SLAC-PUB-9179
September 2001

The *BABAR* Database: Challenges, Trends and Projections

I. Gaponenko[1], *A. Mokhtarani*[1], *S. Patton*[1], *D. Quarrie*[1], *A. Adesanya*[2], *J. Becla*[2], *A. Hanushevsky*[2], *A. Hasan*[2], *A. Trunov*[2]

[1] Lawrence Berkeley National Laboratory (LBNL), Berkeley, USA
[2] Stanford Linear Accelerator Center (SLAC), Stanford, USA

Abstract

The *BABAR* database, based upon the Objectivity[1] OO database management system, has been in production since early 1999. It has met its initial design requirements which were to accommodate a 100Hz event rate from the experiment at a scale of 200TB per year. However, with increased luminosity and changes in the physics requirements, these requirements have increased significantly for the current running period and will again increase in the future. New capabilities in the underlying ODBMS product, in particular those of multiple federation and read-only database support, have been incorporated into a new design that is backwards compatible with existing application code while offering scaling into the multi-petabyte size regime. Other optimizations, including the increased use of tightly coupled CORBA servers and an improved awareness of space inefficiencies, are also playing a part in meeting the new scaling requirements. We discuss these optimizations and the prospects for further scaling enhancements to address the longer-term needs of the experiment

Keywords: OO Database, ODBMS, Objectivity, Scaling, CORBA

1 Introduction

The *BABAR* experiment at SLAC was one of the first HEP experiments to use an object oriented database management system for its primary event store. The design of the *BABAR* database has been detailed before, most recently in [1] and several contributions to the CHEP 2000 Conference [2]. The database has been in production use since the experiment turned on in 1999 and has met its initial design goals which were to accommodate a 100Hz event rate at a scale of 200TB per year. However, several deficiencies became apparent during the first 18 months of running and it was clear that a major upgrade would be necessary in order to address these, and in order to scale to the increased luminosity running that was expected during 2001 and 2002.

The major deficiencies that were identified were:

- The architectural limitation for no more than 64k database files in a federation meant that very large files were used, particularly for the raw and reconstructed data where 20GB files were used. This had adverse operational impact, both at SLAC, but also in terms of data distribution to remote sites. The use of such large database files also meant that events from many different runs were mixed together within database files, causing the distribution of event samples to remote sites to also be inefficient.
- The architectural limitation on a single federation limited to 64k database files meant that there was no way of scaling to multiple petabytes of data, as was expected to be generated over the lifetime of the experiment.
- Correlated with the creation of large databases, some of the smaller databases used many containers, and scaling problems were seen when the number of containers in a database became a significant fraction of the architectural limit of 32k.
- The throughput of the Online Prompt Reconstruction (OPR) farm of processing and server nodes responsible for the pseudo real-time reconstruction of the event data was limited by saturation of the

[1] http://www.objectivity.com/

Work supported in part by the Department of Energy contract DE-AC03-76SF00515.
Presented at the International Conference on Computing in High-Energy and Nuclear Physics (CHEP'01), 9/3/2001 - 9/7/2001, Beijing, China

Objectivity lockserver[2]. This restricted the throughput to 100Hz on a farm of 150 processing nodes. Much of the lockserver traffic was determined to be related to resizing containers within a database.

- Job startup and closedown effects meant that although the design goal of a steady-state processing rate of 100Hz was achieved, the throughput integrated throughout the day fell significantly short of that steady state. This was particularly true for the case of the reprocessing farms, which are described in more detail in Chapter 4, where the finalize time was sometimes very large.
- Database creation significantly perturbed the steady-state processing and the throughput to the database server machines (AMS servers) was poor.
- Space utilization was poor, with the event size being considerably larger (300-400%) than the design goal.

A major review of the *BABAR* computing model was held in the fall of 2000. This arrived at the following conclusions and requirements for 2001 and the future:

- In order to improve the downstream processing of physics analysis samples and the efficiency of data distribution to remote sites, the output from the OPR farm should be split into 21 output streams rather than the 4 that were originally provided.
- The demands of data distribution would be best served by restricting most databases to about 500MB in size, with a maximum of 2GB for bulk data (*e.g.* raw).
- The throughput of OPR needed to scale by at least a factor of 2 during 2001, with a factor of 3-4 for 2002 being predicted.
- Support for multiple petabytes of data would be necessary over the lifetime of the experiment.

