Письма в ЭЧАЯ. 2013. Т. 10, № 2(179). С. 253–267

АНАЛИЗ ЭФФЕКТИВНОСТИ И ПРОИЗВОДИТЕЛЬНОСТИ АЛГОРИТМА РАСПОЗНАВАНИЯ ТРЕКОВ В STS-ДЕТЕКТОРЕ ЭКСПЕРИМЕНТА CBM НА МНОГОЯДЕРНОМ СЕРВЕРЕ ЛИТ ОИЯИ

И. С. Кулаков [а, б], *С. А. Багинян* [в], *В. В. Иванов* [в], *П. И. Кисель* [в]

[а] Франкфуртский университет им. И.-В. Гете, Франкфурт-на-Майне, Германия

[б] Киевский национальный университет им. Т. Шевченко, Киев

[в] Объединенный институт ядерных исследований, Дубна

Распознавание траекторий заряженных частиц является ключевой проблемой в задаче реконструкции событий в эксперименте CBM (GSI, Германия). Высокая множественность событий, интенсивный фон, неоднородное магнитное поле и необходимость реконструкции всех событий в режиме реального времени потребовали не только развития новых подходов для решения рассматриваемой задачи, но и максимального использования потенциала современных многоядерных архитектур CPU/GPU. В настоящей работе приведены результаты анализа эффективности и производительности алгоритма распознавания треков на основе клеточного автомата и фильтра Калмана в STS-детекторе эксперимента CBM на многоядерном сервере ЛИТ ОИЯИ.

The results of the tests for the tracks reconstruction efficiency, the speed of the algorithm and its scalability with respect to the number of cores of the server with two Intel Xeon E5640 CPUs (in total 8 physical or 16 logical cores) are presented and discussed.

PACS: 29.40.-h

ВВЕДЕНИЕ

Пучки тяжелых ионов высокой интенсивности, которые будут предоставляться на будущих ускорителях FAIR (Facility for Antiproton and Ion Research), в сочетании с готовящимся экспериментом CBM (Compressed Baryonic Matter) открывают превосходные возможности для изучения барионной материи при сверхвысоких плотностях и умеренных температурах в лабораторных условиях [1]. Физическая программа CBM нацелена на изучение структуры и поведения барионной материи при плотностях, сопоставимых с плотностями в центре нейтронных звезд. Она включает в себя: 1) установление фазовой границы между адронной и партонной материями, 2) определение критической конечной точки, 3) поиск указаний на начало восстановления киральной симметрии при высоких чистых барионных плотностях.

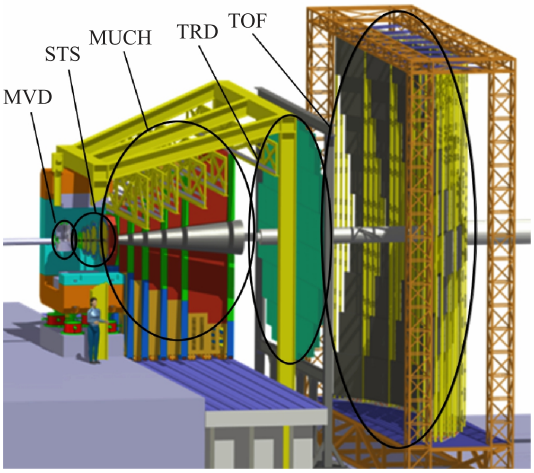

Рис. 1. Схема экспериментальной установки CBM для регистрации мюонных распадов

На рис. 1 представлена схема экспериментальной установки CBM, ориентированной на регистрацию мюонных распадов. На этом рисунке непосредственно за мишенью, в сверхпроводящем дипольном магните, располагается вершинный трековый детектор, состоящий из двух плоскостей монолитных микропиксельных детекторов MVD (Micro-Vertex Detector) и восьми кремниевых микростриповых детекторов STS (Silicon Tracking System). Вершинный детектор предназначен для реконструкции треков заряженных частиц и восстановления с максимальной точностью вторичных вершин в условиях высокой плотности треков. Информация с вершинного детектора будет также использоваться для определения импульсов заряженных частиц с точностью не хуже 1 %. Мюонная станция MUCH (MUon CHamber) состоит из чередующихся слоев железных поглотителей (для подавления адронов и выделения мюонов) и координатных детекторов. Детекторы MUCH будут использоваться для реконструкции траекторий регистрируемых мюонов. Эти треки будут «связываться» с треками, восстановленными с помощью вершинного детектора, что позволит определять импульсы детектируемых мюонов. Детектор переходного излучения TRD (Transition Radiation Detector) предназначен для: 1) реконструкции треков, 2) идентификации электронов и позитронов в условиях доминирующего фона от пионов. Для идентификации адронов используется система измерения времени пролета TOF (Time-Of-Flight), а с помощью электромагнитного калориметра ECAL (Electromagnetic CALorimeter) будут идентифицироваться электроны и фотоны.

Распознавание траекторий заряженных частиц с помощью вершинного детектора является ключевой проблемой в задаче реконструкции событий в эксперименте CBM (GSI, Германия). Высокая множественность событий (до 1000 вторичных треков в каждом ядро-ядерном соударении), интенсивный фон (до 85 % фоновых отсчетов в STS-детекторе), неоднородное магнитное поле и необходимость реконструкции всех событий в режиме реального времени (до 10^7 событий в секунду) потребовали не только развития новых подходов для решения рассматриваемой задачи, но и максимального использования потенциала современных многоядерных архитектур CPU/GPU.

