INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Fast Analysis & Storage of Big Data Using HDF5

Dheeraj Joshi
Department of Computer Science, Mumbai University, DJSCOE, Mumbai, India
Krima Shah
Department of Computer Science, Mumbai University, DJSCOE, Mumbai, India
Harshan Andrews
Department of Computer Science, Mumbai University, DJSCOE, Mumbai, India
Lakshmi Kurup
Professor, Department of Computer Science, Mumbai University, DJSCOE, Mumbai, India

Abstract:
Data is growing at a tremendous rate with an overall increase in the digital universe with petabytes of data flowing from different sources. The simplest explanation of this big data phenomenon is that, on one hand it's about large amounts of data, while on the other hand it is the difficulty to analyze these large data sets. The analytics require some specific data to be retrieved. This usually involves methods with large time and space complexities. In this paper, we describe the HDF5 data model and some of its performance enhancing capabilities that minimize the complexities involved in large data handling.

Key words*: Big Data, HDF5, Analytics*

1. Introduction

Big data has emerged as a buzzword in business IT over the past few years. Big data is a term that can be applied to some very specific characteristics in terms of scale and analysis of data. Analysis of data is a process of inspecting, transforming and modeling data with the goal of discovering useful information that supports decision making. Relatively small organizations can improve the business value and gain insight to revenue generating future strategies from analysis of their own data sets. This promotes business on a large scale and gives a thrust to global economies as well.

Data sets consist of data generated through business transactions, mergers, acquisition, etc. Various reports are generated for audits and acceptance procedures from these data sets. These data sets are initially in megabytes but as the business expands, the amount of data generated through that business also expands at an expeditious rate. Hence, the process of generating reports and deriving statistical information through analysis of these data sets becomes a tedious task to accomplish. Files of more than 2GB in size become almost impossible to parse, transform and retrieve records from. The optimum system resources available get exploited to their breaking point and generally collapse. This has adverse effects not only on the integrity and accuracy of these datasets but the entire business as a whole. Though, such analytic work involves retrieval of data records from large data sets, such operations require exorbitant resources for execution and are thereby not suitable for small organizations. In order to make analysis feasible for business of all statures, the HDF5 library proves to be an admirable solution.

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of data types, and is designed for flexible and efficient input and for high volumes of complex data. It is portable and is extensible, allowing applications to evolve in their use of HDF5.

2. Related Work

Large data sets consist of millions of data records. Hence, retrieval of some specific records from a pool of data requires a large amount of time as well as space due to its sequential structure. In order to store records in such a sequential order, pointers are used between them. These pointers increase traversal time thus, exhausting the available resources.

Over a period of time, new records get stacked up at the end of the structure with new pointers. Hence, the count of these pointers is directly equal to the number of records stored in the file. When a query is processed, the conditions that determine the result set have to be evaluated for each record. Let's assume that the time consumed for such evaluation of each record is 1 microsecond. Hence, for a record placed at n^{th} place, it would take minimum of n-1 microseconds to evaluate the condition, retrieve the record and place it into

the result set. It is clearly evident that processing queries on such large data sets makes the system incapable for other operations and at times the entire system fails leading to disastrous losses. This compromises with data integrity and atomicity.
This led to the development of libraries in order to minimize such space and time complexities. Hence, the HDF Library came into existence and made great strides in the field of Large Data Processing. Various versions of HDF have been developed in the past decade such as HDF4, HDF5, etc. with each improving the efficiency of traversal and storage of data.

3. Proposed Method

The solution, as proposed is using HDF5, (Hierarchical Data Format 5) which involves stacking up records in a tree structure rather than a linear structure. Classifying these records into groups or datasets and developing a hierarchy among these records makes it the unrivalled solution to our problem.

- The HDF5 File Format Specification is organized in three parts:
- Level 0: File signature and super block
- Level 1: File infrastructure
- Level 1A: B-link trees and B-tree nodes
- Level 1B: Group
- Level 1C: Group entry
- Level 1D: Local heaps
- Level 1E: Global heap
- Level 1F: Free-space index
- Level 2: Data object
- Level 2A: Data object headers
- Level 2B: Shared data object headers
- Level 2C: Data object data storage

