File Optimization Project

Aneesh Reddy
Correspondence: Managing Partner Mark Kerzner [Houston, TX] & Partner Sujee Maniyam [San Jose, CA]

Background:
Files are central to the modern world. Whether it be a word document, a text file, or a columnar binary storage unit, files run the way we organize data. But to understand files, it is necessary to interpret data analytics. Data Analytics require data formatting, specifically the Extract, Transform, and Load (ETL) process. ETL can either be memory-intensive or process-intensive. However, Companies want no reduction in processing speed. Thus, they usually decide a universal file format for all their data storage. But not all companies can spend thousands of dollars for IT consultants that "devise" the proper file format.
Keywords: ETL process, data analytics, memory-intensive, process-intensive

Topic Introduction:
To construct a file format optimizer using HDFS - Hadoop Distributed File System- so corporations can minimize their ETL processing time as well as potentially reduce byte-allocation. This would also provide research data on what file formats are more suitable for specific ETL tasks.

Methodology:
The first step would be to design HDFS on an Amazon EC2 instance. After HDFS is instantiated, their needs to be a way to safely shutdown and startup the instance without damaging the distributed system. File systems are unique in that they don't store data based on the actual information within the data. A traditional database would store data with relations (tables in a sense). But, HDFS stores data in small chunks (typically 128 MB) spread out over several data nodes. Constructing an HDFS first requires creating the datanodes and the subsequent Name node.

The next step involves collecting large datasets of traditional file formats: XML, JSON, CSV, TXT Logs, Binary XML (Solr), Avro, Parquet, etc. File sizes of ~5-10 GB are necessary for the ETL-process optimization to be worthwhile.

Once data formats are collected, a typical MapReduce program must be run to perform ETL in HDFS. The times will be noted for each format, and averaged for same file formats of different data. Along with these ETL process times, the ETL times for converting between file formats will be noted. Finally, a file format converter used in a previous project will be applied to help optimize the data format. It will take the original file, and "trial run" ETL on a small subset of the data (about 10% of the original file). Using the previously determined time values, the file format converter will result in the preferred output that a company *should* use as its universal file format.

References:

1. Matthew, Sajee (2014). "Overview of Amazon Web Services", https://d0.awsstatic.com/whitepapers/aws-overview.pdf

2. Cloudera, (2015). "CDH5 Installation Guide", http://www.cloudera.com/content/www/en-us/documentation/cdh/5-0-x/CDH5-Installation-Guide/CDH5-Installation-Guide.html

3. Schmid, Chris (2011). "balesio Native Format Optimization Technology (NFO)", http://www.balesio.com/pdf/whitePapers/eng/balesio-native-format-optimization-technology.pdf