В качестве примера на рис. 2, *б* приведено типичное для эксперимента CBM событие, сгенерированное методом Монте-Карло (МК) в детекторах MVD и STS с помощью

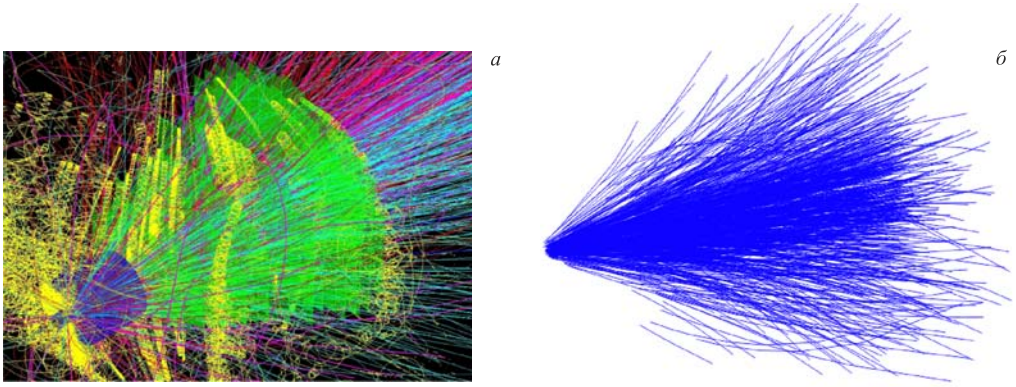

Рис. 2. Пример типичного для эксперимента CBM события, сгенерированного методом MK для центрального столкновения $\mathrm{Au}+\mathrm{Au}$ при энергии 25 ГэВ/нуклон: *а*) сгенерированное событие в детекторах MVD и STS; *б*) реконструированное событие, содержащее 729 треков

пакета GEANT3 [13] в среде CBM-ROOT [14] для центрального столкновения $\mathrm{Au}+\mathrm{Au}$ при энергии 25 ГэВ/нуклон.

В настоящей работе приведены результаты тестирования алгоритма распознавания треков в STS-детекторе эксперимента CBM на многоядерном сервере ЛИТ ОИЯИ, содержащем два процессора Intel Xeon E5640: суммарно восемь физических, или 16 логических ядер. В основу этого алгоритма положены клеточный автомат (КА) и фильтр Калмана, что позволило распараллелить обработку событий как на уровне оптимального использования всех ядер сервера, так и на уровне обмена данными.

1. МЕТОД КЛЕТОЧНОГО АВТОМАТА ДЛЯ РЕШЕНИЯ ЗАДАЧИ РАСПОЗНАВАНИЯ ТРЕКОВ

Физическая модель траектории заряженной частицы в декартовой системе координат, принятой в эксперименте CBM (0 — начало координат, ось OZ совпадает с направлением падающего пучка частиц), и при известных z_i-координатах STS-станций описывается вектором состояния $(x_i, y_i, t_{xi}, t_{yi}, q/p)$. Здесь x_i, y_i — координаты места пересечения частицей плоскости i-го STS-детектора (далее мы будем называть их также хитами, от английского слова hit); t_{xi}, t_{yi} — тангенсы углов наклона трека к оси OZ; q/p — отношение заряда частицы к ее импульсу. Задача распознавания треков, зарегистрированных в одном событии эксперимента CBM, состоит в отборе хитов, относящихся к траекториям отдельных частиц, и формировании соответствующих векторов состояния.

Упрощенная схема, иллюстрирующая работу алгоритма КА распознавания треков, представлена на рис. 3. На нем изображено событие в STS-детекторе из шести координатных станций, содержащее всего два трека. Алгоритм КА включает два последовательных этапа [2–5].

1.1. Первый этап: формирование треков-сегментов (триплетов). В качестве входной информации в алгоритме КА используются хиты (*0* на рис. 3). На первом шаге из хитов

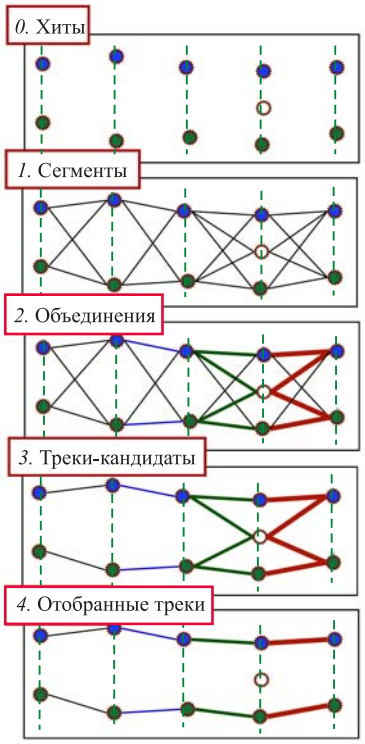

Рис. 3. Упрощенная схема, иллюстрирующая работу алгоритма КА распознавания треков: *0* — измеренные хиты; *1* — формирование треков-сегментов; *2* — объединение треков-сегментов с соседними треками-сегментами; *3* — выделение треков-кандидатов; *4* — отбор лучших треков-кандидатов

формируются треки-сегменты (триплеты), которые содержат хиты трех соседних STS-станций. На рис. 3, для наглядности, эта процедура представлена упрощенной моделью трека-сегмента, который содержит всего два трека (*1* на рис. 3).

С целью уменьшения перебора хитов при формировании триплетов используются как геометрические, так и физические ограничения, накладываемые экспериментом. Так, например, с помощью МК-моделирования эксперимента CBM было установлено, что подавляющее большинство регистрируемых частиц летят из области первичной вершины (место взаимодействия пучка с мишенью) под небольшими углами относительно направления падающего пучка. Кроме того, частицы, несущие полезную информацию о физических процессах, происходящих во время ядро-ядерных соударений, имеют импульсы больше 0,1 ГэВ/c. В связи с этим частицы с импульсами, меньшими 0,1 ГэВ/c, исключаются из рассмотрения.

Каждый триплет содержит хиты с трех соседних станций STS. Так как один хит включает (x, y)-координаты места пересечения частицей плоскости конкретной станции, то хиты с трех STS-станций образуют набор из шести измерений, позволяющих однозначно определить все пять параметров, описывающих рассматриваемый элемент трека, а также оценить χ^2-отклонение реальных измерений от реконструированного трека. Как показал последующий анализ, получаемая таким образом χ^2-оценка трека позволяет отбросить большинство случайных триплетов.

