Fermi National Accelerator Laboratory

FERMILAB-Conf-88/16-E
[E-740]

On Micro Vax Farms and Shower Libraries - Monte Carlo Techniques Developed for the D0 Detector*

Rajendran Raja
Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

January 1988

*Invited talk given at the 1987 Workshop on Detector Simulation for the SSC, Argonne, Illinois, August 24-28, 1987. To be published in proceedings of same.

Operated by Universities Research Association Inc. under contract with the United States Department of Energy

On Micro Vax farms and Shower libraries - Monte Carlo techniques developed for the D0 detector[1]

Rajendran Raja
Fermi National Accelerator Laboratory[2]
Batavia, Illinois, 60510, USA

January 1988

Abstract

In order to predict correctly the effects of cracks and dead material in a nearly hermetic calorimeter, hadronic and electromagnetic showers need to be simulated accurately on a particle by particle basis. Tracking all the particles of all showers in the calorimeter leads to very large CPU times (typically 5 Hours on a VAX780) for events at $\sqrt{(s)} = 2TeV$. Parametrizing the energy deposition of electromagnetic particles in showers with energy below 200 MeV [1] results in event times of the order of 1 Hour on a VAX780. This is still unacceptably large. The D0 collaboration then employed a farm of 16 MicroVax II 's [2] to get acceptable throughputs. The calorimeter hit patterns of each individual track was output , to be summed up by a later job. These individual hit patterns were entered into a random access shower library file[3], which was then used for subsequent Monte Carlo simulations. This shower library technique results in further speed-ups of a factor of 60 without degrading the quality of simulation significantly.

---

[1]Invited talk given at the 1987 Workshop on SSC detector Simulation, Argonne National Laboratory

[2]Fermilab is operated by the Universities Research Association Inc. under contract with the U.S Department of Energy

Running a D0 GEANT in a Farm of large MicroVAX–II Computers under VAXELN

The event–based nature of data in high–energy physics lends itself naturally to parallel processing of individual events which may be done in a "farm" of computers coupled together via some loose network to a host computer. This parallel event–processing scheme can be used as part of data acquisition; for example, one may use such a farm for fast online filtering of events, with the nodes capable of running filter programs which are written in high level languages. Such an arrangement is the basis for the D0 data acquisition system which has been described elsewhere[4] . The combined hardware/software solution chosen for D0 is based on Digital Equipment Corporation's MicroVAX processors and VAXELN software package. In order to gain experience with running large FORTRAN programs in this environment, a first try at operating such a farm for Monte Carlo simulations was made in late 1984. The program used then was ISAJET, a Monte Carlo program which generates high–energy physics collsions. This work provided experience in assembling the control programs and other utilities needed for such an operation.

In the summer of 1986, the D0 experiment had an urgent need for a large number of simulated events to finalize the detailed design of the detector and to investigate triggering schemes based on this design. A Monte Carlo program to generate the events as seen by the D0 detector had been developed based on the GEANT package from CERN [5]. But there was great difficulty finding enough resources to run such a very CPU–intensive program (about 1 hour of CPU–time per event on a VAX–780). None of the institutions within D0 (including three national labs) could dedicate sufficient conventional computing capacity to dedicate to the generation of the required 10,000 events on a short timescale, and because of code incompatibility and other problems neither the Fermilab ACP multiprocessor system nor an outside supercomputer could be used. The collaboration turned to the option of running the program on MicroVAXes under VAXELN. This special version of D0GEANT which was to run under VAXELN turned out to be easily put together; beginning with the VMS version of GEANT we added various control features and with a few weeks of effort had a pro-

duction version running. Most of the time was spent developing general aspects which are useful for setting up any application to run in the farm.

Hardware Layout

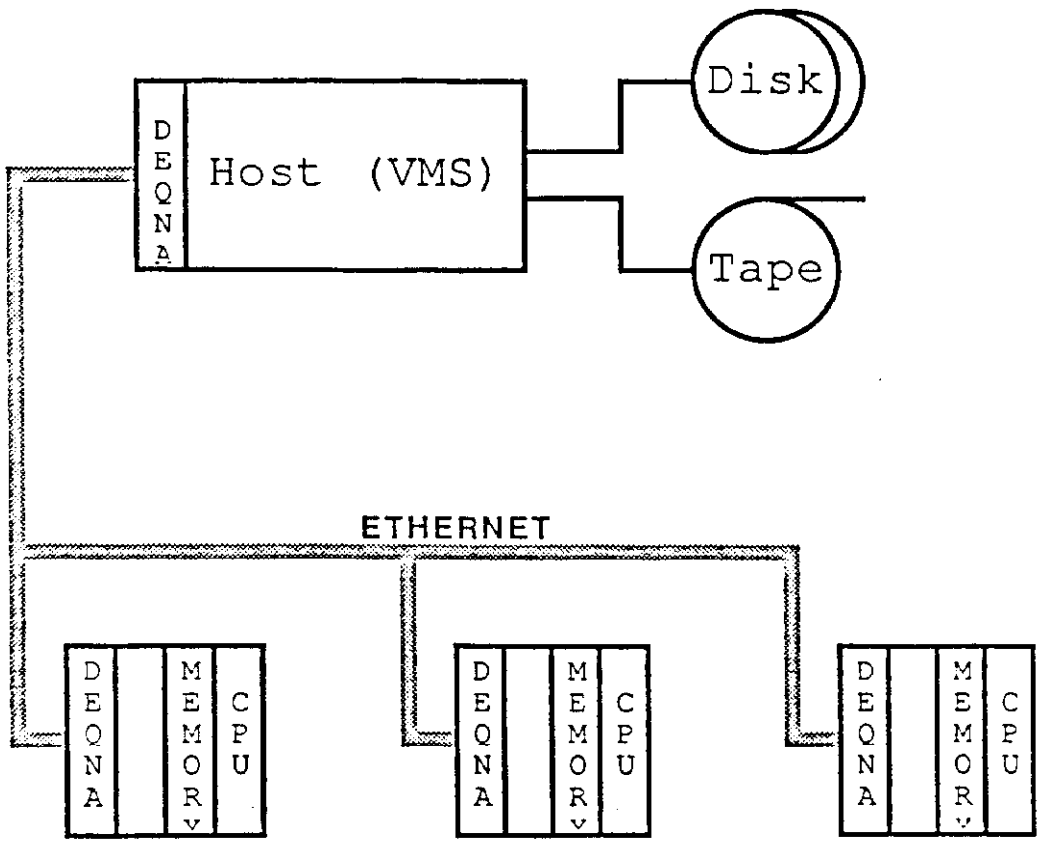
The basic hardware used in this farm consists of MicroVAX–II CPU's (in either the KA–630 as used in VAXstation–II's or the KA–620 real–time version which may only run VAXELN), external memory (most nodes used in this run had 4MB added to the 1MB on the processor board), and DEQNA ethernet interfaces. In addition to the farm nodes, we used as the host a MicroVAX–II system with 2 RA81 420MB disks and a TU81+ 6250 bpi tape drive. Data transfers between the host and the nodes utilized Ethernet, whose support is automatically included with VAXELN systems. Since data was actually transferred only at the beginning and at the end of each of the very CPU–intensive events, the restriction of the Ethernet bandwidth did not pose any limitations. The farm nodes were housed in various backplane configurations: in custom setups used for data acquisition test work, in rack mounted BA–23 chassis and even in a fully configured VAXstation–II. The basic hardware setup for the GEANT run is shown in Figure (1)

For the run described the farm consisted of up to 16 VAXELN nodes. At present, several innovations are affecting the basic hardware described above. A new custom backplane has been developed commercially [6] for the D0 data acquisition farm and will also be used in a slightly modified version for an offline farm. High speed communication channels are also being developed for use in the host–to–node data transfers. This path (40 MB/second per channel) is based on the dual–ported Q–Bus memory boards [6] devloped for the D0 data acquisition system.

VAXELN

VAXELN is a "software product for the development of dedicated and/or real–time systems for VAX processors". It is not an operating system, but allows the user to have full control of all services on a target processor. The development of these application systems takes place under VAX/VMS, using the same compilers, the same linker and all the other facilities for ease of

Hardware Layout of Farm



VAXELN nodes.
'Infinite' number may be added.

ETHERNET/DEQNA communication may be replaced by high speed I/O bus as in DØ data aquisition system.

