 doi:10.20944/preprints201904.0281.v1

Choice of Cluster Computing System Hadoop and Apache Spark for Network Systems

Vasiliy S. Elagin, Vladislav I. Karpov, Aleksandr S. Kravchenko, Aleksandr B. Goldstein, Andrei G. Vladyko
The Bonch-Bruevich Saint-Petersburg State University of Telecommunications,
Saint-Petersburg, 193232, Russia; elagin.vas@gmail.com, dincrash@yandex.ru , first.alsekr@gmail.com, agold@niits.ru , vladyko@sut.ru
* Correspondence: vladyko@sut.ru; Tel: +7-921-915-0920

Abstract: The article provides detailed information about the new technologies based on cluster computing Hadoop and Apache Spark. The experimental task of processing logistic regression with the help of these technologies is considered. The findings on the comparison of the performance of cluster computing of Hadoop and Apache Spark are revealed and substantiated.

Keywords: Cluster computing, Big Data, Spark, Hadoop.

1. Introduction

Today, medium and large enterprises have large costs in connection with the use, storage, and transmission of a multitude of information. To solve this problem, new technologies, such as Hadoop (MapReduce) and Spark, are needed to speed up and shorten the information processing time. MapReduce and its variants were very successful in applying them in large applications, using a large amount of data on clusters. [4]

2. Computing cluster

To date, the cluster computing model has become widespread, it runs parallel computing of data on interconnected computers that automatically provide task scheduling, fault tolerance and load balancing. Cluster computing has become a hot topic of research among academic and industrial communities, including system developers, network developers, and algorithm developers. The use of clusters as a computing platform is not limited only to scientific and engineering applications;

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

There are many business applications that can benefit from the use of clusters. Research and development is carried out in many areas, but there are several that are of particular interest. The first is, without a doubt, a network. Fig. 1. Shows the structure of Clusters are based on communication between nodes, and the creation of networks with fast and low latency is a prerequisite for clusters to become the configuration of the future.[2],[5]

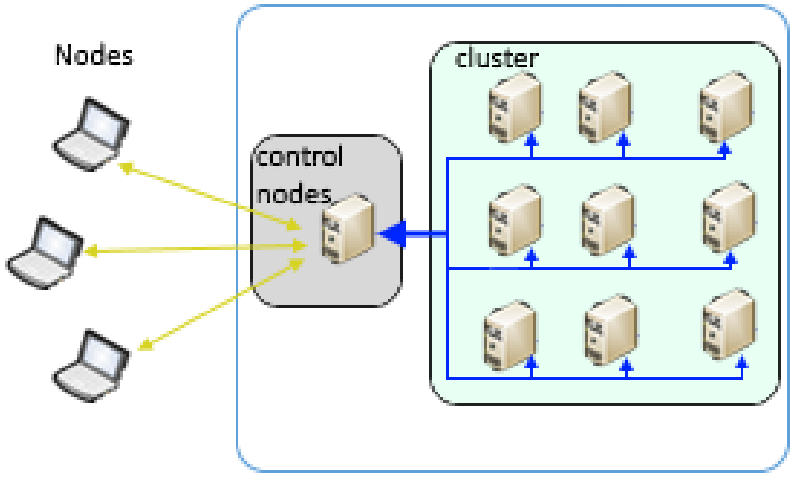

Fig. 1. Structure of Cluster Computing.

Apache has developed Apache Hadoop to create robust, scalable, and distributed computing.

3. Hadoop is a new cluster computing model

Hadoop is an open source software environment for storing and processing large data distributed across multiple storage nodes in a Hadoop cluster. In fact, it performs two tasks: data storage and fast processing.

The Hadoop kernel consists of HDFS and a Hadoop MapReduce implementation.

HDFS is a distributed file system on Fig. 2. show block diagram of HDFS cluster. HDFS serves as an external client and looks like a regular file system, with it you can create, delete, move, rename files.[3]

The NameNode serves as the HDFS metadata services, and the DataNode nodes serve as HDFS storage units.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

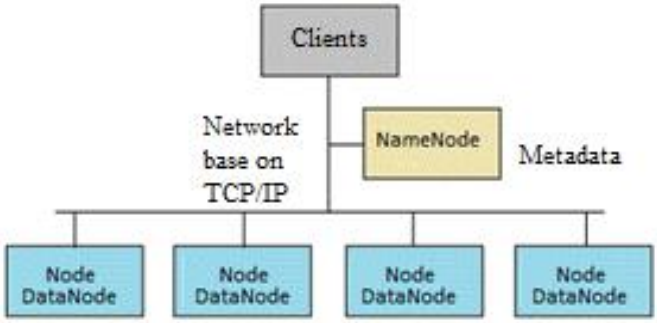

Fig. 2. Block diagram of HDFS cluster.

