IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 4, Issue 2, April 2021
Impact Factor: 4.819

Edge Computing for Optimizing Data Locality Access for Analysis

Mrs. S. R. Sowmiya[1] and Ms. R. Ratchani[2]
Assistant Professor, Department of Computer Science and Engineering[1]
Master of computer Applications[2]
Dhanalakshmi Srinivasan Engineering College, Perambalur, TamilNadu

Abstract: *In traditional scientific applications are computationally intensive, recent applications requires more data-intensive analysis and visualization to extract knowledge from the explosive growth of scientific information and simulation data. In existing system ODDS leverages a distributed file system (DFS) to provide scalable data access for scientific analysis. In proposed system the edge computing is combined with an ODDS which prove to be effective for data accessibility. The file system is first converted into compute centric to data centric. Then the edge system is used to make these data available for local access.*

Keywords: Data accessibility, Data security, Distributed file storage, Edge computing.

I. INTRODUCTION

In traditional scientific applications are computationally intensive, recent applications require more data-intensive analysis and visualization to extract knowledge from the explosive growth of scientific information and simulation data. As the computational power and size of compute clusters continue to increase, the I/O read rates and associated network for these data-intensive applications have been unable to keep pace. These applications suffer from long I/O latency due to the movement of "big data" from the network/parallel file system, which results in a serious performance bottleneck. To address this problem, we proposed a novel approach called "ODDS" to optimize data-locality access in scientific data analysis and visualization. ODDS leverages a distributed file system (DFS) to provide scalable data access for scientific analysis. Through exploiting the information of underlying data distribution in DFS, ODDS employs a novel data-locality scheduler to transform a compute-centric mapping into a data-centric one and enables each computational process to access the needed data from a local or nearby storage node. ODDS is suitable for parallel applications with dynamic process-to-data scheduling and for applications with static process-to-data assignment. To demonstrate the efficacy of our methods, we present and evaluate ODDS in the context of two state-of-the art, scientific-analysis applications—mpiBLAST and ParaView —along with the Hadoop distributed file system (HDFS) across a wide variety of computing platform settings.

1.1 Proposed Algorithm

In proposed algorithm the edge computing is combined with ODDS which proves to be effective for data accessibility. The file system is first converted into compute centric to data centric. Then the edge system is used to make these data available for local access. The security for data is enhanced.

1.2 Focal Point

- Performance overhead is reduced.
- The size of compute clusters decreased.
- The data stored in local nodes are secured.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 4, Issue 2, April 2021
Impact Factor: 4.819

II. Environment Details

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use.

Software Specification:

- Front end = java 8 and above
- Back end = mysql 5 and above
- Tool = netbeans 8.0 and above

Hardware Specification:

- Processor = intel i3 and above
- Ram = 4 gb and above
- Hard disk = 250 gb and above

2.1 System Architecture

1. Distributed file system creation
2. Location monitoring and data centric computation
3. Edge system construction
4. Data security

2.1.1 Distributed File System Creation

- In the distributed file system there will be a data server, render server, and client.
- The data server reads in files from shared storage and processes data through the pipeline to the render server that renders this processed data to present the results to the client.
- The data server can exploit data parallelism by partitioning the data and assigning each data server process a part of the dataset to analyze.
- By splitting the data, Para View is able to run data processing tasks in parallel

2.1.2. Location Monitoring and Data Centric Computation

- A data locality mapping scheduler which transforms a compute centric mapping into a data-centric one so that a computational process always accesses data from a local or nearby computation node
- A data location-aware monitor to support the mapping scheduler for obtaining the physical data distribution in the underlying file system

III. Edge System Construction

- Edge systems are constructed so that the data from different source can be provided to user locally.
- These edge system are temporary storage for this data