1.2. Второй этап: создание треков-кандидатов. На втором этапе из всех триплетов, сформированных для рассматриваемого события, проводится объединение

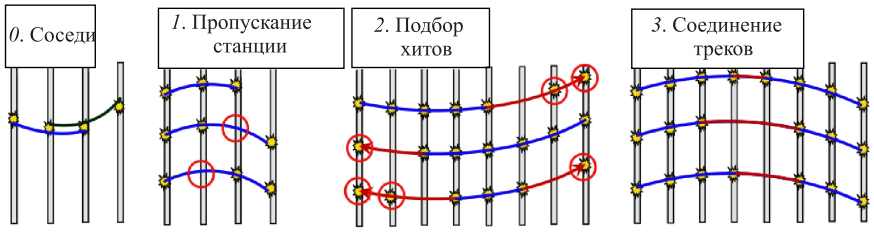

Рис. 4. Схема, иллюстрирующая работу алгоритма КА по формированию треков-кандидатов: *0* — соседние триплеты; *1* — три разных типа триплетов; *2* — «подбор» дополнительных хитов для реконструированного трека-кандидата; *3* — объединение треков-кандидатов

треков-сегментов с соседними треками-сегментами и строятся треки-кандидаты (см. *2–4* на рис. 3).

Вначале алгоритм КА, «двигаясь» слева направо, определяет для каждого триплета его соседей — триплеты, которые, в соответствии с моделью трека, могут быть продолжениями рассматриваемого триплета. Соседние триплеты должны иметь два общих хита и совпадающие в пределах ошибок модули импульсов (см. *0* на рис. 4).

В процессе определения соседей у каждого триплета формируется так называемый уровень триплета — величина, равная числу триплетов, примыкающих к рассматриваемому триплету слева и образующих непрерывную цепочку триплетов-соседей. Уровень триплета определяет его положение на треке в цепочке соседей — чем большее значение он принимает, тем более длинный трек из него может быть построен.

Знание уровней триплетов позволяет легко и быстро связать их в полноценные треки-кандидаты. Начиная с триплета, имеющего максимальный уровень, соединяем его с соседними, у которых уровень строго на единицу меньше. Их, в свою очередь, также соединяем с соответствующими соседями. Продолжаем эту процедуру до тех пор, пока не дойдем до триплетов, не имеющих соседей слева. При этом образуется древовидная структура потенциальных треков-кандидатов, из которых с помощью критерия χ^2 отбирается лучший трек.

Таким образом, для каждого триплета с максимальным уровнем строится трек-кандидат. Затем все треки-кандидаты попарно анализируются на предмет наличия в них общих хитов. Если треки-кандидаты имеют общий хит, то предпочтение отдается треку-кандидату с наименьшим χ^2. Процедура окончательного отбора реконструированных треков построена на сортировке по критерию χ^2, после чего, начиная с лучшего, треки-кандидаты обозначаются как восстановленные треки, а все хиты, относящиеся к этим трекам, метятся как использованные. Если очередной трек-кандидат содержит использованные хиты, он отбрасывается. После того как рассмотрены все треки-кандидаты с максимальной длиной, такая же процедура обработки применяется для треков-кандидатов, содержащих меньшее число соседей, и т. д.

Двухэтапная процедура, описанная выше, применяется к каждому событию в три прохода: 1) поиск треков от быстрых ($p > 0{,}5$ ГэВ/c) квазипервичных (испускаемых из области мишени) частиц; 2) поиск треков от медленных ($0{,}1 < p < 0{,}5$ ГэВ/c) квазипервичных частиц; 3) поиск треков от вторичных частиц с произвольным импульсом. После каждого прохода все измерения, принадлежащие найденным трекам, метятся и исключаются из дальнейшего рассмотрения. При таком подходе последовательное ослабление критериев при поиске треков приводит к уменьшению количества хитов. Это обеспечивает возможность быстрого и надежного восстановления треков при высоких множественностях событий и плотностях измерений в вершинном детекторе эксперимента CBM.

В процессе построения и отбора треков-кандидатов применяется ряд ограничений, позволяющих существенно уменьшить количество неверно реконструированных треков. Кроме того, для учета неэффективной работы отдельных станций вершинного детектора в алгоритм КА включены дополнительные возможности (см. *1, 2* и *3* на рис. 4): 1) задание триплетов, позволяющих пропускать одну неэффективно работающую станцию; 2) экстраполяция треков-кандидатов через неэффективные станции и добор дополнительных хитов; 3) связывание элементов трека частицы, реконструированных независимо друг от друга.

На рис. 2 приведен результат обработки события, сгенерированного методом MK для центрального столкновения Au + Au при энергии 25 ГэВ/нуклон: реконструированное событие (рис. *б*) содержит 729 треков.

2. МЕТОД ФИЛЬТРА КАЛМАНА В ЗАДАЧЕ РЕКОНСТРУКЦИИ ТРЕКОВ ЧАСТИЦ

При аппроксимации (фитировании) измерений трека заряженной частицы методом наименьших квадратов (глобальное фитирование) в идеальных условиях, а именно при однородном магнитном поле и отсутствии многократного рассеяния, дает наилучшую оценку параметров модели траектории частицы. Однако ситуация резко меняется в случае неоднородного магнитного поля и наличия многократного рассеяния, которое вносит заметный вклад в ошибку измерения:

$$\boldsymbol{\xi} = \boldsymbol{\xi}_{\mathrm{er}} \bigcup \boldsymbol{\xi}_{\mathrm{ms}}, \tag{1}$$

где $\boldsymbol{\xi}_{\mathrm{er}}$ — измерительная ошибка детектора; $\boldsymbol{\xi}_{\mathrm{ms}}$ — ошибка, связанная с многократным рассеянием.