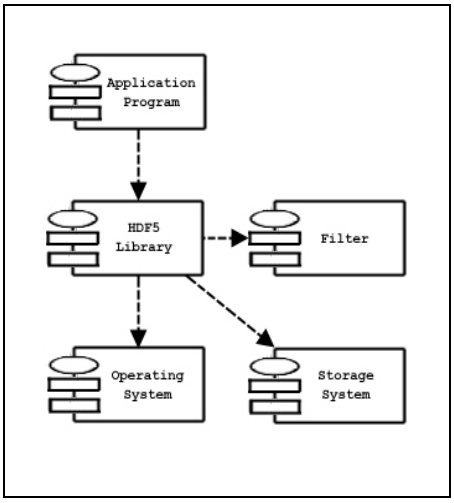

Figure 1

The Level 0 specification defines the header block for the file. Header block elements include a signature, version information, key parameters of the file layout (such as which VFL file drivers are needed), and pointers to the rest of the file. Level 1 defines the data structures used throughout the file: the B-trees, heaps, and groups. Level 2 defines the data structure for storing the data objects and data. In all cases, the data structures are completely specified so that every bit in the file can be faithfully interpreted.
It is important to realize that the structures defined in the HDF5 file format are not the same as the abstract data model: the object headers, heaps, and B-trees of the file specification are not represented in the abstract data model. The format defines a number of objects for managing the storage including header blocks, B-trees, and heaps. The HDF5 File Format Specification defines how the abstract objects (for example, groups and datasets) are represented as headers, B-tree blocks, and other elements.
The HDF5 Library implements operations to write HDF5 objects to the linear format and to read from the linear format to create HDF5 objects. It is important to realize that a single HDF5 abstract object is usually stored as several objects. A dataset, for example, might be stored in a header and in one or more data blocks, and these objects might not be contiguous on the hard disk.

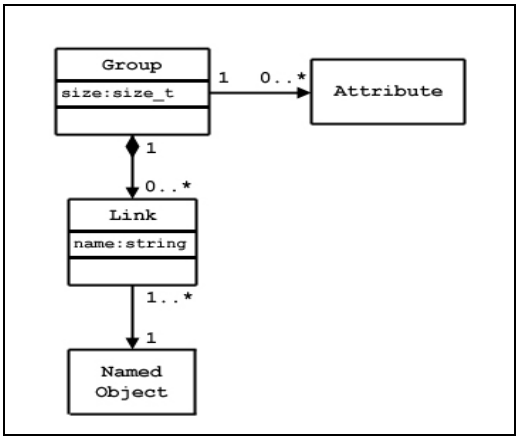

Figure 2

By creating several objects we minimize the structural information to be stored. The data structure is specified prior to the storage definition. The data are then classified and stored in these objects based on the structure pre-defined. Once the data have been allocated to the subdivided groups, pointers are initialized between these groups and the hierarchical model is erected. Thus, we can rescue the space complexities of large data sets with linear structure.

The creation of these files can be programmed using Python, C++ or C# language for reusability. As these languages consist of libraries developed using C, the speed of operation and processing of such files is quick. Using the inbuilt functions of the HDF5 library, creation of .h5 files becomes easier.

HDF5 has been well tested and modularized for Python and hence, it is the most widely used HDF libraries.

The second complexity is the retrieval of records from a large pool of data. Now, once we have the organized hierarchical structure in place, the large number of records can be placed as per classification of the records into groups or datasets. Each node has a data structure and according to the type of structure, the classification is developed. For example, for an employee dataset, a classification on the basis of gender will require a group structure. Hence, we can have two sub groups to the root node as male and female. Now, the records are filtered on the basis of gender resulting in two reduced datasets. This in turn reduces the number of pointers present in these records. Considering a traversal operation, the traversal up to the sub-group node, would cost 1 microsecond. The time required for traversal across the dataset would obviously be reduced as it consists of lesser number of records. Hence, the total time is considerably reduced with minimum utilization of resources. This makes the system stable and reliable for query processing and evaluation thereby making the process of business analysis faster, reliable as well as economical.

4. Future Scope

This solution has an evolving nature due to its hierarchical structure. One way of alleviating the tedious task of classification is by means of clustering. Clusters will result in a distribution of data into subsets and classifying them into mini datasets. Smaller the dataset's, lower the time for traversing across the records. Various mining techniques can also be developed to smartly classify data in such a way that it leads to stellar performance and faster operation.

Also, the HDF5 library should become a permanent component of programming modules in order to promote the use of HDF5 for data storage. In today's world, space complexities hardly matter but the most arduous challenge that is still prevalent is the time required for operations on large data.

The optimization of the tree structure will also result in some efficiency of allocating classifications to respective nodes. Data structure selection should also be developed based on the attributes of the data and their properties. This will supplement creation time of such files thereby improving business operations.

5. Conclusion

To sum it up, the existing systems are heavily reliable on resources rather than solutions with high performance. The proposed method combats the two biggest challenges faced by organizations while handling data of humungous magnitude. Therefore, in order to promote analysis throughout the enterprise chain, the proposed solution seems to be peerless. The open source licensed HDF library has developed at a meteoric rate and will continue to value high performance and cost-effectiveness.

6. References

1. The HDF Group http://www.hdfgroup.org/HDF5/
2. HDF5 Brochure 2012 http://www.hdfgroup.org/about/HDF5Brochure_2012.pdf
3. Atmospheric Science Data Centre https://eosweb.larc.nasa.gov/HBDOCS/hdf.html
4. Wolfram Reference Portal http://reference.wolfram.com/mathematica/ref/format/HDF5.html
5. UIUC, Lecture-1, Hierarchical Data Format http://www.cis.rit.edu/class/simg726/lectures/HDF/HDF_Lecture.pdf
6. H5PY, Python interface to the HDF5 binary data format. https://code.google.com/p/h5py/