EXPERIENCE IMPROVING THE PERFORMANCE OF READING AND DISPLAYING VERY LARGE DATASETS*

Ted D'Ottavio, Bartosz Frak, John Morris, Seth Nemesure
BNL, Upton, New York, U.S.A.

Abstract

There has been an increasing need over the last 5 years within the BNL accelerator community (primarily within the RF and Instrumentation groups) to collect, store and display data at high frequencies (1-10 kHz). Data throughput considerations when storing this data are manageable. But requests to display gigabytes of the collected data can quickly tax the speed at which data can be read from storage, transported over a network, and displayed on a users computer monitor. This paper reports on efforts to improve the performance of both reading and displaying data collected by our data logging system. Our primary means of improving performance was to build a Data Server – a hardware/software server solution built to respond to client requests for data. Its job is to improve performance by 1) improving the speed at which data is read from disk, and 2) culling the data so that the returned datasets are visually indistinguishable from the requested datasets. This paper reports on statistics that we've accumulated over the last two years that show improved data processing speeds and associated increases in the number and average size of client requests.

INTRODUCTION

Over the last three years, we have put substantial effort into improving the speed at which we can read and display logged data. The need for this improvement has resulted from a huge increase in the amount of data that has been logged (8 TB in 2009 to over 100 TB in 2013) and a subsequent increase in the amount of data users need to view with each display request.

The work reported here should be considered a follow-up to work reported in a previous ICALEPCS paper titled "Improving Data Retrieval Rates Using Remote Data Servers" presented in 2011 [1]. That paper describes the construction and early testing of a specialized server, which we call a Data Server, designed to improve the processing and delivery of logged data. Since that first paper was written, we have begun to use the Data Server operationally and can now report on usage and performance data collected during the most recent operational time period of our RHIC collider.

As described in the 2011 paper, the Data Server is an enterprise grade middleware application server based on Java EE6 and Glassfish 3.1. It communicates with clients via standard HTTP protocols and has a high speed connection to the file system holding the logged data. Requests to the server from our logging display program, called LogView, are split up and distributed to 8 separate modules that read and process the logged data, which is then returned to the client.

Work supported by Brookhaven Science Associates, LLC under Contract No. DE-AC02-98CH10886 with the U.S. Department of Energy.

The client does not receive the full set of logged data requested, but a culled dataset designed to be virtually indistinguishable to the user on a standard scatter plot. This allows requests for tens or hundreds of millions of data points to be reduced to culled datasets consisting of 20 thousand points. All of this makes it possible to read, transport and display huge amounts of logged data in a few seconds. A visual representation of this process and its benefits compared to a client reading data directly is shown below in Figure 1.

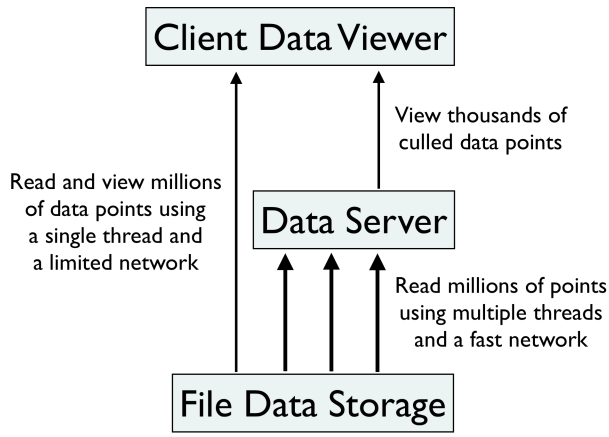

Figure 1: Benefits of using an intermediate Data Server vs. reading and displaying data directly from disk.

RESULTS

We began using the Data Server operationally in 2012. Our LogView data viewer was modified with a menu option that determined how the Data Server was to be used. The "Auto" default option left it up to the program to determine whether to use the Data Server or not. LogView was then programmed to use the Data Server for high volume requests and to read data directly from disk for low volume requests. Options for using the Data Server "Always" or "Never" are also available. These options proved useful in determining how to transition the Data Server into our system and helped to diagnose problems as they occurred.

