A Scalable Framework for the Collaborative Analysis of Scientific Data

Jaliya Ekanayake, Shrideep Pallickara and Geoffrey Fox
Department of Computer Science
Indiana University, Bloomington, IN 47404, USA
{jekanaya,spallick,gcf}@indiana.edu

Abstract

Data analysis involving the processing of large amounts of distributed data is becoming increasingly common. This is especially true in the scientific community where voluminous data is routinely produced and consumed by instruments, experiments, models and simulations. Despite the ever increasing compute capabilities of modern processors, the distributed nature and the sheer volume of the data that needs to be processed presents unique challenges that need to be addressed in a distributed fashion. The composition property is a feature often found in many such data analysis; here, the analysis can be divided into multiple subtasks, which are performed in parallel and the results produced therein are then merged to form the final result. Scientists often work as teams and typically collaborate over the analysis of data. In this proposal we present a scalable, collaborative data analysis framework for analysis tasks that have the composition property.

1. Introduction

In recent years there has been an exponential growth in the amount of data that needs to be processed by scientific applications. The producers, consumers and the processors of this data are all distributed. Scientists cite two core requirements for data analysis frameworks: scale and collaboration. Scale corresponds to the ability to deal with increases not only in data volumes but also in the number of distributed components. Collaboration corresponds to the ability to perform this data analysis collaboratively in groups with fluid memberships while at the same time retaining control of authorizations to be part of a collaborative session. Two other requirements of this framework are security and fault-tolerance. Security corresponds to the ability to have confidential and authorized interactions with entities, while at the same time dealing with issues related to tampering and denial of service attacks. The fault tolerance aspect corresponds to the ability to sustain failures that take place within the system. This thesis focuses on the design of a scalable framework for the secure, fault tolerant collaborative analysis of scientific data.

In this thesis we focus on two domains viz. Particle Physics and Information Retrieval. Data processing in these domains often follow the *composition* property, wherein a task can be split into smaller sub-tasks, and the results from these sub-tasks composed to constitute the final result.

To delve a little deeper into the composition property, consider a common practice in many particle physics data analysis where the aim is to identify particular type of events within a collection of millions of events. The result of such analysis would be a histogram of possible events. The analysis task is to go through all the events available and identify a particular set of events. We can easily break the initial set of events into smaller subsets and run the same analysis software on these subsets concurrently. The resulting histograms can then be combined to produce the final result. This approach is much more scalable than running the entire analysis as a single task, which would preclude the possibility of concurrent analysis.

Information retrieval is another example where we can apply the composition property to accrue performance gains. A typical information retrieval system comprises information collection, indexing and query processing subcomponents which normally operate on a single collection of documents. The documents can be simple textual data, multimedia documents or documents with linked structures such as web pages. Querying the same document collection with multiple queries or querying different document collections with same query are two scenarios where we can apply the composition property to gain scalability. In a typical information retrieval scenario, merging the results may involve computing new relevance ranking values for the information retrieved from either multiple sources or using multiple similar queries.

Apart from these examples, there are other data analysis tasks that have the composition property. Image analysis tasks in astronomy and the genome sequence analysis in biology are data analysis tasks which have the composition property. However, there

are other analysis tasks that do not have this composition property. Distributed simulations, for example, would typically not be composable in this fashion.

A data analysis task that has the composition property has two distinct phases.

1. Analysis of the data subsets.
2. Merging results obtained from subset processing and applying various models.

These two phases fit very well the compute capabilities available within computational grids. The data analysis phase can be best handled by using multiple machines or clusters or grid computers while the less computationally intensive task of merging results and fitting deferent models can be handled at the user machines as depicted in Figure 1. In addition, the results produced by these data analyses may not only be of interest to the scientist who originally ran the analysis, but also to many other researchers in disparate geographical locations who are doing similar research may want to see these results. An infrastructure capable of handling both the analysis aspects and also the collaborative aspects of the analysis would benefit the scientists involved in similar scientific experiments.

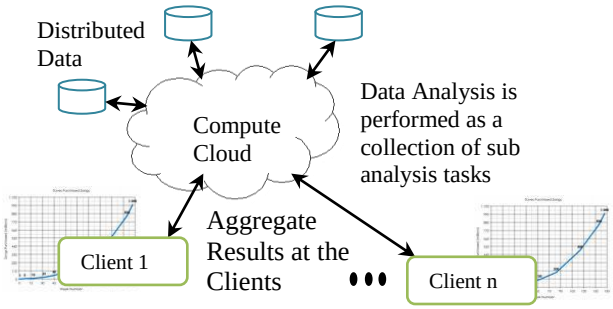

Figure 1. Composition property in data analysis.