2 Multi-petabyte support

It was clear that addressing the simultaneous requirements for smaller databases but for multiple petabytes of data could only be met through a major architectural enhancement of the ODBMS. Several possibilities were examined in conjunction with the database vendor, with the selected one being support for multiple federations [3]. Thus it is now possible for a single application to simultaneously have access to multiple federations, each of which is still limited to 64k databases. This solution has the additional advantage of being scalable in terms of the number of lockservers, since multiple of them can be assigned to the ensemble of federations. Our plan is to have a pool of such lockservers, newly created federations being assigned in a round-robin fashion.

In fact the *BABAR* database architecture already mapped well onto multiple federation support since it had the concept of domains. The event store was one domain, the conditions database containing the time-varying response of the detector was another, and several others were defined. These domains could now be mapped onto different federations rather than being mapped onto regions of a single federation. This concept of *divested domains* relies upon a main federation that has links to the federations corresponding to the other domains, some of which might still reside in the primary federation. It is also currently necessary that the schema for all federations be identical, but existing operational procedures within *BABAR* ensure this. One possibility that is still being investigated is that of having different page sizes for the various federations, optimized to the particular domain. This might allow us to increase the page size for the event store federations, while retaining the existing page size for the conditions federation which needs to exist for the lifetime of the experiment and is already populated with data

In a multiple federation environment, it is necessary to be able to navigate between objects in different federations. Such a built-in long reference was not initially available, and backwards compatibility with our existing 100TB of event data meant that such references could not be widely used. We therefore decided to restrict their use to the event collections, which provide conventional references to individual events. The operational model within *BABAR* is for applications to iterate over the set of events provided by an input event collection (or set of collections), where these collections are hierarchical. Thus each node of the OPR farm writes out its events into a set of output collections (one per output stream), and a hierarchical collection for each stream binds these node collections into collections that contain the events for each run.

[2] The lockserver computer manages lock traffic associated with ensuring the integrity of the ODBMS in a multi-process environment

Any event can itself reside in multiple such collections.. *Bridge collections* are collections of event collections, where the collections can reside in multiple federations. Thus an application iterating over the set of events in a bridge collection will transparently switch between the corresponding federations.

One major implication of this restricted use of cross-federation references is that any output event data has to be written to the same federation as the input data if it needs to refer back to input data. Operationally this translates into the use of a new event store federation every few weeks in order to allow for later reprocessing of data. Of course it is possible to make a complete copy of the event through its transient representation such that it can be copied to a different federation.

3 Scaling the number of processing nodes

One short-term impact of increasing the number of output streams from 4 to 21 was an increase in the lockserver traffic that caused us to have to reduce the number of processing nodes from 150 to 100 in order to avoid saturation of the lockserver. This was unforeseen, but results from the number of locks being increased by a factor of 5 as a result of the increased number of containers being written to by the increased number of output streams, even though less data was written to any one stream. Interestingly, the lock traffic is essentially independent on the throughput for a given number of processing nodes and output streams. An upgrade to a processing node with 2.5 times better performance that is in progress is not therefore expected to significantly impact the lockserver traffic, while providing almost 2.5 times the overall throughput. The number of database servers is being increased in order to cope with this increased throughput. Tests with a lockserver machine having a faster clock speed have shown the expected improvement (approximately linear scaling against lockserver performance) and several such machines have been ordered.

A CORBA-based *Clustering Server* [4] has been developed to improve handling of the metadata that determines when new databases are created. These are now created asynchronously in advance of their use, and the same server allows the containers within these databases to be presized and reused across processing runs. This has significantly reduced the overall lock traffic since extending the size of containers was a major factor in that traffic. It also smoothed out the steady-state processing rate since the creation of new databases had previously been a stall-point that caused noticeable disruption to the steady-state processing. Reusing containers also means that fewer are necessary per database, which addresses the scaling problem that was seen when the number of containers approached the architectural limit.

