Computing Platforms with ultimate file system approaches, Heterogeneous Mass Storage Devices and an Open Distributed Hierarchical Mass Storage Management System. This contribution will provide an overview on how far we got and what the next steps will be.

BR9737140

ABS_96

Data Handling and Post-Reconstruction Analysis at Next Generation HEP Experiments

M. FISCHLER (FNAL)
M. Isely, M. Galli, S. Lammel, D. Sachs, J. Mack (FNAL)

A new generation of experiments in high energy physics is approaching. With the approval of the Large Hadron Collider at CERN and the revised Main Injector plans at Fermilab, high statistics experiments will start operation within five and 10 years. With luminosities of 10^{33} and $10^{34} 1/cm^2s$ and several hundred thousand readout channels data from these experiments cannot be handled and analysed in the traditional old-fashioned HEP way. This paper discusses approaches to the data intensive phase of post-reconstruction analysis, characterised by a wide variety of physicists intitiating multiple scans through large bodies of data.

Approaches to analysis can be characterised by the type of investigations they support efficiently, the nature of selection criteria the user can supply, and the data and results delivered. Currently, experiments handle raw data in banks that are handled by FORTRAN memory management packages. A reconstruction pass creates objects of interest, again stored sequentially in files. In a second "pre-analysis" production pass events are sorted by characteristics like the existance of lepton candidates. Early data-intensive analysis handles the event information much like these production passes, fragmenting data and controlling organization through memory management packages. In the next analysis step, specific information per event is collected and saved in n-tuples. This information is then rapidly available for histogramming, visual- ization, and statistics. While bank headers provide limited information about the object stored, the organization of n-tuples is defined by the code that produced them. Access to data other than that extracted for n-tuples can be very time consuming. One approach to future analysis is to retain the current data handling methodolgy, and rely on improved computing hardware to cope with the factor of about 20 increase in data volume.

PAW and PIAF tools are already in use for data analysis. All hot information is collected and stored in n-tuples, distributed over several CPUs. Users then extract their desired information - histograms or further n-tuples - from this master n-tuple. Future analyses could be based on this second approach, which becomes especially attractive once n-tuples have been formed.

The Fermilab CAP project is focusing on a third approach, suitable where limitations imposed by n-tuple methods impede analysis. Data are loaded into the CAP system and hot information for event selection is stored in an object oriented format. Queries can provide full or summary event information; the volume of data extracted is assumed to be rather small and thus manageable on a standard workstation. An Event Query Language extends the familiar PAW style, automating conditions involving multiple particles or objects within the event and links between objects. The intricate hierarchical structure present in experiment data is expressed in a C++ -based paradigm of "physics objects". Efficient processing, avoiding repeated scans of the full data set, is achieved by grouping the physics objects of each kind into stores which can be scanned individually as needed. To implement relations between objects, and smoothly handle the data organization, lightweight persistent object management software is used. Starting from the Ptool persistent object API, we have created a Physics Object Persistency Manager (POPM) designed to support optimum query rates on a large parallel system. New features include data parallelism, low-overhead access to persistent data by locked "physical pointers", and read- ahead logic to achieve very high performance on individual queries. The DST data for D0 Run1B has been brought into the CAP system and can be "mined" via EQL queries.

A fourth approach manages the data in an even earlier phase of the analysis cycle. All information about an event, whether from the data aquisition system, the reconstruction passes, or even generated during the analysis phase, is collected in a database system. This data is available for distribution to processors requiring the information. Frequently accessed information stays at the workstation or analysis cluster; additional data is also present so that when small changes in selection criteria are applied, the missing information is made available to the user quickly and transparently. This approach is being investigated on the same testbed hardware as the CAP project.

The common feature of any approach to this phase of analysis is the need for high-performance access to large datasets. We discuss pros and contras of these four different analysis approaches currently being considered for the next Collider and Fixed Target runs at Fermilab. Ideally, various methods could be integrated on one or more systems, preferably sharing one copy of the event data on disk. Since efficient implementations of each approach have distinct data organization needs, a compromise must be reached trading some duplication of data for some efficiency improvements. We illustrate instances where efficient data sharing can be implemented, and present problems which may prevent this in other cases.

ABS_88

BR9737141

The Nile System Data Model

M. Ogg (Texas)

The Nile Collboration (affiliated with CLEO), K. Birman, D. Cassel, R. Helmke, D. Kreinick, D. Riley, M. Rondinaro (Cornell) A. Calkins, K. Marzullo (U. California) M. Athanas, P. Avery, W. Hong, T. Johnson (U. Florida) C. Chan, A. Ricciardi, E. Rothfus (Texas)

NILE is a multi-disciplinary project building a distributed computing environment for HEP. Nile will provide fault-tolerant, integrated access to processing and data resources for collaborators of the CLEO experiment, though the goals and principles are applicable to many domains. Nile currently has three main objectives: a realistic distributed system architecture design, the design of a robust data model, and a Fast-Track implementation providing a prototype design environment which will also be used by CLEO physicists. We will discuss the Data Model, its design issues, and its interactions with the Nile System Architecture.

The purpose of the Data Model is to allow the application layers to access any arbitrary data attribute with low latency. In the case of CLEO II, the accumulated data is about 10 TB (growing by a factor of 10 over the next 4 years). The quantity of data and processing resources require the latency to be no more than a few milliseconds At the same time, the Data Model must hide the details of the data storage from the application, and not prejudice analysis schemes, present or future.

The principal challenge with a distributed architecture is to balance the intelligence in query servicing between the extremes of client or application-centered (as is mostly the case today in HEP), and server-centered (as with a traditional database). The Nile Data Model is layered: the lower layers handle the physical storage, while the upper layers interact with the application. In this way, we gain flexibility and do not have to make a fixed choice between the two extremes.

At the lowest level, data are partitioned into "chunks" which are selections of events, and projections of data attributes. Associated with each chunk is a descriptive chunk table. The ensemble of chunk tables provides a complete description of the current data location. A chunk may be controlled by a hierachical storage manager; if part of a chunk that is not on disk is needed, the entire chunk will be cached to disk.

Higher up, the data server processes requests (from jobs or job managers) and can do restricted database operations, sending the final data stream to the client. Ideally, only attributes that are actually needed are sent to a client, and the data can be pre-fetched to overcome latency.

We are now designing the storage manager, and considering the appropriate caching strategies. Issues that are being investigated include the optimum data layout, the assignment of events and attributes to chunks, chunk size, query language, and user interface. It is an open question whether the best performance will come from a relational or object-oriented approach, or a combination of the two.

ABS_83

BR9737142

ZARAH - the Central Computing Facility for the ZEUS Experiment

O. Manczak (DESY)

O. Derugin, D. Gilkinson (DESY)

In this paper we would like to present most important technical issues concerning design and implementation of ZARAH – the central computing facility for physics analysis for the ZEUS experiment. We would like to present to the HEP community our achievements, open problems, and ideas for the future concerning transparent access to the hierarchical mass storage, indexed data access method, data compression, ORACLE based data bookkeeping and data processing in a high performance distributed "batch" environment.

Our approach has proved itself to be very successful for the ZEUS experiment. However, we belive that our ideas and concepts are truely "reusable" (i.e. not ZEUS-specific) and our experience can be profitable for other experiments and the whole HEP community.

ABS_80