IV. Data Security

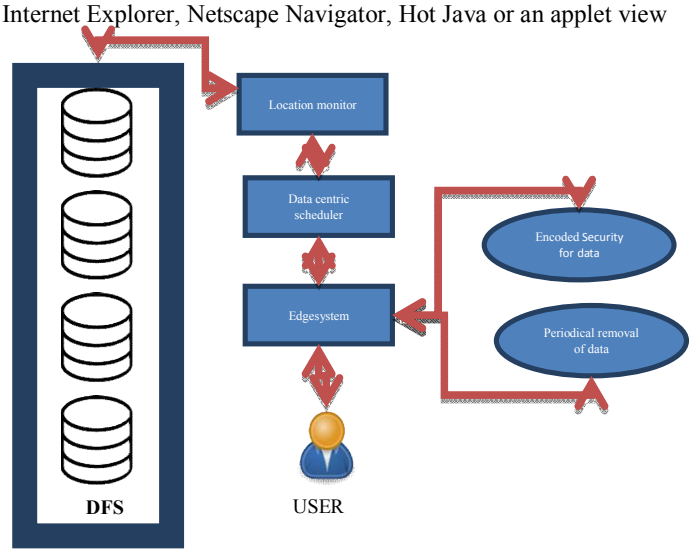
- Data security is ensured by encoded data at edge systems.
- These data are replaced with fake data when the edge system is removed.

4.1 Software Overview

Java is a platform Independent. Java is a high level programming language Introduced by Sun Microsystems in June 1995 Java is becoming a standard for Internet Applications. It provides for interactive processing and for the use of graphics and animation on the Internet. Since the Internet consists of different types of computers and operating

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 4, Issue 2, April 2021
Impact Factor: 4.819

systems, a common language was needed to enable computers to run programs that run on multiple platforms. Java is an object oriented language built upon C and C++.It derives its syntax from C and its object-oriented features are influenced by C++. Java can be used to create applications and applets. An application is a program that runs on the user's computer, under its operating system. An applet is a small window based program that runs on HTML page using Java enabled browser like Internet Explorer, Netscape Navigator, Hot Java or an applet view



4.2 Features of JAVA

4.2.1 Simple

- Java Language constructs are easy to learn and use. It takes care of memory management. Though
- Java was developed from C++, the complexities associated with C++ have been eliminated in Java.

4.2.2 Object-Oriented

Java is designed around the object-oriented model. In Java the focus is on the 'data' and the 'methods' that operate on the data in an application and not just on the procedures. The data and methods together describe the state and the behaviour of an object in Java.

4.2.3 Robust

Java is a robust language since it has strict compile time and run time checking of code. This minimizes programming errors. Error handling and recovery is taken care of in Java by the 'exception- handling' feature.

4.2.4 Secure

Java is language that focuses on the network. Java security features ensure that its programs that run are safe. Programmers cannot manipulate memory in Java. This is a good defense mechanism against malicious code that may flow in from the network. Java programs running on the Web cannot open, read, write or delete files on the user's system or run other programs on it.

4.2.5 Distributed

Java can be used to develop applications that are portable across multiple platforms, operating systems and graphical user interfaces. Java is designed to support network applications. Thus Java is widely used tool in an environment like the Internet where there are different platforms.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 4, Issue 2, April 2021
Impact Factor: 4.819

4.2.6 Multithreaded

Java programs can do many tasks simultaneously by a process called 'multithreading'. Java provides the master solution for synchronizing multiple processes. Therefore, interactive applications on the Net can run smoothly. This is made possible by the built-in support for threads.

- Running Java File with single command
- New utility methods in String class
- Local-Variable Syntax for Lambda Parameters
- Nested Based Access Control
- HTTP Client
- Reading/Writing Strings to and from the Files
- Flight Recorder

4.3 Technology Infrastructure

4.3.1 Core Java

Java can be used to create two types of programs: application and applet. An application is a program that runs on your computer, under the operating system of that computer. That is, an application created by java is more or less like one created using C or C++. When used to create application, java is not much different from any other computer language. Rather, it is java's ability to create applets that makes it important. An applet is an application designed to be transmitted over the internet and executed by a java-compatible Web Browser. An applet is actually a tiny java program, dynamically downloaded across the network, just like an image, sound file, or video clip. The important difference is that an applet is an intelligent program, not just an animation or media file. In other words, an applet is a program that can react to user input and dynamically change-not just run the same animation or sound over and over.