В этом случае удобно воспользоваться хорошо зарекомендовавшим себя фильтром Калмана, который применяется для нахождения оптимальной оценки параметров дискретной линейной динамической системы [7,8], описывающей изучаемый процесс. В нашем случае процедуру фитирования трека можно представить как эволюционный процесс указанной системы [9], где в качестве вектора состояний системы в k-й момент времени выступает вектор истинных значений параметров трековой модели $\mathbf{p}_k = (x^k, y^k, t_x^k, t_y^k, q/p)^T$ при прохождении трека через k-й детектор, а в качестве вектора измерений системы — хиты, зарегистрированные k-м детектором:

$$\mathbf{m}_k = (x^k, y^k, z^k)^T. \tag{2}$$

Таким образом, трек представляется как дискретная линейная динамическая система, которая описывается уравнением эволюции системы:

$$\mathbf{p}_k = F_k \mathbf{p}_{k-1} + \boldsymbol{\omega}_k \tag{3}$$

и уравнением проекции пространства параметров трековой модели в пространство измерений:

$$\mathbf{m}_k = H_k \mathbf{p}_k + \boldsymbol{\xi}_k. \tag{4}$$

Здесь F_k — так называемая матрица перехода, описывающая связь между параметрами модели трека в $(k-1)$-м детекторе со значениями параметров в k-м детекторе; H_k — матрица проекции параметров трековой модели в пространство измерений. В нашем случае в качестве измерения используется положение стрипа и матрица $H_k = \{\sin\phi, \cos\phi, 0, 0, 0\}$, где ϕ — угол наклона стрипа к оси OX; $\boldsymbol{\omega}_k$ и $\boldsymbol{\xi}_k$ — ошибки, вносимые в процесс многократным рассеянием и измерительной системой k-го детектора соответственно. Случайные векторы $\boldsymbol{\omega}_k$ и $\boldsymbol{\xi}_k$ независимы друг от друга и имеют нормальное распределение $N(0, Q_k)$ и $N(0, V_k)$, где Q_k и V_k — ковариационные матрицы векторов $\boldsymbol{\omega}_k$ и $\boldsymbol{\xi}_k$ соответственно.

Введем следующие обозначения:

— $\tilde{\mathbf{p}}_k$ — оценка вектора параметров системы на k-м шаге; фактически это математическое ожидание вектора $\mathbf{p}_k$ при условии, что уже известны k векторов измерений $E(\mathbf{p}_k \mid \mathbf{m}_k, \mathbf{m}_{k-1}, \ldots, \mathbf{m}_1)$;

— $\tilde{\mathbf{p}}_k^{k-1}$ — предсказанное значение вектора параметров системы на k-м шаге, если известна оценка вектора параметров на $(k-1)$-м шаге, $\tilde{\mathbf{p}}_k^{k-1} = F_k\tilde{\mathbf{p}}_{k-1}$;

— $C_k = \operatorname{cov}(\tilde{\mathbf{p}}_k - \mathbf{p}_k)$ — ковариационная матрица ошибок оценки вектора параметров системы на k-м шаге;

— $C_k^{k-1} = \operatorname{cov}(\tilde{\mathbf{p}}_k^{k-1} - \mathbf{p}_k)$ — ковариационная матрица ошибки предсказанного вектора.

Процедура фильтра Калмана позволяет получить значение оценки вектора параметров системы на k-м шаге $\tilde{\mathbf{p}}_k$, исходя из значения оценки вектора параметров на $(k-1)$-м шаге $\tilde{\mathbf{p}}_{k-1}$ и вектора измерений $\mathbf{m}_k$ на k-м шаге.

Основные формулы, которые нужны нам (в приведенной ниже последовательности) для построения алгоритма фильтра Калмана:

1) $\tilde{\mathbf{p}}_k^{k-1} = F_k\tilde{\mathbf{p}}_{k-1}$,

2) $C_k^{k-1} = F_kC_{k-1}F_k^T + Q_k$,

3) $K_k = C_k^{k-1}H_k^T(H_kC_k^{k-1}H_k^T + V_k)^{-1}$, где K_k — матрица преобразования Калмана, которая переводит ошибку из пространства измерений в пространство параметров,

4) $\tilde{\mathbf{p}}_k = \tilde{\mathbf{p}}_k^{k-1} + K_k(\mathbf{m}_k - H_k\tilde{\mathbf{p}}_k^{k-1})$ — это и есть основная формула одношагового предиктора Калмана,

5) $C_k = (I - K_kH_k)C_k^{k-1}$, здесь I — единичная матрица,

6) $\chi_k^2 = \chi_{k-1}^2 + (\mathbf{m}_k - H_k\tilde{\mathbf{p}}_k^{k-1})^T(H_kC_k^{k-1}H_k^T + V_k)^{-1}(\mathbf{m}_k - H_k\tilde{\mathbf{p}}_k^{k-1})$.

Фильтр Калмана представляет собой итерационный процесс, и для его запуска, исходя из имеющейся априорной информации и/или проведенного МК-моделирования, определяются стартовые значения для $\tilde{\mathbf{p}}_0$ и C_0. Затем организуется цикл:

1) $\tilde{\mathbf{p}}_k^{k-1} = F_k\tilde{\mathbf{p}}_{k-1}$,

2) $C_k^{k-1} = F_kC_{k-1}F_k^T + Q_k$,

3) $K_k = C_k^{k-1}H_k^T(H_kC_k^{k-1}H_k^T + V_k)^{-1}$, здесь K_k — преобразователь Калмана,

4) $\tilde{\mathbf{p}}_k = \tilde{\mathbf{p}}_k^{k-1} + K_k(\mathbf{m}_k - H_k\tilde{\mathbf{p}}_k^{k-1})$,

5) $C_k = (I - K_kH_k)C_k^{k-1}$

по всем станциям STS-детектора, начиная с первой и до последней: $k = 1, \ldots, n$.

По завершении вычислений с помощью фильтра Калмана на выходе алгоритма мы получаем оптимальную аппроксимацию трека заряженной частицы в неоднородном поле дипольного магнита вместе с восстановленным импульсом.

3. КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ И ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

Современные компьютерные технологии позволяют организовать одновременное выполнение многих операций, т. е. распараллеливание вычислений в рамках одной вычислительной среды: суперкомпьютера, многопроцессорного/многоядерного сервера и даже на отдельном персональном комьютере или ноутбуке. Они предоставляют возможность существенно (в 10, 100 и более раз) сократить астрономическое время выполнения конкретного времяемкого вычислительного алгоритма.

