XSearch: Benchmark the state-of-the-art Distributed Search Platforms (on disk)

CS-554: Data-Intensive Computing

Project Proposal

People Involved:

- **Team Members:**

Ashish Ryot	A20405230	aryot@hawk.iit.edu
Nilesh Jorwar	A20405042	njorwar@hawk.iit.edu

- **Project Mentor:**

Alexandru Iulian Orhean aorhean@hawk.iit.edu

Abstract:

With the continuous growth in scientific and unstructured data, it has become necessary to bring the parallel and distributed search platform into work to perform indexing. Each day as the size of data grows exponentially, so does the demand for the faster information retrieval thus necessitating to build the robust and faster search engine. While currently available search engines perform better on distributed file systems and well-structured data, they fail to provide timely response and support for complex and lengthy search queries coming from the users for the retrieval of data generated by HPCs for scientific purposes. This project employs the use of custom data structures allowing faster indexing and searching capability in multithreaded distributed environment. We thus tend to reduce the computation time needed for improving indexing throughput of the search platforms. Impact of storing the indexes on memory and disk will be compared against the parallel file systems by measuring the bottlenecks.

Keywords- distributed filesystems, indexing, information retrieval, unstructured data

Background Information:

Computer clusters, supercomputers, grid systems and cloud systems have become popular platforms for performing scientific research, pushing forward not only the boundaries of scientific discovery but also leading the advancement of computer technology. Applications that generally run on these platforms can take hours or days to complete and commonly access and process data sets that reach sizes of hundreds of terabytes. In order to store and access large quantities of data computing centers and data centers employ parallel and distributed file systems at large scales, such as: IBM Spectrum Scale, Lustre, OrangeFS, the Hadoop File System and Ceph. While in literature the focus is on developing techniques and strategies for systems and applications to store, access and process data in a serene manner and in a reasonable amount of time, searching and aggregating information over the large storage systems are challenges that have not been thoroughly explored.

Large-scale distributed and parallel file systems are the basic in practice data management infrastructure in super-computing, HPC (High Performance Computing) environments and Big Data. Compared to a variety of database solutions, including traditional SQL databases, NoSQL databases, and NewSQL databases, file systems generally perform better and offer better scalability and flexibility (support for structured and unstructured data schemas as

well as unfixed data schemas). Therefore, a large proportion of analytical applications like MapReduce and scientific applications are still using file systems to access data.
The data organization by de coupling the hierarchical namespaces from the actual data gives the file system its superior performance. This de-coupling of data and organization provides opportunities to optimize I/O access patterns and increasing parallelism. [3]
But for large volumes of complex datasets, like system logs, web clicks, financial transaction records, and scientific experimental data, the concept of hierarchical file system namespace is beginning to get old and is unable at managing these complex datasets. The quantity of unstructured information that the HPCs can process and store is simply increasing the complexity of the information retrieval systems. This data is transferred to the distributed systems for storage and retrieval. Systems like Hadoop Distributed File System and Ceph provide high performance for this data. The enterprise level search platforms like the Apache Solr and the Elasticsearch are used to index and query this data effectively.

Problem Statement:

This project is based on the two well established domains of computer science namely information retrieval and distributed systems [4]. These days most of the research on information retrieval is focused on small well controlled homogeneous collections such as collections of news stories on a related topic and other structured data. Our project focuses on benchmarking the state-of-the-art search platforms like the Apache Solr and Elasticsearch on the file systems found in scientific computer systems (HPC), dominated by numerical data, high resolution images and specialized file formats that are usually distributed across many storage nodes and accessed in parallel [5]. Metrics like throughput, latency, scalability, availability, reliability and economics are significant in determining the efficiency and efficacy of such system [5]. In the first and second part of our project we will be focusing on metrics such as indexing latency, throughput and scalability, index size and distribution as they are the essential performance evaluation factors for Apache Solr and Elasticsearch. Following the same context of distributed unstructured information retrieval problem, the third part of the project takes a look at the custom implementation of the search platform and library namely GUFI (Grand Unified File Index). Here the comparisons between the performance of the enterprise level search platforms and GUFI are to be made and we will look at ways to push the limits of these state-of-the-art search platforms to find their bottlenecks and also find ways to boost or get good performance out of them.

Related Work:

Now-a-days various distributed indexing platforms are available that allow search and extract information from large scale systems but lacks in faster information retrieval. It becomes need to have faster indexing in short amount of time with submission of complex queries and without interference of external applications. However, there are search engines available, a slow search, primarily deals with high quality search experience while relaxing speed requirements [1]. Though this search engine gives us new search experience with high quality result but does not focus on the time factor considering the vast amount of data to search for indexing.
An alternative solution available, was to use of dynamic indexing in search engine, where indexer facilitates data organization for information retrieval and minimizes the time of query [2]. Indexing process is distributed over the cluster of computers in grid computing to improve the performance through distributed load. Though the process of distributing the indexed data over distributed memory helps faster indexing for small amount of data, say few petabytes of files, but significantly reduces the chances of faster data retrieval considering the millions of data resided on distributed storage systems. In addition to this, there is overhead of maintaining continuous index updates from crawled document to the master node in distributed system.

