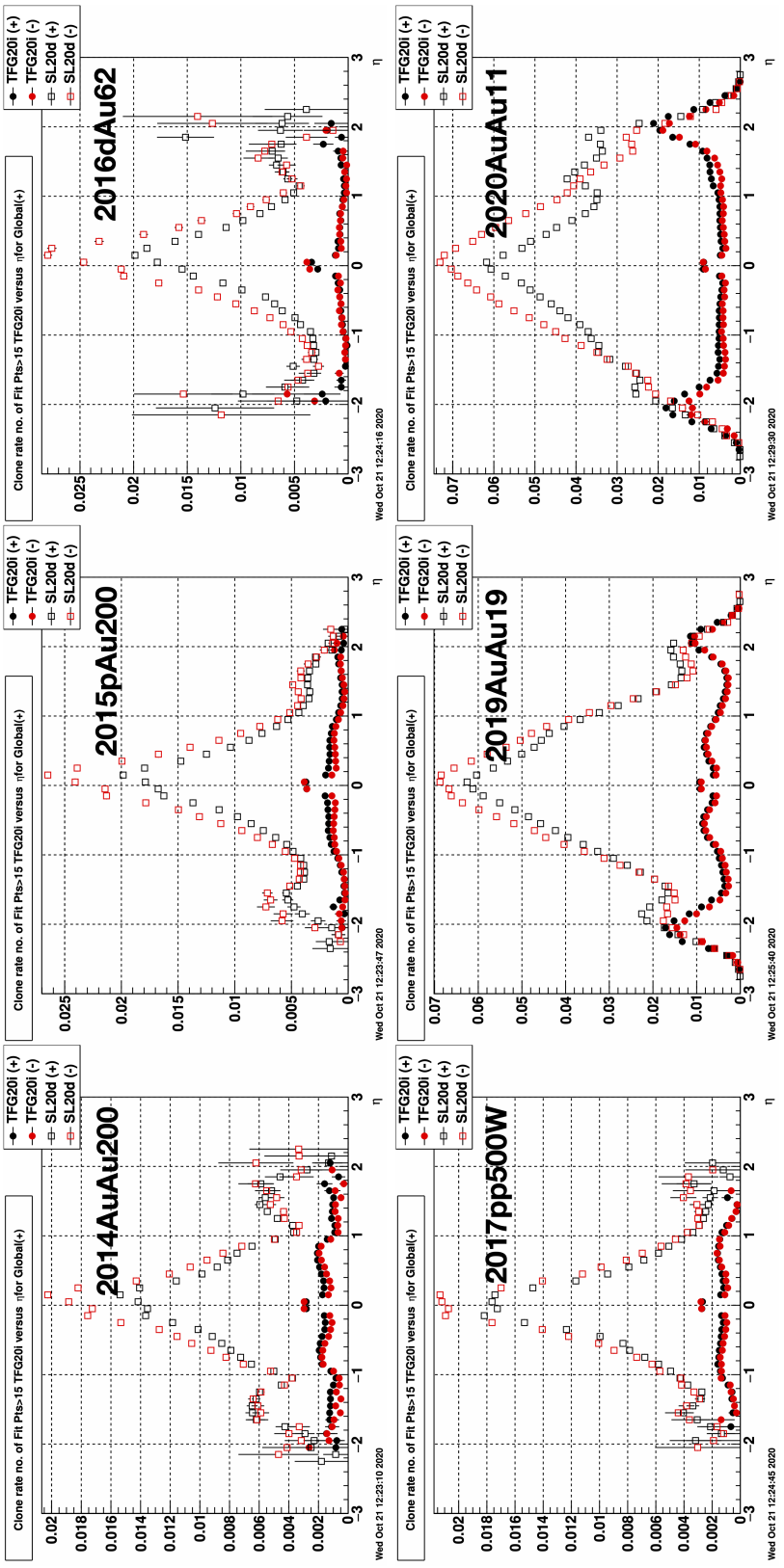


Figure 6.10: Comparison of the clone rate of track reconstruction algorithms that include the old and new versions of the TPC CA Track Finder when processing data from different years, with different energies and different collision systems. The relative clone rates are given depending on the pseudorapidity $\eta$ for global tracks.

Figure 6.6 demonstrates the difference in the relative efficiency of the considered tracking options depending on the transverse momentum $p_T$ and the angles $\eta$ and *phi*. The study was carried out using data obtained in 2020 during Run20 in Au+Au collisions at energies of 9.2 GeV. Estimates of the actual efficiencies for the real data cannot be made, but the comparison makes it possible to understand the differences in the algorithms. As you can see from the above plots, the updated TPC CA Track Finder allows us to find more tracks in almost all the ranges reviewed. The smallest difference in the efficiency — 1–1.5% with the same $\phi$ — is typical for primary tracks. Coming directly from the vertex, they usually have higher energy and straight trajectories, making it easier to find them, regardless of the algorithm used. For all global tracks, the difference in efficiency for the same values of the angle $\phi$ grows to 3–5%. The difference is noticeable practically over the entire measurement range. Still, the greatest differences are characteristic for very low values of $p_T$, as a result of successful reconstruction of loopers, and for high $\eta$.

In addition to the most recent data, earlier results were used in testing, including various years, energies, and collision systems. As can be seen from Figure 6.7, the nature of the results differs depending on the experiment's specifics, but the superiority of the new tracker version remains unchanged in each of the variants. The above results demonstrate a qualitative leap in the efficiency of the particle trajectories reconstruction, especially in the region of very small momenta, which became possible due to the TPC CA Track Finder update.

It should also be noted that the reconstruction efficiency of tracks with the highest momentum has increased. This effect is most noticeable at low collision energies when the probability of the formation of high-energy particles goes down. The loss of each of these tracks becomes more noticeable in comparative analysis. To better understand the reasons for the change in efficiency, one should also consider the clone rate in each case, which indicates reconstruction problems associated with the formation of several disconnected segments related to the same real track. It is not possible to use Monte Carlo information for real experimental data, so the relative clone rate is considered. In other words, when one track from one reconstruction is associated with two or more tracks from another reconstruction.

As can be seen from the graphs shown in Figure 6.8, the highest clone level when using the old version of the algorithm falls on particles with momentum less than 0.2 GeV. These mainly include loopers that form several turns inside

the detector. The new looper merger allows multiple such short segments to be combined into complete tracks, clearing numerous clones and improving reconstruction efficiency. The effect of this procedure is clearly visible in all the cases under consideration, but it becomes especially important for the data collected in 2019–2020, after the transition to the new iTPC geometry. The greater number of pad rows increased the reconstructability of the looper segments, filling the inner part of the detector with short clone tracks and making it difficult to physically analyze the results. The test results indicate that the developed algorithm for combining such segments successfully fulfills its task.

The lowest clone level is characteristic of particles with energies from 0.2 to 1 GeV. This figure tends to zero when using the old TPC geometry and grows slightly with the increasing number of pad rows. This is due to the forthcoming possibility of finding those tracks in the inner part of the detector, which previously simply could not leave enough hits. In such conditions, the updated tracker allows both to better restore such trajectories at the sector tracking stage and combine them more successfully during merging.

And finally, one more area of clone formation is energies greater than 1 GeV. This problem also becomes more noticeable when using the new geometry due to the more complete reconstruction in the inner part of the detector. At the same time, the merging of high-momentum tracks is difficult due to the peculiarities of the Kalman Filter used for fitting, which performs calculations with single precision. This is enough to fit the tracks, but when the vector of parameters is rotated into the coordinate system of the neighboring sector, the errors in determining the momentum can increase significantly, which leads to the rejection of pairs of merging candidates, even if they are part of the same track. This problem was solved algorithmically in the new version of the algorithm without complicating the KF. This made it possible to achieve an increase in efficiency and a reduction in clone level without the additional cost of time.

Based on the data shown in Figure 6.6, the dependence of efficiency on $\phi$ demonstrates the superiority of the new algorithm but does not show its significant features. At the same time, the distributions of efficiency versus pseudorapidity $\eta$ have a more interesting structure. Figure 6.9 shows the efficiency distributions depending on $\eta$ for the data of different years and collision systems, corresponding to those in the previous images. The use of the updated algorithm made it possible not only to increase the overall computational efficiency for almost all values of $\eta$ but also to expand the range of reconstruction in the

regions of the minimum inclination angle relative to the beamline. The changes are especially noticeable at higher collision energies at which a larger number of reconstructed particles with a small deflection angle are produced. Looking at the corresponding histograms of the clone ratio (Fig. 6.10), we can see that with the updated tracker, most clones in the area of small $\eta$ disappeared. Low values of pseudorapidity are typical for loopers' coils. Thus, changes in the central part of the histograms characterize, in general, the work of the looper merger. You can also notice an increase in the number of clones with values of $\eta$ about $\pm 2$, associated with the complexity of merging such tracks. The significant gap between the segments and the inaccurate determination of the momentum of such tracks make it difficult to merge, regardless of the algorithm version used. The decrease in efficiency and additional clone rate at near-zero pseudorapidity are associated with the division of the detector into the Western and Eastern parts. If a track that is practically perpendicular to the beamline hits the central region of the detector, it can make it impossible to register or combine it using the merger.