MapReduce is a processing technology and software model for java-based distributed computing. The MapReduce algorithm Fig. 3. contains two important tasks: map and reduce. A map takes a data set and converts it into a data set, where the individual elements are divided into composition (key / value pairs). Secondly, reduce, which outputs the data from the map as input and combines these pairs of values. As follows from the sequence of the MapReduce name, the reduce task is always executed after setting the map.[6]

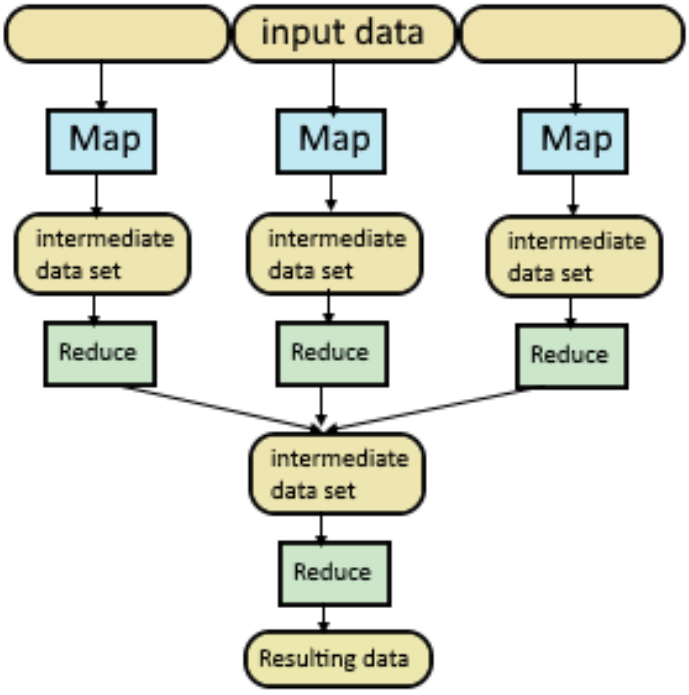

Fig. 3. Block diagram execution MapReduce.

The main advantage of MapReduce is the scaling and processing of data across multiple compute nodes. Within the MapReduce model, data processing

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

primitives are called mapper and reducer. Decomposing a data processing application into a mapper and reducer is sometimes nontrivial. But, as soon as we write an application in the form of MapReduce, application scaling is possible on hundreds, thousands or even tens of thousands of computers in a cluster - it is just a configuration change. This simple scalability is what attracted many programmers to use the MapReduce model.[9]

The structure described above allows you to manage the basic planning system and respond to errors without user intervention. The data flow programming model is useful for most applications, but there are applications that cannot effectively use it (iterative tasks, machine learning algorithms). Each iteration can be expressed as MapReduce work, each task must reload data from the disk, which leads to a significant decrease in performance. Each Hadoop request takes

a) significant latency (tens of seconds) since it works like
b) separate MapReduce task and data read from disk.

4. Spark, as a replacement for Hadoop

To solve these problems and improve performance, a new cluster computing structure called Spark appeared, which at the same time provides the same scalability and fault tolerance, but storing data in RAM, not like Hadoop on the disk.

Due to this, data processing is much faster. Authors and Affiliations

Apache Spark consists of:

a) Core spark
b) Spark SQL
c) Spark streaming
d) MLlib
e) Graphx

The Spark core is the basic engine for large-scale distributed and parallel data processing.[10]

1) The kernel serves to:

a) memory management and recovery after failures

b) scheduling, distributing and tracking tasks on a cluster

c) storage system interactions

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

2) Spark SQL allows you to use data queries using SQL.
3) Spark Streaming allows you to process streaming data in real time.
4) MLlib is a machine learning library that includes various algorithms.
5) GraphX is a library of manipulating graphs and using parallel operations with them.

