Presented at the 5th International Linear Collider Workshop (LCWS 2000), Oct 24-28, 2000, Batavia, IL SLAC-PUB-10799

Physics and Detector Simulations

Norman A. Graf

Stanford Linear Accelerator Center

Abstract. A brief summary and synthesis of topics presented during the tools parallel sessions.

INTRODUCTION

The simulation tools session was divided into three main areas of concentration: physics event simulation and event generators, full detector simulation and event reconstruction frameworks, and fast simulation and physics analysis frameworks. Although the primary purpose of the plenary session talk was to summarize the contents of the parallel session presentations for those unable to attend, the intent of this write-up is to attempt to point out features in common among the ongoing efforts and present a personal view of some goals for the future. The reader is directed to the individual write-ups for details on the specific topics discussed during the workshop.

GOALS

As the linear collider community gets closer to the realization of an actual accelerator and detectors, the demands placed upon the simulation tools become more stringent. Design reports require detailed plans demonstrating both the technical feasibility and cost worthiness of the various accelerator and detector components. Physics analyses should be based on a full consideration of not only the signals of interest, but also on the backgrounds which can mimic or spoil such signatures. The analyses should also reflect the limitations of the hardware and software in accumulating and providing the data to the physicists. To answer all of these questions in a timely and accurate manner requires very sophisticated tools, which oftentimes are not immediately available. The common goals of the linear collider physics simulation tools group are, therefore, to provide full simulation of the physics program: event generators for physics signals and backgrounds, full detector simulations and reconstruction and analysis frameworks. One needs the flexibility to account for new theoretical processes, detector geometries and technologies and different reconstruction and analysis algorithms. Limited resources demand efficient solutions and focused effort. As is demonstrated by the breadth and depth of the presentations during the tools sessions, much of that functionality is now available to physicists to assist in their investigations. In the following sessions, the broad objectives of the three session branches are presented in more detail.

Work supported by the Department of Energy contract DE-AC02-76SF00515.

Stanford Linear Accelerator Center, Stanford University, Stanford, California, 94309

Physics Event Generators

Monte Carlo event generators should accurately represent signal events from various theoretical models as well as Standard Model backgrounds.

To correctly model the backgrounds arising from Standard Model processes requires the inclusion of higher-order processes in a gauge-invariant fashion. This leads to the inclusion of an enormous number of often-redundant Feynman diagrams. One approach to the solution of these problems was presented by Thorsten Ohl. O'mega is a tree-level scattering-amplitude generator which factorizes the sum of Feynman diagrams into a series of one-particle off-shell wave functions. The resulting expression is numerically more stable, since the sensitive gauge-induced cancellations have been accounted for more naturally. The electroweak sector of the Standard Model has been implemented, with Fortran source generation. Extensions beyond the Standard model and source generation in C++ and Java were promised. Having generated the scattering amplitudes, one still needs to efficiently sample from them to provide unweighted events for the event generator. The phase-space sampling program WHIZARD was described as the solution to this problem. The driver also allows for leading-order initial-state-radiation (ISR), beamstrahlung (via CIRCE) and beam polarization.

Recent additions to the de facto standard event generator PYTHIA were outlined by Torbjörn Sjöstrand. There now exists a complete framework for photon-photon interactions as well as improved parton shower descriptions. Extensions to SUSY, Technicolor, compositeness and extra dimensions were advertised. The implementation of PYTHIA in object-oriented C++ was mentioned as a collaborative effort with the authors of HERWIG.

An object-oriented approach to providing a generic framework for the generation of events specific to the linear collider environment was presented by Michael Peskin. Its aim is to provide the infrastructure necessary to efficiently generate both Standard Model and exotic events, fully accounting for polarization effects, beamstrahlung and ISR and spin correlations and asymmetries. The Pandora event generator is written in object-oriented C++. The modular, extensible character is intended to allow easy implementation of arbitrary hard scattering processes. It is currently interfaced to PYTHIA to handle parton evolution and showering.