In this thesis we will focus on providing a framework for scalable, collaborative analysis of datasets within the Particle Physics domains. We will also investigate the applicability of our approach to data analysis tasks in other fields such as information retrieval and astronomical image processing.

The rest of the paper is organized as follows. In section 2 we describe the problem of data analysis in Particle Physics. Section three describes the three main software subsystems that we intend to use in our design. Section four explains the proposed solution elaborating on the key design goals and the section five describes the work done so far in this regard. Related work is the section six and in section seven we give our conclusions.

2. High Energy Physics Data Analysis

Particle accelerators and colliders play a major part in most of the High Energy Physics(HEP) experiments and analyzing the large amount of data produced by these instruments is a significant challenge. The Large Hadron Collider (LHC) [1], a particle accelerator and collider, currently under construction will produce an enormous amount of data once it is in operation. It is expected to produce tens of Petabytes (PB) of data per year even after multiple layers of filtration. Processing these large sets of data is one of the many challenges faced by particle physicists. For example, the estimated processing power required for reconstruction, simulation and analysis of the data produces by the Compact Muon Solenoid (CMS) [2] in LHC is 2^7SPECint2000 or equivalent of ~15000 of today's processors. In addition, the CMS has been built as a collaboration involving several countries and collaborative data analysis is a requirement.

Data analysis in HEP experiments usually requires processing of all the reconstructed events (even after multiple layers of online filtrations) produced during an experiment to identify events with some topology of interest. The *de facto* standard analysis tool used today in HEP is the ROOT framework [3]. According to Holtman[4] a typical data analysis that a physicist wants to do, would be:

"*run this next version of the system I am developing to locate the Higgs events, and create a plot of these parameters that I will use to determine the properties of this version.*"

Another possible task would be to perform a similar analysis not by a single physicist but by an operator on behalf of a group of collaborative physicists. The physicist needs to perform two main tasks:

1. Develop an analysis function to identify features in experimental data and test it using sample data.
2. Locate a relevant data set and run the analysis function on each data file of this data set.

Although these two steps simply explain the analysis task at hand, it hides one critical requirement that is the movement of data from its original location to the physicist's desktop computer or the cluster. It would be ideal if the data and the computing power are located at the same place. However, in most situations this data movement is inevitable and the accessibility of processing power and the necessary software are key factors in deciding where to move the data. The movement of data can be minimized if the analysis can be performed at a close (network) proximity to the data. This requires a mechanism for physicists to

access the computing power available in those locations and run their analysis on those computers.

The Clarens [5] Grid-Enabled Web Services Framework is a "portal" for ubiquitous access to data and computational resources provided by computational grids. Clarens provides libraries to utilize ROOT functionalities at both the server and the client. Typical steps that a physicist should execute to analyze data using Clarens are as follows.

- Physicist writes and tests the data analysis function on sample data which normally involves functions written using ROOT.
- Physicist then uses Clarens' client libraries to upload the analysis function to the Clarens server and execute it over the available data files using web service interfaces.
- After the analysis, the client library can be used to retrieve the resulting histograms of the analysis tasks and merge them to produce the final result.

Although Clarens provides a solution to accessing computing resources and the movement of data, it does not provide a solution for analyzing data from multiple geographic locations or the collaborative analysis of data.

As mentioned in the introduction section, the main goal of this research is to investigate data analysis tasks that can be found in various scientific domains have the composition property, and to design a software system and the necessary algorithms capable of analyzing such data in secure, fault-tolerant and collaborative fashion. To achieve this we incorporate the following widely known software subsystems: Clarens, NaradaBrokering [6] and the ROOT Analysis Framework. We would also provide a detailed analysis on the applicability of the designed system in similar data analysis.

3. Software Sub Systems

3.1. ROOT Analysis Framework

ROOT is an object oriented data analysis framework capable of analyzing large amounts of data in a very efficient manner. The ROOT framework includes support for one, two and three dimensional histograms and also various curve fitting, function evaluation, minimization, graphics and visualization techniques in its class libraries. C++ is the programming language for ROOT and it also comes with a built-in C++ interpreter named CINT [7] which makes prototyping very easy by providing C++ based scripting language. Capabilities of the ROOT framework can easily be extended by linking external libraries, a feature we will use in our design as well, making it the most popular data acquisition, analysis and simulation framework for scientific data.