The main feature in Spark is the flexible distributed data set (RDD) Spark's fundamental data structure. This is a fixed distributed collection of objects. Each data set in RDD is divided into logical partitions that can be computed on different nodes of the cluster. RDDs can contain any type of Python, Java, or Scala object, including custom classes. Formally, RDD is a readable, divided record set. RDDs can be created using deterministic stable storage data or other RDDs. RDD is a fault-tolerant set of elements that can work in parallel. There are two ways to create RDD — parallelizing an existing collection in your driver program, or linking to a data set in an external storage system, such as a shared file system, HDFS, HBase, or any data source that offers Hadoop input format. Spark uses the RDD concept to achieve faster and more efficient MapReduce operations.[1],[2]

Part of the code Fig. 4. is processed on the cluster (Resource Manger),

Fig. 4. Block diagram of RDD flow.

as part of the code on the driver Fig. 5.(Driver Program).

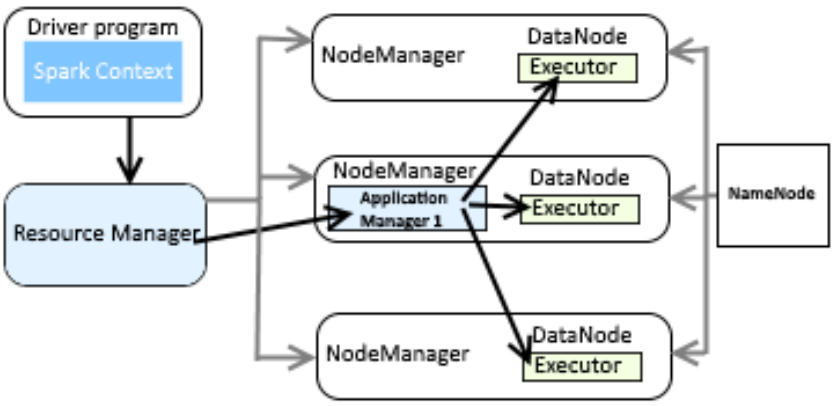

Fig. 5. Structure of Spark cluster.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

In order to show which of the frameworks will be the most efficient, an experiment was conducted comparing Apache Spark and Apache Hadoop.

5. Experimental part

The purpose of the experiment.

As part of a study comparing the performance of Apache Hadoop and Apache Spark, the task of performing a logistic regression on nodes and outputting it as a graph was selected.

Selection of tools used for the experiment.

To solve the problem, it took 2 nodes (Worker Node), a personal computer, the Mesos cluster management system, which will allow allocating CPU and memory resources in a cluster, as well as Apache Spark and Apache Hadoop.

The scheme of the experiment.

First, the driver creates a task in our case, this is a logistic regression, then the task is added to the task scheduler. Mesos determines which machine handles the task. Because Mesos takes into account the structure when planning these short-term tasks, several Frameworks can coexist in the same cluster without resorting to static resource sharing.

6. The result of the experiment

Below on Fig.6. are the results of the experiment, which show clustering of computing power to perform the task - logistic regression work.

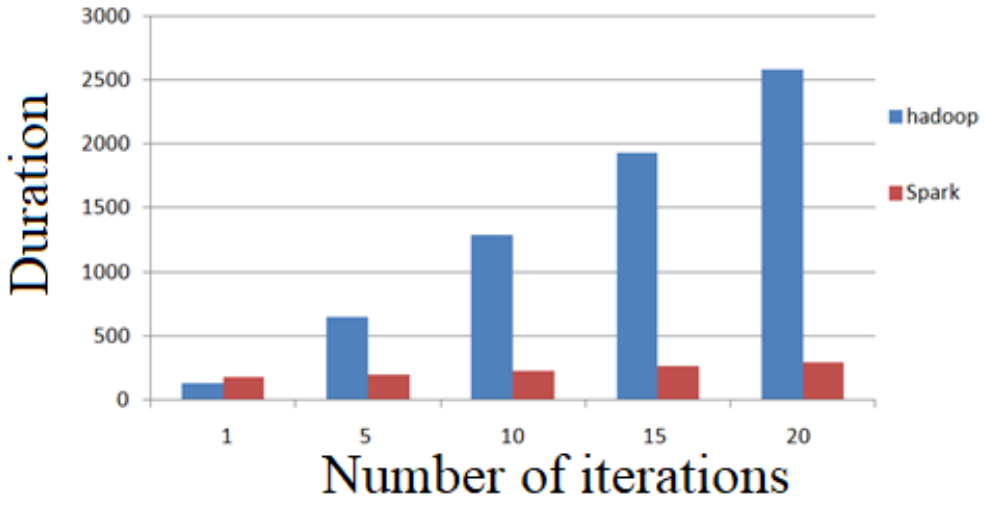

Fig. 6. Logistic regression execution in Spark and Hadoop

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2019 doi:10.20944/preprints201904.0281.v1

