Analyzing High Energy Physics Data Using Databases: A Case Study

R. Grossman, X. Qin, D. Valsamis, W. Xu
Laboratory for Advanced Computing
University of Illinois at Chicago

C. T. Day, S. Loken, J. F. MacFarlane, D. Quarrie
Lawrence Berkeley Laboratory

E. May, D. Lifka, D. Malon, L. E. Price
Argonne National Laboratory

A. Baden
Department of Physics
University of Maryland

L. Cormell, P. Leibold, D. Liu, U. Nixdorf, B. Scipioni, T. Song
Superconducting Supercollider Laboratory

Abstract

We describe the initial work of the PASS Project which uses techniques from distributed object managment to analyze experimental data from high energy physics. At this time, we have designed two prototypes to analyze high energy physics data from the CDF experiment at Fermi Lab. The data from this experiment consists of "events" which describe particle collisions. Each event consists of several hundred numerical attributes and occupies approximately 10K in a compressed format. We describe our experience analyzing this data using a relational database, an object oriented database, and a persistent object manager.

1 Introduction

We summarize the results of the first phase of the PASS Project [1]. This is an extended abstract of the report [5]. The goal of the project is to use techniques from database management and distributed object managment to analyze experimental data from high energy physics. The basic challenge is to design software which scales to handle data in the terabyte range and beyond and which supports complex, numerically intensive queries.

At this time, we have designed two prototypes (the Mark 0 and the Mark 1 System) which use high energy physics data from the CDF experiment at Fermi Lab. The data consists of "events" which describe particle collisions. Each event consists of several hundred numerical attributes and occupies approximately 10K in a compressed format. Data analysis require the examination and filtering of a data sets containing anyone from 10 thousand to 10 million events. At the present time, there is approximately 1.5 Terabytes of data. The majority of the tape is on 8mm tape.

Of course it is possible to spin a terabyte of disk. Our interest, though, is in systems in which data and budget requirements require that the majority of data reside in tertiary storage, such as tape or some type of CD-ROM. In particular, we are interested in how to interface distributed object managers and databases to hierarchical storage systems and best to exploit such systems in high performance computing environments. Our work is simplified because by the nature of our application, the data is frequently read, but infrequently updated.

During the first phase of the project, we designed and developed a system called the Mark 0 System, which is described in more detail below. With this system, we created a test bed containing high en-

0-8186-6610-2/94 $04.00 © 1994 IEEE

ergy physics data and queried it using a commercial relational database, a commercial object oriented database, and an object manager called ptool [3] that was developed at UIC. There were significant performance differences compiling, populating, and querying the stores between these different technologies. Three queries chosen to be typical of the anticipated query patterns were selected and developed into benchmarks. Table 1 contains a summary of the query times. During this phase of the project, we

1. developed a series of benchmarks which were convincing to physicists and could quantify the performance advantages of using database and object management technologies;
2. demonstrated that our proof-of-concept implementation allowed analyses to be completed about an order of magnitude faster than existing production code;
3. discovered that existing commercial and experimental systems had problems scaling up to the sizes required in our project.

During the second phase of the project, we designed and developed a system called the Mark 1 System with the goal of understanding how to interface a database or distributed object management system to a hierarchical storage system. With this system, we scaled up the data sets by about an order of magnitude, and more significantly, coupled databases and object managers to hierarchical storage systems in a variety of manners. We also developed several additional benchmarks and gained some experience analyzing data in parallel computing environments. On the basis of this experience, we completed our work on an open reference model [6] which we feel is suitable for the initial design of a system designed to manage Terabytes of scientific data. This work is described in [4].

2 Background

Current and proposed high energy physics experiments share the following features:

- Data is collected about particle collisions or *events*. The data is historical in the sense that it is written once, but read often. There is so much data that approximately 90% of it is on tertiary storage. Currently tape is used.
- Analysis of the data is done in several passes. The first pass subjects all of the data to some type of preliminary analysis and selects certain events for further study. These objects are written to tape or disk. Further passes subject these selected objects to more expensive tests, often statistical in nature. Each pass produces additional derived data.
- Queries consist of complicated Fortran or C programs. By tuning the parameters in these programs, events are selected for further study.

