A Survey on Compression Algorithms in Hadoop

Sampada Lovalekar
Department of IT
SIES Graduate School of Technology
Nerul, Navi Mumbai, India
sampada.lovalekar@gmail.com

***Abstract*—**Now a days, big data is hot term in IT. It contains large volume of data. This data may be structured, unstructured or semi structured. Each big data source has different characteristics like frequency, volume, velocity and veracity of the data. Reasons of growth in the volume is use of internet, smart phone ,social networks, GPS devices and so on. However, analyzing big data is a very challenging problem today. Traditional data warehouse systems are not able to handle this large amount of data. As the size is very large, compression will surely add the benefit to store this large size of data. This paper explains various compression techniques in hadoop.

Keywords*-bzip2, gzip ,lzo, lz4 ,snappy*

*****

I. INTRODUCTION

The volume of big data is growing day by day because of use of smart phones, internet, sensor devices etc. The three key characteristics of big data are volume, variety and value. Volume can be described as the large quantity of data generated because of use of technologies now a day. Big data comes in different formats like audio, video, image etc. This is variety. Data is generated in real time with demands for usable information to be served up as needed. Value is the value of that data whether it is more or less important.

Big data is used in many sectors like healthcare, banking, insurance and so on. The amount of data is increasing day by day. Big data sizes vary from a few dozen terabytes to many petabytes of data.

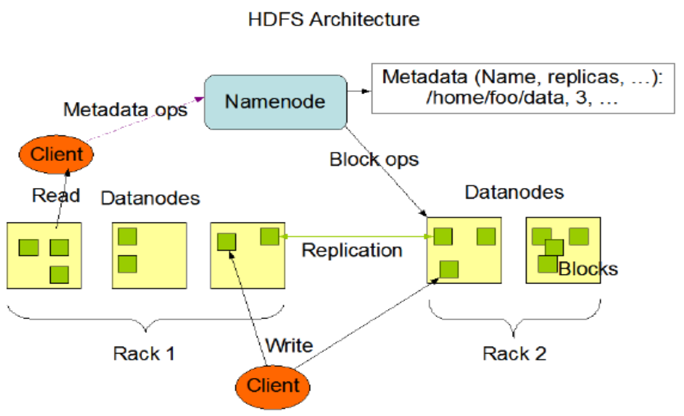
Big data doesn't only bring new data types and storage mechanisms, but new types of analysis as well. Data is growing too fast. New data types are added. Processing and managing big data is a challenge in today's era. With traditional methods for big data storage and analysis is less efficient. So, there is difference between analytics of traditional data and big data. The challenges comes with big data are data privacy and security, data storage, creating business value from the large amount of data etc. Data is growing too fast. Following points should be considered [1].

- In 2011 alone, mankind created over 1.2 trillion GB of data.
- Data volumes are expected to grow 50 times by 2020.
- Google receives over 2,000,000 search queries every minute.
- 72 hours of video are added to YouTube every minute.
- There are 217 new mobile Internet users every minute.
- 571 new websites are created every minute of the day.
- According to Twitter's own research in early 2012, it sees roughly 175 million tweets every day, and has more than 465 million accounts.

As the size of big data is growing, compression is must. These large amounts of data need to be compressed. The advantages of compression are [2]:

- Compressed data uses less bandwidth on the network than uncompressed data.
- Compressed data uses less disk space.
- Speed up the data transfer across the network to or from disk.
- Cost is reduced.

II. BIG DATA TECHNOLOGIES



Hadoop is an open source framework for processing, storing, and analyzing massive amounts of distributed unstructured data. Hadoop has two main components. These are:

Figure 1. HDFS architecture

A HDFS: Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is a distributed file system designed to store very large data sets , and to stream those data sets at high bandwidth to user applications.

It cn be easily portable from one platform to another.Figure 1[3] shows HDFS architecture.

This architecture contains Namenodes and Datanodes. HDFS has a master/slave architecture.[3] HDFS consists of single NameNode. There are number of DataNodes which manage storage attached to the nodes that they run on. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

B MapReduce

Hadoop Map/Reduce is a software framework which process vast amounts of data. The Map/Reduce framework consists of a single master JobTracker and one slave TaskTracker per cluster-node. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master. Following figure[13] shows example of Hadoop MapReduce.