Figure 1: The basic hardware setup for the GEANT run

program development found in VMS. The programs may be written in any high level language and linked together to form separate programs. VAXELN comes with its own "dialect" of Pascal, called EPASCAL, which is a superset of standard Pascal with extensions for interfacing to the VAXELN services for multitasking, network services, and real–time resource management and device control. VAXELN also provides such an interface for VAX–C, using the standard VAX–C compiler. FORTRAN is supported with a Run–Time Library which makes essentially all VAX–FORTRAN extensions available under VAXELN, although there are a few restrictions as to some types of I/O operations (use of indexed files), and logical file names are in general not supported.

To run a program in a VAXELN target node, the code is first compiled under VMS, and then linked with the VAXELN libraries. Next, a system file is built with the small, menu–driven facility, EBUILD, included in the VAXELN tool–kit. Such a system file may then be used to download to a target node, and will become the only system running there. Particular consideration has to be given to the system size, as VAXELN does not swap pages in and out of memory (no local disk); the whole image therefore has to fit in memory at once.

The programmer has full control over which services like drivers etc. are included in the system. This makes for a very efficient run–time environment with little system overhead. The VAXELN tool–kit also includes a remote debugging facility, EDEBUG, which runs under VAX/VMS but connects to the remote node and allows the programmer to debug his/her code directly as it is running there. EDEBUG is similar to the VAX/VMS debugger and is fully symbolic, with all variables for example available to the programmer for interactive inspection and possible modification.

Changes Needed to Make a FORTRAN Program Run under VAXELN

Very few changes have to made in the FORTRAN code to make a FORTRAN program run under VAXELN. Most of the changes have to do with OPEN statements: making sure OPENs have explicit host node and disk specifications. Also, any refrences to system service (SYS$, LIB$ etc.) rou-

tines should be removed and/or replaced by calls to VAXELN equivalent routines.

Control Program for Farm Operation, FARMRUN

The program FARMRUN was devloped to control the running of this farm. Principally, FARMRUN provides server functions for the input and output event streams. These servers run separately as batch jobs, and command files are used to start them and keep track of which files to use etc. An operator controls the farm by running FARMRUN interactively and may then give START, STOP, PAUSE and CONTINUE commands. The program is also able to interrogate each node and obtain a status message describing the current state of the node. This program uses two other program packages to perform its task. For interfacing with the operator, FARMRUN employs the command/menu package COMPACK developed for general use in the D0–experiment. COMPACK is a general purpose command/menu interface package which has both a "line"–mode and a full–screen "menu"–mode of operation. The user may switch between the two modes at any time. COMPACK also has a command file mode and may be run in batch mode without modifications. It is written in FORTRAN with a subset in FORTRAN–77 for ease in transport to other machines. The full–screen part of COMPACK uses the SMG$ set of routines to perform screen I/O. To send commands to each of the nodes in the farm and retrieve status messages, FARMRUN uses a small package of FORTRAN–callable routines, ELNCON. ELNCON simplifies the system programming needed to perform DECNET I/O under VAX/VMS and provides a Fortran interface to the VAXELN services for network I/O.

Programs in each Farm Node

The programs running in each farm node are shown schematically in Figure (2).

The main program itself, via the call to the initialization routine, starts a few subprocesses used for control and for message passage. These processes go into wait states and hibernate until signaled by software flags.

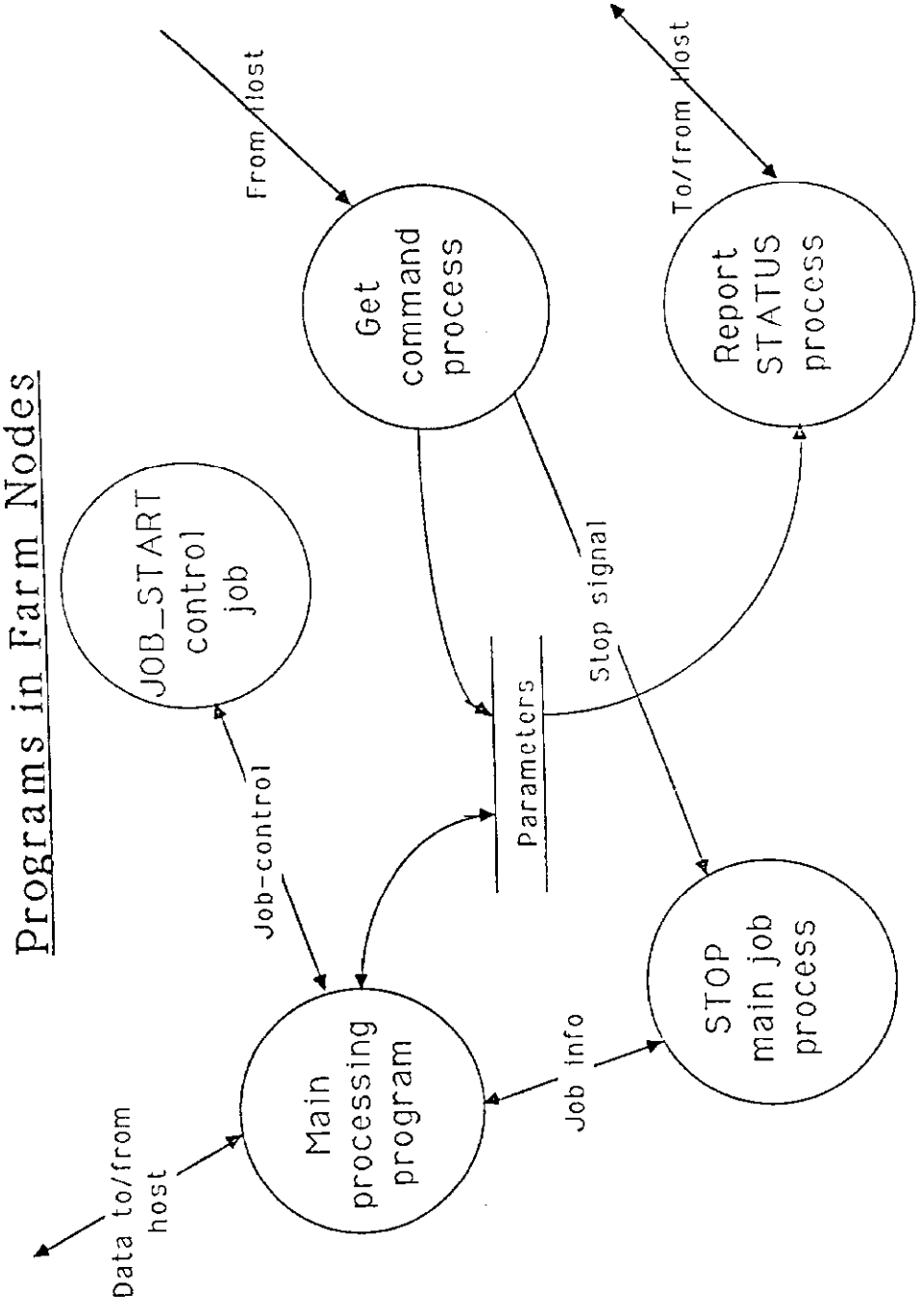


Figure 2: Schematic diagram of the programs running in each farm node

This technique avoids polling–type statements in the main code. The subprocesses take care of control/status messages to and from the host, and may stop the main program if such a command is passed to the node.

The first program which gets control when a node is booted, is JOB__START, a small control program using the VAXELN feature which lets one program control how and when another program is running. JOB__START is set up to start the main program with certain input parameters and then go to sleep until the main program disappears for some reason (ends naturally or is aborted by operator intervention). When the main program goes away, a new copy is immediately started by JOB__START. This action, which does not involve any downloading of the image (the code was already in memory), cuts down on network traffic when many nodes are stopped to be restarted again with new parameters or for other reasons.

Various utilities were also developed to make the computers in the farm work together and make the operator intervention easy. For control and downloading of parameters etc., a call to a special initialization routine is put into the main program. It returns the DECNET–number of the host node and other program–specific parameters used at startup. The name and the number of the node itself which are useful for setting up file specifications for OPEN statements or setting seeds etc., are also available. In D0GEANT, input and output event files were opened via calls to a special control routine which could access status variables in a common block.