It is clear that the complexity of the simulations has increased quite markedly over the past few years, and that the trend in event generators is toward more modular construction and automatic calculation of complex processes. Whether one can manage to incorporate the various functionalities within a single framework remains to be seen, but collaborative efforts have begun and should be encouraged. One major topic which has yet to be addressed is the question of persistence. The value of a common, experiment-independent data format for storing Monte-Carlo events cannot be overstated.

Full Detector Simulations and Event Reconstruction

Although many studies can be conducted using "back of the envelope" calculations and simplified simulations, in order to convincingly design a realistic detector one

needs the ability to fully simulate the response of the system as a whole to physics events and machine backgrounds. Detector elements need to be represented in their entirety, including support structures and readout elements. In addition to detailed detector descriptions, a complete accounting of the physics processes needs to be in place. Particle interactions with the materials and fields of the detector need to be correctly simulated, from propagation of tracks through central magnetic fields, to showering of particles in calorimeters. Finally, the signals recorded by the various readout technologies, including noise and inefficiencies, should be accurately reproduced. All of these components are essential for detector development. However, the data must also be useful for physics analysis, so the reconstruction framework and implementation of algorithms needs also to be in place, assuring that the technology ultimately chosen is capable of delivering the desired level of physics. In principle, this involves pattern recognition from the raw, digitized data through to the reconstructed four-momenta of initial-state partons. In practice, such a level of complexity is normally only achieved much later in an experiment's lifetime. It is the difficult task of the simulation groups to provide as much of the desired functionality as soon as possible, and in as user-friendly a fashion as possible; ideally without constraining the eventual completion and extension of the basic framework. (In practice, the main use of the full reconstruction for physics analyses has been to provide the parameterizations for the fast Monte Carlo simulations.) The three regional groups have taken different paths in their realizations of these requirements. A brief overview is given of each approach, hoping to recognize elements in common to all, and point perhaps to a universal solution.

The JLC full simulation package, JIM, is based on GEANT3, using ZEBRA for memory management, but has been encapsulated within the ROOT analysis framework. Two detector possibilities have been simulated, with realistic detector components. The reconstruction programs write ntuples as their output, which are then further analyzed within the ROOT framework. Work is currently underway to create detector descriptions using GEANT4.

The TESLA full simulation package, BRAHMS, was narrowly focused to complete a technical design report based on a well-defined detector. A conscious effort was made to re-use as much existing software as possible in order to arrive at the TDR within a short time. The detector was modeled in GEANT3 and reconstruction and analysis code was written in Fortran. Plain-text ASCII files were used for a platform-independent IO format. Most of the reconstruction code was adapted from the LEP experiments: TPC pattern recognition from ALEPH, VTX track-finding from OPAL, global track fitting from DELPHI, similarly for much of the calorimeter software. As such, relatively mature algorithms were available for event reconstruction. Additionally, the calorimeter was successfully modeled using GEANT4 in a C++ environment. Very sophisticated algorithms for calorimeter cluster reconstruction and association with central tracks are available, forming the basis for the energy-flow technique of jet reconstruction.

The LCD group decided early on to develop a purely object-oriented simulation and analysis framework using C++ and Java. As GEANT4 was unavailable, the full simulation was done using the Gismo package. Flexibility was one of the primary design criteria for the framework, since it was intended that a variety of detector

models should be compared. The models are described by XML files, which are parsed into different geometries by a single executable. Hit smearing is done at the reconstruction level, allowing maximal flexibility for modeling different detector responses. Compressed ASCII files are the model for data persistence. The full-reconstruction takes place within the Java Analysis Studio (JAS). Reconstruction within the ROOT framework can also be undertaken, limited only by manpower resources.

Fast Detector Simulations and Physics Analysis

Although the full simulation and reconstruction capabilities for all three regions are quite advanced, most of the physics analyses have been conducted using so-called "fast" detector simulations.