Despite some peculiarities, the given comparisons indicate that the updated track reconstruction algorithm based on the Cellular Automaton is superior to its predecessor in terms of efficiency and reconstruction quality in all considered situations. It allows particle trajectories and loopers to be stably reconstructed while remaining fast enough to be used even in the online data processing.

As mentioned earlier, in order to expand the functionality of the High-Level Trigger of the STAR experiment, it was supplemented with a chain of algorithms that make it possible to perform fast reconstruction and physical analysis of the data in real-time using the power of the HLT computer farm. This mechanism, which allows both to control the quality of the received data and to evaluate the expected physical results, is called eXpress production. (Fig. 6.11). Express data calibration is performed using the same algorithms as the final calibration, which allows you to use the preliminary results later, speeding up the offline calibration process. On the other hand, it should be noted that the process of adopting algorithms for use in express production is much easier than in offline reconstruction. This allows the implementation to use more advanced versions of software products as they are developed. So, the KF Particle Finder package, which performs physical analysis of the events, in the HLT is more advanced in relation to the offline version and significantly surpasses it in terms of functionality, including such useful features as the Missing Mass Method [102, 103], which makes it possible searching of the short-lived particles which decay into a charged fragment

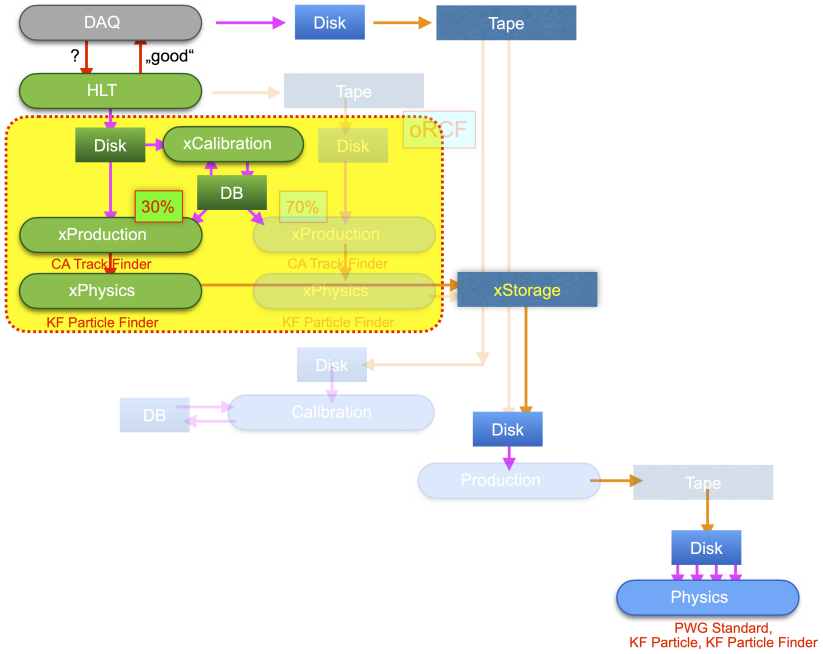


Figure 6.11: Scheme of online data analysis in their express processing on the HLT cluster of the STAR experiment [102].

and a neutron. In this case, the qualitative reconstruction of short tracks of the mother particle and the decay product is very important. Since the neutron does not carry a charge, the trajectory of the daughter particle will have a slight deviation from the mother. Accordingly, there is a risk of erroneous combination such tracks into one during the reconstruction or merging procedure. The results presented in the above articles indicate that the updated CA tracker successfully coped with the task, providing high-quality reconstruction even in rather difficult situations.

The updated TPC CA Track Finder, which includes the improvements and enhancements described above, was also included in the express production reconstruction chain and was successfully tested in the processing of the real experimental data. Despite the incomplete calibration and alignment of the detector, carried out in a fast mode, the efficiency and accuracy of the reconstruction proved to be sufficient for carrying out physical analysis and stably finding the particles of interest. It should be noted that express production uses the capacity of the HLT cluster to a limited extent since fast reconstruction should not interfere with the execution of the main trigger function — tracking particle col-

lisions and initiating data collection. But, even in cramped conditions, the high speed of the algorithms used, including the tracker, allows processing about 30% of incoming events online. If it is necessary to increase the volume of quickly analyzed data or conduct additional research on the express production results, the information can be copied to a more powerful RCF computing farm for the so-called fast offline data processing using the standard reconstruction chain.

Using the same or very similar algorithms in the express production and the standard offline production leads to the fact that preliminary data can serve as a reliable estimate of the expected final result that should be obtained after completing the full calibration, reconstruction, and analysis.

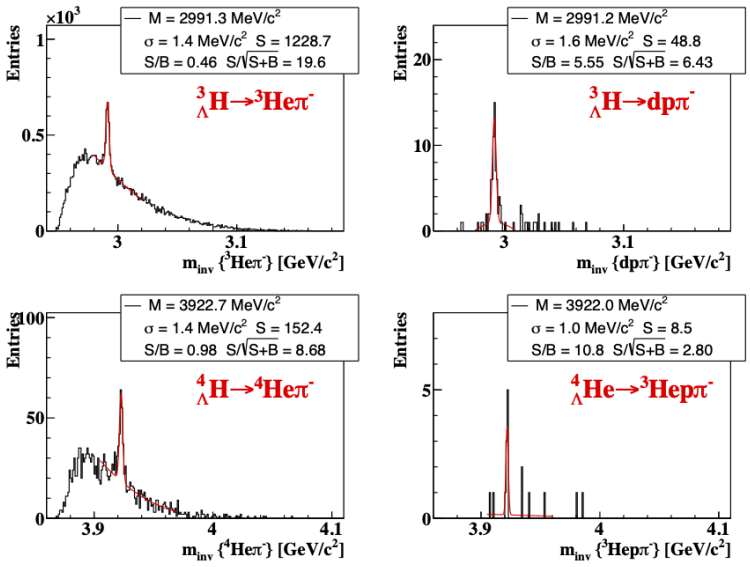


Figure 6.12: Distribution of invariant masses of hypernuclei decays ${}^{3}_{\Lambda}H$, ${}^{4}_{\Lambda}H$ and ${}^{4}_{\Lambda}He$, obtained as a result of the analysis of 140 million collisions in the TPC detector working in the barrel mode at the energy of $\sqrt{s_{NN}}$=7.7 GeV using express production in 2021 [104].

The express production mechanism, which includes updated algorithms for track reconstruction and physical analysis, has already made it possible to obtain significant physical results with a high degree of reliability. Particular attention was paid to the search and study of hypernuclei — nucleus-like systems containing hyperons in addition to protons and neutrons. The particle search results during the fast reconstruction and analysis of the express production are shown in Fig-

ure 6.12. The data were obtained as a result of fast online processing of about 140 million Au+Au collisions with an energy of $\sqrt{s_{NN}}$=7.7 GeV in the standard barrel mode of the TPC detector. It should be noted that for the results to be recognized as reliable, significance — the ratio of $Signal/\sqrt{Signal + Background}$ — must be greater than 5, where $Signal$ is the histogram inputs, presumably corresponding to the desired particle, and $Background$ — other particles. Thus, even when analyzing less than 30% of the data obtained, it was possible to accurately trace the decays of ${}^{3}_{\Lambda}H \rightarrow {}^{3}He\pi^{-}$, ${}^{3}_{\Lambda}H \rightarrow dp\pi^{-}$ and ${}^{4}_{\Lambda}H \rightarrow {}^{4}He\pi^{-}$ whose significance values exceed the minimum required value. The significance of the much rarer ${}^{4}_{\Lambda}He \rightarrow {}^{3}Hep\pi^{-}$ decay is 2.8, which is not sufficient for stable particle fixation and must be confirmed by offline data analysis with full statistics.