Operation of the Farm

Figure(3) shows schematically how the farm operation occurs. While several components of FARMRUN (Input Server, Output Server) may be running in batch mode at the same time, only the Input Server is really needed for the farm to operate. The input and output records in D0GEANT operation are simply files on the host containing one input or one output event. A node opens and reads an event when it is ready for another and then deletes the single input event file. The Input Server wakes up every so often to check if the input event file has disappeared in the meantime. If so, it reads the next event off the file of input events and writes a new single

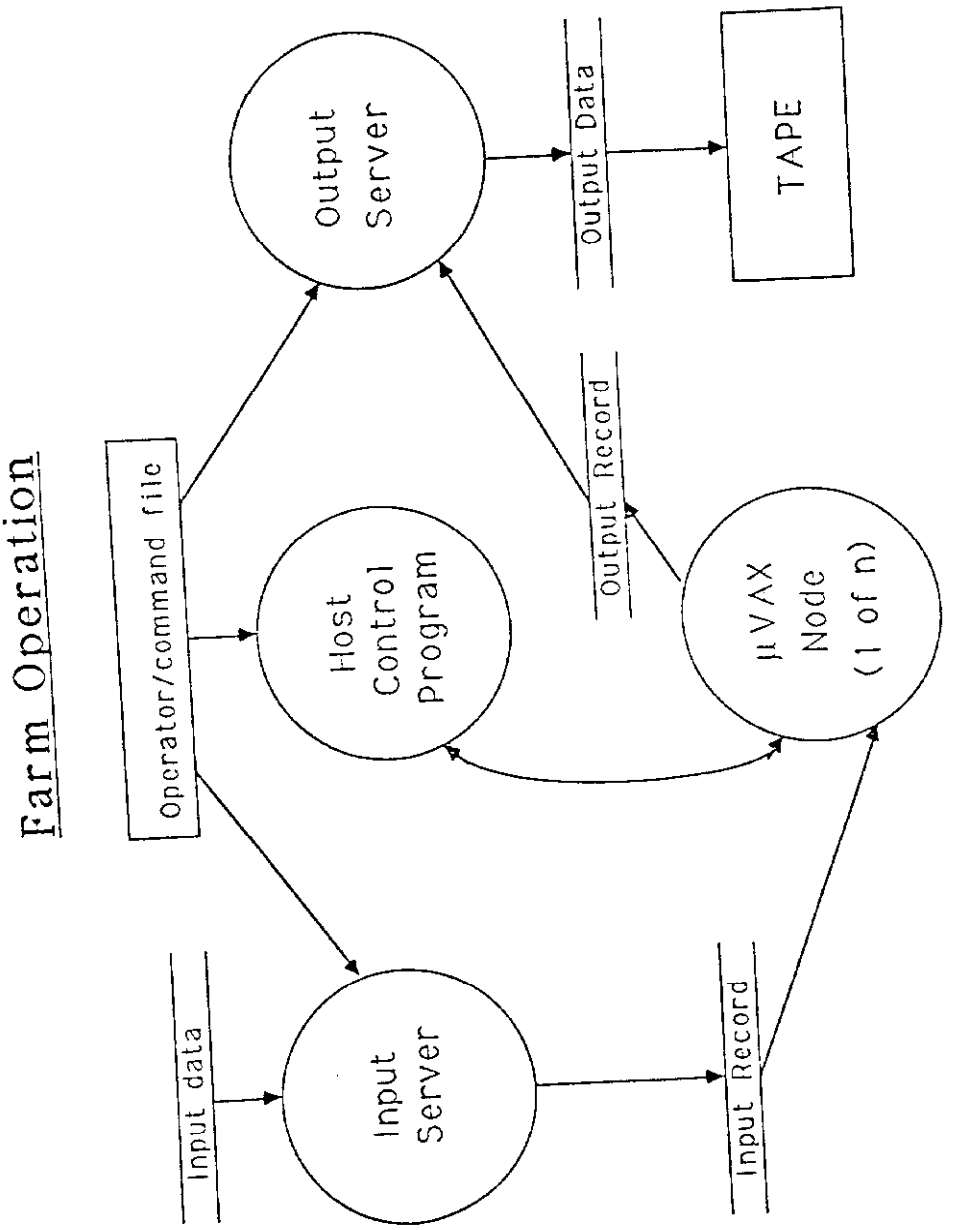


Figure 3: Schematic diagram of how the farm operation occurs

event file. When a node is done processing an event, it opens a new file on the host disk and writes the event to it. The Output Server also wakes up every so often to see if any new events have been written in the meantime. If so, it reads in the event, checking that its format is in order and writes it out to a file of collected output events. When a certain size of the output file is reached, the Output Server closes it, and restarts itself using a new file for the next set of output events. Thus collected output events can be copied to tape at any time thereby freeing up valuable disk space. The FARMRUN component which interactively controls the run and provides status messages is started only when such an operation is needed.

Conclusions

While fulfilling a vital need for the D0 Experiment, we have demonstrated that MicroVAX–II computers running under VAXELN are very well suited for event–based data processing tasks in high–energy physics. The amount of system programming involved to operate such a farm is small and easily managed, even for massive software packages like GEANT. Much of the convenience of our system derives from utilities built for the D0 data acquisition system. Ethernet communications, included in the VAXELN nodes, provides a very natural medium for data transfers. Ethernet bandwidth has proved acceptable for CPU–intensive jobs like GEANT, but larger farms or jobs with less computation per event may rather employ hardware developed for the D0 data acquisition farm. Our experience in using the MicroVAX farm to provide Monte Carlo events for D0 has been highly successful and should be a model for similiar applications elsewhere.

Shower Libraries for D0 GEANT Monte Carlo

Introduction

The full simulation of hadronic showers, while desirable for realistic evaluation of detector performance, is very time-consuming of computer resources. For example, a full simulation of a two-jet event using the standard version of D0 GEANT, where the shower is generated using the GHEISHA shower code, can take several hours of VAX-780 time. High transverse momentum

jet events are particularly slow to generate. It is clearly necessary to be able to generate events faster than this, and various approaches are possible. Typical of the early attempts is the UA1 parametrisation scheme whereby the energy deposition of the particle is parametrized in longitudinal and transverse dimensions[7] in terms of interaction (radiation) lengths for hadronic (electromagnetic) showers. The resulting Monte Carlo is very fast but fails to account for correlations in energy depostion both longitudinally and transversally. This approach fails badly when there are large scale inhomogeneities in the showering medium. It is not clear how to extend the parametrization over calorimeter boundaries going across cryostats and air into a subsequent medium.

D0 has adopted a more conservative parametrisation, namely to parametrise only the electromagnetic particles when they fall below a cut-off energy, set in D0 to 200 MeV. This speeds up shower development considerably (a factor of 3–5 depending on event type) and the boundary problems for low momentum electromagnetic particles can be ignored. The speed up results from the large number of low energy electromagnetic particles present in both electromagnetic and hadronic induced showers [1]. Detector inhomogeneities prevent the raising of the threshold energy in this method to gain greater speed.

Another idea to speed up event generation, used in the simulation of electromagnetic showers in BGO in the L3 experiment[8] and in lead-glass in OPAL[9], is to use a library of 'frozen showers' for low-energy electromagnetic depositions. The showers are generated by EGS Monte Carlo and copied into the event output as required when the shower reaches the threshold energy. These frozen showers also suffer from boundary problems, since one is never quite sure how to re-shape the shower to a region of the detector that is different from the one in which the shower was generated.

In D0, it is possible to generate large numbers (about 10,000) of events through the full showering using a 16 node MicroVAX farm each of whose elements has the computing power of 0.9 VAX-780. This generation takes roughly 1 month of running after the above parametrisation is employed. During the generation, the energy deposition in the calorimeter cells due to individual tracks is saved individually. The summing is done later after full calorimeter response corrections not contained in GEANT are introduced.

The principle of the shower library is to store these showers in a random access library and re-use them for subsequent Monte Carlo runs with great increase in speed. The axial symmetry of the D0 detector aids us in generating fewer tracks than would be otherwise necessary. Boundary problems are totally absent in this method since tracks simulated in one rapidity range are used for the same rapidity range. This results in a minimum of computations after the shower is fetched.