Proposed Solution:

To benchmark the stat-of-art distributed search platforms on disk, we are implementing Apache Solr, Elastic Search and custom search platform i.e. GUFI on distributed system. These platforms will be tested against the various parallel distributed systems such as Luster, OrangeFS, HDFS and Ceph. To achieve high performance in indexing and searching by using various information retrieval libraries like lucene, lucene plus plus, and GUFI while adhering

to multithreaded environment setup, it becomes essential to look for the custom data structures that helps reducing storage space and improves indexing.

Apache Solr -
This search engine will be implemented in Java. Solr is a wrapper for the Apache Lucene library that Lucene classes to generate this index, called Inverted index. When we query the Solr, it breaks the query and submits the query to different chunks or entities and matches them with the inverted index of documents created earlier. In the inverted index, all the search terms will be having associated document ids. Once the user issues a query, it will search for the terms and the associated documents. It is the optimized way to get the fast search results from the search engine. The documents obtained as a result of Solr's search are based on the similarity class and other constraints defined in the Schema.xml and Solr.config files. In order to achieve higher performance, the custom data structure will be implemented in Apache Solr for indexing.

Elasticsearch -
The second part of the project deals with implementation of real time search platform where there is slight latency between the time you index a document until it becomes searchable. Unlike Solr, Elasticsearch is schema-less table structure and built on Lucene but natively it is JSON + Restful and has Query DSL to make Lucene's query syntax accessible to the users though JSON syntax.

GUFI -
Project also includes the analysis of the proposed custom implementation of the search platform i.e. GUFI. We are planning to analyze performance of this implementation by identifying the bottlenecks performed over different parallel distributed systems.

Evaluation:

For the evaluation of the Apache Solr, Elastic Search and custom implementation of GUFI, we will be running the multi-node scalability experiments on Chameleon cluster to evaluate the indexing throughput, index size and query throughput, indexing latency when the index is stored on the disk. These experiments will be conducted on increasing number of files and increasing file size using multithreaded environment setup.

TimeLine:

Project Proposal	Week 1
Understanding and Building Search Platforms – Apache Solr and Elastic Search	Week 2
Benchmark Apache Solr and Elastic Search platforms on the Chameleon	Week 3-4
Analysis of Apache Solr and Elastic Search platforms	Week 5
Mid Term Progress Report	Week 6
Environment Setup for GUFI (Grand Unified File Index)	Week 7
Analysis and Performance Evaluation of GUFI (Grand Unified File Index)	Week 8-9
Preparation of Formal Final Report	Week 10
Final Presentation	Week 11

Deliverables:

- Mid-Semester Progress Report
- Final Formal Project Presentation
- Source Code
- Final Project Report

Conclusion:

In this project, we expect to learn and find ways to tune the parameters of the system, like finding just right number of threads, the buffer and file size to reduce the indexing latency and increasing indexing throughput. Use of custom data structures and their impact on indexing is expected to learn. Additionally, we will also check the type of documents and files support by the search platforms along with their capability for load balanced queries and query type support. We also look forward to learn the impact of data and index storage locations on indexing throughput and latency on various parallel distributed systems. In a comparison-based study, we will be measuring the performance that the state-of-art distributed search platforms has while running on typical text data vs system metadata from real supercomputers. We also hope to evaluate and compare the results obtained from these search platforms with other enterprise-level information retrieval libraries and would be thinking of improving the quality of search results.

References:

[1] Jaime Teevan, Kevyn Collins-Thompson, Ryen W. White, Susan T. Dumais & Yubin Kim, “Slow Search: Information Retrieval without Time Constraints” in Proceedings of HCIR 2013.

[2] M. E. Elaraby, Omaima Nomir, M. Sakre, “Dynamic and Distributed Indexing Architecture in Search Engine using Grid Computing” in International Journal of Computer Applications (0975 – 8887), Volume 55– No.5, October 2012.

[3] Lei Xu, Ziling Huang, Hong Jiang, Lei Tian, David Swanson, "VSFS: A Searchable Distributed File System" in Proceeding PDSW '14 Proceedings of the 9th Parallel Data Storage Workshop Pages 25-30

[4] S. Brin and L. Page, “Reprint of: The anatomy of a large-scale hypertextual web search engine,” Computer networks, vol. 56, no. 18, pp. 3825–3833, 2012.

[5] Alexandru Iulian Orhean, Kyle Chard, Ioan Raicu, "XSearch: Distributed Information Retrieval in Large-Scale Storage Systems" Illinois Institute of Technology, 2018