В настоящее время существует три основных класса компьютерных архитектур, которые реализуют разные принципы выполнения операций и обмена данными:

1) SISD (Single Instruction, Single Data stream) — одна команда для одной скалярной величины; все операции выполняются последовательно;

2) SIMD (Single Instruction, Multiple Data streams) — одна команда для многих данных; используется для одновременного выполнения одной операции с многими данными;

3) MIMD (Multiple Instruction, Multiple Data) — много команд для многих данных; одновременно может выполняться несколько команд над многими данными.

Кроме того, в современных компьютерах поддерживаются технологии многопоточности (Multithreading и Hyper-threding), которые позволяют распределять вычислительный процесс сразу между несколькими виртуальными процессорами (ядрами) и более эффективно использовать вычислительный потенциал каждого отдельного ядра, что бывает эквивалентно использованию двух логических ядер на одном физическом ядре.

3.1. SIMD-архитектуры. Основной элемент SIMD-архитектур — это векторный регистр, с помощью которого выполняются операции одновременно (парраллельно) над несколькими числами (данными). Поэтому вычисления на SIMD-архитектурах часто называют векторными. Исходные данные, над которыми выполняются операции, представляются словами, кратными одному байту; чаще всего они содержат 4 байта. И прежде чем проводить какую-либо операцию над числами, их нужно занести (упаковать) в векторный регистр [6, 10]. Одна из основных характеристик SIMD-архитектуры — это количество слов, обрабатываемых одновременно. В частности, если используется 128-разрядный векторный регистр, то можно параллельно выполнять операции сразу над четырьмя 32-разрядными действительными числами с одинарной точностью (рис. 5).

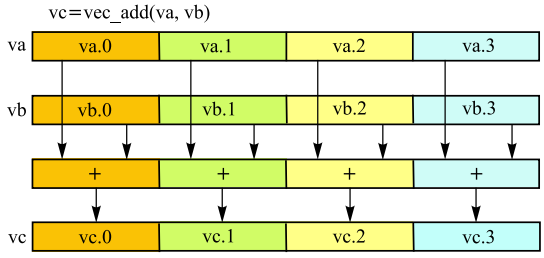

Рис. 5. Иллюстрация одновременного выполнения одной операции сразу над четырьмя числами

SIMD-архитектуры используются практически на всех существующих процессорах. На процессорах Intel SIMD-архитектуры для работы с целыми числами и числами с плавающей запятой реализованы в технологиях MMX и SSE.

Технология MMX (MultiMedia eXtension) — это набор SIMD-команд, разработанный Intel и реализованный в 1997 г. на микропроцессорах Pentium MMX. В MMX-микропроцессорах в архитектуру предшествующего процессора были добавлены восемь новых векторных регистров, каждый из которых представляет собой 64-разрядное целое число.

Технология SSE (Streaming SIMD Extensions) была включена в 1999 г. в процессоры Pentium III. SSE добавляет восемь новых 128-разрядных регистров, известных как регистры XMM. Каждый такой регистр вмещает четыре 32-разрядных числа с плавающей запятой и одинарной точностью. Введенная в Pentium IV технология SSE2 добавила на

тех же 128-разрядных регистрах XMM новые математические команды для действительных чисел с двойной точностью (64 разряда) и 8-/16-/32-разрядных целых чисел. Таким образом, SSE2 обеспечила программисту возможность применять на SIMD-архитектурах математические операции фактически над любым видом (от 8-разрядного целого до 64-разрядного действительного) чисел. Технология SSE3 — это дальнейшее развитие SSE2: добавлены математические команды DSPoriented, такие как сложение и вычитание многих чисел, которые сохраняются в одном векторном регистре. Технология SSE4 — это четвертое поколение технологии SSE, которое разработано для новых микроархитектур, создаваемых компанией Intel.

Векторные регистры SSE могут быть реализованы в четырех разных вариантах: 1) шестнадцать 8-разрядных слов со знаком или без знака, 2) восемь 16-разрядных слов со знаком или без знака, 3) четыре 32-разрядных целых числа, 4) четыре 32-разрядных числа с плавающей запятой.

В набор SEE включены команды для контроля кэш-памятью, которые позволяют максимально повысить эффективность использования кэш-памяти во время работы с потоками данных. Начиная с технологии SSE2 и выше можно оперировать 64-битными действительными числами с двойной точностью, а также обмениваться данными между векторными регистрами и оперативной памятью.

В задаче распознавания треков технологии SIMD используются на этапе построения триплетов и их аппроксимации. Это оказалось возможным благодаря тому, что каждый триплет обрабатывается независимо от других триплетов и обработка выполняется с помощью одного алгоритма — фильтра Калмана. Параметры фрагментов треков и измерения упаковываются в соответствующий SIMD-формат, после чего к ним применяются специально подготовленные функции, реализующие параллельно все этапы фильтра Калмана (коррекцию параметров трека при включении очередного измерения, экстраполяцию, учет многократного рассеяния).

Применение SIMD-архитектур в задаче распознавания треков позволило получить ускорение вычислений с помощью фильтра Калмана примерно в четыре раза, а алгоритма клеточного автомата — приблизительно в два раза.

3.2. Многопоточность. Уже достаточно продолжительное время для ускорения работы программ используют многопоточное программирование. В этом случае определенные вычисления выделяют в отдельный поток (thread). Поток занимает свои регистры процессора и имеет свой стек (часть памяти, содержащая адрес для возвращения в подпро-

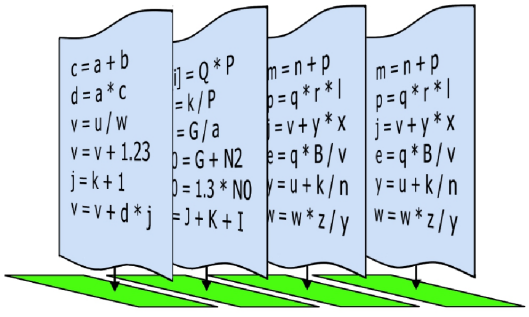

Рис. 6. Иллюстрация одновременного выполнения операций с использованием технологии многопоточности

цедуры и локальные изменения в этих подпроцедурах). При этом адресное пространство рассматриваемого потока является общим с другими потоками процессора. Потоки запускаются независимо и так же, как и процессы в операционной системе, ни один поток не знает о существовании и состоянии других потоков [11]. Однако благодаря использованию общей памяти можно сделать так, чтобы все потоки эффективно трудились над одной и той же задачей.