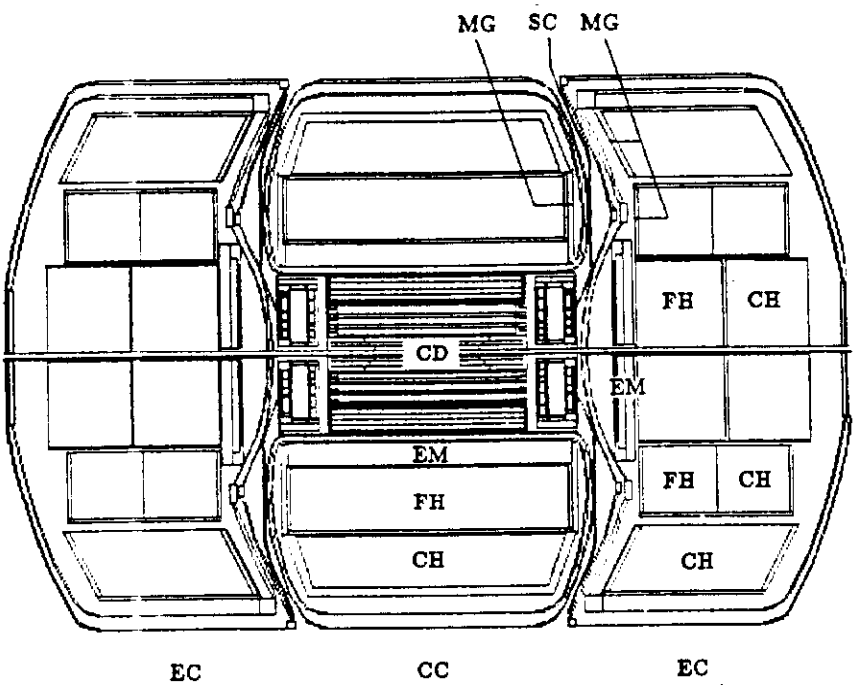
The D0 Detector

The D0 detector for the Fermilab proton-antiproton collider is currently under construction. The principal feature of the detector is the use of a very large uranium-liquid argon calorimeter surrounding the interaction point. The design of the calorimeter is shown in Figure (4). The calorimeter is enclosed in three cryostat vessels — a central toroidal vessel which surrounds the central tracking system, and two endcaps. The calorimeter is divided into an electromagnetic section (four layers, 24 radiation lengths), a hadronic section (three or four layers, out to 4–5 interaction lengths) and a leakage section (one layer, a further 2 interaction lengths). The leakage section is made of copper and steel to save cost. Projective towers are employed. To lessen the deleterious effects on energy resolution of the thick cryostat walls, thin 'massless' liquid argon readout gaps are placed in both central and endcap vessels near the transition region, and a scintillator array is mounted in the space between the cryostats.

Library Features

When using the shower library, it is obviously impractical to have entries ready for retrieval corresponding to absolutely every possible incident momentum, rapidity and particle species. Some rescaling of energies and binning of phase space must therefore be accepted. The chosen solution for D0 has been to rely on the detector features to determine the bin size. Tracks are divided into electromagnetic and hadronic showers. The detector granularity being $\Delta\phi \times \Delta\eta = 0.1 \times 0.1$, tracks are divided into 37 bins of pseudorapidity η (with $\Delta\eta = 0.1$ from 0 to 3.2, increasing therafter). There is no binning in ϕ or $\pm z$, the detector is assumed perfectly sym-

DØ Detector



Longitudinal section through the central part of the DØ detector parallel to the beamline, showing the central tracking system, the central and endcap uranium/liquid argon calorimeters, the 'massless gaps' and the scintillator detectors between the calorimeter cryostats.

Key: CC = central calorimeter cryostat; EC = endcap calorimeter cryostats; EM = electrogmagnetic calorimeters; FH = uranium hadronic calorimeters; CH = leakage hadronic calorimeters; CD = central detectors (vertex chamber, transition radiation detector, and central and forward drift chambers); SC = scintillator planes; and MG = massless argon gaps.

Figure 4:

metric in these variables. Four bins of primary vertex position are used, for the ranges $z < -25$, $-25 < z < 0$, $0 < z < 25$ and $z > 25$ cm. Tracks are binned according to incident momentum lying in the ranges: 100–500 MeV/c, 500 MeV/c – 1 GeV/c, 1–4, 4–10, 10–20, 20–50 and $p > 50$ GeV/c. Tracks with momenta less than 100 MeV/c are ignored entirely.

With this binning scheme, an appropriate library entry to simulate the shower for a given track is chosen by retreiving one with the correct values of the key vector `KEY(1:4)`, where:

`KEY(1)` $\Rightarrow$ Momentum range
`KEY(2)` $\Rightarrow$ Vertex range
`KEY(3)` $\Rightarrow$ Pseudorapidity range
`KEY(4)` $\Rightarrow$ Electromagnetic or hadronic

The library file uses the RZ keyed-access storage system of Zebra which gives fast retrieval of the record with a specified key vector. In the shower library, the retrieved record is a linear structure of zebra banks, one for each stored shower. A random number determines which actual shower is chosen from the data structure to be used in event generation.

A considerable space penalty must be paid for the convenience of RZ, however: a test library of 20k showers occupies 115k blocks of VAX storage. This is perhaps one tenth of the size library that would be desirable for production running.

Library Contents

For each shower, the following information is stored:

- p_x, p_y, p_z, E, m of incident track.
- CC massless gap energy (total).
- EC massless gap energy (total).
- Scintillator hits (total).
- Dead material energy.

- NHITS, the number of calorimeter tower hits stored. This can be up to 20, but where fewer than 20 hits encompass over 95% of the calorimeter energy for the shower then that number is used. Hits are stored in descending order of energy starting with the highest.
- Calorimeter energy EMISS that was missed by using only the NHITS highest energy hits instead of all hits.
- An array dimensioned (1:NHITS) of hit tower positions and a corresponding array of hit energies.

The showers are linked together into 2072 linear data structures, one for each key vector value.

The library is written using a program that reads the showers from full simulation GEANT events which have been written earlier. In these tests the library has been compiled from showers taken from p_T 120–160 GeV/c events processed on the Brown University MicroVAX farm. The library compilation obviously need only be done once.

Event Generation

When generating events, the particles are tracked normally by GEANT up to the start of the calorimetry, at which point the track is killed, and an entry is chosen at random from the library according to rapidity, energy and primary vertex position. The library shower is rotated in ϕ and/or flipped in z to match the track entering the calorimeter. The energy of each calorimeter hit is scaled so that the total energy is increased by the amount EMISS that was missed by restricting the number of hits. An additional scaling of each hit is then performed by the ratio of the momentum of the incident track to that of the library entry track, so that the shower matches the new incident energy. In future, we envisage an algorithm where the missing energy EMISS of the track is sprinkled around the track direction in some appropriate manner. The energies are then smeared using the standard smearing factors (FUNCTION SMRFAC) and copied into the calorimeter working common block /CTWRK/ using the normal storage routine DHSTOR. The massless gap energies, dead energy and scintillator hits are scaled by the ratio of momenta (without the factor for EMISS).

Generation method	*Standard Gheisha*	*Gheisha with param.*	*Shower Library*
Full D0 detector p_T 40 − 80 *GeV*/c	20560	6800	128
Full D0 detector p_T 120 − 160 *GeV*/c	25600	8070	136
D0 calorimeter only p_T 40 − 80 *GeV*/c:	20000	5440	131
D0 calorimeter only p_T 120 − 160 *GeV*/c:	25800	7540	

Table 1: Comparison of mean program time taken for event generation for various D0 GEANT options. All times are in VAX-780 equivalent seconds (approximately MIP-seconds).

Because the 'generation' of each shower from the library is so much faster than full simulation, the routine `CALBLD` which generates the track Zebra banks from the working common was found to be unacceptably slow. Shower library generation therfore calls this routine once per event rather than once per track as is usual, which means that calorimeter information is not kept for individual tracks but only for the whole event.

Speed

The gain in program speed obtained by the use of shower library generation can be very large. Table (1) lists the mean time taken (in VAX-780 eqivalents) to generate two-jet D0 events with transverse momentum in the ranges 40–80 and 120–160 GeV/c. Because of the variability of the topology and multiplicity of events, the generation time is subject to large fluctuations from event to event. The RMS of the time is typically 40% of the mean. Also listed in Table (1)are the corresponding times to generate events in the D0 calorimeter only, without the central detectors present (`DCEN=0`); these are essentially identical to those for the full detector, show-

ing that tracking particles through the central detectors before the shower library is called is not a substantial drain of time. The speed gains obtained by generating events by shower library compared to the GHEISHA options are large: factors of 161 and 53 over full and parametrisation versions of GHEISHA for p_T 40–80 GeV/c and 188 and 59 respectively for p_T 120–160 GeV/c.

Figure (5) shows how the time per event is broken down by routine. The majority of the time is occupied with the RZ I/O (`RZIODO`, `RZINK`) and none of the event processing routines is a time drain. Altogether the routines handling RZ I/O account for 37% of the execution time.

How Well are Events Generated?

