ARCHITECTURE FOR BUILDING THE SYSTEMS OF STORAGE AND ANALYSIS OF BIG DATA

Shukurov Shokhrukh Tolibjonovich, Ibragimov Ulugbek Muradilloyevich

Bukhara engineering-technological institute

Keywords: big data, Hadoop, Apache, database, NoSQL, open source, visualization, analytics.

Abstract

In this article, we will look at common architectures for building clusters for analytical processing of big data. List of technologies and their application in each node of the cluster.

At present, there are a huge number of architectural solutions and tools used for big data. The most common basis for building big data systems is the Hadoop ecosystem, a project of the Apache Software Foundation, which is a set of freely distributed utilities, libraries and frameworks for developing distributed programs and running them on clusters [1]. Distributed file systems are used to store big data, which have the following differences:

- the ability to store files that are larger than the size of a single drive on the storage server;
- stored files are automatically replicated to storage servers, allowing parallel processing and creating redundancy to ensure reliable cluster operation;
- system can be easily scaled using the principle of horizontal scaling.

The most common distributed file system is the Hadoop File System. Other examples of DFS are Red Hat ClusterFS, QuantCast File System, Ceph File System[2,5].

Big data storage requires database systems and data access facilities. Due to known limitations, the use of classic relational databases such as Oracle SQL, DB2 is not possible. In systems for analyzing and storing big data, systems called NoSQL and their further development NewSQL are used. These database systems will be discussed in more detail below. Here we note that in addition to the database systems themselves, systems for collecting, converting batch and streaming data are needed.

Data integration tools, as the name suggests, are used to merge data by moving data from one source to another. As noted above, there are two main classes of big data solutions for processing batch and streaming data. Examples of batch processing solutions are Apache Sqoop. Examples of a solution for use in streaming data processing are the eccentricity of the task has led to the emergence of technologies such as Apache Flume, Apache Storm and Apache Kafka[3].

After moving the data to a distributed file system, it is needed to move on to extracting information from the data. For these purposes, in the process of analyzing big data, methods of applied mathematics, mathematical statistics, and machine learning are used. However, an important difference between the algorithms used in big data analysis is that the algorithms must be distributed. To date, many libraries and frameworks implement these algorithms. Examples of programming languages actively used in this task are Python, Java, R. For example, for Python there are libraries such as:

- NLTK – Natural Language Toolkit is natural language data processing library; Scikit-learn is one of the most famous machine learning libraries;
- TenzorFlow – deep machine learning library from Google. Apache Spark is an example of a real-time machine learning system.

Distributed programming frameworks simplify the implementation of distributed algorithms because they implement “low-level distributed” tasks while hiding them from the programmer. These tasks include redistribution of tasks in case of their failure on the computing node, communication between subprocesses. Examples of distributed programming frameworks are Apache Thrift, Zookeeper [8].

Scheduling tools are used to automate repetitive tasks. For example, running MapReduce tasks when a new dataset is available. Representatives of this group are Hadoop YARN. Benchmarking tools are used to optimize big data infrastructures by using standardized profiles. Each profile is based on a specific set of tools for storing and processing big data. Any information system for storing and analyzing big data should be based on a specific hardware and software architecture. There is a generalized architectural scheme for big data applications that defines the big data tech stack [2].

The load layer loads the final relevant information without noise into the Hadoop distributed storage layer. Algorithms at this level must validate, clean, transform, reduce, and integrate data into the big data technology stack for further processing.

Data source level		
Identification	Filtration	Validation
Noise reduction	Transformation	Compression
	Integration	
Data source level		
NoSQL	HDFS	

Fig. 1. The role of the data load layer in the architecture of applications of big data storage and analysis

Load layer architectural patterns describe solutions to common data source problems in terms of impact on the load layer. These solutions can be selected based on performance, scalability and availability requirements.