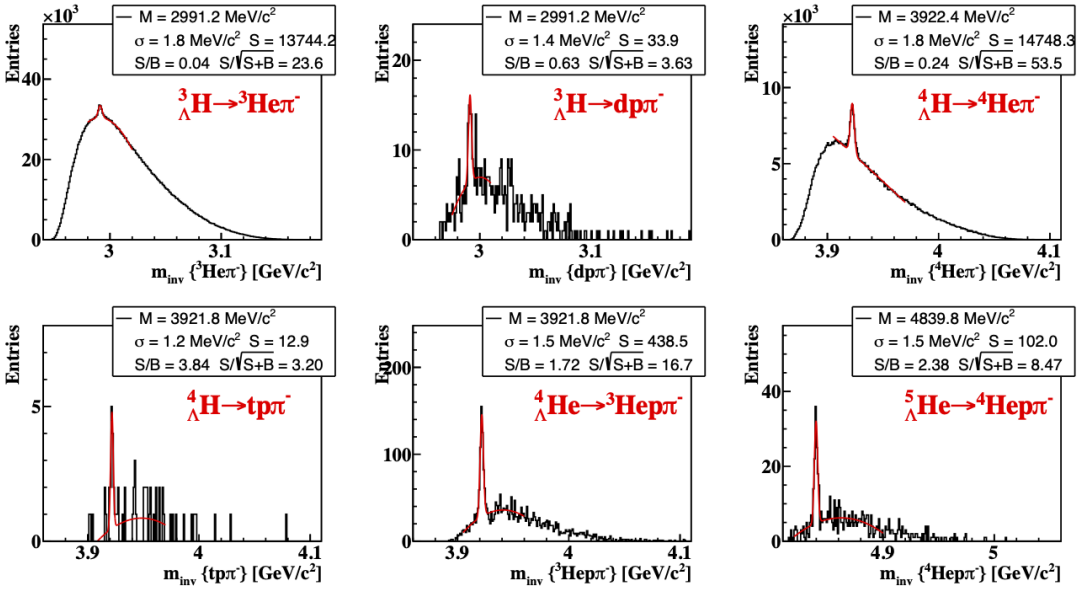


Figure 6.13: Distribution of the invariant masses of hypernuclei decays ${}^{3}_{\Lambda}H$, ${}^{4}_{\Lambda}H$, ${}^{4}_{\Lambda}He$ and ${}^{5}_{\Lambda}He$ obtained from the analysis of 393 million collisions in the TPC detector working in the fixed target mode at the energy of 3.85$A$ GeV using express production in June 2021 [104].

Reducing the beam energy at the RHIC facility is a difficult task. Implementation of particle collisions in the mode of colliding beams at $\sqrt{s_{NN}}$<7.7 GeV was not possible. Therefore, within the framework of the Beam Energy Scan II program, the detection system of the STAR experiment was improved to operate in the fixed target (FXT) mode. Thus, it was possible to reduce the collision

energy to 3.85$A$ GeV, which corresponds to the single beam. In addition, in June 2021, the interaction rate was increased from 2 to 3 kHz to obtain the maximum statistics in the minimum time. Considering that the TPC detector's operation speed is about 800 Hz, this significantly increased the pileup level. This, in turn, increased the loading of the tracking algorithm, complicating the reconstruction process. Due to the peculiarities of the detector's functioning — electron drift in the gas — the pileup tracks of the different events mostly do not overlap each other but shift along the Z axis, increasing the multiplicity of particles per event. Many pileup tracks are discarded after the reconstruction and do not participate in the physical analysis, but they still participate in the track searching process.

Despite the difficult reconstruction conditions and the non-standard detector operation mode, the improved TPC CA Track Finder proved to be excellent while maintaining high computation speed and efficiency rates. This is evidenced by the results of the physical data analysis using the KF Particle Finder software package as part of the express production running on the HLT computer farm. About 393 million of the 17 billion collisions registered were processed online. As well as in the case of operating in the barrel mode, fast express production algorithms were able to successfully reconstruct and distinguish several hypernucleus decays of high scientific interest. As can be seen from the distributions of the invariant mass shown in Figure 6.13, fast reconstruction is still able to provide a stable physical result, despite the increase in the background, which is especially noticeable during the ${}^{3}_{\Lambda}H \rightarrow {}^{3}\ He\pi^{-}$ and ${}^{4}_{\Lambda}H \rightarrow {}^{4}\ He\pi^{-}$ decays. The growth in the statistics dramatically increases the significance of the previously considered decays and makes it possible to trace other variants that should be confirmed after analyzing a larger number of events.

The presented results clearly indicate the high quality of the express reconstruction, which makes it possible to perform the successful physical analysis of the real experimental data rather than simulations [105]. This proves the considered approach effectiveness, at least for processing the results of collisions at the energies typical for the STAR BES-II and CBM experiments. Thus, on the example of the ongoing experiment, the reliability of the concept underlying the CBM FLES is confirmed. This is an important step in the transition to real-time data processing in HEP, which should provide scientists with many new opportunities to observe and study extremely rare particles and phenomena.

Starting in 2021, the online Event Display (Fig. 6.14) has also been transferred to a demonstration of events reconstructed using the TPC CA Track Finder.

Previously, a simpler track search method based on conformal mapping was used for this task, which allows reconstructing only the primary tracks. Event Display demonstrating the complete reconstruction of particle trajectories in the TPC detector of the STAR experiment in real-time is available to everyone on the official website of the experiment [106].

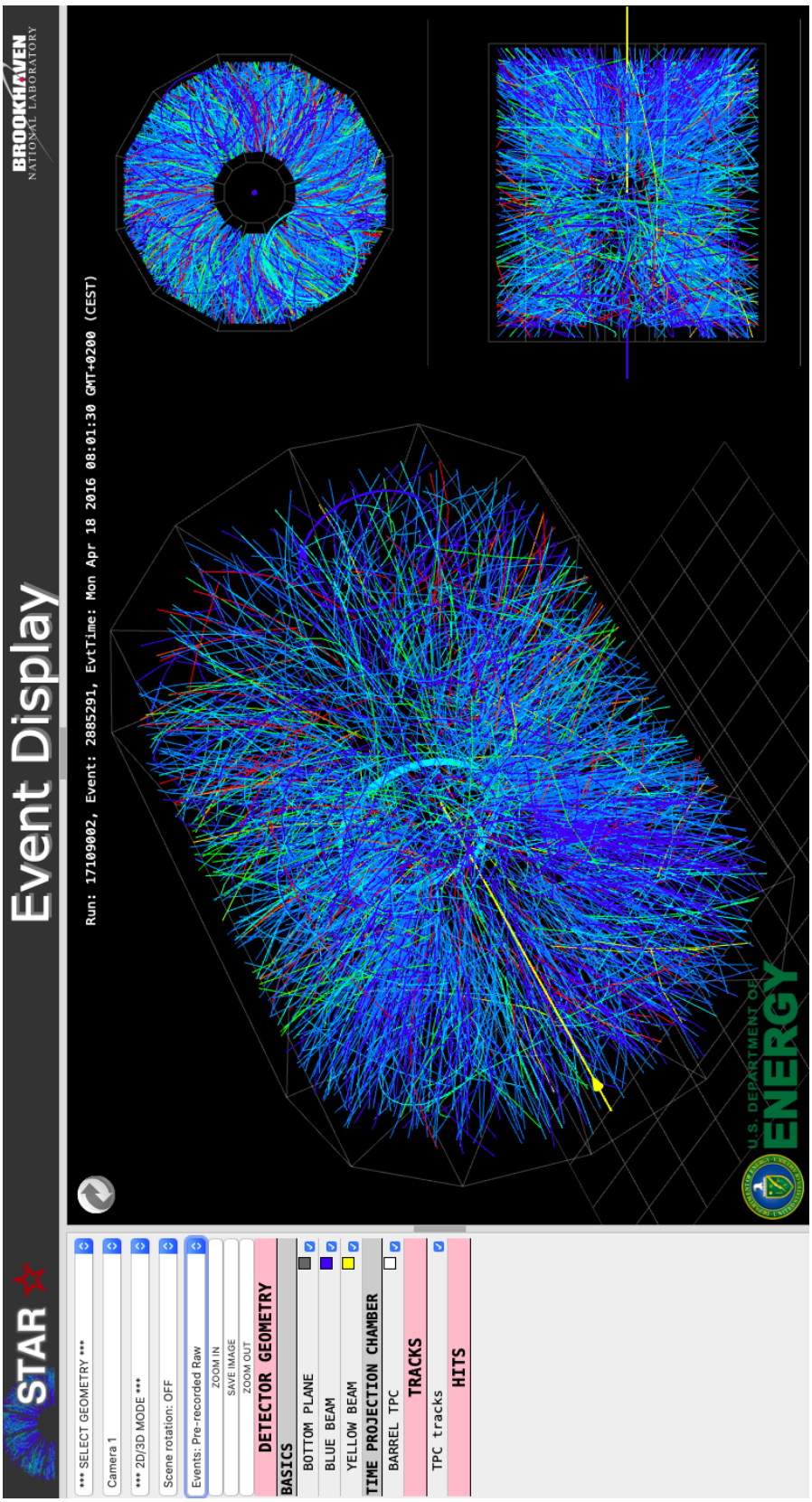


Figure 6.14: TPC CA Track Finder based Online Event Display in the STAR experiment [106].