Shower library events have been generated using the 20k-shower test library for comparison with standard GHEISHA events run on the Brown University Microvax farm. The mean and standard deviation of various kinematical quantities are compared in Table (2) for shower library events and GHEISHA events (two-jet events with transverse momentum 40-80 and 120-160 GeV/c).

The distributions of kinematical quantities for two-jet events in the same two momentum ranges are compared in Figures (6) and (7). It will be seen that the agreement is reasonably good even in the tails of the distributions such as (Missing E_T)^2, which is very encouraging. The p_T 40-80 GeV/c events have a slight excess of GHEISHA events over the shower library in the tails of the distributions of x and (Missing E_T)^2. This is not very significant because the GHEISHA sample contained only 1007 events in this momentum range and there are only a few excess events in this sample. The curve has been rescaled to compare with the 2140 events of the shower library sample which appears to enhance the statistical significance of this difference. This accounts for the differences between the mean and RMS of these distributions as compared in Table (2).

The energy quantities are compared in Figures (8) and (9). Again, the agreement between shower library and GHEISHA events is good. Only the scintillator hits distribution is not terribly well simulated, which should come as no real surprise given the crudity of scaling the number of hits by

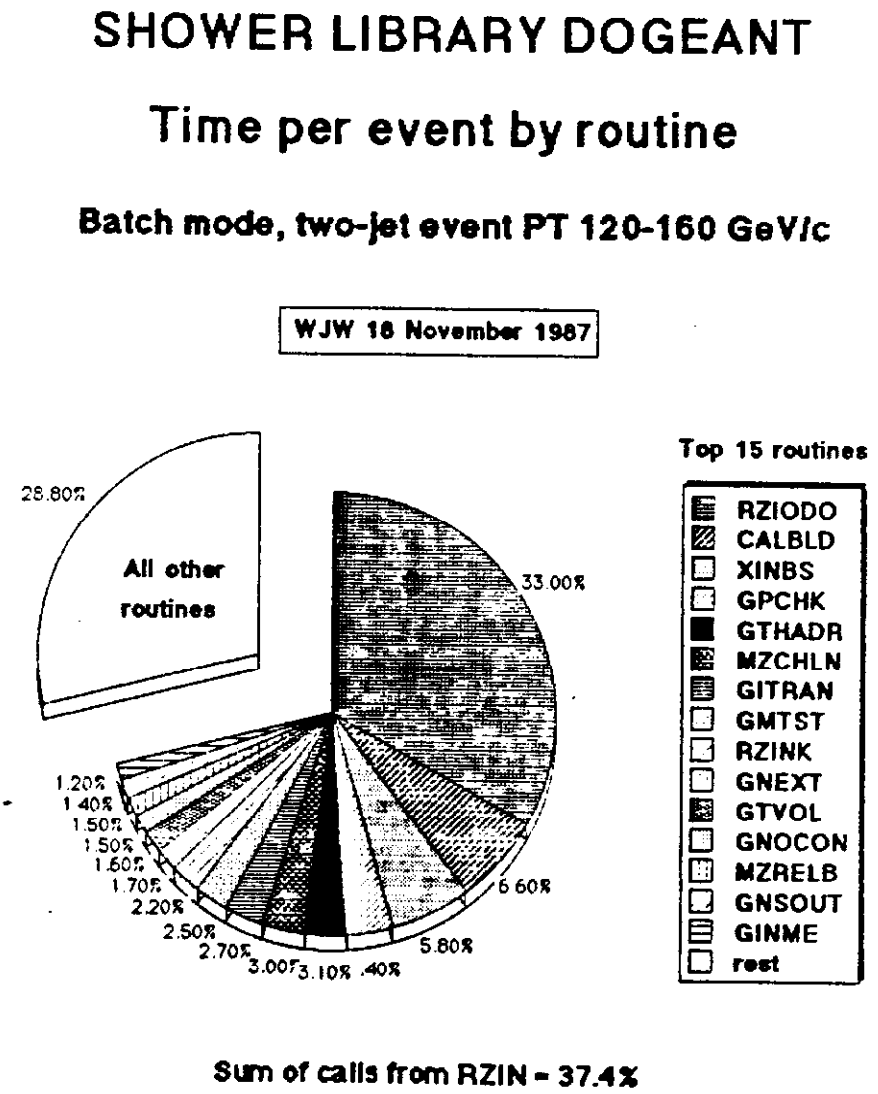


Figure 5: Output from VAX-PCA analysis of time spent per routine in shower library event generation. One two-jet event with transverse momentum 120–160 GeV/c was analysed.

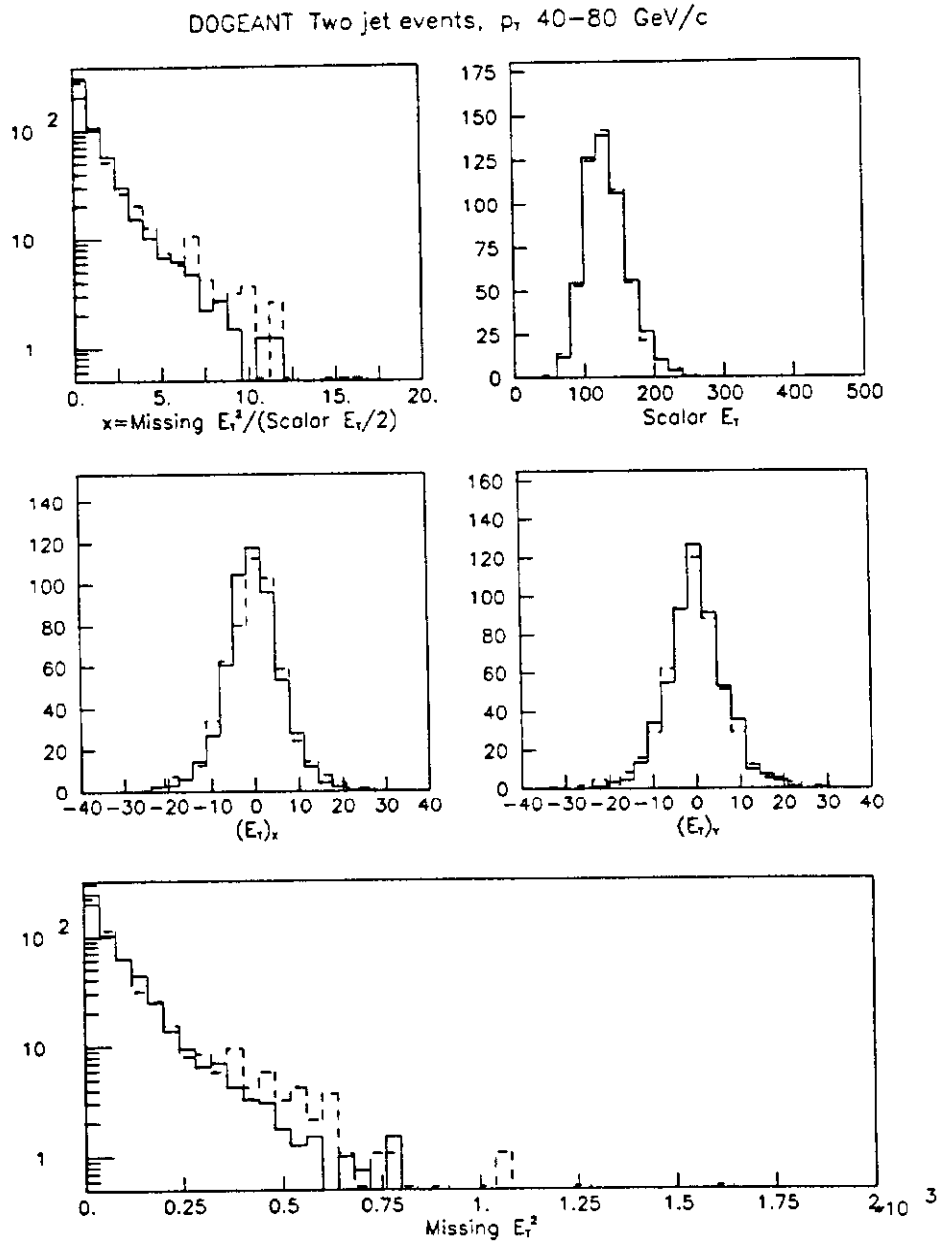


Figure 6: Distributions of various kinematic quantities for D0 events generated by shower library (solid line, 2140 events) and by GHEISHA (dashed line, 1007 events, renormalised by number of events). Two-jet events with transverse momentum 40–80 GeV/c. All energies are in GeV.

