Компьютерное моделирование физических процессов в детекторах с использованием пакета Geant4

М. Демичев, А. Жемчугов

2011

Контактная информация

Жемчугов Алексей Сергеевич
zhemchugov @ jinr,ru
ЛЯП корпус 4, комната 325
телефон 62363

Демичев Михаил Андреевич
mdemichev @ jinr,ru
ЛЯП, корпус 4, комната 332
телефон 63462

Содержание курса

Цели моделирования физических процессов в детекторах

Основы метода Монте-Карло

Устройство и работа с пакетом Geant4

Написание простой программы моделирования

Литература

Метод Монте-Карло

И.М.Соболь Численные методы Монте-Карло. Наука, М. 1973

Е.Бюклинг, К.Каянти Кинематика элементарных частиц, Мир, М. 1975
http://mipt.jinr.ru/base/lectures.html

Документация Geant4

Geant4 User's Guide for Application Developers
Geant4 Physics Reference Manual
Geant4 User's Guide for Toolkit Developers

http://geant4.web.cern.ch/geant4/support/userdocuments.shtml

Цели и способы моделирования процессов в детекторе

Методы Монте-Карло

Методы Монте-Карло – это численные методы решения прикладных математических задач при помощи моделирования случайных величин.

Идея родилась при работе над “Манхэттенским проектом”

S.M. Ulam, J. von Neumann, “On combination of stochastic and deterministic processes”.
Bull. Amer. Math. Soc. 53 1120 (1947)

S.M. Ulam, N. Metropolis, “The Monte-Carlo method”, J. Amer. Statist. Assoc. 1949 , 44 Vol 247, 335-341

Применение метода Монте-Карло в физике частиц (1)

• Эксперимент в физике частиц представляет собой, как правило, многократное измерение совокупности случайных процессов – взаимодействия частиц и прохождения их через детектор

• Моделирование случайных процессов сводится к моделированию дискретных случайных величин с последующим преобразованием

• Метод Монте-Карло дает удобный способ расчета результатов такого измерения

Применение метода Монте-Карло в физике частиц (2)

Задачи, решаемые методом Монте-Карло

• Моделирование наблюдаемых величин (импульс, энегрия, точка рождения ...) на основе теоретических предсказаний – генераторы событий

• Моделирование искажения наблюдаемых величин в результате прохождения частиц через вещество детектора

• Моделирование отклика детектора для отладки алгоритмов реконструкции событий

Применение метода Монте-Карло в физике частиц (2)

Задачи, решаемые методом Монте-Карло

• Моделирование наблюдаемых величин (импульс, энегрия, точка рождения ...) на основе теоретических предсказаний – генераторы событий

• Моделирование искажения наблюдаемых величин в результате прохождения частиц через вещество детектора

• Моделирование отклика детектора для отладки алгоритмов реконструкции событий

Цели моделирования

- Оптимизация элементов детектора
- Отладка алгоритмов реконструкции событий
- Расчет ожидаемых значений сигнала и фоновых процессов. Оценка ожидаемой точности измерений
- Отладка методики анализа данных, определение наилучших критериев отбора событий

История развития специализированных программ моделирования

- **1950е - середина 60х** – первые программы расчета взаимодействия частиц в веществе
- **середина 60х – начало 70х** – программы моделирования э/м и адронных каскадов (EGS, GHEISHA, FLUKA ...)
- **1974 –** GEANT
- **1982** – GEANT3
- **1993** - GEANT4

В настоящее время в физике частиц наиболее широко применяются

- **GEANT3**
- **FLUKA**
- **GEANT4**

Однако существует множество других программ (в том числе коммерческих) для расчета прохождения частиц через вещество

MCNP (Monte-Carlo N-Particle Transport Code) – написана в LANL (Лос-Аламос), первоначально применялась для расчета реакторов

MARS – расчет ливней, FNAL

CALOR95 – расчет ливней и источников нейтронов, ORNL (Оак-Ридж)

CORSIKA, AIRES – расчет широких атмосферных ливней

многие другие ...

FLUKA (FLUktuirende KAskade)

- Первоначально разрабатывалась для расчета защиты в проекте SPS в ЦЕРН (1962-1978)
- В настоящее время универсальная программа расчета взаимодействия частиц с веществом
- Написана на языке Фортран.
- Хорошее моделирование адронных ливней
- Сложность описания геометрии
- Лицензионные ограничения

MCNP (Monte-Carlo N-particle transport code)

- Разработана в Лос-Аламосе для расчета реакторов
- В настоящее время универсальная программа расчета взаимодействия частиц с веществом
- Написана на языке Фортран.
- Хорошее моделирование нейтронных процессов, и процессов при низких энергиях
- Оригинальный подход к описанию геометрии
- Лицензионные и экспортные ограничения
- После выхода версии 4 разделилась на две ветви: MCNP5 и MCNPX (MCNP+LAHET)