Другим важным преимуществом распределения одной задачи между несколькими потоками является возможность избежать простоя при операциях считывания/записи. Считывание из основной памяти — очень времязатратная процедура и может занимать до нескольких сотен тактов, при этом программа прекращает вычисление и ожидает необходимую информацию. Таким образом, процессор может простаивать достаточно длительное время. Если же мы имеем одновременно несколько запущенных потоков, то можно, сохранив все регистры для потока, который в настоящее время занят какими-то другими операциями (например считыванием из основной памяти), перейти на выполнение вычислений для другого потока.

Для использования описанного подхода алгоритм распознавания треков был распараллелен в части поиска триплетов. Для этого те участки алгоритма, которые отвечали отдельным этапам поиска триплетов, а именно нахождению одно-/двух-/трехотсчетных фрагментов, были выделены из общего цикла и разбиты на три соответствующие части, которые выполняются последовательно. Каждая из указанных частей содержит цикл по STS-детекторам. Отдельные итерации этих циклов были сделаны независимыми, что дало возможность их одновременного запуска на нескольких потоках. Поиск триплетов занимает 84 % времени работы алгоритма, потому ее распараллеливание теоретически (при использовании большого числа процессоров) позволяет ускорить выполнение алгоритма примерно в шесть раз.

Однако более эффективным оказался другой подход, заключающийся в запуске поиска треков в отдельных событиях на нескольких потоках, при этом одному событию выделяется только один поток. Для выполнения программ распознавания треков с применением технологии многопоточности использовался набор процедур из библиотеки Intel Threading Building Blocks (ITBB) [12].

4. ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ

Основной величиной, характеризующей работу алгоритма распознавания треков, является его эффективность. Для ее определения вводятся понятие найденного трека частицы — реконструированного трека, и понятие частицы, которая потенциально может быть найдена — потенциально реконструируемая частица. Часть множества потенциально реконструируемых частиц, которая находится алгоритмом, называется эффективностью алгоритма поиска. Множество потенциально реконструируемых частиц определяется задачами и аксептансом детектора, в котором проводится поиск. При условиях, которые будут иметь место в центральных столкновениях при энергиях порядка 25 ГэВ/нуклон в детекторах MVD и STS эксперимента CBM, невозможно отличить комбинации трех отсчетов, образованных одной частицей, от тех, которые случайно расположились вдоль траектории, подобной физической. Более того, число последних комбинаций превышает число первых на несколько порядков. Похожая проблема имеет место для треков

с импульсом до 0,1 ГэВ/c — они имеют большую кривизну и вылетают из чувствительного объема детектора, пересекая недостаточное количество станций. Учитывая это, для оценки алгоритма распознавания треков потенциально реконструируемыми было решено считать только частицы, которые пересекают четыре или большее число детекторных станций, а также имеют импульс выше 0,1 ГэВ/c.

Для тестирования алгоритма была проведена реконструкция сгенерированных с помощью МК-моделирования 100 центральных столкновений Au + Au при энергии налетающего ядра 25 ГэВ/нуклон, что отвечает максимально сложному сценарию (из-за количества образующихся вторичных частиц) для задачи распознавания треков. В качестве примера на рис. 7 приводится результат реконструкции для одного центрального события.

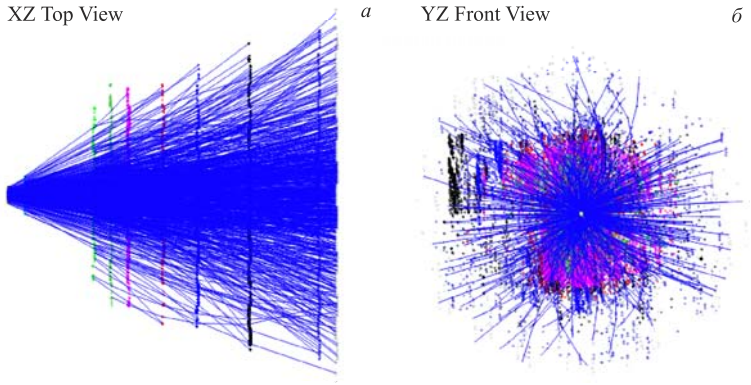

Рис. 7. Хиты и реконструированные треки для одного центрального столкновения Au + Au при энергии налетающего ядра 25 ГэВ/нуклон: число реконструированных треков равняется 729. *а*) Вид сверху, *б*) вид спереди. Серыми точками обозначены хиты, отвечающие местам прохождения реконструируемых частиц, черными — фоновые хиты. Реконструированные треки показаны синими ломаными линиями (цветная версия рисунка доступна на сайте www1.jinr.ru)

Следует отметить, что подавляющее большинство ядро-ядерных соударений в эксперименте CBM будет носить периферийный характер. По сделанным оценкам, доля центральных событий не превысит 1 %. В этой связи нами были также сгенерированы и реконструированы 1000 событий со случайным прицельным параметром. В дальшейшем такие (minimum bias) события будем называть смешанными.

4.1. Эффективности распознавания треков. Эффективности распознавания треков в зависимости от импульса регистрируемой частицы для смешанных и центральных событий представлены на рис. 8 и в таблице.