The JLC Quicksim package allows the geometry, resolutions and efficiencies to be varied via run-time parameter files. Monte Carlo particles are swum through the tracking volume and smeared by the effects of multiple coulomb scattering. Tracks in the vertex detector and central drift chamber are smeared covariantly as appropriate for the material and number of measurements and measurement resolutions. Smeared hits are created in the intermediate tracking chamber. The response of the calorimeter is simulated by smearing the particle energy and distributing the energy exponentially over towers in the immediate lateral neighborhood of the point of entry. Electrons and photons deposit energy only in the EM calorimeter, whereas hadrons cause a signal only in the hadronic section. Muons leave no energy in the calorimeters. The simulated data is output in ntuple format, available for physics analysis, which takes place within the ROOT framework.

TESLA's parametric fast Monte Carlo, SIMDET, smears the detector response to particles using functional forms determined from the full BRAHMS simulation and reconstruction. Tracks are smeared (albeit not covariantly) as a function of azimuthal angle and momentum. The calorimeter response is modeled by smearing the energy and distributing it laterally amongst the neighboring towers. Reconstruction code is incorporated to find and resolve calorimeter clusters, which are then linked to smeared tracks to provide energy-flow objects, which form the constituents of the jet-finding algorithms. Rudimentary particle ID, based on the presence or absence of tracks and hadronic or EM clusters, is simulated with efficiencies and fake rates determined from the full reconstruction. The results are then written out in an ASCII format and serve as input to the analysis stage, normally within the PAW framework.

The North American fast simulations are supported within both the JAS and ROOT frameworks. Tracks are smeared using the full covariance matrix appropriate for the energy and angle of the Monte Carlo particle. Calorimeter clusters are then created with smeared energy and position. Further analysis, such as jet-finding, topological vertex finding and jet flavor tagging, proceed as for the full reconstruction.

Visualization

Event displays are critical when implementing new detector setups or testing reconstruction algorithms. Accordingly, visualization is an integral part of all the

analysis packages, allowing physicists to view geometries, reconstructed tracks, calorimeter clusters, jets, etc. There is, however, no common visualization package or event graphics format. Granting that no single technology is likely to be adopted, I would still argue that a common interface needs to be developed which would allow interoperability of the different solutions.

THE FUTURE

Although the fast simulation programs have been used quite extensively to make the physics case for a linear collider, I believe the need is now to optimize the detector designs using more realistic physics studies based on full simulations and event reconstruction. Realistic backgrounds arising from both the machine and the detectors need to be considered. Inefficiencies arising from detector readout, hit merging, etc. should also be correctly accounted for. Pattern recognition errors, leading to missing or fake tracks, must be incorporated into the design of robust tracking detectors. Doing all this realistically, while changing detector designs and reconstruction algorithms is a daunting task. With limited resources, there should be a concerted effort amongst the groups to share not only ideas and algorithms but also code. Object-oriented methodology provides the necessary flexibility to efficiently study multiple designs within a single framework, allowing fair, objective comparisons to be made. GEANT4 is seen, correctly, as the standard tool for full detector simulations, and various efforts have been undertaken by the regional groups to model their detector designs using it. There has been, however, very little direct collaboration between the groups and the resulting code has not been reusable or shareable. I would argue strongly that the time is ripe to draw up a set of design requirements which would satisfy all the groups. I believe that a software package could be developed which would be sufficiently flexible, at run-time, to fulfill the full simulation needs of all three regional groups. The benefits of a common executable and data format are well worth the time and effort to design the interfaces. Currently, there are two main reconstruction and analysis packages suggested as replacements for the Fortran-based PAW package: ROOT, based on C++, and JAS, which uses Java. Here again, some design considerations early on can simplify the interaction between the two systems. Being able to access data or methods from either of the packages would go a long way towards efficiently utilizing scarce manpower resources.

ACKNOWLEDGEMENTS

I would like to thank the speakers of this session for their contributions and the session co-conveners, Keisuke Fujii and Martin Pohl, for their efforts in organizing the presentations.