CDF Monte Carlo'2000

P.Murat, CDF colalboration

Fermilab Batavia IL 60510

Abstract. We present status of CDF Run II Monte Carlo project which is being developed for about 3 years. We discuss general architecture of the project, its major components, performance and future prospects.

INTRODUCTION

Detector simulation code written for any detector reflects specifics of this detector. The heart of the CDF Run II detector (1) is its excellent tracking system. The CDF tracker consists of the Si vertex detector (SVX) with 8 layers of double-sided silicon, inner of which is positioned at $R \simeq 1.5\ cm$ from the beam and the outer - at $R \simeq 30\ cm$q. SVX provides resolution of about $7 \oplus 21/P_t$ microns in impact parameter. It is positioned inside the central drift chamber (COT) which extends up to R=135cm in radius and provides up to 96 space measurements with the single-hit resolution of about 150 microns. The COT is surrounded by the layer of TOF counters with the resolution of 120ps which significantly improve particle ID capabilities of the detector. CDF has modest but adequate sandwich calorimeter with EM resolution of $\frac{13.5\%}{\sqrt(E)}$ and HAD resolution of $\frac{80\%}{\sqrt(E)}$ and a robust muon system which covers the calorimeter with 4-8 layers of drift chambers. Luminosity is monitored by 2 hodoscopes of the Cherenkov counters with the timing resolution of about 50ps. To simulate the detector performance adequately, the CDF Monte Carlo does detailed hit-level simulation of the tracker and muon systems, and, to improve the performance uses fast parametrizations to simulate the calorimeter response.

OVERALL DESIGN

CDF Analysis Framework, AC++ (2), is written in C++. One of the key features of AC++ is its modularity, meaning that each data processing algorithm is implemented as a software module and one can easily plug the algorithms in to and out of the event loop. Following this scheme the detector simulation code is implemented as an AC++ module and every MC event generator also has a corresponding module as well. All the event generator modules and the detector simulation module are combined into a common CDF MC executable. Data structures produced by the event generators have unified format and the detector simulation takes these structures as input. Such organisation significantly simplifies debugging and also allows to avoid writing out intermediate files with the event generator output. However the functionality of writing out the files with the events generated but not simulated in the detector is fully preserved. When the reconstruction code becomes more stable, we plan to run event generation, detector simulation and reconstruction in a single job.

GEOMETRY MODELLER / TRACING ENGINE

CDF MC is using GEANT3 (3) as an underlying geometry modelling and tracing engine. To interface GEANT3 to CDF offline framework, we put a thin layer of C++ code on the top of GEANT3. Currently this interface is given by TGeant3 class (4), which is substantially shared with the ALICE collaboration. Digitization code has been implemented in C++ from the very beginning. The choice of GEANT3 which has its algorithms debugged over the decades, allowed CDF to build its detector simulation framework on a very solid foundation. Presently we are making an effort to unify geometry description in the simulation and the reconstruction code by using GEANT4 (5) geometry classes on top of GEANT3. The goal of this effort is to use GEANT3 tracing utilities for extrapolating the muons in very complex geometry of the CDF muon system and also to provide a migration path to GEANT4.

PARAMETRIZED CALORIMETRY SIMULATION

CDF has a conventional iron(lead)/scintillator sandwich calorimeter. To simulate its performance we are using GFLASH (6) - a parametrized calorimetry simulation code developed within H1 collaboration. GFLASH is based on a parametrized shower model, both longitudinal and transversal profiles are modelled as functions of the incident particle energy and properties of the tracking medium. Important feature of GFLASH implementation is that it is fully integrated with GEANT3: it uses GEANT3 geometry and other data structures, such that given a tracking volume it is up to user to decide whether to simulate shower development in this volume using fast GFLASH or detailed GEANT3 algorithms. Another feature of GFLASH is that it has many handles to tune the performance. We found that with the proper choice of constants it is even possible to use GFLASH to simulate the response of the CDF showermax detectors - the scintillation strips in the plug calorimeter and the gas chambers in the central calorimeter.

It is also worth to comment on a common limitation of the fast calorimetry simulations: because of the approximations used they usually do not work well in complex inhomogeneous geometries with the calorimeter-type compartments separated in space.