Starting in late 2012, we began measuring and storing performance data for each request to view logged data. If LogView read the data directly, it recorded the size of the data collected and the time it took to read and display the data. If the request was sent to the Data Server, the server stored the same information. As our logging system is heavily used, we quickly amassed a lot of performance data. Data for a 5 month time period from March through July of 2013 (our most recent RHIC running period) is shown in Table 1.

ISBN 978-3-95450-139-7

Table 1: Logged Data Display Statistics

	Client Direct to Disk	Client via Data Server to Disk	Client via Data Server to Cache
Total Number of Data Requests	75K	22K	1.1M
Total Amount of Data Requested	0.18 TB	2.4 TB	11.8 TB
Average Processing Time	4.6 sec	3.7 sec	1.2 sec
Average Amount of Data Requested	6.4 MB	133 MB	92 MB
Average Data Processing Speed	3.4 MB/sec	61 MB/sec	285 MB/sec

DISCUSSION

Table 1 shows three columns of data. The methods used to collect and display the data represented by the three columns is described below.

- **Client Direct to Disk** – This column represents data requests that were completely satisfied by our LogView client. That is, LogView read all of the data directly from disk, stored it in memory, then displayed a culled dataset to the user.
- **Client via Data Server to Disk** – Data requests to LogView were routed to the Data Server, which read the data from disk, culled the data, and returned the culled dataset to LogView for display.
- **Client via Data Server to Cache** – Data requests were routed to the Data Server, which retrieved the data from its memory cache, culled the data, and returned it for display. These usually represent client requests to zoom into previously displayed datasets.

As can be seen from the data in Table 1, LogView satisfied about three times as many requests itself as it sent to the Data Server, but the average request size was much smaller (6.4 MB vs 133 MB). This was primarily due to LogView automatically routing only large data requests to the Data Server. This was an attempt to make sure the Data Server was not overloaded with requests that could easily be satisfied without it.

More importantly, this data shows that the Data Server is able to handle much larger requests (average 133 vs. 6.4 MB) while reducing the time the user needs to wait before the data is displayed (average 3.7 vs. 4.6 secs). The average data processing speed shows an improvement of about a factor of 18 (61 vs. 3.4 MB/sec). More detailed investigations show that processing speeds for both of these scenarios are dominated by the time it takes to read the data from disk and that the display times scale fairly linearly with the size of the data request. Note that the logged data read during these tests is data that had been compressed before disk storage (average 4x compression). The data sizes and rates reported, though, represent sizes measured after the data had been uncompressed.

Improvements and Future Directions

Over the last two years, we have made a couple of noteable additions to the Data Server. We added the capability to filter data requests based on machine-specific contexts such as "only return data taken when the RHIC collider was filled with beam" or "only return data taken when the collider was ramping", etc. We also added a second, identical Data Server to improve performance and reliability, with a proxy server receiving the data requests and routing them to the least busy Data Server. A java client interface was also constructed so that our java-based applications could also read logged data via the Data Server.

Although we are very happy with the performance gains we have achieved with the Data Server, if history is any guide, it won't be long before we will need even higher performance. One promising approach is probably contained within the data shown in the third column of Table 1. Note how much better the performance of the Data Server is if the data it is reading is already available in its memory cache (285 vs. 61 MB/sec). So getting the data off disk is clearly a significant bottleneck. Improvements in this area can be made by using a more distributed data storage and processing approach, by using improved network speeds, and/or by storing more data in static RAM memory.

SUMMARY

The speed at which large amounts of logged data can be read and displayed can be signficantly improved by routing these requests to a server specifically designed to 1) read the data quickly, and 2) return an intelligently culled dataset to the client for display. A server of this type used operationally at BNL over the last year has improved performance by about a factor of 18, keeping the time to read and display requests of hundreds of megabytes of data to a few seconds.

REFERENCES

[1] T. D'Ottavio, B. Frak, S. Nemesure, and J. Morris, "Improving Data Retrieval Rates Using Remote Data Servers", ICALEPCS (2011), MOMAU002, http://accelconf.web.cern.ch/accelconf/icalepcs2011/papers/momau002.pdf

ISBN 978-3-95450-139-7