One of the levels is the Monitoring Layer. Monitoring systems are used to track the status of distributed clusters and collect information about the operating systems, hardware, etc. used. To accomplish this task, machines must communicate with the monitoring tool through high-level protocols such as XML instead of machine-specific binary formats. Monitoring systems must also provide tools for data storage and visualization. Open source tools such as Ganglia and Nagios are widely used to monitor big data stacks. Applications of the analytical level (Analytics Engine) are used to perform search queries on distributed data, perform analytical text processing, statistical analytics, and perform intelligent algorithms on data.

Visualization tools (Analytics Engine). As a rule, “raw” analytical output cannot be used to solve business problems. It is necessary to translate analytical data into tabular or graphical form, as well as the ability to look at the data from different angles. Therefore, visualization tools are an integral part of big data storage and analysis systems. Visualization tools work on top of consolidated and aggregated outputs. In the case when it is required to study the results of analytics in real time, real-time mechanisms working on the mechanisms of complex event processing (CEP) and event-driven architectures (EDA) can be used.

Visualization tools			
BI tools		Analytical tools of big data	
Operating XD	Data warehouses	Data Scoop	Data lakes
Regulating DBMS		NoSQL DBMS	
Structured data		Partially structured and unstructured data	

Fig. 2. Conceptual architecture of the visualization layer

As the performed review of the architectures and tools of systems of big data storage and analysis shows, there is a huge set of possible construction options. The number of possible solutions can be very large, if we take into account the developed lists of tools (Landscapes) for creating storage systems and analyzing big data. An example is an Internet resource dedicated to the study of solutions in the field of big data [4]. The main purpose of Big Data applications is to help companies make more informed business decisions by analyzing large amounts of data. This data may include web server logs, internet clickstream data, social media content and activity reports, customer email texts, mobile phone call details, and technical data captured by IoT sensors. As noted above, these can be structured, partially structured, and unstructured [6,7, 9].

Conclusion

Systems of big data storage and analysis are built according to two architectural types: systems for batch and/or stream processing of big data.

There are a large number of tools used to create systems for storing and analyzing big data and form a big data ecosystem, which includes groups of tools: distributed file systems, deployment tools, NoSQL databases, etc.

There are four types of NoSQL databases: column, document, key-value, and graph. Each of the types of NoSQL databases has its own advantages and the best applicability for solving a certain range of tasks. Deploying systems for storing and analyzing big data is a complex technical and engineering challenge. To facilitate the deployment of storage systems and analysis of big data, special distributions can be used, made in the form of virtual machines or cloud services.

References:

1. Hadoop. [Electronic resource] - Access mode: https://ru.wikipedia.org/wiki/Hadoop (accessed 5.27.2019).

2. Sawant N., Shah H. Big Data Application Architecture //Big data Application Architecture Q & A. – Apress, Berkeley, CA, 2013. – pp. 9-28.

3. Cielen D., Meysman A., Ali M. Introducing data science: big data, machine learning, and more, using Python tools. – Manning Publications Co., 2016.

4. Big Data Analytics Landscape 2019. [Electronic resource] - Access mode: https://www.learnbigdatatools.com/big-data-analytics-landscape-2019 (accessed 27.05.2019).

5. HDFS shell [Electronic resource] - Access mode: https://github.com/avast/hdfs-shell (accessed 27.05.2019).

6. IMF DATA. [Electronic resource] - Access mode: https://www.imf.org/en/Data (accessed 27.05.2019).

7. Lambda architecture. From Wikipedia, the free encyclopedia. [Electronic resource] - Access mode: https://en.wikipedia.org/wiki/Lambda_architecture (accessed 27.05.2019).

8. Marr Bernard. Big Data And AI: 30 Amazing (And Free) Public Data Sources For. / Bernard Marr 2018 [Electronic resource] - Access mode: https://www.forbes.com/sites/bernardmarr/2018/02/26/big-data-and-ai-30-amazing-and-free-public-data-sources-for-2018/#57d218e05f8a (accessed 27.05.2019).

9. Muradilloyevich, U. I. (2019). use of computer modeling in the process of teaching the general professional and special disciplines in higher educational institutions. European Journal of Research and Reflection in Educational Sciences Vol, 7(12).