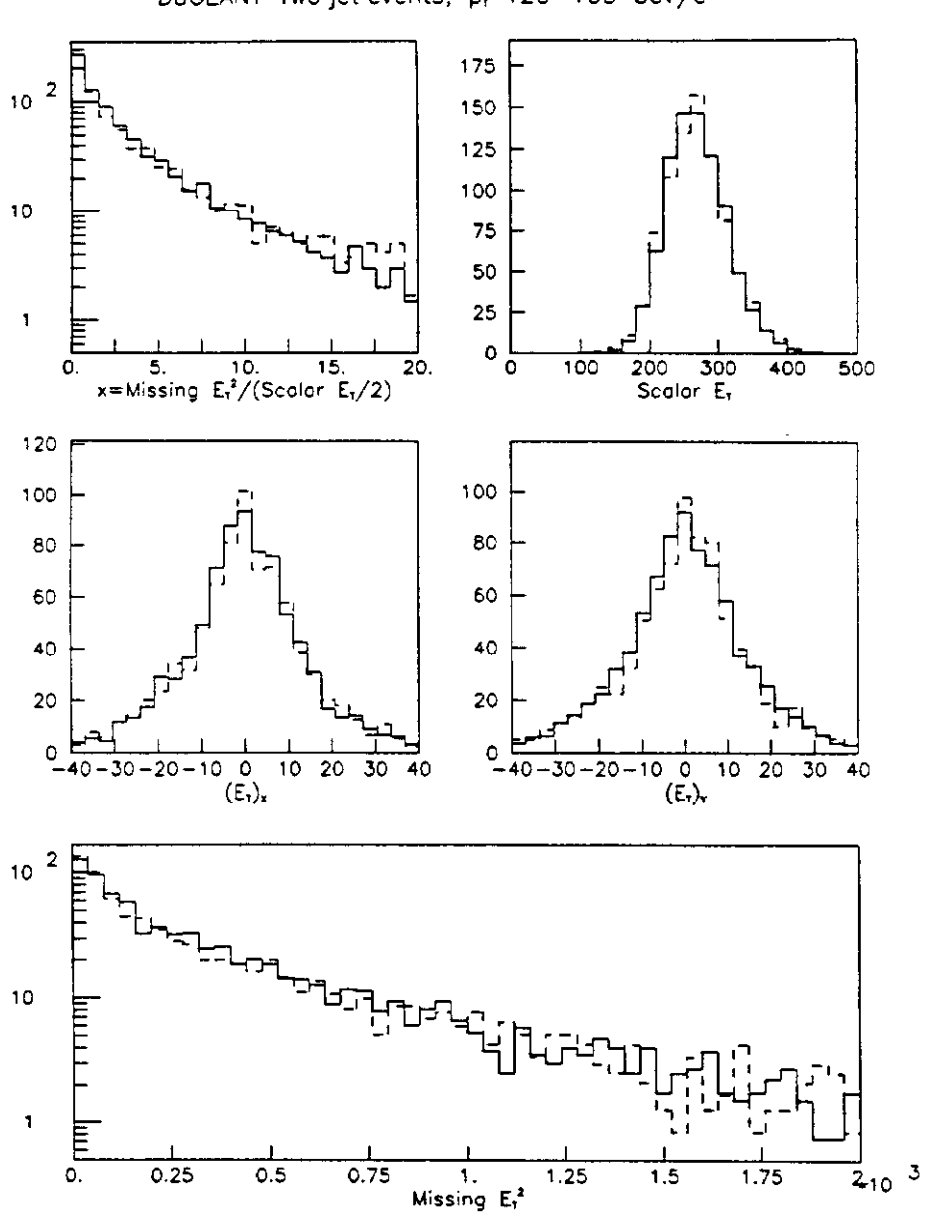


Figure 7: Distributions of various kinematic quantities for D0 events generated by shower library (solid line, 3303 events) and by GHEISHA (dashed line, 1958 events, renormalised by number of events). Two-jet events with transverse momentum 120–160 GeV/c. All energies are in GeV.

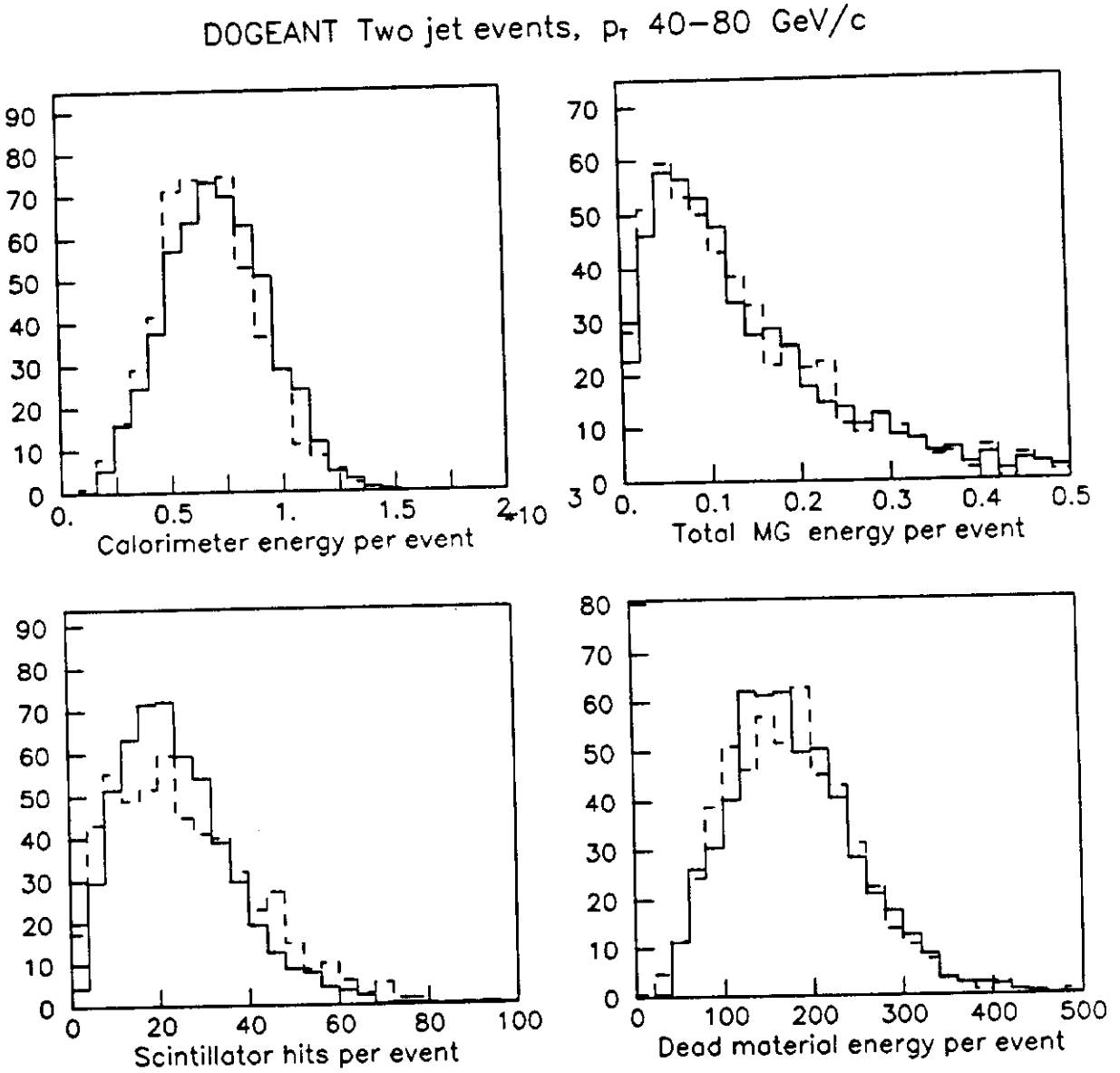


Figure 8: Distributions of energy in calorimeter, dead material, massless gaps and number of scintillator hits for D0 events generated by shower library (solid line, 2140 events) and by GHEISHA (dashed line, 1007 events, renormalised by number of events). Two-jet events with transverse momentum 40–80 GeV/c. All energies are in GeV.