# Chapter 7

# Summary

Charged particle trajectories reconstruction is one of the most important and most complicated stages of processing the results of experiments in High Energy Physics. Growing the experiments' complexity and the transition to the study of increasingly rare and difficult-to-register phenomena, an increase in requirements for algorithms of reconstructing and analyzing the obtained data entails. In such conditions, the existing software products meet the requirements worse and need to be improved or replaced.

Increasing the number of particles per collision creates a demand for algorithms that can maintain high quality and efficiency of reconstruction regardless of the tracks' multiplicity. At the same time, a rise in the particle interaction speed and events reading leads to the accumulation of an increasing amount of data that must be processed in a limited time. This makes the high speed of reconstruction algorithms, especially tracking, a critical condition for the success of the experiment, which cannot be achieved without cutting-edge high-performance computing technologies usage.

The abundance of incoming data brings to a new level the importance of processing results in real-time. Express production in the STAR experiment makes results evaluation possible with high accuracy directly during the experiment and, if necessary, adjust the conditions for conducting it. At the same time, the CBM experiment currently being prepared cannot be successfully carried out at all without complete transfer of the reconstruction process online since the expected speed of data taking will exceed the equipment's capabilities to save and store them.

Track reconstruction algorithms based on the Cellular Automaton are an ideal

solution for data processing in modern HEP experiments. They combine both the original high speed and efficiency and the wide possibilities for parallelizing calculations, using the capabilities of up-to-date computer systems.

The thesis considers two different approaches to track reconstruction using CA, which correspond to the features of the detecting systems used in the STAR experiment. These are the HFT CA Track Finder, similar to the CBM experiment tracker and designed for reconstruction in detectors with an explicit station structure. The TPC CA Track Finder is applicable to the TPC detectors family and is based on the ALICE experiment's CA tracker. The STAR experiment has an established solution for online data analysis, the High-Level Trigger. In addition, the Beam Energy Scan-II program assumes particle collisions at energies similar to the CBM experiment. All this makes this platform an ideal testing ground for real-time reconstruction feasibility testing under the FAIR Phase-0 program.

Work on the HFT tracker was focused on maximizing its calculation speed while maintaining a high reconstruction efficiency. As a result, its speed was increased hundreds of times. When improving the program, various techniques for optimization such reconstruction algorithms were applied and studied in detail. In addition to classical approaches, new solutions have been developed that now allow effectively using the capabilities of CPUs concerning parallelism at the data level. Work on the algorithm was stopped after the HFT was excluded from the detection chain of the STAR experiment. Still, the results achieved allow, if necessary, to reconfigure the track finder for work with other detectors with a similar barrel configuration with minimal labor.

The TPC CA Track Finder in the STAR experiment was initially developed for use in online reconstruction as part of the High-Level Trigger. Therefore, the work on the algorithm was focused on increasing the quality of tracking while maintaining a high computational speed. As part of this task, several shortcomings were corrected. The Track Finder has been timely optimized for the new conditions after upgrading the inside of the iTPC detector. This made it possible not only to save but also to improve the quality of the track reconstruction, keeping the speed of calculations at a level acceptable for working in the online mode.

In addition to the development of the existing functionality of the tracker, the reconstruction chain was supplemented with a new software module that was not available before. The developed Looper Merger allowed segments of very

low-momentum swirling tracks to be effectively merged. This made it possible to significantly reduce the clone rate, decreasing the level of the background tracks and simplifying the further physical analysis of the events.

The updated TPC CA Track Finder was included in the express production chain and successfully used to reconstruct events in real-time during the run21. The results presented in the dissertation testify to the high quality of the updated tracker even in conditions of increased interaction rate and the presence of the pileup tracks. This confirms both the possibility of successful online reconstruction and physical analysis of events when working with real experimental data and the ability of CA-based algorithms to perform this task qualitatively and quickly.

In addition, the conducted research and comparison of the old and new tracker versions showed a noticeable advantage of the new version over its predecessors, not only when working with the new data, that includes the updated iTPC, but also when reconstructing old archived data of the past years. Therefore, besides the online analysis, the improved tracking algorithm is currently being reviewed for inclusion in the standard offline data processing chain. Thus, the updated TPC CA Track Finder, which is the fast and efficient mechanism for tracking and reconstructing events with any particles multiplicity, will take its place both in online and offline chains of processing the results of the STAR experiment.

# List of Figures

# List of Tables

# Bibliography


[1] Langacker P., "The standard model and beyond," CRC Press (2010) 670

[2] Sakurai J. J., "Theory of strong interactions," Annals of Physic, Vol. 11 (1960) 1–48

[3] Wolfenstein L., "The strenghh of the weak interactions," Annual Review of Nuclear and Particle Science, Vol. 54 (2004) 1–17

[4] Greiner W., Schramm S., Stein E., "Quantum chromodynamics; 2nd ed," Springer Science and Business Media (2002) 551

[5] Aoki Y., Endrodi G., Fodor Z., Katz S. D. and Szabo K. K., "The Order of the quantum chromodynamics transition predicted by the standard model of particle physics," Nature 443 (2006) 675

[6] Collins J. C. and Perry M. J., "Superdense Matter: Neutrons Or Asymptotically Free Quarks?", Phys. Rev. Lett. 34 (1975) 1353

[7] CBM Collaboration, "Challenges in QCD matter physics –The scientific programme of the Compressed Baryonic Matter experiment at FAIR," Eur.Phys.J.A 53 no.3 (2017) 60. T.Galatyuk, NPA982 (2019), update (2021)

[8] Fodor Z. and Katz S. D., "Critical point of QCD at finite T and mu, lattice results for physical quark masses," JHEP 0404 (2004) 9

[9] Sorensen P. and Shen C., "Sketch of relativistic heavy-ion collisions," (2014) `https://u.osu.edu/vishnu/`

[10] https://home.cern/about

[11] ALICE Collaboration, "W and Z boson production in p-Pb collisions at $\sqrt{s_{NN}}$=5.02 TeV," JHEP 02 (2017) 077

[12] ALICE Collaboration, "Measurement of $Z^0$-boson production at large rapidities in Pb-Pb collisions at $\sqrt{s_{NN}}$=5.02 TeV," Phys.Lett.B 780 (2018) 372–383

[13] ALICE Collaboration, "The ALICE experiment at the CERN LHC," JINST 3 (2008) S08002

[14] STAR Collaboration, "The STAR experiment at the relativistic heavy ion collider," Nucl. Phys. A 566 (1994) 277C

[15] STAR Collaboration, "The RHIC Beam Energy Scan program in STARand what's next...", Phys.: Conf. Ser. 455 (2013) 012037

[16] STAR Collaboration, "The STAR BES-II and Forward Rapidity Physics and Upgrades," Nucl. Phys. A 982 (2019)

[17] P.Senger for the CBM collaboration, "Probing dense QCD matter in the laboratory — The CBM experiment at FAIR," Phys.Scripta 95 (2020) 7

[18] RHIC, "The Relativistic Heavy Ion Collider," https://www.bnl.gov/rhic/

[19] B. Mohanty, for the STAR Collaboration, "Results from STAR experiment at RHIC," PRAMANA — journal of physics, Vol. 67, No. 5 November 2006 pp. 927–935

[20] J. Adams *et al*, "Experimental and Theoretical Challenges in the Search for the Quark Gluon Plasma: The STAR Collaboration's Critical Assessment of the Evidence from RHIC Collisions," Nucl.Phys.A757 (2005) 102-183

[21] Nu Xu for the STAR Collaboration, "An overview of STAR experimental results," Nuclear Physics A, Vol. 931, November 2014, Pages 1–12

[22] D. Tlusty, "The RHIC Beam Energy Scan Phase II: Physics and Upgrades," arXiv:1810.04767 (2018)

[23] G. Odyniec for the STAR Collaboration, "Beam Energy Scan Program at RHIC (BES I and BES II) — Probing QCD Phase Diagram with Heavy-Ion Collisions," CORFU2018, 31 August – 28 September 2018

[24] D. McDonald, "Overview of results from phase I of the Beam Energy Scan program at RHIC," EPJ Web of Conferences 95, 01009 (2015)

[25] C. Yang for the STAR Collaboration, "The STAR beam energy scan phase II physics and upgrades," Nuclear Physics A 967 (2017) 800–803

[26] R.L. Brown *et al.*, "The STAR Detector Magnet Subsystem," 1997 Particle Accelerator Conference, Vancouver, USA, May 12–16 1997

[27] W.J. Llope *et al.*, "The STAR Vertex Position Detector," Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Vol:759 (2014) 23–28

[28] W.J. Llope *et al.*,"The TOFp/pVPD Time of Flight System for STAR," Nucl.Instrum.Meth. A, vol. 522 (2004) 252–273