2.2. Clarens Server

Clarens is a grid enabled web service framework that supports most common web service protocol stacks comprising HTTP, SOAP/XML-RPC and SSL/TLS encryption and X.509 certificate-based authentication implemented in Python. Although the server implementation of Clarens is completely Python based, it provides client libraries for other languages such as Python, Iguana, JavaScript, and most importantly the C++ based interpreted language supported by the ROOT Analysis framework. Support for all the above features as well as the integrated support for ROOT makes Clarens a key framework for particle physics data analysis.

2.3. Naradabrokering

A content distribution infrastructure based on the publish/subscribe paradigm. The NaradaBrokering substrate itself comprises a distributed network of cooperating broker nodes. It can be used as a message oriented middleware or as a notification framework. Communication within NaradaBrokering is asynchronous and the substrate places no constraints either on the size, rate or scope of the interactions encapsulated within events or the number of entities present in the system. Also, it provides support for wide verity of transport protocols making it an ideal messaging infrastructure for heterogeneous distributed systems.

4. Proposed Solution

Before proceeding to a solution, we would like to present a set of features that the system should provide to support the above data analysis task as follows:

1. Functionality to create an experiment workspace to keep track of relevant content such as analysis scripts, configuration and, if required, resulting data files.
2. Upload analysis scripts into the experiment's workspace.
3. Execute the analysis in multiple geographic locations where the data is available.
4. Combine the resulting histograms from the data analysis subtasks.
5. Apply different models on the results to see how they fit into the results.

6. Perform the above scenarios in a secure and fault tolerant manner.

4.1. Distributed Analysis

A system capable of supporting the above requirements can be architected by incorporating a messaging substrate with the Clarens Server and a set of agents which keep track of the available Clarens Servers and the list of ongoing experiments. Figure 2 shows a high level architecture diagram of the system.

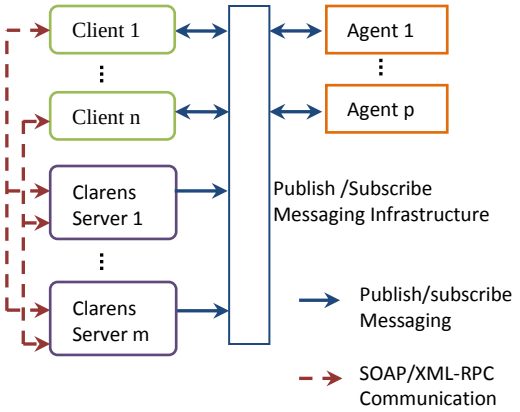

Figure2. Architecture of the proposed data analysis framework.

At startup, each Clarens server notifies its availability to the agents using a messaging infrastructure, and the agents start keeping track of the available servers thereafter. We have given complete details of a scalable approach to tracking entities in a distributed system in a secure and authorized manner in Ref [8]. The physicist that initiates the analysis uses his/her client-software (simply client hereafter) to locate the available servers and browse to see the data available for the analysis. In order to discover available Clarens servers the client first has to discover an agent and query it to get a list of available servers. After the discovery phase, the physicist can proceed to create an experiment space to upload the analysis scripts and start invoking them against the selected data. During the creation of the experiment space, the client application notifies the agent about the experiment details such as, the data files, the analysis files and other settings such as communication topics. During these steps the publish/subscribe infrastructure is used for communications between the client and agents and also to receive notifications from Clarens servers (shown in thick arrows in Figure 2). The client uses SOAP/XML-RPC communication to invoke web services in the Clarens server.

4.2. Collaboration

Once an experiment is set up by a physicist, other participants can join the experiment using the same client software. In this scenario, they will simply discover experiments and subscribe to receive notifications from the ongoing experiments or simply retrieve results from completed experiments. The client software can be configured to merge and apply a "fit" function for the received events. The client software can download these functions from the experiment space or can be configured to use a different one. The shared event based collaboration allows all the participants to receive the results of the individual analysis tasks in real time.

4.3. Security

Security is another area of research within this framework. The Clarens Server provides authentication [9] via the Public Key Infrastructure, which is based on the X509 certificates issued by a trusted Certificate Authority, and authorization via a *gridmap* file similar to other grid frameworks such as the GlobusToolkit [10]. Communication between the clients and different Clarens Servers uses transport layer security using SSL. In a companion paper S. Pallickara et al. [11] have given a detailed description of NaradaBrokering's secure and authorized end-to-end delivery capability of streams. Integrating these software systems so that the security infrastructure of the framework is transparent to the user is an important design aspect. We will be designing the necessary components to integrate these two security schemes so that the overall system is compliant with grid security standards such as Grid Security Infrastructure [12].