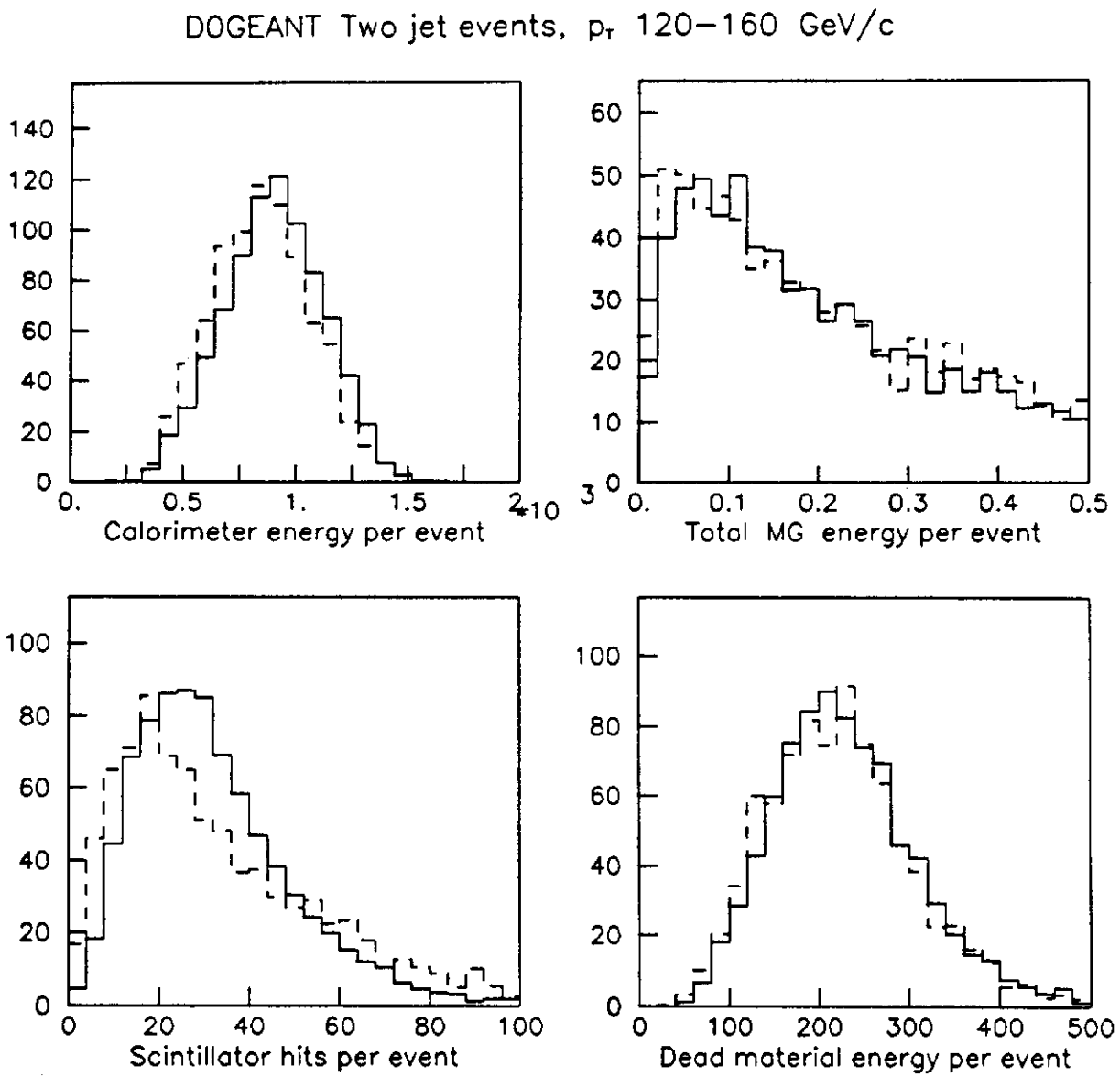


Figure 9: Distributions of energy in calorimeter, dead material, massless gaps and number of scintillator hits for D0 events generated by shower library (solid line, 3303 events) and by GHEISHA (dashed line, 1958 events, renormalised by number of events). Two-jet events with transverse momentum 120–160 GeV/c. All energies are in GeV.

	Shower Library	*GHEISHA*	*Ratio*
•p_T 40 − 80 GeV/c	2140 *events*	1007 *events*	
$(E_T)_x$ *RMS* :	6.54	7.17	0.91
$(E_T)_y$ *RMS* :	6.70	7.47	0.90
Scalar E_T *mean* :	133	133	1.00
RMS :	30.5	30.8	0.99
(*Missing* E_T)^2 *mean* :	88	107	0.82
RMS :	118	157	0.75
$x = (ME_T)^2/(SE_T/2)$ *mean* :	1.37	1.60	0.85
RMS :	1.87	2.20	0.85
Mean calorimeter energy :	714	677	1.05
Mean massless gap energy :	0.139	0.137	1.02
Mean number scintillator hits :	24.5	26.2	0.94
Mean dead material energy :	180	177	1.02
•p_T 120 − 160 GeV/c	3303 *events*	1958 *events*	
$(E_T)_x$ *RMS* :	13.7	13.9	0.98
$(E_T)_y$ *RMS* :	13.8	13.9	0.99
Scalar E_T *mean* :	269	268	1.00
RMS :	44.3	45.6	0.97
(*Missing* E_T)^2 *mean* :	376	367	1.00
RMS :	415	426	0.98
$x = (ME_T)^2/(SE_T/2)$ *mean* :	3.03	3.22	0.94
RMS :	3.75	4.11	0.91
Mean calorimeter energy :	904	856	1.06
Mean massless gap energy :	0.193	0.194	0.99
Mean number scintillator hits :	32.0	32.7	0.98
Mean dead material energy :	229	223	1.02

Table 2: Comparison of event properties from shower library and GHEISHA generation. Two-jet events with transverse momentum 40–80 and 120–160 GeV/c. All energies are in GeV.

the ratio of track momenta. Scaling by the ratio of logarithms was also tried, but fared worse.

Correlations

Figures (10) and (11) show scatter plots illustrating the correlations between the energy deposition in the massless gaps, in dead material, and the number of scintillator hits. The correlations with dead energy are weaker than that between the massless gap energy and the scintillator hits because no selection has been performed to require jets pointing toward the massless-gap/scintilator region of the detector. The shower library reproduces well the degree of correlation seen in the GHEISHA events.

Advantages and shortcomings of this technique

The shower library offers a way of generating events very much faster than full simulation. It handles fluctuations correctly (assuming that they were done properly in the events used to create the library) and avoids the problems with inhomogeneities that plague parametrisations. For a library that has roughly 1000 tracks per bin, one can expect to simulate jet-jet fluctuations that happen at the level of typically one part per million, if the fluctuations are due to two tracks each fluctuating at the level of one part per thousand. Thus, this method enables us to go beyond the intial numbers used for generating the shower library.

On the negative side, the large library file must be written and stored. We see perhaps a full RA81 disk being dedicated to the shower library in D0. For an experiment such as D0, or for SSC detector designs where the geometry is not yet fixed, this means that a run of full simulation events to compile a library is needed each time a substantial change is made to the geometry of the detector. In addition there are shortcomings introduced by the granularity of the library binning. The use of bins of 0.1 in η means that electromagnetic showers taken from the library may peak in the wrong cell of the EM3 layer, where the cell size is 0.05. This can however be corrected by re-distributing the EM3 layer hits according the true position of the parent track. Tracks entering cells near cracks may also have problems, as