[29] M. Anderson *et al.*, "The STAR time projection chamber: a unique tool for studying high multiplicity events at RHIC," Nucl.Instrum.Meth. A, vol. 499, no. 2-3 (2003) 659–678

[30] H.Matis *et al.*, "STAR Geometry," STAR Note CSN229B (1999), 15

[31] F. Sauli, "Principles of Operation of Multiwire Proportional and Drift Chambers," Experimental Techniques in High Energy Physics (1987) 79–188

[32] F. Shen, S. Wang, C. Yang and Q. Xu, "MWPC prototyping and testing for STAR inner TPC upgrade," Journal of Instrumentation, v. 12(06) (2017)

[33] B. Bonner *et al.*, "A single Time-of-Flight tray based on multigap resistive plate chambers for the STAR experiment at RHIC," Nucl. Instr. Meth. A 508 (2003) 181

[34] STAR Collaboration, M. Beddo *et al.*, "The STAR barrel electromagnetic calorimeter," Nucl. Instrum. Meth. A499 (2003) 725–739

[35] L.Ruan *et al.*, "Perspectives of a mid-rapidity dimuon program at the RHIC: a novel and compact muon telescope detector," Journal of Physics G: Nuclear and Particle Physics, vol. 36, no. 9 (2009)

[36] W.J. Llope, "Multigap RPCs in the STAR experiment at RHIC," Nucl. Instrum. Meth. A 661 (2012) 110–113

[37] D. Beavis *et al.*, "The STAR Heavy Flavor Tracker Technical Design Report," https://drupal.star.bnl.gov/STAR/starnotes/public/sn0600

[38] G. Contin *et al.*, "The STAR MAPS-based PiXeL detector," Nucl. Instrum. Meth. A 907 (2018) 60–80

[39] J. Schambach *et al.*, "The STAR Heavy Flavor Tracker (HFT)," DOI: http://dx.doi.org/10.3204/DESY-PROC-2014-04/83

[40] L. Arnold *et al.*, "The STAR silicon strip detector (SSD)," Nucl. Instrum. Meth. A 499 (2003) 652–658

[41] E.G. Judd *et al.*, "The evolution of the STAR Trigger System," Nucl. Instrum. Meth. A 902 (2018) 228–237

[42] F.S. Bieser *et al.*, "The STAR trigger," Nucl. Instrum. Meth. A 499 (2003) 766–777

[43] J.SLange *et al.*, "The STAR level-3 trigger system," Nucl. Instrum. Meth. A 453 (2000) 397–404

[44] P. Yepes, "A fast track pattern recognition," Nucl. Instrum. Meth. A 380 (1996) 582–585

[45] A.Tang, "HLT and CA Tracker," CBM-STAR Joint Workshop, CCNU Wuhan, September 23, 2017

[46] M. Zyzak, "Online selection of short-lived particles on many-core computer architectures in the CBM experiment at FAIR," Doctoral Thesis, Johann Wolfgang Goethe-Universitat, 2016

[47] Bruce H. McCormick, "The Illinois Pattern Recognition Computer-ILLIAC III," IEEE Transactions on Electronic Computers ( Volume: EC-12, Issue: 6, Dec. 1963)

[48] T. Pabst, "The Intel Pentium II ('Klamath') CPU," Tom's Hardware, March 1, 1997

[49] M. J. Flynn, "Some Computer Organizations and Their Effectiveness," IEEE Transactions on Computers, vol. C-21, no. 9 (1972) 948

[50] M. Mittal, A. Peleg, U. Weiser, "MMX Technology Architecture Overview," Intel Technology Journal (1997)

[51] G. Kroah-Hartman, A. Rubini, J. Corbet, "Chapter 5. Concurrency and Race Conditions," Linux Device Drivers, 3rd Edition, O'Reilly Media (2005)

[52] John von Neumann, "First Draft of a Report on the EDVAC," University of Pennsylvania (1945)

[53] J. Mellor-Crummey, "Bitonic Sort," Rice University

[54] T. Cormen, C. Leiserson, R. Rivest, C. Stein, "Introduction to Algorithms 3rd Edition," The MIT Press (2009)

[55] https://www.intel.com/content/www/us/en/products/details/processors/movidius-vpu.html

[56] https://en.wikichip.org/wiki/movidius/microarchitectures/shave_v2.0

[57] https://www.research.ibm.com/articles/brain-chip.shtml

[58] https://en.wikichip.org/wiki/intel/loihi

[59] E. Paxon Frady *et al.*, "Neuromorphic Nearest-Neighbor Search Using Intel's Pohoiki Springs," arXiv:2004.12691, (2020)

[60] https://www.nvidia.com/en-gb/data-center/volta-gpu-architecture/

[61] https://www.anandtech.com/

[62] https://www.nvidia.com/en-us/data-center/a100/

[63] https://www.nvidia.com/en-us/data-center/ampere-architecture/

[64] https://www.nvidia.com/en-us/data-center/dgx-a100/

[65] https://www.amd.com/system/files/documents/amd-cdna-whitepaper.pdf

[66] https://www.amd.com/en/products/server-accelerators/instinct-mi100

[67] N. P. Jouppi, "In-Datacenter Performance Analysis of a Tensor Processing Unit," ACM SIGARCH Computer Architecture News, Volume 45, Issue 2, May 2017, 1–12

[68] https://en.wikichip.org/wiki/nervana/nnp/nnp-t_1300

[69] https://habana.ai

[70] https://en.wikichip.org/wiki/habana/microarchitectures/goya

[71] https://en.wikichip.org/wiki/habana/microarchitectures/gaudi

[72] https://www.mobileye.com/our-technology/evolution-eyeq-chip/

[73] https://www.microsoft.com/en-us/research/blog/second-version-hololens-hpu-will-incorporate-ai-coprocessor-implementing-dnns/

[74] https://apt.cs.manchester.ac.uk/projects/SpiNNaker/

[75] https://www.kip.uni-heidelberg.de/vision/previous-projects/facets/neuromorphic-hardware/waferscale-integration-system/hicann/

[76] https://web.stanford.edu/group/brainsinsilicon/neurogrid.html

[77] https://aws.amazon.com/ru/machine-learning/inferentia/

[78] https://en.wikichip.org/wiki/arm_holdings/microarchitectures/mlp

[79] https://fuse.wikichip.org/news/2721/alibaba-launches-dc-inference-accelerators/

[80] https://github.com/hollance/neural-engine

[81] https://www.qualcomm.com/products/artificial-intelligence

[82] https://news.samsung.com/global/samsung-electronics-introduces-a-high-speed-low-power-npu-solution-for-ai-deep-learning

[83] https://www.huawei.com/en/news/2019/8/huawei-ascend-910-most-powerful-ai-processor

[84] https://en.wikichip.org/wiki/tesla_(car_company)/fsd_chip

[85] M. Kretz, "Vector classes," https://github.com/VcDevel/Vc

[86] The OpenMP API specification for parallel programming https://www.openmp.org

[87] POSIX threads programming https://computing.llnl.gov/tutorials/pthreads

[88] M. Hansroul, H. Jeremie, and D. Savard, "Fast Circle Fit With The Conformal Mapping Method," Nucl. Instrum. Methods Phys. Res., A 270 (1988) 498–501

[89] P.V.C. Hough, "Machine Analysis Of Bubble Chamber Pictures," 2nd International Conference on High-Energy Accelerators and Instrumentation, HEACC 1959 : CERN, Geneva, Switzerland, September 14–19, 1959, 554–558

[90] R. Mankel, "Application of the Kalman Filter Technique in the HERA-B Track Reconstruction," HERA-B public note (1995)

[91] https://geant.org

[92] H. Jung, "Simulations in high energy physics, summerstudent lecture," DESY, Hamburg, Germany, Aug 2008

[93] S. Wolfram, "Statistical Mechanics of Cellular Automata," Reviews of Modern Physics, 55 (1983) 601–644

[94] M. Gardner, "Mathematical games: The fantastic combinations of John Conways new solitaire game Life," Sci. Amer., 223 (1970) 120–123

[95] S. Gorbunov *et al.*, "Fast Cellular Automaton tracker for the ALICE High Level Trigger," Computer Science (2009)

[96] S. Gorbunov *et al.*, "ALICE HLT high speed tracking on GPU," Real Time Conference (RT), 2010

[97] **G.Kozlov**, I.Kisel, "Speed up approaches in CA track finder," 29th CBM Collaboration Meeting, March 2017

[98] **G.Kozlov**, I.Kisel, "Speed up approaches in Cellular Automaton (CA) track finder," MMCP 2017, Dubna, Russia, July 2017

[99] Y. Fisyak, I. Kisel, I. Kulakov, J. Lauret, M. Zyzak, "Track Reconstruction in the STAR TPC with a CA Based Approach," GSI scientific report (2011)