For example, the CDF colliding beam experiment at the Fermi National Accelerator Laboratory measures the radiation products from the collision of particle beams at rates of up to 285 kHz. Collisions, or events, are recorded in detectors which provide approximately 150 KBytes of digital information. A small fraction (approximately 1 Hz) of the total number of events are recorded on magnetic tape for data analysis. Currently 1.5 TBytes of data are stored on tape. The first pass typically produces a tape (the data summary tape) containing approximately 2 GBytes of data. This is subjected to further analysis which selects $\sim 10,000$ events. Attributes in these events are histogrammed. A peak in the histogram indicates a particle.

3 Mark 0 PASS System

We developed a proof-of-concept system and analyzed approximately 100 MBytes of data from the CDF experiment at Fermi National Laboratory.

Data model

The CDF data is stored in a sequential IO format and accessed by a Fortran 77 common-based memory manager system called YBOS. The Fortran common is divided into variable length banks to store the objects of interest. Fortran variables act as crude pointers to store variable length data. The data used in this study was approximately 5000 event records, with each event record containing one or more bank types (typically 11 banks). The original CDF data model was well conceived and can be roughly mapped as "banks to relations" or "banks to classes" for either a relational data model or object-oriented data model, respectively. The actual number of relations was higher, since some of the banks having repeating-groups. The actual number of classes was higher, since additional classes were added reflecting structures in the data. The relational data schema consisted of 18 relations, containing a total of 484 attributes. The

object-oriented schema consisted of 25 classes, containing 465 attributes.

Design and Implementation

The data we analyzed was first translated from a compressed VAX binary format (YBOS) into a neutral format consisting of ASCII data arranged in C-structures. This data was then loaded into a commercial relational database, a commercial object oriented database, and ptool [3] and the benchmarks run. A front end was also developed to provide specialized visualization tools to examine the events returned and a uniform means of querying the data. Benchmarking the system with actual data, rather than simulated data, was very labor intensive, but crucial to our understanding of the requirements of the actual problem. It was also felt that it was essential to develop application specific benchmarks reflecting actual high energy physics queries and not to use general object-oriented benchmarks, such as in [2].

Benchmarks

In this section we briefly describe our three benchmarks and their implementation for each of the database engines.

Benchmark 1.

1. Find and label all muons having tracks with assigned electric charge.
2. Calculate a 4 vector for each selected muon.
3. Calculate the effective two-particle mass for all selected pairs of distinct muons using the 4 vector information.
4. Order and histogram the calculated mass.

Benchmark 2:

1. Select all events which have a muon pair mass in the range 3.0 to 3.2 GeV.
2. Calculate a mass constrained 4 vector for a J/psi particle.
3. For all particle tracks which are not muons, calculate a 4 vector with the assumption it is labeled a Kaon.
4. For each event which contains a J/psi and 1 (or more) Kaons, calculate the effective mass for all combinations of J/psi and Kaon pairs.
5. Order and histogram the mass.

Benchmark 3:

1. Calculate the effective mass for all distinct pairs of pions in each event.

Using the YBOS data format and tool kit, these queries are carried out using a fortran program of typically 300 lines of code. All 11 banks are read in from disk via a serial read for each event. These are loaded into a Fortran common. Appropriate banks and data values are obtained by index offset pointers into the common treated as a large fortran array. Almost all of the time is spent in the fortran serial read and loading the common with the banks. This is how the current production software works.

Using the relational data base, the banks were mapped into tables. SQL queries were constructed to selected the needed attributes to make the indicated calculations. The resulting table was sent to a separate histogramming package. Benchmarks 1 and 2 used three separate SQL queries each, while benchmark 3 required only one SQL select. The SQL provided a concise and compact statement of the query compared with the fortran program. The execution times were typically dominated by joins. In Benchmark 1, a self-join on a table of about 9000 rows; in Benchmark 2, the join of small table (5000 rows) with a large table (100000 rows); and in Benchmark 3, a self-join on a large (100000 row) table.

Both the object-oriented database and the object manager model the CDF data with C++ classes corresponding to the YBOS banks. Collections of events and other objects were implemented as linked lists, since queries generally touched each element in a collection. The queries consisted of C++ code fragments, which are compact and quite fast, but somewhat complex because of the depth of the pointers required.

