New Approaches to Big Data Processing and Analytics

Contributing authors: David Floyer, David Vellante
Original publication date: February 12, 2013

There are number of approaches to processing and analyzing Big Data, but most have some common characteristics. Namely, they take advantage of commodity hardware to enable scale-out, parallel processing techniques; employ non-relational data storage capabilities in order to process unstructured and semi-structured data; and apply advanced analytics and data visualization technology to Big Data to convey insights to end-users.

Wikibon has identified three Big Data approaches that it believes will transform the business analytics and data management markets.

Hadoop

Hadoop is an open source framework for processing, storing and analyzing massive amounts of distributed, unstructured data. Originally created by Doug Cutting at Yahoo!, Hadoop was inspired by MapReduce, a user-defined function developed by Google in early 2000s for indexing the Web. It was designed to handle petabytes and exabytes of data distributed over multiple nodes in parallel. Hadoop clusters run on inexpensive commodity hardware so projects can scale-out without breaking the bank. Hadoop is now a project of the Apache Software Foundation, where hundreds of contributors continuously improve the core technology. ***Fundamental concept: Rather than banging away at one, huge block of data with a single machine, Hadoop breaks up Big Data into multiple parts so each part can be processed and analyzed at the same time.***

How Hadoop Works

A client accesses unstructured and semi-structured data from sources including log files, social media feeds and internal data stores. It breaks the data up into "parts," which are then loaded into a file system made up of multiple nodes running on commodity hardware. The default file store in Hadoop is the Hadoop Distributed File System, or HDFS. **File systems such as HDFS are adept at storing large volumes of unstructured and sem-structured data as they do not require data to be organized into relational rows and columns.**

Each "part" is replicated multiple times and loaded into the file system so that if a node fails, another node has a copy of the data contained on the failed node. A Name Node acts as facilitator, communicating back to the client information such as which nodes are available, where in the cluster certain data resides, and which nodes have failed.

Once the data is loaded into the cluster, it is ready to be analyzed via the MapReduce framework. The client submits a "Map" job -- usually a query written in Java – to one of the nodes in the cluster known as the Job Tracker. The Job Tracker refers to the Name Node to determine which data it needs to access to complete the job and where in the cluster that data is located. Once determined, the Job Tracker submits the query to the relevant nodes. **Rather than bringing all the data back into a central location for processing, processing then occurs at each node simultaneously, or in parallel. This is an essential characteristic of Hadoop.**

When the each node has finished processing its given job, it stores the results. The client initiates a "Reduce" job through the Job Tracker in which results of the map phase stored locally on individual nodes are aggregated to determine the "answer" to the original query, then loaded on to another node in the cluster. The client accesses these results, which can then be loaded into one of number of analytic environments for analysis. The **MapReduce** job has now been completed.

Once the MapReduce phase is complete, the processed data is ready for further analysis by Data Scientists and others with advanced data analytics skills. Data Scientists can manipulate and analyze the data using any of a number of tools for any number of uses, including to search for hidden insights and patterns or to use as the foundation to build user-facing analytic applications. The data can also be modeled and transferred from Hadoop clusters into existing relational databases, data warehouses and other traditional IT systems for further analysis and/or to support transactional processing.

Hadoop Technical Components

A Hadoop "stack" is made up of a number of components. They include:

- Hadoop Distributed File System (HDFS): The default storage layer in any given Hadoop cluster;
- Name Node: The node in a Hadoop cluster that provides the client information on where in the cluster particular data is stored and if any nodes fail;
- Secondary Node: A backup to the Name Node, it periodically replicates and stores data from the Name Node should it fail;
- Job Tracker: The node in a Hadoop cluster that initiates and coordinates MapReduce jobs, or the processing of the data.

- Slave Nodes: The grunts of any Hadoop cluster, slave nodes store data and take direction to process it from the Job Tracker.

In addition to the above, the Hadoop ecosystem is made up of a number of complimentary sub-projects. NoSQL data stores like Cassandra and HBase are also used to store the results of MapReduce jobs in Hadoop. In addition to Java, some MapReduce jobs and other Hadoop functions are written in Pig, an open source language designed specifically for Hadoop. Hive is an open source data warehouse originally developed by Facebook that allows for analytic modeling within Hadoop.

Following is a guide to Hadoop's components:

Hadoop Distributed File System: HDFS, the storage layer of Hadoop, is a distributed, scalable, Java-based file system adept at storing large volumes of unstructured data.

MapReduce: MapReduce is a software framework that serves as the compute layer of Hadoop. MapReduce jobs are divided into two (obviously named) parts. The “Map” function divides a query into multiple parts and processes data at the node level. The “Reduce” function aggregates the results of the “Map” function to determine the “answer” to the query.

Hive: Hive is a Hadoop-based data warehousing-like framework originally developed by Facebook. It allows users to write queries in a SQL-like language caled HiveQL, which are then converted to MapReduce. This allows SQL programmers with no MapReduce experience to use the warehouse and makes it easier to integrate with business intelligence and visualization tools such as Microstrategy, Tableau, Revolutions Analytics, etc.