[100] **G. Kozlov**, I. Kisel, M. Zyzak, "Speedup approaches in a TPC Cellular Automaton track finder," CHEP 2018, Sofia, Bulgaria, July 2018

[101] H. Ke, "Tracking group status report," STAR Collaboration Meeting, September 2021

[102] Y. Fisyak, V. Ivanov, H. Ke, I. Kisel, P. Kisel, **G. Kozlov**, S. Margetis, A. Tang, I. Vassiliev, M. Zyzak, "Application of the missing mass method in the fixed-target program of the STAR experiment", EPJ Web of Conferences 251, 04029 (2021)

[103] Y. Fisyak, V. Ivanov, H. Ke, I. Kisel, P. Kisel, **G. Kozlov**, S. Margetis, A. Tang, I. Vassiliev, M. Zyzak, "Missing Mass Method for reconstruction of short-lived particles in the CBM and STAR experiments", Proceedings of the 9th International Conference "Distributed Computing and Grid Technologies in Science and Education" (GRID'2021), Dubna, Russia, July 5-9, 2021

[104] Y. Fisyak, I. Kisel, I. Vassiliev, M. Zyzak, "Hypernuclei in the 2021 Xdata," STAR collaboration meeting, 15 September 2021

[105] Y. Fisyak, V. Ivanov, H. Ke, I. Kisel, P. Kisel, **G. Kozlov**, S. Margetis, A. Tang, I. Vassiliev, M. Zyzak, "Online reconstruction of long- and short-lived particles in the STAR experiment", 20th International Workshop on Advanced Computing and Analysis Techniques in Physics Research, ACAT-2021, Daejeon, South Korea, December 2021 (to be printed)

[106] https://online.star.bnl.gov/aggregator/livedisplay/

# Zusammenfassung

## Motivation

Die Untersuchung der Struktur von Materie auf Mikroebene ist eine der wichtigsten Komponenten des Wissens über die uns umgebende Welt. Die Kenntnis über die Struktur der Materie und der Gesetze der Interaktion ihrer kleinsten Elemente spielt eine grundlegende Rolle sowohl für den technischen Fortschritt als auch für das Verständnis der Gesetze des Universums. Das fundamentale Wissenschaftsgebiet in diesem Bereich ist die Hochenergiephysik (HEP). Die Kollision beschleunigter Teilchen ermöglicht es, verschiedene Phänomene unter Laborbedingungen zu reproduzieren, deren Beobachtung in der Realität schwierig oder unmöglich ist. Unter den verschiedenen Bereichen der Hochenergiephysik nehmen die Experimente im Zusammenhang mit der Kollision schwerer Ionen, mit hoher Energie und der Bildung heißer und dichter Kernmaterie, einen besonderen Platz ein. Dadurch ist es möglich, unter Laborbedingungen eine breite Palette von Materiezuständen zu erzeugen, von der Ähnlichkeit des frühen Universums bis hin zur extrem dichten Materie von Neutronensternen.

In den letzten Jahren wurden zahlreiche Forschungsarbeiten im Zusammenhang mit der Kollision schwerer Ionen durchgeführt, die es ermöglichten, wichtige Bereiche des Phasendiagramms der Quantenchromodynamik (QCD) zu untersuchen. Eine Reihe von Materiezuständen, die für die Wissenschaftler von Interesse sind, sind jedoch noch unzureichend erforscht. Gleichzeitig erfordert der Erfolg von Experimenten eine zuverlässige Registrierung von immer selteneren Teilchen, die einen bestimmten physikalischen Prozess charakterisieren. Daraus ergibt sich die Notwendigkeit, die Statistik der Experimente zu verbessern, was äußerst schwierig ist, ohne die Häufigkeit der Teilcheninteraktion und die Rate der Informationserfassung zu erhöhen. Die modernen Rechner sind jedoch nicht in der Lage, den wachsenden Bedarf an Verarbeitungsgeschwindigkeit und Datenspeicherkapazität zu decken. Daher muss eine Vorauswahl getroffen werden, die

den Datenfluss auf eine verarbeitbare Anzahl von Ereignissen reduziert, ohne dass Ereignisse von Interesse verloren gehen.

Das Compressed Baryonic Matter (CBM)-Experiment, das in den kommenden Jahren am im Bau befindlichen Beschleunigerkomplex Facility for Antiproton and Ion Research (FAIR) gestartet werden soll, soll die höchste Kollisionsrate in der Geschichte der Schwerionenkollisionen erreichen. Ziel des Experiments ist die Suche nach extrem seltenen Teilchen in Gegenwart einer großen Anzahl von Hintergrundfragmenten. Unter solchen Bedingungen ist eine schnelle Auswahl interessanter Ereignisse anhand einfacher Kriterien nicht möglich. Daher geht man für das CBM-Experiment davon aus, auf einen Auslösemechanismus für das Sammeln von Ereignisdaten zu verzichten. Stattdessen erfolgt die Datenreduzierung und Ereignisauswahl durch eine vollständige Online-Ereignisrekonstruktion und -analyse. Dies erfordert extrem schnelle, effiziente und zuverlässige Algorithmen, die auf einem Hochleistungsrechner (HPC)-Cluster laufen.

Der innovative Ansatz für die Arbeit am CBM-Experiment erfordert eine sorgfältige Voruntersuchung, um mögliche Schwachstellen im Voraus erkennen und korrigieren zu können. Diese Möglichkeit bietet das FAIR Phase-0 Programm. Dabei werden verschiedene Elemente des CBM-Experiments in anderen aktiven HEP-Experimenten erforscht und getestet. Auf diese Weise können die Fähigkeiten von Hardware- und Softwarekonzepten bei der Arbeit mit realen Lasten und Daten bewertet werden.

Einer der FAIR Phase-0 Beteiligten ist das Solenoid Tracker At RHIC (STAR)-Experiment. Im Rahmen des Beam Energy Scan-II (BES-II)-Programms arbeitete das Experiment mit ähnlichen Energien wie das CBM-Experiment bei einer deutlich niedrigeren Kollisionsrate. Außerdem wurde das FiXed Target (FXT)-Programm implementiert, das es dem STAR-Experiment ermöglichte, ähnlich wie das CBM-Experiment in einem Fixed-Target-Modus zu arbeiten. All dies ermöglichte es, dieses Experiment als Testgelände für technische Lösungen wie den CBM-Flugzeitdetektor (Time-of-Flight (ToF)) und den Ansatz der Online-Rekonstruktion von Ereignissen und der physikalischen Analyse zu nutzen.

## Rekonstruktion von Teilchenflugbahnen

Das Ergebnis der Teilchenkollision in HEP-Experimenten ist eine Reihe von Reaktionen von Detektoren, die dafür konstruiert sind, die durch die Kollision entstandenen Fragmente zu erfassen. Für sich genommen sind diese Daten für die

Forscher praktisch unbrauchbar und müssen in eine für die Analyse geeignete Form gebracht werden. Dieser Prozess umfasst eine Reihe von Schritten, bei denen die Detektorreaktionen in die räumlichen Koordinaten des Schnittpunkts von Teilchen und Sensoren umgewandelt werden müssen — so genannte Treffer. Aus diesen Treffern müssen dann mit größtmöglicher Effizienz die Flugbahnen der Partikel (Spuren) rekonstruiert werden. Nach der Anpassung und Schätzung der Spurparameter können diese Daten schließlich zur Erkennung seltener Teilchen verwendet werden.

Der Kalman-Filter (KF) ist optimal für die Schätzung der Parameter von Partikelflugbahnen. Dabei handelt es sich um eine rekursive Methode zur Analyse linearer diskreter dynamischer Systeme, die durch einen Vektor von Parametern, dem sogenannten Zustandsvektor, beschrieben werden können. Der Zustandsvektor kann sowohl zur Anzeige der Parameter der endgültigen Flugbahn, als auch direkt bei der Rekonstruktion der Spur verwendet werden, um die Möglichkeit der Kombination von Treffern auf die eine oder andere Weise zu bewerten. Darüber hinaus kann der KF sowohl auf der Daten- als auch auf der Thread-Ebene parallelisiert werden, wodurch das Potenzial moderner Hochleistungssysteme optimal genutzt wird.

Die Spurwiederherstellung (Tracking) ist der komplexeste und zeitaufwändigste Teil der Ereignisrekonstruktion. Die große Anzahl von Teilchen, die in modernen HEP-Experimenten erzeugt werden, führt zu einer hohen kombinatorischen Komplexität des Prozesses ihrer Kombination. Die Rekonstruktion von Ereignissen in Echtzeit stellt besonders hohe Anforderungen an die Geschwindigkeit des Tracking-Prozesses. Die optimale Lösung des Problems ist in diesem Fall ein Algorithmus zur Rekonstruktion der Spur, der auf einem Zellulären Automaten (CA) basiert und den KF zur Schätzung der Spurparameter in verschiedenen Phasen der Rekonstruktion verwendet.