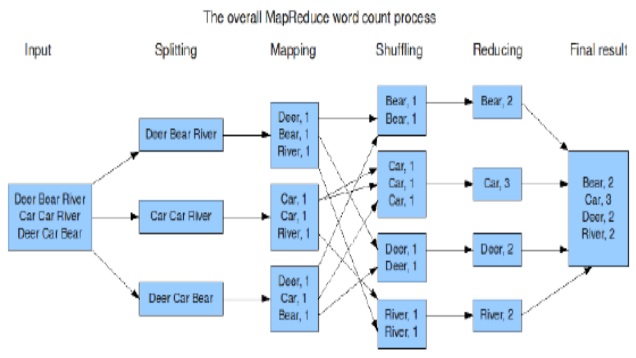


Figure 2. Example of Hadoop MapReduce

It can be seen that [13]the Hadoop system captures datasets from different sources and then performs functions such as storing, cleansing, distributing, indexing, transforming, searching, accessing, analyzing, and visualizing. So, semi-structured and unstructured data are converted into structured data.

III. COMPRESSION TYPES IN HADOOP

Compression will the reduce I/O and decrease network usage. Compressed data uses less bandwidth on the network than uncompressed data. With compression more data can be saved in less space. Big data contains complex and unstructured data. So compression of this data is important. Codec represents implementation of compression and decompression algorithm. Some compression formats are splittable**.** Performance is better for large files if the algorithm is splittable**.** Common compression algorithms supported by hadoop are listed as,

- LZO,
- Gzip
- Bzip2
- LZ4, and
- Snappy.

A. LZO

The LZO compression format is composed of many smaller blocks of compressed data allowing jobs to be split along block boundaries. Block size should be same for compression and decompression. This is fast and splittable. LZO is a lossless data compression library written in ANSI C.LZO has good speed. Its source code and the compressed data format are designed to be portable across platforms. Decompression is very fast. Characteristics of LZO are [4, 11]:

- Data compression is similar to other popular compression techniques, such as gzip and bzip.
- It enables very fast decompression.
- It requires no additional memory for decompression except for source buffers and destination buffers.
- Includes compression levels for generating pre-compressed data which achieve a quite competitive compression ratio.
- There is also a compression level which needs only 8 kB for compression.
- Algorithm is thread safe.
- Algorithm is lossless.
- LZO is portable.

Lzop is a file compressor which uses LZO for compression services. Lzop is the fastest compressor and decompressor.

B GZIP

It is GNU zip. gzip is naturally supported by Hadoop. gzip is based on the DEFLATE algorithm, which is a combination of LZ77 and Huffman Coding. GZIP will generally compress better than LZO though slower. Java will use java's GZIP unless the native Hadoop libs are available on the CLASSPATH; in this case it will use native compressors instead. [5]. gzip compression works by finding similar strings within a text file, and replaces those strings temporarily to make the overall file size smaller. The second occurrence of a string is replaced by a pointer to the previous string, in the form of a pair (distance, length). Literals or match lengths are compressed with one Huffman tree, and match distances are compressed with another tree. The trees are stored in a compact form at the start of each block. Deflate is compression algorithm and inflate is decompression algorithm. Gzip files are stored with .gz extension. The *gzip* sources, written in C, are available here in various formats. [6]

- tar
- shar
- zip
- tar.gz
- tar.z

C.*Bzip2*

bzip2 [7, 8] is a freely available high-quality data compressor. It typically compresses files to within 10% to 15% of the best available techniques. Bzip2 compresses data in blocks of size between 100 and 900 kB. Bzip2 performance is asymmetric, as decompression is relatively fast. The current version is 1.0.6. It supports (limited) recovery from media errors. If you are trying to restore compressed data from a backup tape or disk, and that data contains some errors, bzip2 may still be able to decompress those parts of the file which are undamaged. It's very portable. It should run on any 32 or 64-bit machine with an ANSI C compiler. bzip2 compresses large files in blocks. The block size affects both the compression ratio achieved, and the amount of memory needed for compression and decompression. The header of bzip2 data starts with the letter “BZ”.

D LZ4

LZ4 is a lossless data compression algorithm that is focused on compression and decompression speed. It provides compression speed at 400 MB/s per core fast decoder, with speed in multiple GB/s per core [10].