Pig: Pig Latin is a Hadoop-based language developed by Yahoo. It is relatively easy to learn and is adept at very deep, very long data pipelines (a limitation of SQL.)

HBase: HBase is a non-relational database that allows for low-latency, quick lookups in Hadoop. It adds transactional capabilities to Hadoop, allowing users to conduct updates, inserts and deletes. EBay and Facebook use HBase heavily.

Flume: Flume is a framework for populating Hadoop with data. Agents are populated throughout ones IT infrastructure – inside web servers, application servers and mobile devices, for example – to collect data and integrate it into Hadoop.

Oozie: Oozie is a workflow processing system that lets users define a series of jobs written in multiple languages – such as Map Reduce, Pig and Hive -- then intelligently link them to one another. Oozie allows users to specify, for example, that a particular query is only to be initiated after specified previous jobs on which it relies for data are completed.

Flume: Flume is a framework for populating Hadoop with data. Agents are populated throughout ones IT infrastructure – inside web servers, application servers and mobile devices, for example – to collect data and integrate it into Hadoop.

Ambari: Ambari is a web-based set of tools for deploying, administering and monitoring Apache Hadoop clusters. It's development is being led by engineers from Hortonworoks, which include Ambari in its Hortonworks Data Platform.

Avro: Avro is a data serialization system that allows for encoding the schema of Hadoop files. It is adept at parsing data and performing removed procedure calls.

Mahout: Mahout is a data mining library. It takes the most popular data mining algorithms for performing clustering, regression testing and statistical modeling and implements them using the Map Reduce model.

Sqoop: Sqoop is a connectivity tool for moving data from non-Hadoop data stores – such as relational databases and data warehouses – into Hadoop. It allows users to specify the target location inside of Hadoop and instruct Sqoop to move data from Oracle, Teradata or other relational databases to the target.

HCatalog: HCatalog is a centralized metadata management and sharing service for Apache Hadoop. It allows for a unified view of all data in Hadoop clusters and allows diverse tools, including Pig and Hive, to process any data elements without needing to know physically where in the cluster the data is stored.

BigTop: BigTop is an effort to create a more formal process or framework for packaging and interoperability testing of Hadoop's sub-projects and related components with the goal improving the Hadoop platform as a whole.

Hadoop: The Pros and Cons

The main benefit of Hadoop is that it allows enterprises to process and analyze large volumes of unstructured and semi-structured data, heretofore inaccessible to them, in a cost- and time-effective manner. Because Hadoop clusters can scale to petabytes and even exabytes of data, enterprises no longer must rely on sample data sets but can process and analyze ALL relevant data. Data Scientists can apply an iterative approach to analysis, continually refining and testing queries to uncover previously unknown insights. It is also inexpensive to get started with Hadoop. Developers can download the Apache Hadoop distribution for free and begin experimenting with Hadoop in less than a day.

The downside to Hadoop and its myriad components is that they are immature and still developing. As with any young, raw technology, implementing and managing Hadoop clusters and performing advanced analytics on large volumes of unstructured data requires significant expertise, skill and training. Unfortunately, there is currently a dearth of Hadoop developers and Data Scientists available, making it impracticale for many enterprises to maintain and take advantage of complex Hadoop clusters. Further, as Hadoop's myriad components are improved upon by the community and new components are created, there is, as with any immature open source technology/approach, a risk of forking. Finally, Hadoop is a batch-oriented framework, meaning it does not support real-time data processing and analysis.

The good news is that some of the brightest minds in IT are contributing to the Apache Hadoop project, and a new generation of Hadoop developers and Data Scientists are coming of age. As a result, the technology is advancing rapidly, becoming both more powerful and easier to implement and manage. An ecosystems of vendors, both Hadoop-focused start-ups like Cloudera and Hortonworks and well-worn IT stalwarts like IBM and Microsoft, are working to offer commercial, enterprise-ready Hadoop distributions, tools and services to make deploying and managing the technology a practical reality for the traditional enterprise. Other bleeding-edge start-ups are working to perfect NoSQL (Not Just SQL) data stores capable of delivering near real-time insights in conjunction with Hadoop.

NoSQL

NoSQL

A related new style of database called NoSQL (Not Only SQL) has emerged to, like Hadoop, process large volumes of multi-structured data. However, where as Hadoop is adept at supporting large-scale, batch-style historical analysis, NoSQL databases are aimed, for the most part (though there are some important exceptions) at serving up discrete data stored among large volumes of multi-structured data to end-user and automated Big Data applications. This capability is sorely lacking from relational database technology, which simply can't maintain needed application performance levels at Big Data scale.

In some cases, NoSQL and Hadoop work in conjunction. The aforementioned HBase, for example, is a popular NoSQL database modeled after Google BigTable that is often deployed on top of HDFS, the Hadoop Distributed File System, to provide low-latency, quick lookups in Hadoop.