Das CA-Konzept geht von einer diskreten Änderung des Zustands einzelner Zellen eines gleichmäßigen Gitters in Abhängigkeit von den Zuständen der Elemente in einer bestimmten Nachbarschaft aus. Bei der Rekonstruktion von Spuren gewährleistet dies die Datenlokalität, bietet umfangreiche Möglichkeiten für den Einsatz moderner HPC-Methoden und macht den Algorithmus stabiler gegenüber Detektorineffizienzen und Messfehlern. Der KF wiederum ermöglicht es, sowohl die Bedingungen für die CA-Evolution zu bestimmen, als auch die Korrektheit der einen oder anderen Trefferkombination während des Betriebs des Algorithmus schnell bewerten zu können.

All dies macht den CA Track Finder zu einem optimalen Algorithmus für die Rekonstruktion der Flugbahnen geladener Teilchen in Echtzeit. In dieser Arbeit werden zwei Varianten der Implementierung des CA-basierten Trackings untersucht, die im STAR-Experiment zur Online-Rekonstruktion von Teilchenflugbahnen verwendet werden. Es werden Möglichkeiten zur Verbesserung der Algorithmen und zur Erzielung valider Ergebnisse ohne vollständige Offline-Verarbeitung erforscht.

## HFT CA Track Finder im STAR-Experiment

Der Heavy Flavor Tracker (HFT)-Detektor im STAR-Experiment hat eine zylinderförmige Struktur, die aus vier Schichten und Stationen besteht. Er ähnelt technologisch einigen der CBM-Detektoren. Daher wurde der CA-basierte Trackingmechanismus des CBM-Experiments als Grundlage für das Online-Tracking im HFT genommen. Er beinhaltet die Bildung eines Spurparametervektors in der frühesten Phase der Rekonstruktion. Die Kombination aus einem Treffer und einem Parametervektor wird als Singulett bezeichnet. Singuletten werden zu Dubletten — Kombinationen aus zwei Treffern — zusammengefasst, die auf der Ähnlichkeit der Koordinaten und der Richtung relativ zum Kollisionspunkt — dem „Primary Vertex“ — basieren. Dubletten mit einem gemeinsamen Treffer können Tripletten bilden — das minimale Segment zur Bestimmung der Krümmung, die das Momentum des Teilchens widerspiegelt. Zudem können Triplette mit ähnlichen Momenta und ein paar gemeinsamen Treffern Spurkandidaten bilden.

Der HFT-Detektor verfügt über insgesamt vier zylindrische Stationen. Das bedeutet, dass die erfolgreiche Suche nach einer Spur die Einbeziehung von Treffern auf jeder dieser Stationen erfordert. Die inneren Stationen des Detektors sind viel langsamer als die äußeren. Dies führt zu einer Anhäufung von Treffern aus aufeinanderfolgenden Ereignissen und zur Bildung von Rauschen (Überlagerung), dessen Menge um ein Vielfaches größer ist als die Anzahl der Treffer bei einem Zusammenstoß. All dies wirkte sich äußerst negativ auf die Geschwindigkeit der Rekonstruktion aus, so dass der Algorithmus nicht für den Echtzeitbetrieb im Rahmen des High-Level-Trigger (HLT) des STAR-Experiments eingesetzt werden konnte.

Im Zuge der Arbeiten zur Optimierung und Beschleunigung des Algorithmus wurden die Möglichkeiten und Auswirkungen der Anwendung verschiedener

Verbesserungen für Algorithmen dieser Klasse untersucht. Der HLT-Rahmen erlaubt keine Parallelität auf Thread-Ebene bei der Rekonstruktion von Spuren in einem einzigen Ereignis. Daher konzentrierte sich die Arbeit auf die Verbesserung der Mechanik und die Parallelisierung auf Datenebene.

Traditionell entspricht die Richtung der Rekonstruktion von Partikelflugbahnen ihrer Bewegung im Detektor. Die Lokalität des CA erlaubt es, Treffer an benachbarten Stationen in beliebiger Reihenfolge zu kombinieren. Durch die Umkehrung der Trackingrichtung konnte die Auswirkung der Überlagerungs-Treffer auf die Geschwindigkeit der Dublettbildung erheblich reduziert werden, insbesondere bei einer geringen Teilchenzahl. Die Effizienz der Rekonstruktion blieb auf dem gleichen Niveau. Bei größeren Detektoren kann allerdings die Rekonstruktionsrichtung für verschiedene Detektorbereiche getrennt optimiert werden.

Die Nutzung der Gitterstruktur zur optimalen Anordnung der Treffer im Speicher und schneller Zugriff auf einzelne Elemente machte es möglich die Anzahl der Speicherzugriffe deutlich zu reduzieren und der Algorithmus konnte um ein Vielfaches beschleunigt werden. Durch eine Veränderung der Größe des relevanten Bereichs der Gitterstruktur bei der Treffersuche können die Prioritäten des Algorithmus leicht verändert werden, sodass man zwischen höherer Effizienz und hoher Geschwindigkeit entscheiden kann.

Die maximale Geschwindigkeit des Algorithmus wurde durch die Verwendung von SIMD-Instruktionen erreicht. Der im Rahmen einer Studie entwickelte Vektorisierungsmechanismus ermöglichte es, den Effekt der kombinatorischen Aufzählung von Treffern bei der Konstruktion von Dubletten und Tripletten zu minimieren. Dieser Ansatz stellt gleichzeitig sicher, dass die SIMD-Vektoren vollständig gefüllt sind, maximiert die Anzahl der aufeinanderfolgenden Operationen mit denselben Daten und reduziert den Overhead durch zufällige Speicherzugriffe.

Die endgültige Version des HFT CA Track Finder wurde von einer Laufzeit von über 1 Minute pro Ereignis auf weniger als 7 ms beschleunigt, wobei die Effizienz bei über 90% liegt. Alle Verteilungen der Residuen und Pulls sind frei von systematischen Fehlern. Die Breite der Residuen ist übereinstimmend zu der Auflösung des Detektors, die Breite der Pulls liegt nahe dem erwarteten Wert von Eins. Dies ist ein Zeichen für die hohe Qualität der rekonstruierten Spuren.

## TPC CA Track Finder im STAR-Experiment

Der Time Projection Chamber (TPC)-Detektor unterscheidet sich vom HFT und den Detektoren des CBM-Experiments durch seinen Aufbau und sein Funktionsprinzip. Die Anzahl der Pad-Reihen in jedem Sektor des TPC-Detektors ist sehr groß und der Abstand zwischen ihnen ist gering. Unter diesen Bedingungen ist die Anwendung des zuvor diskutierten Tracking-Ansatzes nicht ratsam. Ein ähnliches Projekt aus dem ALICE-Experiment diente als Grundlage für den STAR TPC CA-Tracker. Nach einer Neuausrichtung auf die Eigenschaften des TPC-Detektors im STAR-Experiment wurde dieser Algorithmus mit Hilfe von SIMD-Operationen und der Vc-Bibliothek vektorisiert. In dieser Form wurde er in die Online- und Offline-Rekonstruktionsketten des STAR-Experiments aufgenommen. Die Vektorisierung des Algorithmus führte nicht zu einer signifikanten Beschleunigung, ermöglichte es aber, die Anforderungen für den Einsatz des Algorithmus im HLT zu erfüllen.

In dieser Arbeit betrachten wir auch die Möglichkeit der Verbesserung des Algorithmus, um die Qualität der Rekonstruktion unter Beibehaltung einer hohen Rechengeschwindigkeit zu steigern. Der TPC-Detektor des STAR-Experiments ist in 24 Sektoren unterteilt. Der Track Finder ist eine Kombination aus einem Sektor-CA-Tracker, der die Teilchenbahnen in jedem der Sektoren sequentiell rekonstruiert, und einem so genannten Merger, der die sektoralen Spuren zu globalen Spuren kombiniert. Während der Arbeit an dem Algorithmus wurden die meisten Arbeitsschritte des Algorithmus optimiert und ein weiterer Arbeitsschritt wurde hinzugefügt.

Der TPC-Aufbau wurde in Vorbereitung auf das BES-II-Programm mit einem verbesserten Detektor-Innenteil (iTPC) aufgerüstet. Dadurch ergab sich die Notwendigkeit, die Rekonstruktion der Spuren zu optimieren und es konnten bessere Ergebnisse erzielt werden. Nach der Aufrüstung wurde die Anzahl der Pad-Reihen im iTPC von 13 auf 40 erhöht, so dass die Gesamtzahl nun 72 beträgt. Aufgrund der höheren Beobachtungsdichte konnte der Detektor erfolgreich im FXT-Modus arbeiten und Teilchen mit geringer Energie besser aufspüren.