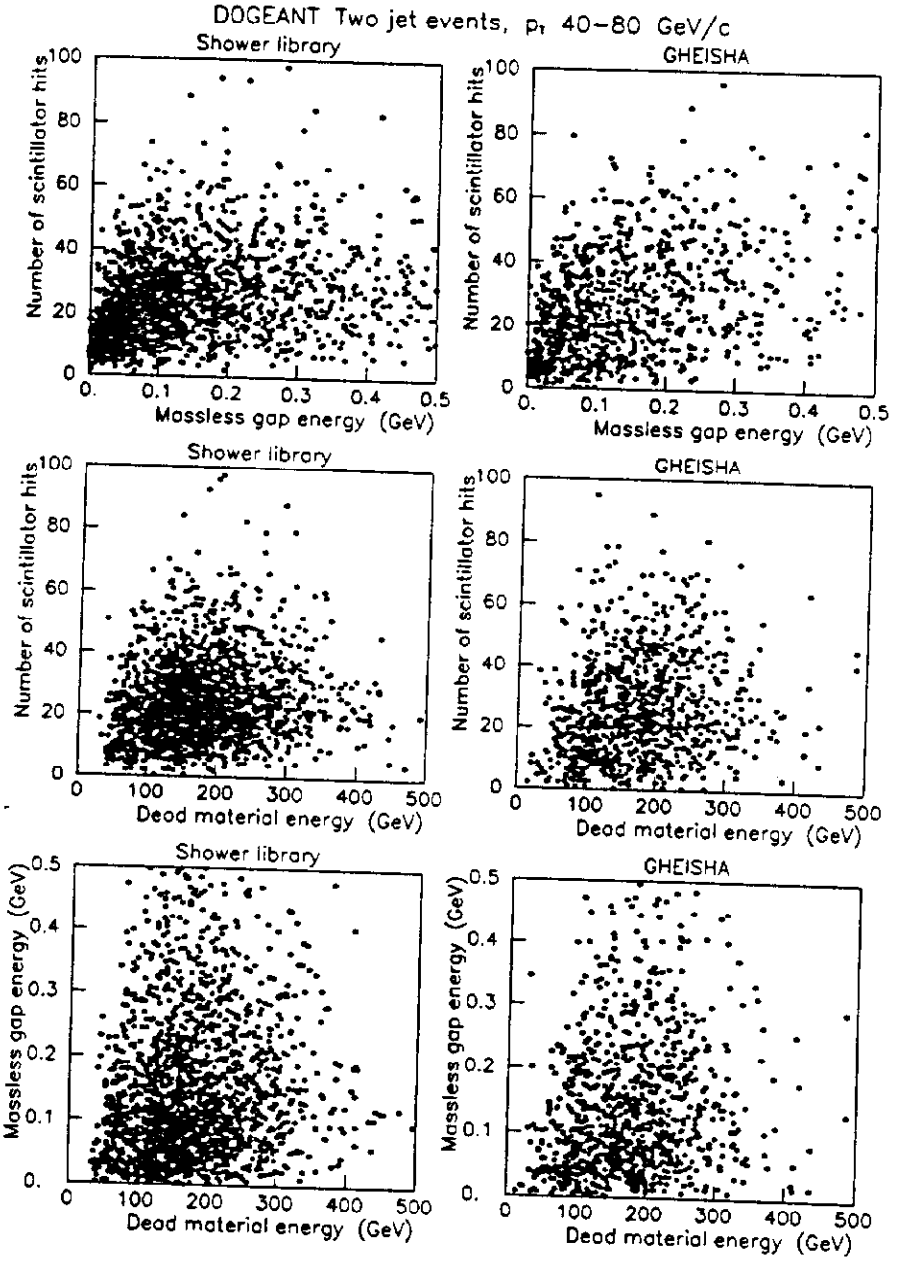


Figure 10: Correlations between energy in dead material, massless gaps and number of scintillator hits for D0 events generated by shower library (2140 events) and by GHEISHA (1007 events). Two-jet events with transverse momentum 40-80 GeV/c.

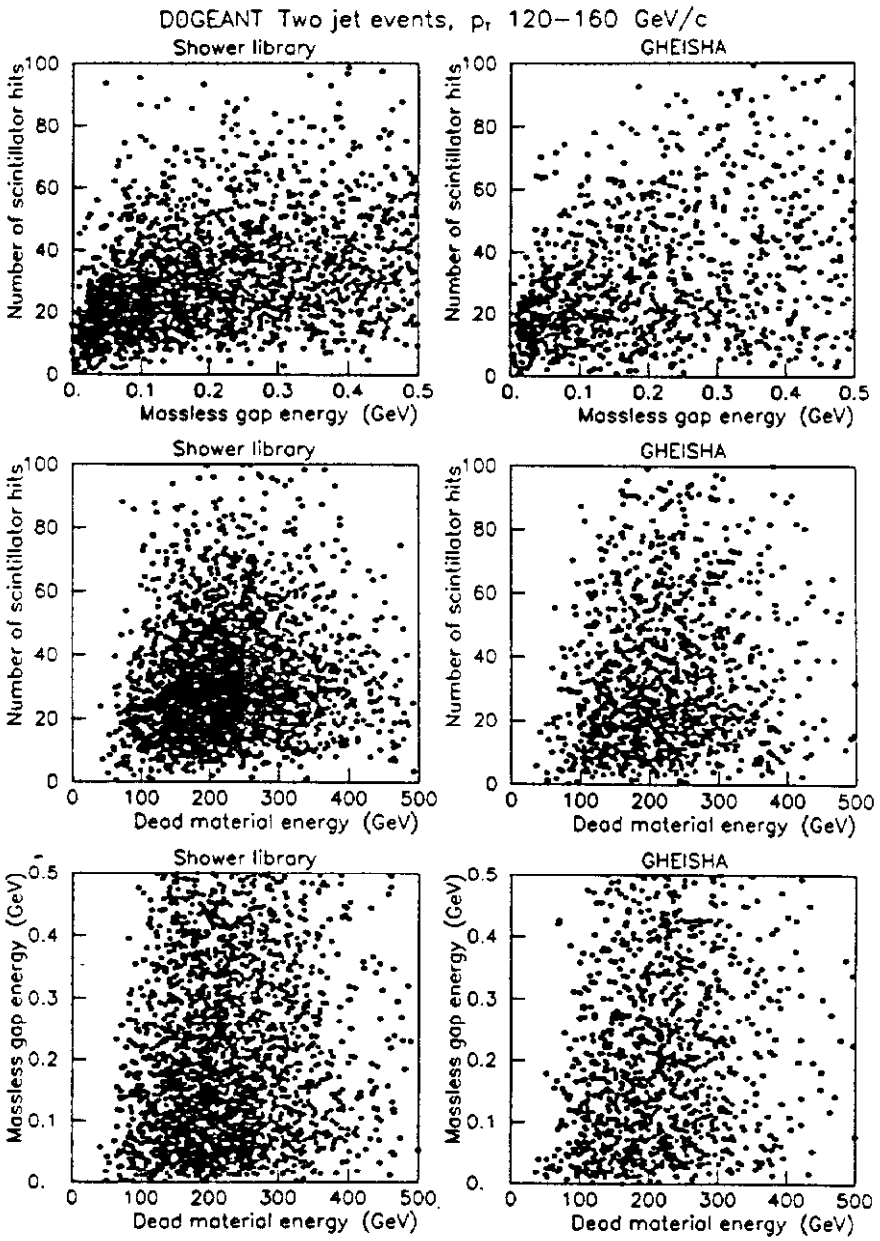


Figure 11: Correlations between energy in dead material, massless gaps and number of scintillator hits for D0 events generated by shower library (3303 events) and by GHEISHA (1958 events). Two-jet events with transverse momentum 120-160 GeV/c.

the library entry is chosen without regard to whether its track come from near the center of a tower or from the crack region.

Conclusions

In general however, despite the problems listed above, the shower library simulates D0 events well, with a vast saving in generation time. The technique has obvious application to the generation of events for future detectors and is in no way restricted to D0.

Acknowledgements

We wish to acknowledge Ray Zeller and Chris Johnson of ZRL for their vital contributions to the D0 data acquisition hardware and software and to the operation of the system described here. We also wish to thank Digital Equipment Corporation for their interest and support of our application of MicroVAX-based VAXELN farms. The work on the MicroVax farms was done mainly by D. Cutts and J. Hoftun. The shower library idea is due to the speaker and its implementation due to J. Womersley. This work was suported in part by the U.S. Department of Energy under contract DE–AC02–76ER03130.A022–TC.

References

[1] S.L.Linn, Description of shower parametrization in D0GEANT program

[2] D.Cutts, J.S.Hoftun, Running a large Monte Carlo program in a farm of MicroVAX-II computers under VAXELN, D0 Note 652.

[3] W.J.Womersley, R.Raja and A.M. Jonckheere, Shower Libraries for D0 GEANT Monte Carlo. D0 Note 650.

[4] The MicroVAX Based Data Acquisition System for D0 ,D. Cutts et al. Proceedings of the Fifth Conference on Real–Time Applications in Nuclear, Particle and Plasma Physics, IEEE Transactions on Nuclear Science, Vol. NS–34 (August 1987).

[5] Simulations of D0 and Hermiticity Studies, A.M. Jonckheere, Talk at this workshop.

[6] Zeller Research Ltd., 8 Rushton Drive, Cranston, RI 02905

[7] R.K. Bock, T. Hansl-Kozanecka and T.P. Shah, Nucl. Inst. & Meth **186** (1981) 533

[8] E.S. Chen, L3 internal memorandum

[9] D.R. Ward, OPAL internal memorandum, "Bootstrap" method for simulation of the OPAL electromagnetic endcap in GOPAL, February 1987