The downside of most NoSQL databases today is that they traded ACID (atomicity, consistency, isolation, durability) compliance for performance and scalability. Many also lack mature management and monitoring tools. Both these shortcomings are in the process of being overcome by both the open source NoSQL communities and a handful of vendors -- such as DataStax, Sqrrl, 10gen, Aerospike and Couchbase -- that are attempting to commercialize the various NoSQL databases.

NoSQL databases currently available include:

- HBase
- Cassandra
- Aerospike
- MongoDB
- Accumulo
- Riak
- CouchDB
- DynamoDB

Next Generation Data Warehousing

Unlike traditional data warehouses, Next Generation Data Warehouses are capable of quickly ingesting large amounts of mainly structured data wit minimal data modeling required and can scale-out to accommodate multiple terabytes and sometimes petabytes of data. Most importantly for end-users, Next Generation Data Warehouses support near real-time results to complex SQL queries, a notable missing capability in Hadoop, and in some cases the ability to support near real-time Big Data applications. The fundamental characteristics of a Next Generation Data Warehouse include:

Massively parallel processing, or MPP, capabilities: Next Generation Data Warehouses employ massively parallel processing, or MPP, that allow for the ingest, processing and querying of data on multiple machines simultaneously. The result is significantly faster performance than traditional data warehouses that run on a single, large box and are constrained by a single choke point for data ingest.

Shared-nothing architectures: A shared-nothing architecture ensures there is no single point of failure in Next Generation Data Warehousing environments. Each node operates independently of the others so if one machine fails, the others keep running. This is particularly important in MPP environments, in which, with sometimes hundreds of machines processing data in parallel, the occasional failure of one or more machines is inevitable.

Columnar architectures: Rather than storing and processing data in rows, as is typical with most relational databases, most Next Generation Data Warehouses employ columnar architectures. In columnar environments, only columns that contain the necessary data to determine the "answer" to a given query are processed, rather than entire rows of data, resulting in split-second query results. This also means data does not need to be structured into neat tables as with traditional relational databases.

Advanced data compression capabilities: Advanced data compression capabilities allow Next Generation Data Warehouses to ingest and store larger volumes of data than otherwise possible and to do so with significantly fewer hardware resources than traditional databases. A warehouse with 10-to-1 compression capabilities, for example, can compress 10 terabytes of data down to 1 terabyte. Data compression, and a related technique called data encoding, are critical to scaling to massive volumes of data efficiently.

Commodity hardware: Like Hadoop clusters, most Next Generation Data Warehouses run on off-the-shelf commodity hardware (there are some exceptions to this rule, however) from Dell, IBM and others, so they can scale-out in a cost effective manner.

In-memory data processing: Some, but certainly not all, Next Generation Data Warehouses use Dynamic RAM or Flash for some real-time data processing. Some, such SAP HANA and Aerospike, are fully in-memory, while others use a hybrid approach that blends less expensive but lower performing disk-based storage for "colder" data with DRAM or flash for "hotter" data.

Next Generation Data Warehouses do have some blind spots, however. Most notably, Next Generation Data Warehouses are are not designed to ingest, process and analyze semi-structured and unstructured data that are largely responsible for the explosion if data volumes in the Big Data Era. In order to make use of the totality of an enterprise's data assets, therefore, a combination of Hadoop/NoSQL and Next Generation Data Warehousing is often required.

Complimentary Big Data Approaches

Hadoop, NoSQL and Next Generation Data Warehouses are not mutually exclusive. Far from it, Wikibon believes the three approaches are complimentary to each other and can and should co-exist in many enterprises. Hadoop excels at processing and analyzing large volumes of distributed, unstructured data in batch fashion for historical analysis. NoSQL databases are adept at storing and serving up multi-structured data in near-real time for web-based Big Data applications. And Next Generation Data Warehouses are best at providing near real-time analysis of large volumes of mainly structured data.

Analysis done in Hadoop can be ported into Next Generation Data Warehouses for further analysis and/or integration with structured data, for example. Insights gleaned from Big Data Analytics can (and should) be productionized via Big Data applications. Enterprises should aim for flexible Big Data architectures to enable the these three technologies/approaches to share data and insights as seamlessly as possible. There are a number of pre-built connectors to help Hadoop developers and administrators perform such data integration, while a handful of vendors -- among them EMC Greenplum and Teradata Aster -- offer Big Data appliances that bundle Hadoop, NoSQL and Next Generation Data Warehousing with preconfigured hardware for quick deployment with minimal tuning required. Others, namely a start-up called Hadapt, offers a single platform that provides both SQL and Hadoop/MapReduce processing on the same cluster.

In order to fully take advantage of Big Data, however, enterprises must take further steps. Namely, they must employ advanced analytics techniques on the processed data to reveal meaningful insights. Data Scientists perform this sophisticated work in one of a handful of languages or approaches, including SAS and R. The results of this analysis can then be operationalized via Big Data applications, either homegrown or off-the-shelf. Other vendors are developing business intelligence-style applications to allow non-power users to interact with Big Data directly.