Im Gegensatz zum HFT-Tracker erfolgt die Bildung von Tripletten ohne Verwendung des Kalman-Filters. Das Verfahren basiert auf der Suche nach Tripletten mit der geringsten Steigungsdifferenz ihrer Teile, das heißt den meisten geraden Segmenten. Dann bilden Tripletette auf benachbarten Pad-Reihen, die gemeinsame Trefferpaare haben, Ketten. Am Ende des Sektortrackings werden

diese Ketten mit dem KF angepasst und wenn möglich erweitert. Diese grundlegenden Sektortrackinroutinen wurden optimiert, um der aktualisierten Detektorgeometrie zu entsprechen. Um die Effizienz der Rekonstruktion im iTPC zu erhöhen, wurde dem Algorithmus ein spezielles Verfahren hinzugefügt, das es erlaubt, Triplette, die nicht an Ketten gebunden sind, zu vollwertigen Spurkandidaten zu erweitern.

Ein Anstieg der Zahl der rekonstruierten Spuren mit sehr geringem Momentum erschwert die anschließende Analyse der Ergebnisse. Solche Spuren, so genannte Looper, können auf ihrem Flug durch den Detektor mehr als 10 Schleifen hinterlassen. Das im TPC CA verwendete Spurmodell rekonstruiert sie als individuelle Partikelbahnen. Um die Anzahl solcher Formationen zu reduzieren und die Rekonstruktion von Teilchen mit geringer Energie zu vervollständigen, wurde ein neues Verfahren — der Looper-Merger — entwickelt und dem Programm hinzugefügt. Unter Verwendung verschiedener Merkmale von Segmenten mit niedrigem Momentum konnte der Looper-Merger etwa 80% der Looper kombinieren, wobei die Reinheit der resultierenden Spuren über 90% betrug.

Tests der eigenständigen Version des Programms mit simulierten Daten zeigten eine Steigerung der Effizienz bei der Rekonstruktion von Spuren der meisten Typen um mehr als 3% im Vergleich zur alten Variante des Algorithmus. Die Verbesserungen des Mergers und die Integration des Looper-Merger haben auch das Klon-Niveau deutlich reduziert - die Rekonstruktionsfälle mehrerer separater Segmente, die mit demselben Partikel verbunden sind. Die Verteilungen der Residuen und Pulls bestätigen die hohe Qualität der gefundenen Spuren. Die Breite der Pulls liegt nahe bei 1; die Eigenheiten der Daten erklären einige Abweichungen.

Der aktualisierte Algorithmus wurde in den HLT integriert und erfolgreich für die Online-Rekonstruktion von Ereignissen im Rahmen der „STAR-Express-Production“ eingesetzt. Abbildung 7.1 zeigt einen Vergleich der relativen Effizienz der alten und der neuen Version des Algorithmus bei der Rekonstruktion von realen Daten aus dem Jahr 2020. Der Algorithmus weist deutlich sichtbare Verbesserungen auf, was die Testergebnisse mit simulierten Monte-Carlo-Ereignissen bestätigen. Die Untersuchung des Tracking-Algorithmus an den früheren Daten ab 2014 bestätigt die Überlegenheit des neuen, verbesserten Algorithmus in vollem Umfang.

Die Qualität der TPC CA Track Finder-Rekonstruktion erwies sich als völlig ausreichend für die physikalische Echtzeitanalyse von Ereignissen mit dem

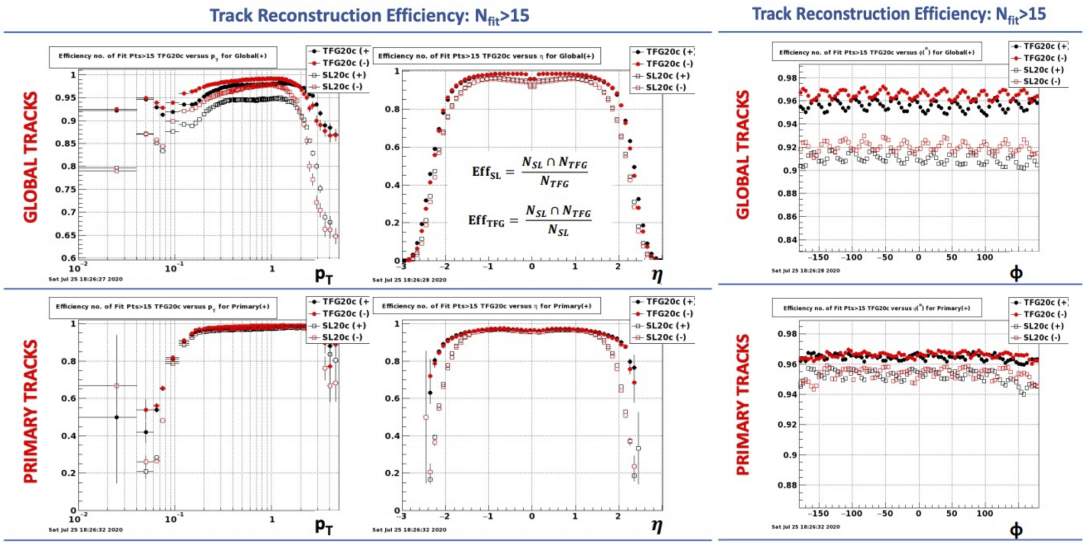


Figure 7.1: Vergleich der Effizienz von Spurrekonstruktionsalgorithmen, die auf der alten (**SL20c**) und der neuen (**TFG20c**) Version des TPC CA Track Finders basieren, bei der Verarbeitung von Run20-Daten aus dem Jahr 2020. Die relativen Effizienzen sind in Abhängigkeit vom Transversalimpuls $p_T$, den Winkeln $\eta$ und $\phi$ für primäre und globale Spuren angegeben.

KF Particle Finder-Paket. Bei der Analyse von weniger als 30% der Ereignisse war es möglich, die Zerfallsprozesse kurzlebiger Teilchen sowohl mit klassischen Ansätzen als auch mit der Missing-Mass-Methode eindeutig zu bestimmen, was ihre Wirksamkeit in den realen Daten bestätigte. Die Effektivität der Idee der Online-Rekonstruktion und -Datenanalyse, die die Basis des CBM Experiments bildet, wird somit unter realen Bedingungen vollstens bestätigt. Gleichzeitig sind die Ressourcen der HLT-Computerfarm, die für die Express-Rekonstruktion vorgesehen sind, im Gegensatz zu der geplanten Rechenleistung, die für die CBM zur Verfügung stehen wird, stark begrenzt.

In Anbetracht der gesteigerten Rekonstruktionsleistung und der zusätzlichen Funktionalität wurde beschlossen, den veralteten Tracking-Algorithmus auch in der Standard-Offline-Datenverarbeitungskette durch den neuen zu ersetzen. So wird der aktualisierte TPC CA Track Finder, der schnelle und effiziente Mechanismus für das Tracking und der Rekonstruktion von Ereignissen mit beliebiger Teilchenanzahl, seinen Platz sowohl in der Online- als auch in der Offline-Verarbeitungskette des STAR-Experiments einnehmen.

# Acknowledgements

I want to express my sincere gratitude to all the people who helped me and supported me throughout my scientific activity and the preparation of this thesis.

My special thanks to Prof. Dr. Ivan Kisel, who not only guided my scientific activity but also helped me progress as a scientist and specialist. Working with such a supervisor, I always felt supported and knew that I could rely on his help in any situation. I also want to say thank you to all the organizers and staff of the Helmholtz Graduate School for the Heavy Ion Research (HGS-HIRe), without whom this work most likely would never have taken place.

I am very grateful to the people who led me into science and helped me take my first steps in this area. I want to say thanks to Prof. Dr. Gennady Ososkov for opening this path for me, Prof. Dr. Victor Ivanov for the opportunity to move forward, Dr. Andrey Lebedev for teaching the basics and guidance.

Many thanks to everyone who shared their knowledge and skills with me, giving me the opportunity to learn new things and expand my knowledge. I feel grateful to Dr. Iouri Vassiliev for his informative and entertaining explanations of the physical processes of the CBM experiment and Dr. Yuri Fisyak, who helped to understand the features of the STAR experiment. I want to especially thank Dr. Maxim Zyzak for his invaluable help with algorithms and software. Thanks to Robin Lakos for helping me with the translation in German. And many thanks to all my colleagues for their support and the consistently positive atmosphere in the team, especially to Pavel Kisel, Artemiy Belousov, and Dr. Valentina Akishina.

And, of course, I sincerely thank my family and friends, whose unwavering support has accompanied me all these years, giving me the strength to go to the goal.