These improvements have allowed the number of processing nodes to be increased from 100 to 220. Combining these improvements with use of multiple event store federations and bridge collections should allow scaling to approximately 500-600 processing nodes, at which point it is expected that the startup and closedown effects will start to impact the overall scaling.

A change in the operating model to allow processing to use the calibrations, not from the previous run as is currently the case, but from the run prior to that, would allow us to operate two independent OPR farms in leap-frog mode (operating on alternate runs), each of 500-600 processing nodes, with only a slight penalty in latency. Alternatively, a more major reorganization could allow for the required calibrations to be derived prior to the processing in OPR. This again would cost some latency, but in fact would allow data to be reconstructed with the “correct” calibrations. This would be feasible if not every event within a run were necessary for sufficient statistics to be made.

Finally since the lock traffic is heavily dependent on the number of output streams, another option would be to just do filtering and tagging in OPR so that all events were written to a single stream, deferring the splitting into different output streams to another fan-out stage. This would reduce the lock traffic on the initial cpu-bound stage, allowing more processing nodes to be added. Fewer nodes would be necessary during the fan-out stage since that would not be cpu-bound.

4 Reducing the startup and closedown effects

In addition to the OPR farm, there are other farms that are used for reprocessing of earlier data using the most recent production algorithms and calibration information. These farms are configured somewhat

similarly to the OPR farm, and the number of such farms has varied from one to three depending on the operational requirements.

One characteristic of having multiple farms is that the conditions information must be merged between the farms so that a consistent set of calibrations are available within each of them, as well as for physics analysis. The processing rate as a function of time for a typical OPR job is shown in Fig. 1. It is characterized by having a ramp up to a flat top, and finally a closedown time. A side-effect of the merging was to considerably elongate the closedown time for reprocessing under some conditions. Contributions to the ramp-up and closedown times have been analysed in the OPR and reprocessing environments. The major contribution to the ramp-up was discovered to be the processing nodes all attempting to locate and download the conditions information for the current run. A CORBA-based *OID Server* [4] has been developed to improve the performance for locating the conditions data by caching the results of the earliest requests. The conditions data itself has been distributed across more servers in order to improve the download performance.

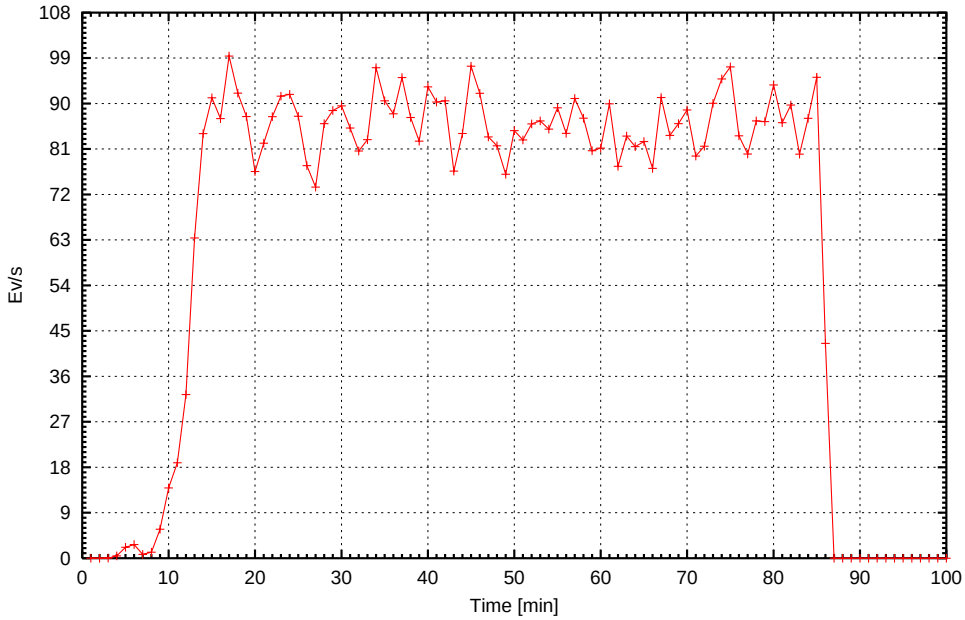

Fig 1 A processing rate for a typical OPR job

The major contributor to the closedown time is combining the partial results of all processing nodes and creating the summary conditions data that needs to be in place for the start of the next run. A third CORBA-based server is being developed in order to parallelize the writing of the 17 pieces of conditions information across multiple nodes rather than writing them serially. Finally the closedown time resulting from the merging of conditions information between the OPR and reprocessing farms has been addressed by a redesign of the merging procedure.