GEANT3
(GEometry ANd Tracking)

- Первая версия появилась в 1974 году в ЦЕРН
- Описание физических процессов основано на программах EGS (э/м ливни) и GHEISHA (адронные ливни)
- Пакет GEANT3 появился в 1982 году и был использован для моделирования детекторов в экспериментах на LEP
- Написан на языке Фортран
- Основной инструмент моделирования в физике частиц на протяжении 30 лет

GEANT4

- Объектно-ориентированная программа с функциональностью GEANT3
- Первая версия пакета появилась в 1995 году
- Написана на языке C++
- Первое “боевое” применение – эксперимент BaBar
- С 2004 года – основная программа моделирования в экспериментах на LHC (кроме ALICE)
- Широкое применение в физике частиц, космонавтике (ESA), радиационной медицине.

Пакет программ GEANT4

GEANT4 представляет собой набор программ для моделирования прохождения частиц через вещество
Включает в себя инструменты для гибкого описания геометрии
Содержит множество физических моделей взаимодействия частиц с веществом

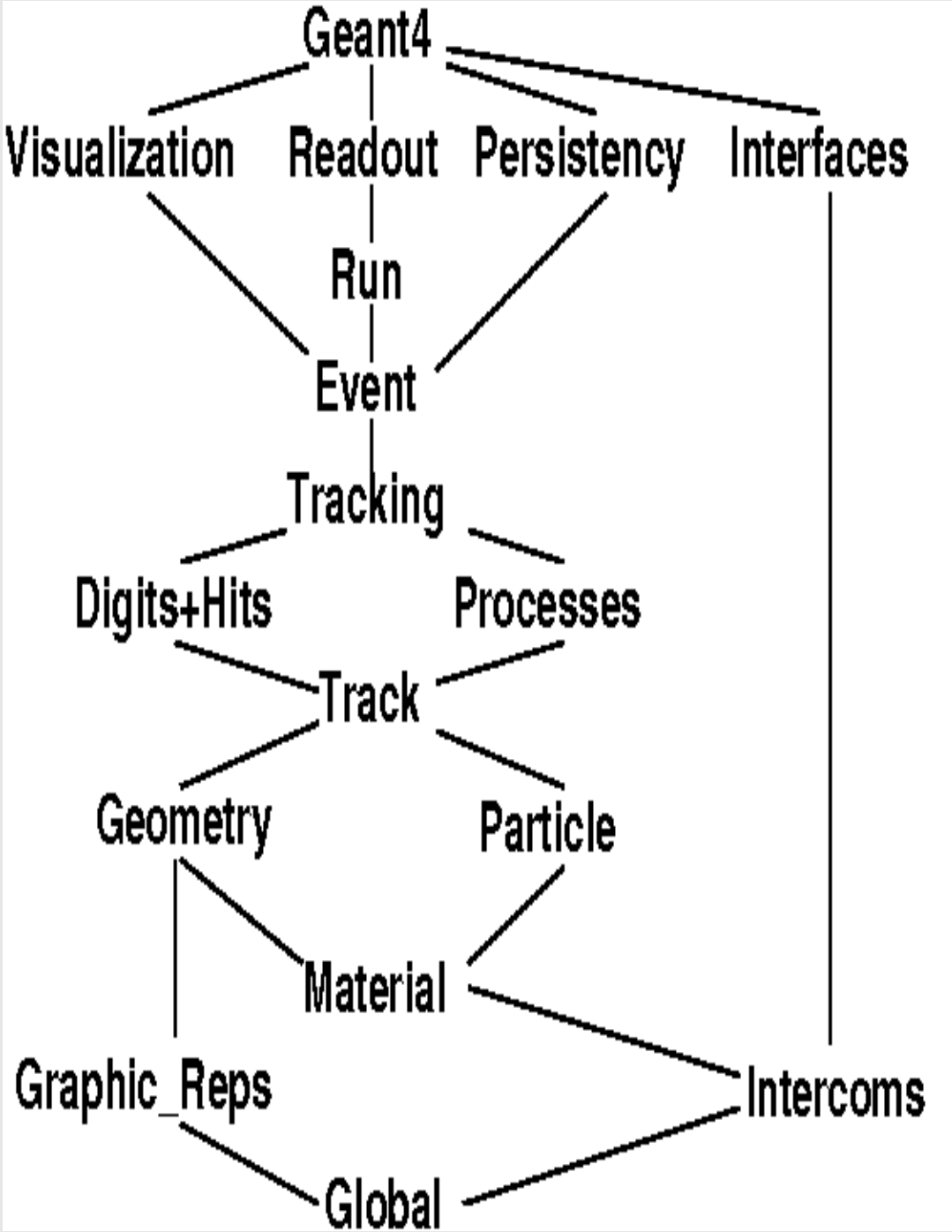
- **Электромагнитные процессы**
- **Адронные процессы**
- **Фотон-адронные и лептон-адронные процессы**
- **Процессы с участием оптических фотонов**
- **Моделирование распадов**
- **Параметризация ливней**
- **Методики использования статистических весов**