Для детального анализа работы алгоритма распознавания треков все частицы и треки были разбиты на несколько категорий. Прежде всего, сгенерированные методом МК частицы были разделены: 1) по импульсу — на быстрые ($p > 1$ ГэВ/c) и медленные ($0{,}1 < p < 1$ ГэВ/c); 2) по месту образования — на первичные, которые отвечают частицам, образовавшимся в первичной вершине, и вторичные, которые образовались в результате распада короткоживущих частиц. Реконструированный трек ассоциируется с определенной частицей в случае, если не менее 70 % его отсчетов было образовано взаимодействием этой частицы с детектором. Частица считается найденной, если она была ассоциирована по крайней мере с одним реконструированным треком. Если одной

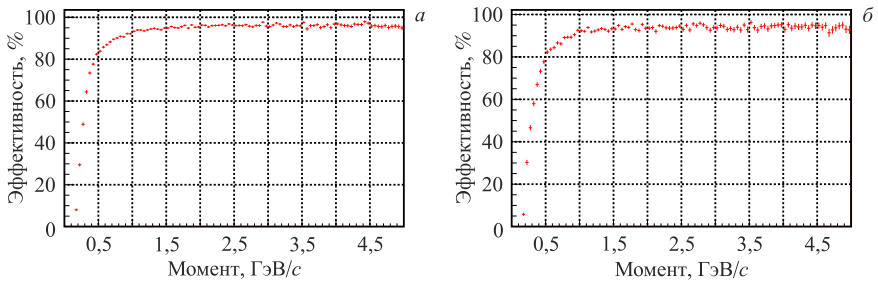

Рис. 8. Эффективности распознавания треков в зависимости от импульса частицы для смешанных (*а*) и центральных (*б*) событий

Эффективности распознавания для различных категорий треков, уровень клонов и гостов, число реконструированных треков, времена выполнения алгоритма

Параметр	Смешанные	Центральные
Эффективность, %		
быстрые первичные	97,4	95,5
медленные первичные	88,1	86,7
быстрые вторичные	81,0	78,3
медленные вторичные	47,9	45,1
все треки	88,1	86,3
Уровень, %		
клоны	0,3	0,4
госты	1,6	4,4
Число реконструированных треков	153	717
Время на событие, мс	25	220

частице было поставлено в соответствие больше одного реконструированного трека, все дополнительные треки считаются клонами. Реконструированный трек считается найденным некорректно и называется гостом (ghost), если ему не отвечает ни одна реальная частица.

Большинство представляющих интерес (сигнальных) треков — продуктов распадов D-мезонов, J/ψ-, ψ'-, φ- и ρ-частиц — являются высокоэнергичными, вылетают из области первичной вершины, и, таким образом, эффективность их поиска определяется категорией быстрых «первичных» частиц. Доля таких частиц в смешанных событиях составляет 97 %. Вторичные быстрые треки могут образовываться при распадах K^0_s, Λ, а также при каскадных распадах Ξ и Ω. Такие треки образуются вдалеке от мишени, они пересекают меньше станций, и для них невозможно использовать позицию мишени в качестве дополнительной информации. Как следствие, эффективность их поиска несколько меньше, чем для «первичных» частиц, — 81 %. Вторичные низкоэнергичные частицы могут образовываться при распадах Ξ- и Ω-гиперонов. Кроме того, реконструкция треков медленных первичных частиц важна для понижения уровня фона при реконструкции распадов векторных мезонов и J/ψ по электронному каналу. Одной из составляющих такого фона являются электроны, возникающие при распадах Далица (π^0) и γ. Много-

кратное рассеяние в материале детектора и большая кривизна траектории также приводят к дополнительным потерям при реконструкции медленных частиц — эффективности для первичных и вторичных частиц соответственно равны 88 и 48 %.

Усредненная по всем категориям частиц эффективность реконструкции треков составила 88 %, при этом доля клонов равна 0,3 %, а гостов — 2 %.

Центральные события значительно превосходят смешанные как по числу образующихся частиц, так и по плотности отсчетов в STS-детекторе. И, как следствие, они характеризуются более низкими эффективностями корректно реконструированных треков и большим процентом некорректно найденных треков. Усредненная по 100 центральным событиям эффективность распознавания треков составила 86 %, доля гостов выросла до 4 %.

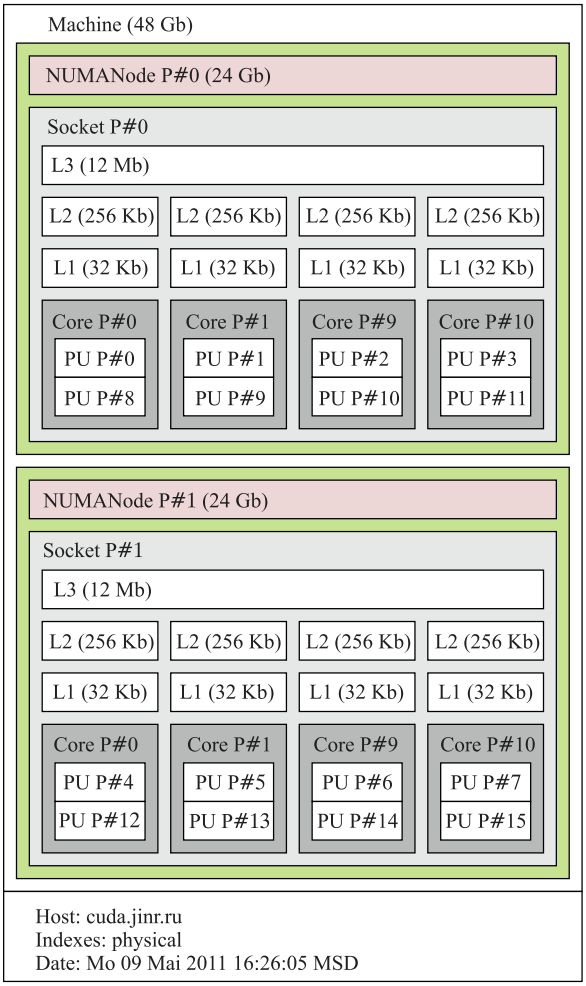

Рис. 9. Структура сервера cuda.jinr.ru: приведены распределения оперативной памяти между процессорами (розовый) и кэш-памяти для первого, второго и третьего уровней (белый). Каждому физическому ядру (темно-серый) отвечают два логических ядра (см. цветную версию рисунка на сайте www1.jinr.ru)

4.2. Масштабируемость алгоритма распознавания треков. Как отмечалось выше, в эксперименте CBM планируется провести полную реконструкцию событий в режиме реального времени. Поэтому скорость выполнения алгоритма распознавания треков, зарегистрированных в STS-детекторе, так же важна, как и эффективность алгоритма. В настоящее время в разработке специализированных высокопроизводительных серверов наблюдается тенденция как по увеличению числа процессоров, так и по росту числа ядер при постоянной тактовой частоте. В этой связи важно, чтобы с ростом числа ядер пропорционально росла скорость обработки событий, в нашем случае это скорость реконструкции треков.