4.4. Fault Tolerance

The proposed architecture does not have any central control point such as a "portal" and hence poses no threat of a single-point-of-failure. The messaging infrastructure can be configured as a set of broker network which supports fault tolerance [13]. Agents keep track of the ongoing experiments and other bookkeeping information relevant to the experiments. We incorporate a discovery mechanism for agents wherein the client is allowed to discover an available agent based on certain criteria such as response time. Once such an agent is discovered, the client uses that agent to run the experiments. Agents also use a gossip protocol to synchronize the experiment's metadata between them: this allows the system to tolerate individual agent failures. Clarens server failures

during the execution of a data analysis task can cause an incomplete result set. However, the composition nature of the data analysis task makes it easy to restart the analysis in a different Clarens server for the data files that have not been analyzed and combine the results from that point on. User intervention is required in selecting a new Clarens server and also for restarting the analysis.

5. Work So far

NaradaBrokering is a messaging substrate developed entirely using Java and we plan to use it as the messaging infrastructure for the above architecture. However, integrating Naradabrokering written in Java with ROOT written in C++ and Clarens written in Python is one of the major challenges we had to face. The solution to the above required:

1. Providing a C++ client library for Naradabrokering so that other C++ programs can utilize publish/subscribe capabilities of NaradaBrokering.
2. Provide a ROOT compatible version of the same so that the interpreted code (interpreted by CINT) can publish messages to and receive subscription messages from the Clarens servers.

We developed the C++ Client for Naradabrokering in a generic form so that other programs can also use it to utilize NaradaBrokering. The Application Programming Interface (API) was kept very simple so that a user can publish a message containing a set of bytes to a topic and subscribe to receive messages from any topic. To use this library with ROOT requires writing the necessary wrapper classes so that the ROOT interpreter can link to the C++ library.

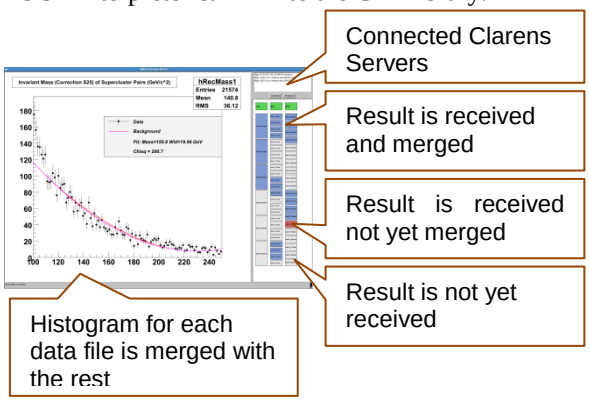

Figure3. Proof of concept implementation.

With the above implementation we were able to develop a proof of concept client written in the CINT interpreted language. The client (if used for submitting analysis jobs) first connects to the available Clarens servers and submits jobs for analyzing data available on those servers. Then it waits on a subscription over a given topic and retrieves partial results, merges with previously received histograms and visualizes the result. Other collaborative clients can use the same software to retrieve, merge and visualize histograms as and when they are available by subscribing to the same topic. This program utilizes Clarens, ROOT and the C++ Client library for Naradabrokering to provide the desired capabilities. Figure 3 shows a user interface of the client application when an analysis is in progress.

6. Related work

Grid-enabled portals [14] have been proven to be an effective mechanism for exposing computing resources and distributed systems to various scientific communities. It is possible to develop a portal based solution for a data analysis task that is similar to HEP data analysis. The architecture, that we propose utilizes the computing power available at the user machines for merging the results of data analysis sub tasks. In a portal based system this would be handled by the portal itself. Portal based approach does not require additional software to be installed at the user machines. However, in our design the shared event based communication between the processing entities and the collaborative clients allows different models to be fit to the results available at the client side. The physicist may be experimenting with his code that fits a model to the results, and he can simply debug it by running it at the client side, without needing to upload it to the server. Our approach will be able to provide a better fault tolerance over the portal based approach since it does not have a single point of failure.

PROOF [15, 16] provides a cluster based scalable solution for analyzing large amount of particle physics data files in parallel. PROOFs analysis is controlled by a master node and it also supports multi-tier masters. The tree structured master-participant connections may cause a single point of failure where as in the proposed architecture we do not impose any such single point of control over the data analysis. Also the metadata regarding the analysis jobs are maintained by multiple agents. Client can always connect to a different agent if the one that it was connected to fails. Also, PROOF does not support collaborative data analysis.