5 Space Optimizations

An exhaustive investigation of inefficiencies in the size of persistent objects and their placement into pages and containers has resulted in major space gains. Part of this came from a redesign of the infrastructure objects that provide navigational access to event data, and part from an exhaustive redesign of many of the data objects themselves. In some cases pre-sizing persistent arrays rather than extending them has significantly improved space utilization. These gains were only feasible after an extensive investigation into the detailed placement of objects within pages, containers and databases. The bulk data has been reduced by approximately 50% and somewhat smaller gains have been achieved in the summary data.

The database (AMS) servers utilize a file system backend developed by *BABAR*. Although the primary motivation for this is to provide support for staging and migration of files to and from tape, it has allowed

for extensive optimizations to be performed. One of these that is about to enter production is the capability to compress database files using a run-length encoding algorithm. This reduces the typical database size by approximately 50% at the cost of a slight cpu peformance penalty in the AMS server machines themselves.

The result of these optimizations is that the original design goals for space utilization are being achieved.

6 Trends and Challenges

It is clear from the above discussion that there is no single "silver bullet" for improving performance, whereas a single technical enhancement (multi-federation support) has been responsible for most of the scaling enhancements. In trying to understand performance limitations many possible effects have to be exhaustively examined. Once one limiting factor has been identified and an appropriate response has been designed and implemented, in most cases a totally different factor will determine the next limit. Some of these become obvious only with hindsight. File descriptor limits are an example of those. Others have taken painstaking detective work to uncover the source of the limit. One example of the latter resulted in changes to the way that databases were created and the containers presized, this significantly reducing the cpu load on the AMS server machines.

The above discussion has focused on the major improvements, but many others have had a small, but significant impact on the performance. These include the availability of read-only databases which reduce the lock traffic in the physics analysis federations and optimizations in the I/O to the RAID disks on the database servers for these same federations, taking advantage of the mainly read-only access to the databases in this environment.

One of the trends is to identify technology that is complementary to the ODBMS in order to minimize database access from multiple clients. The extensive use of CORBA servers to cache access to metadata information is an example of this. This is particularly true for processing farms where each processing node requires essentially the same setup and configuration information. Offloading some of the bookkeeping information (*e.g.* which databases are used by the specified event collection) to a relational database management system loaded by automated applications has reduced the load on the physics analysis federations, such that they can focus on their primary objective, which is to allow rapid access for physics analysis.

The extensive upgrades that are presently being installed are expected to provide performance scaling of approximately a factor of 10 relative to that available at the end of 2000, and to provide capacity scaling up to multiple petabytes. The latter is sufficient for the lifetime of the detector. The former should be adequate for the next 1 to 2 years, but thereafter the continuing improvements to the accelerator luminosity are expected to require another round of improvements. Some preliminary R&D has already been done in this direction, and it is expected that hardware improvements will also play an important part. There is an ongoing effort to optimize the ODBMS in conjunction with the vendor, and releases scheduled for the near-term future are specifically targeted at performance issues (*e.g.* reduced lockserver traffic for many database operations).

We believe therefore that at the technical level we will have addressed the short and medium term requirements once the present upgrade is fully installed into production. Multi-federation support is undergoing the final rounds of tests before being installed. Many enhancements are already in production use.

References

[1] The RD45 Collaboration, *RD45 Status Report*, CERN/LHCC 99-28, Sept. 1999

[2] Computing in High Energy and Nuclear Physics Conference, 2000, Padova, Spring 2000

[3] S. Patton *et al*., *Support in the BABAR database for Multiple Federations*, CHEP 2001, Beijing, Sept. 2001

[4] J. Becla *et al*., *Optimizing Parallel Access to the BABAR Database System using CORBA Servers*, CHEP 2001, Beijing, Sept. 2001