Для тестирования временных характеристик алгоритма использовался 8-ядерный сервер cuda.jinr.ru (ЛИТ ОИЯИ) (рис. 9). Сервер имеет два процессора Intel Xeon E5640, каждый процессор включает четыре ядра с частотой 2,66 ГГц и 12 Мбайт кэш-памяти третьего уровня. Оперативная память размером в 45 Гбайт распределена поровну между двумя процессорами. Наличие технологии Hyper-threading делает возможным использование каждого физического ядра в виде двух логических ядер.

Средние времена, затрачиваемые на одно событие при запуске алгоритма на одном ядре сервера, приведены в таблице. Они оказались равными 25 и 220 мс для смешанных и центральных событий соотвественно.

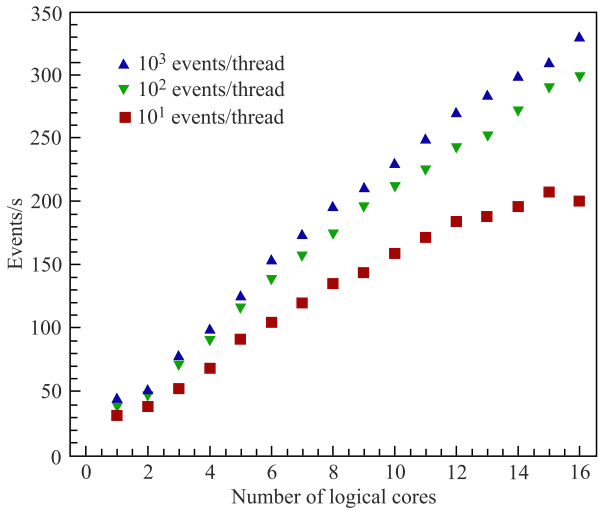

Рис. 10. Масштабируемость алгоритма распознавания треков для смешанных событий

Задача распознавания треков запускалась параллельно на разном числе логических ядер. На каждом ядре выполнялась реконструкция треков для 10, 100 и 1000 смешанных событий. Для организации параллельных вычислений с включением многих ядер использовалась библиотека Intel Threading Building Blocks (TBB) [12]. Полученные результаты масштабируемости алгоритма приведены на рис. 10. Видно, что для небольшого числа событий имеют место дополнительные расходы (overhead) на распределение процессов между ядрами. Для групп с большим числом событий наблюдается требуемая линейная масштабируемость алгоритма.

ЗАКЛЮЧЕНИЕ

На основе смоделированных событий, генерированных с помощью пакета GEANT3 [13] в среде CBM-ROOT [14], выполнена оценка эффективности и производительности алгоритма распознавания траекторий заряженных частиц на многоядерном сервере ЛИТ ОИЯИ. В основу алгоритма распознавания треков положены концепция клеточного автомата и метод фильтра Калмана. Несмотря на высокую множественность событий, интенсивный фон и неоднородное магнитное поле, алгоритм продемонстрировал высокую эффективность реконструкции треков — 96–97 % при низком уровне некорректно найденных треков — 2–4 %. Достигнута высокая скорость обработки событий на одном ядре, равная в среднем 220 мс на одно центральное событие и 25 мс на одно смешанное событие. Показано, что с увеличением числа включаемых в обработку ядер практически линейно растет количество обработанных событий.

Благодарности. Авторы благодарят И. В. Киселя за консультации и полезные обсуждения, Д. В. Белякова и А. М. Рапортиренко за техническую поддержку и помощь при выполнении исследований на сервере cuda.jinr.ru.

СПИСОК ЛИТЕРАТУРЫ

1. *Friman B. et al.* The CBM Physics Book // Lect. Notes in Phys. 2011. V. 814. P. 980.
2. *Kisel I.* Event Reconstruction in the CBM Experiment // Nucl. Instr. Meth. A. 2006. V. 566. P. 85–88.
3. *Bussa M. P. et al.* Application of a Cellular Automaton for Recognition of Straight Tracks in the Spectrometer DISTO // Comp. Math. Appl. 1997. V. 34, No. 7/8. P. 695–701.
4. *Bussa M. P. et al.* Application of CA and NN for Event Recognition in Experiments DISTO and STREAMER // Nucl. Instr. Meth. A. 1997. V. 389. P. 208–209.
5. *Gorbunov S., Kisel I.* Analytic Formula for Track Extrapolation in Non-Homogeneous Magnetic Field // Nucl. Instr. Meth. A. 2006. V. 559. P. 148–152.
6. *Gorbunov S. et al.* Fast SIMDized Kalman Filter Based Track Fit // Comp. Phys. Commun. 2008. V. 178. P. 374–383.
7. *Балакришнан А. В.* Теория фильтрации Калмана. М.: Мир, 1988;
   Balakrishnan A. V. Kalman Filtering Theory. N. Y., 1984.
8. *Kalman R. E., Bucy R. S.* New Results in Linear Filtering and Prediction Theory // J. Basic Engineering. 1960. V. 82. P. 35–40.
9. *Frühwirth R.* Application of Kalman Filtering to Track and Vertex Fitting // Nucl. Instr. Meth. A. 1987. V. 262. P. 444–450.
10. IA-32 Intel Architecture Optimization Reference Manual. Intel, 2005.
11. *Reinders J.* Intel Threading Building Blocks. O'Reilly Media Inc., 2007.
12. Intel Threading Building Blocks. http://www.threadingbuildingblocks.org
13. GEANT — Detector Description and Simulation Tool. CERN Program Library, Long Write-up W5013. 1995.
14. *Bertini D. et al.* The FAIR Simulation and Analysis Framework // Proc. of Intern. Conf. on «Computing in High Energy and Nuclear Physics», Victoria, BC Canada, 2007.

Получено 23 апреля 2012 г.