4.3.2 Security

As you are likely aware, every time that you download a "normal" program, you are risking viral infection. Prior to java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Even so, most users still worried about the possibility of infecting their system with a virus. When you use a java-compatible web browser, you can safely download java applets without fear of viral infection or malicious intent. Java achieves this protection by confining a java program to the java execution environment and not allowing it access to other parts of computer.

4.3.3 Portability

Many types of computers and operating systems are in use throughout the world-and many are connected to the internet. For program to be dynamically downloaded to all the various type of platforms connected to the Internet, some means of generating portable executable code is needed.

4.3.4 Byte Code

The key that allows java to solve both the security and the portability problems just described is that output of a java compiler is not executable code. Rather, it is BYTECODE. Byte code is a highly optimized set of instruction designed to be executed by the java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreted code.

4.4 System Maintenance

System maintenance is an ongoing activity, which covers a wide variety of activities, including removing program and design errors, updating documentation and test data and updating user support. For the purpose of convenience, maintenance may be categorized into three classes, namely:

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 4, Issue 2, April 2021
Impact Factor: 4.819

4.4.1 Corrective Maintenance

This type of maintenance implies removing errors in a program, which might have crept in the system due to faulty design or wrong assumptions. Thus, in corrective maintenance, processing or performance failures are repaired.

4.4.2 Adaptive Maintenance

In adaptive maintenance, program functions are changed to enable the information system to satisfy the information needs of the user. This type of maintenance may become necessary because of organizational changes which may include:

- Change in the organizational procedures,
- Change in organizational objectives, goals, policies, etc.
- Change in forms,
- Change in information needs of managers.
- Change in system controls and security needs, etc.

4.4.3 Perfective Maintenance

Perfective maintenance means adding new programs or modifying the existing programs to enhance the performance of the information system. This type of maintenance undertaken to respond to user's additional needs which may be due to the changes within or outside of the organization. Outside changes are primarily environmental changes, which may in the absence of system maintenance, render the information system ineffective and inefficient.

V. CONCLUSION

In this proposed method ODDS to optimize data-locality access in scientific analysis and visualization. ODDS employs a data-locality scheduler to transform a compute-centric mapping into a data-centric. This enables a computational process to access the needed data from a local or nearby storage node. ODDS is not only suitable for scientific applications with dynamic process-to-data scheduling but also for applications with static process-to-data assignment. ODDS can be adopted for newly developed scientific analysis programs and for existing scientific analysis applications such as ParaView and mpiBLAST. By conducting comprehensive experiments over 128-node clusters with four popular file systems, we found that ODDS can greatly reduce the I/O cost and double the overall execution performance as compared with the schemes that do not consider physical data distribution.

REFERENCES

[1]. (2014, mar.) libhdfs – hadoop wiki. apache software foundation. http://wiki.apache.org/hadoop/LibHDFS.
[2]. Fuse:. http://fuse.sourceforge.net/.
[3]. genomes:. http://aws.amazon.com/1000genomes/.
[4]. E. E. Abola, F. C. Bernstein, and T. F. Koetzle. Protein data bank. Technical report, Brookhaven National Lab., Upton, NY (USA), 1984.
[5]. J. Ahrens, B. Geveci, and C. Law. Para view: An enduser tool for large data visualization. *The Visualization Handbook, 717:731, 2005.*
[6]. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, D. J. Lipman, et al. Basic local alignment search tool.*Journal of molecular biology, 215(3):403–410, 1990.*
[7]. J. C. Bennett, H. Abbasi, P.-T. Bremer, R. Grout, A. Gyulassy,T. Jin, S. Klasky, H. Kolla, M. Parashar, V. Pascucci, P. Pebay, D. Thompson, H. Yu, F. Zhang, and J. Chen. Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12,* pages 49:1–49:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.