EVENT GENERATORS

CDF is using various event generator (EG) packages:

- widely used in HEP event generators, such as ISAJET, PYTHIA, HERWIG, which generate events with equal weights
- VEGAS-type generators, such as VECBOS, DYRAD, WGRAD, which on output produce events with different weights
- different generators of minimum-bias events
- single particle gun, used for debugging purposes

Specifics of the event generator packages is that all of them are written in FORTRAN and require special interfaces to C++. In addition, for each event generator package there exists an AC++ module, interfacing the generator to the CDF offline framework. All the EG modules have unified interface to the particle decay packages TAUOLA (7) and QQ(8) and independent of which primary EG package is used to generate the events, the same switches allow to use QQ and TAUOLA to decay B's and τ's.

RANDOM NUMBER GENERATORS

As large detector simulation programs are using random numbers for many different purposes and to ensure that the code is producing the same results on different platforms it is very important to manage the random numbers on a consistent basis.

CDF Monte Carlo is using a single random number (RN) manager, which maintains multiple RN streams and is responsible for saving their state to a data stream when requested. Thus it is possible to resimulate once simulated events starting from the last one for which the random numbers have been saved and get the identical results.

Many external to CDF packages, such as GEANT3 and MC event generators, are using their own implementations of RN generators. To guarantee the reproducibility of the results, all the external RN generators have been identified and substituted with the implementations based on the CDF RN manager, such that all the random numbers in CDF Monte Carlo code are coming from the streams in a single place. No exception from this rule is allowed.

OUTPUT FORMATS

The typical output of the detector simulation program in HEP is the simulated event data (hits, digits) and also so-called “MC truth” information which links the hits, generated in the detector, to their parent particles. The CDF detector simulation writes the event data out in the format identical to that of CDF DAQ system. The driving requirement here is that CDF Run II DAQ is 100% digital and it should be possible to reconstruct trigger decisions for the simulated events and for the real data using the same procedure.

PERFORMANCE AND MOCK DATA CHALLENGES

We expect the bulk of the Monte Carlo calculations in Run II to be performed on the Linux-based PC farms, so we tested the performance of the code using 500-750MHz PIII Linux boxes with 256-512 MBytes of memory and 100MHz bus. With all the code compiled in debug mode, the size of the simulation executable on disk is about 100 MBytes. When the debug symbols are stripped, the size of the executable goes down to about 45 MBytes. The executable size in memory is about 50 MBytes at startup, 10-20 MBytes are allocated dynamically during the execution. Time necessary to simulate one event ranges from about 1sec for the single 45 GeV electron events to about 10 sec for the $t\bar{t}$ events (all the code compiled in debug mode). We expect further improvement in timing by a factor of about 2 when the code is compiled in optimized mode. This level of performance is adequate for generating any volume of MC statistics necessary for developing and debugging the reconstruction code and for studying the physics signals in various channels.

We'd like to emphasize that simulating a $t\bar{t}$ event in the CDF tracker takes about 2-3 sec, thus to improve the overall performance further one has to use significantly different approach to tracing or much simplier geometry model.

In fall of 1999 and in summer of 2000 the CDF Run II Monte Carlo system has been used to generate input data for the 2 Mock Data Challenges. It turned out to be very stable - millions of events and Terabytes of data were simulated without a single crash.

CONCLUSIONS

In conclusion, CDF experiment has developed and stable Monte Carlo system, which provides detailed hit-level simulation of the tracker and the muon detectors and relies on fast parametrized algorithms to simulate the calorimeter response. Combining detailed and parametrized algorithms allowed to achieve quite impressive level of performance - a $t\bar{t}$ event is fully simulated in about 10 sec on 750MHz Linux box. The system is very stable such that millions of events corresponding to the Terabytes of data are simulated without a single crash and the CDF Monte Carlo code is ready to be tuned when the first Run II data will become available in spring'2001.

REFERENCES

1. The CDF II Collaboration. The CDF-II Detector Technical Design Report, FERMILAB-Pub-96/390-E
2. http://www-cdf.fnal.gov/upgrades/computing/projects/framework/framework.html
3. CERN Program Library Long Writeup W5013, PM0062. An html-based version is available from http://wwwinfo.cern.ch/asdoc/geant_html3/geantall.html
4. http://purdue-cdf.fnal.gov/CdfCode/source/geant_i/TGeant3.h
5. home page of GEANT4 project is http://wwwinfo.cern.ch/asd/geant4/description.html
6. G.Grindhammer, S. Peters, The Parameterized Simulation of Electromagnetic Showers in Homogeneous and Sampling Calorimeters, NIM A290(1990)469-488, also hep-ex/0001020
7. S. Jadach, Z. Was, R. Decker, J.H. Kuehn, The tau decay library TAUOLA: version 2.4 , Comp. Phys. Commun. 76 (1993) 361
8. QQ home page: http://www.lns.cornell.edu/public/CLEO/soft/QQ/index.html