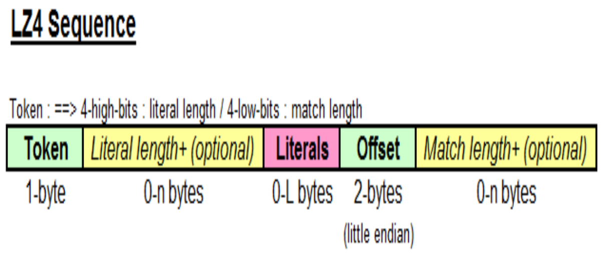


Figure 3. LZ4 sequence

Fig.2 [10] explains working of LZ4 algorithm. It has sequences. The token is a one byte value. The field is literal length. If the value of this field is 0, then there are no literals. If the value is 15, more bytes should be added. Each additional byte then represents a value of 0 to 255, which is added to the previous value to produce a total length. The next field is literals which are uncompressed literals. Offset is the next field. It represents the position of the match to be copied from. 1 means "current position - 1 byte. Maximum value of this field is 65,535. Next field is Match length. Second token field is used with values from 0 to 15. There is a baselength to apply, which is the minimum length of a match, called *minmatch*. This minimum is 4. If the value is 0, there is match length of 4 bytes. If the value is 15 means a match length of 19+ bytes. On reaching the highest possible value (15), we output additional bytes, one at a time, with values ranging from 0 to 255. They are added to total to provide the final matchlength. With the offset and the matchlength, the decoder can now proceed to copy the repetitive data from the already decoded buffer. By decoding the *matchlength*, we reach the end of the sequence, and start another one.

E.Snappy

Hadoop-Snappy[9] is a project for Hadoop that provide access to the snappy compression. Snappy is written in C++. Focus of snappy is on very high speeds and reasonable compression. Requirements to build snappy are gcc c++, autoconf, automake, libtool, Java 6, JAVA_HOME set, Maven3 [9].

IV. SUMMARY OF HADOOP COMPRESSION SCHEMES

There is large amount of data and it is growing day by day. Also properties of unstructured and semi structured data are not similar. Compression reduces this large volume of data. So, it will reduce the storage space. Obviously, it is beneficial. A compression format is commonly referred to as a *codec*, which is short for coder-decoder. The various types of compression algorithms are discussed in this paper. Summary [7] of these techniques can be given as follows. Gzip is general purpose compressor. Bzip compression is better than gzip but it is slower. The LZO compression format is composed of many smaller (~256K) blocks of compressed data, allowing jobs to be split along block boundaries [12]. Speed is good for lzo and snappy but compression is less effective. Decompresses about twice as fast as gzip.It doesn’t compress quite as well as gzip — expect files that are on the order of 50% larger than their gzipped version .Snappy is good for decompression than LZO. Table 1[7] shows summary of these compression algorithms.

TABLE 1: SUMMARY OF HADOOP COMPRESSION FORMATS

Compression format	**Tool**	**Algorithm**	**File extention**	**Splittable**
Gzip	*gzip*	DEFLATE	.gz	No
bzip2	*bizp2*	bzip2	.bz2	Yes
LZO	*lzop*	LZO	.lzo	Yes if indexed
Snappy	N/A	Snappy	.snappy	No

Following table [12] shows a typical example, starting with an 8.0 GB file containing some text-based log data:

TABLE 2: COMPARISON OF DIFFERENT COMPRESSION FORMAT

Compre -ssion	**File**	**Size (GB)**	**Compressin Time (s)**	**Decompressin Time (s)**
None	some_logs	8.0	-	-
Gzip	some_logs.gz	1.3	241	72
LZO	some_logs.lzo	2.0	55	35

V. CONCLUSION

We are in the era of big data. There are various challenges and issues regarding big data. Large amount of data is generated from the various sources either in structured, semi structured or unstructured form. Such data are scattered across the Internet. Hadoop supports various types of compression and compression formats. Different types of compression algorithm are discussed in this paper. These algorithms are summarized. Finally, algorithm comparison is mentioned here.

REFERENCES

[1] "Introduction to Big Data: Infrastructure and Networking Considerations",Juniper Networks ,White Paper

[2] http://doc.mapr.com/display/MapR/Compression

[3] Dhruba Borthakur ,The Hadoop Distributed File System: Architecture and Design, pp.4-5

[4] http://porky.linuxjournal.com:8080/LJ/220/11186.html

[5] http://hbase.apache.org/book/gzip.compression.html

[6] http://www.gzip.org/

[7] http://comphadoop.weebly.com/

[8] http://www.bzip.org/

[9] http://code.google.com/p/snappy/

[10] http://code.google.com/p/lz4/

[11] http://www.oberhumer.com/opensource/lzo/

[12] http://blog.cloudera.com/blog/2009/11/hadoop-t-twitter-part-1- Splittable-lzo-compression/

[13] Jean Yan," Big Data, Bigger Opportunities ", White Paper, April 9, 2013