GRID IR [17, 18] describes a proposed architecture for building an information retrieval system based on grid technology where Matthew at el. explain how the subtask processing can be exploited using grid technology. However, the document does not explain a possible method to combine the retrieved results by

multiple searches. Yangwoo Kim in his memo [19] describes a possible architecture for grid information retrieval based on the concept of virtual organizations. Here the focus is more on allocating the computation resource for three main tasks, information collection, indexing/search and query processing. J. Callan [20] discusses the advantages of using multiple databases to retrieve information with a detailed discussion on merging document rankings from multiple results sets from different databases.

N. Yamamoto et al. [21] discuss worldwide parallel and distributed data analysis in the observational astronomical field based on a network shared file system *Gfarm*. Here the main focus is to allow multiple processors to access data files and their replicas in an efficient manner. Their work does not support collaborative analysis.

8. Conclusion

In this proposal we have presented how data analysis tasks that have the composition property can benefit from using our approach. Many analysis takes can be broken down into multiple subtasks, which are then executed in parallel. The results from these analysis subtasks can then be combined to produce the final result. We have elaborated on a possible solution for large-scale data analysis tasks that are common in many particle physics experiments. We have also discussed the possibility of extending the approach to other similar tasks such as information retrieval.

9. Advisor's Note

Work on this started eight months ago and exploited the work done four months prior to that. The research will be completed in next twelve months or thereabouts.

10. References

[1] The Large Hadron Collider, http://lhc.web.cern.ch/lhc/

[2] Compact Muon Solenoid, http://cms.cern.ch/

[3] ROOT - An Object Oriented Data Analysis Framework, http://root.cern.ch/

[4] K. Holtman, CMS Data Grid System Overview and Requirements, The Compact Muon Solenoid (CMS) Experiment Note 2001/037, CERN, Switzerland, 2001.

[5] The Clarens Grid-Enabled Web Services Framework, http://clarens.sourceforge.net/

[6] The NaradaBrokering Project **@** Indiana University, http://www.naradabrokering.org/

[7] CINT – The CINT C/C++ Interpreter, http://root.cern.ch/twiki/bin/view/ROOT/CINT

[8] S Pallickara, J Ekanayake, G Fox: A Scalable Approach for the Secure and Authorized Tracking of the Availability of Entities in Distributed Systems, IPDPS 2007: 1-10.

[9] Steenberg et al .The Clarens Web Services Architecture., Proceedings of CHEP2003, La Jolla, Paper MONT008, 2003.

[10] The Globus Toolkit, http://www.globus.org/toolkit/

[11] S Pallickara et al: A Framework for Secure End-to-End Delivery of Messages in publish/Subscribe Systems, GRID 2006: 215-222.

[12] Overview of the Grid Security Infrastructure, http://www.globus.org/security/overview.html

[13] S Pallickara, H Bulut, G Fox. Fault-Tolerant Reliable Delivery of Messages in Distributed Publish/Subscribe Systems, Fourth International Conference on Autonomic Computing. 2007 Page(s):19 – 19.

[14] G. Fox, D. Gannon, and M. Thomas, "Editorial: A Summary of Grid Computing Environments." Concurrency and Computation: Practice and Experience, Vol. 14, No. 13-15, pp. 1035-1044 (2002).

[15] PROOF – The Parallel ROOT Facilityhttp://root.cern.ch/twiki/bin/view/ROOT/PROOF

[16] Fons Rademakers, PROOF will analyse LHC data, CERN NEWSLETTER, Feb 9, 2006.

[17] GRID IR - GRID Information Retrieval, http://www.w3c.rl.ac.uk/Euroweb/poster/112/gridir.html

[18] Dovey, M. J. (2002). Music GRID: A Collaborative Virtual Organization for Music Information Retrieval Collaboration and Evaluation. In the MIR/MDL Evaluation Project White Paper Collection (2nd ed., pp. 50--52), Champaign, IL: GSLIS.

[19] Yangwoo Kim, Grid Information Retrieval System for Dynamically Reconfigurable Virtual Organization, Memo for Grid Information Retrieval Working Group (GIR-WG).

[20] J. Callan, Distributed information retrieval, In W.B. Croft, editor, Advances in information retrieval, chapter 5, pages 127-150, Kluwer Academic Publishers, 2000.

[21] Naotaka Yamamoto, Osamu Tatebe, Satoshi Sekiguchi, Parallel and Distributed Astronomical Data Analysis on Grid Datafarm, Proceedings of 5th IEEE/ACM International Workshop on Grid Computing,2004, pp.461-466, 2004.