Ядро Geant4

Классы, описывающие основные понятия:
сеанс, событие, трек, шаг срабатывание, траектория

Базовый механизм

- описания геометрии
- описания физических процессов
- визуализации и интерфейсов пользователя



Системные требования

Основные платформы и компиляторы

Linux + gcc

Windows + VisualC++ (с использованием эмулятора Cygwin)

Solaris + SunCC

MacOS + gcc4

IA32-IA64 + icc8

Внешние библиотеки

CLHEP

графические библиотеки (опционально)

Требует около 1.2 Гб дискового пространства (полная установка)

Требует минимум 128 Мб оперативной памяти для работы

Документация и исходный код

http://cern.ch/geant4

Основные публикации

S. Agostinelli et al.,
Geant4: a simulation toolkit, NIM A 506 (2003) 250-303

J. Allison et al.,
Geant4 developments and applications
IEEE Trans. Nucl. Sci. 53 No. 1 (2006) 270-278

Основные понятия Geant4

Сеанс (Run)

Период набора статистики, в котором не меняются условия проведения эксперимента (параметры пучка, конфигурация и параметры детектора, материал мишени и т.п.)

В Geant4 – самый крупный элемент моделирования, состоящий из последовательности событий. Во время сеанса описание геометрии и набор физических процессов остаются неизменными.

Представлен классом G4Run

Управление осуществляется объектом класса G4RunManager

Событие (Event)

Единичное независимое измерение физического явления детектором.

В Geant4 представлено классом G4Event

G4Event содержит все входные и выходные характеристики (исходные частицы, срабатывания и т.д.) смоделированного события

G4Event создается объектом класса G4RunManager и передается объекту класса G4EventManager, который осуществляет управление событием

Структура события

Первичная вершина и первичная частица

Траектории

Коллекция срабатываний

Коллекция оцифрованных сигналов

Кроме того, G4EventManager управляет объектами G4Track соответствующими данному событию, взаимодействуя с объектами классов G4TrackManager и G4StackManager

Трек (Track) и шаг (Step)

Шаг (G4Step) описывает минимальное продвижение частицы через вещество с учетом различных физических процессов.

Треки представлены классом G4Track, и содержат информацию о последнем шаге.

Объект G4Track, таким образом, описывает полное продвижение частицы в веществе к моменту обращения к данному объекту

Срабатывание (Hit)

Описывает единичное взаимодействие частицы с веществом в детектирующем объеме

Содержит информацию о координате и времени взаимодействия, энергии и импульсе частицы в этой точке, энерговыделении, геометрическую информацию (объем, в котором произошло взаимодействие и т.п.)

Служит исходной информацией для моделирования оцифрованного сигнала

Является “истинной” Монте-Карло информацией (Monte-Carlo truth)

Оцифрованный сигнал (Digi)

Моделируется на основе срабатываний, и содержит информацию в виде “канал-сигнал”

Один оцифрованный сигнал может быть результатом нескольких срабатываний (например, несколько треков прошли через одну ячейку калориметра)

Является полным аналогом измеряемых в реальном эксперименте величин

Будучи необязательным для моделирования, служит в основном для отладки алгоритмов реконструкции

Создание простой программы моделирования

Geant4 – это набор инструментов

Создание исполняемой программы моделирования установки – задача конечного пользователя

Geant4 предоставляет необходимые библиотеки и сервисы, включая визуализацию и возможность интерактивного режима работы

В простейшем случае от пользователя требуется описание геометрии детектора, списка физических процессов, учитываемых в моделировании и генерация первичной вершины

Простая программа моделирования

```
#include "G4RunManager.hh"
#include "G4UImanager.hh"
#include "ExN01DetectorConstruction.hh"
#include "ExN01PhysicsList.hh"
#include "ExN01PrimaryGeneratorAction.hh"
int main()
{
  // construct the default run manager
  G4RunManager* runManager = new G4RunManager;
  // set mandatory initialization classes
  runManager->SetUserInitialization(new ExN01DetectorConstruction);
  runManager->SetUserInitialization(new ExN01PhysicsList);
  // set mandatory user action class
  runManager->SetUserAction(new ExN01PrimaryGeneratorAction);
  // initialize G4 kernel
  runManager->initialize();
  // get the pointer to the UI manager and set verbosities
  G4UImanager* UI = G4UImanager::GetUIpointer();
  UI->ApplyCommand("/run/verbose 1");
  UI->ApplyCommand("/event/verbose 1");
  UI->ApplyCommand("/tracking/verbose 1");
  // start a run
  int numberOfEvent = 3;
  runManager->BeamOn(numberOfEvent);
  // job termination
  delete runManager;
  return 0;
}
```