For each of the benchmarks, the basic operation was to iterate, or loop, over collections of objects and applied simple C++ functions to select objects of interest. For Benchmark 1, one such loop was necessary; for Benchmark 2, three loops; and for Benchmark 3, two loops.

Performance

The performance of the three database engines for each of the benchmarks is reported in Table 1. The benchmarks were chosen to be representative of the queries expected once a working system is in use. The basic picture is clear and expected: for these types of

	YBOS	RDBMS	OODBMS	ptool
Benchmark 1	734	640	354	30
Benchmark 2	759	444	375	42
Benchmark 3	840	1964	440	100

Table 1: The time required to complete the three benchmarks using a commercial relational database (RDBMS), a commercial object oriented database management system (OODBMS), and the PASS persistent object manager ptool is given in seconds. The dataset was 109 Mbytes in size. The benchmarks were done on a Sun Microsystems Sparc Station 1 running Sun OS 4.1.3.

queries, performance improves by moving from a relational to an object-oriented database, and from an object-oriented database to an object manager. An object manager is about ten to twenty times faster than a relational database. Similar performance measures was obtained in the object operations benchmarks of Cattell [2].

We developed these benchmarks over a period of approximately eighteen months—although the precise numbers have changed quite a bit, the basic pattern just described has remained valid throughout the development. In particular, it has continued to remain valid, as we have refined the data models, improved the code implementing the benchmarks, and updated the system software.

4 Conclusion

We have described a proof-of-concept system which compared the analysis of high energy physics data using a commercial relational database, a commercial object oriented database, and a university developed persistent object manager. We also developed a series of benchmarks which were convincing to physicists and could quantify the performance advantages of using database and object management technologies. With the proof-of-concept system, we demonstrated that analysis of high energy physics data could be completed about an order of magnitude faster than with existing production code.

Acknowledgements

The work described here is part of the PASS Project. The PASS Project is funded by the DOE's HPCC Advanced Software Technology and Algorithms Program, DOE grant DE-FG02-92ER25133. In addition the development of ptool was also supported in part by NASA grant NAG2-513 and NSF grants IRI 9224605 and CDA 9303433. For more information, contact Robert Grossman, Laboratory for Advanced Computing, m/c 249, University of Illinois at Chicago, 851 S. Morgan Street, Chicago, IL 60607, grossman@eecs.uic.edu.

References

[1] A. Baden, L. Cormell, C. T. Day, R. Grossman, P. Leibold, D. Lifka, D. Liu, S. Loken, E. Lusk, J. F. MacFarlane, E. May, U. Nixdorf, L. E. Price, X. Qin, B. Scipioni, and T. Song, "Database Computing in HEP—Progress Report," *Computing in High Energy Physics 1992,* to appear.

[2] R. G. G. Cattell, "Object operations benchmark," *ACM Transactions on Database Systems,* to appear.

[3] R. L. Grossman and X. Qin, "Ptool: A Scalable Persistent Object Manager," *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data,* edited by R. T. Snodgrass and M. Winslett, ACM, New York, 1994, page 510.

[4] R. L. Grossman, X. Qin, and D. Valsamis, and D. Lifka, E. May, and D. Malon, and L. Price, "The Architecture of a Multi-level Object Store and its Application to the Analysis of High Energy Physics Data," *Laboratory for Advanced Computing Technical Report,* Number LAC 94-R8, University of Illinois at Chicago. December, 1993.

[5] R. Grossman, X. Qin, D. Valsamis, W. Xu, C. T. Day, S. Loken, J. F. MacFarlane, D. Quarrie, E. May, D. Lifka, D. Malon, L. E. Price, A. Baden, L. Cormell, P. Leibold, D. Liu, U. Nixdorf, B. Scipioni, and T. Song, "Analyzing High Energy Physics Data Using Database Computing," *Laboratory for Advanced Computing Technical Report,* Number LAC 94-R16, University of Illinois at Chicago. January, 1994.

[6] The PASS Collaboration, "An Architectural Model for the Petabyte Access and Storage (PASS) Project ," *Laboratory for Advanced Computing Technical Report,* Number LAC 94-R4, University of Illinois at Chicago. August, 1993.