From this data it can be concluded that:

In Hadoop, each iteration takes 127 seconds, because it works as an independent MapReduce task. Each such MapReduce task must reload data from the disk, resulting in a significant performance degradation. MapReduce has proven to be very effective for complex batch tasks. In addition, it uses a one-pass computation model.

At the first iteration, Spark takes 174s, but then the iterations take only 6 seconds. Spark can cache RDD in memory to speed up work and re-access these data sets.

At 10 iterations, the duration of Apache Hadoop lasts 1288 seconds, compared with the same Apache Spark 228 seconds, which is almost 6 times faster, it follows that Apache Hadoop is positioned as a platform that is not too adapted for low-latency applications and iterative computing, for example, implementing algorithms on graphs or machine learning. Apache Spark was created for cases when data volumes are relatively small, and high latency is required. If the task is not large enough RAM, Spark will not be able to process very large data.[6]

Also, to improve the results of the experiment, it is necessary to analyze the used programming language of writing DriverProgram. Since the programming language Java 8 was used for the current task, languages such as Scala, Java, and Python were chosen for comparison. Apache Spark was written in the Scala language and some functions are not found in other languages, and the appearance of new features is much faster than in other programming languages.

Java does not support the REPL (Read-Evaluate-Print Loop) shell, and this is very important for every engineer working on the Big Data and Data project. Java 8 brings Lambda expressions and streaming APIs that improve performance a bit, but are not yet comparable to Scala expressions. Scala is a static typed language that helps us catch early errors during compilation, while python is a language with dynamic typing. When the Python API shell calls internal Spark functions written in Scala running on a JVM, translation between two different environments and the selected languages can be the source of more errors, problems, and reduced performance. Almost 70% of Apache Spark applications are currently written in the Scala language.[7]

As a result of the analysis of programming languages, it can be concluded that to improve performance, it is most preferable to use the Scala programming

language, which allows reducing the processing time of the task in each iteration.[11]

7. Conclusions

At the moment, developers of systems with big data face the difficult task of choosing the best product for their applications. The studies in this article lead to the following conclusions:

- Spark is the best choice for logistic regression processing. Results were analyzed why Spark is faster than Hadoop. Hadoop is mainly used for hard disk operations using the MapReduce paradigm, and Spark is more flexible but more expensive, memory processing architecture.
- The preferred programming language is Scala, which has the shortest processing time compared to other languages.

It follows from the above that when deciding on the use of a particular system, it is important to understand the features of each of them. [8]

References

1. Andy Konwinski, Holden Karau, Matei Zaharia, and Patrick Wendell. Learning Spark: Lightning-Fast Big Data Analysis. Trans. With the English, DMK Press 2015
2. Holden Karau and Rachel Warren. High Performance Spark: Best Practices for Scaling and Optimizing Apache Spark. Trans. With the English. Peter,2018
3. Chuck Lam. Hadoop in Action. Trans. With the English, DMK Press 2012
4. Joel Grus. Data Science from Scratch. Trans. With the English, BHV-Petersburg, 2018
5. J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. Commun. ACM, 51(1):107–113, 2008.
6. J. Ekanayake, S. Pallickara, and G. Fox. MapReduce for data intensive scientific analyses. In ESCIENCE ’08, pages 277–284, Wa
7. F. Perez and B. E. Granger. IPython: a system for ´ interactive scientific computing. Comput. Sci. Eng., 9(3):21–29, May 2007
8. J. Cohen, B. Dolan, M. Dunlap, J. Hellerstein, and C. Welton. Mad skills: new analysis practices for big data. VLDB, 2009.
9. A. Thusoo et al. Hive-a petabyte scale data warehouse using hadoop. In ICDE, 2010.
10. M. Zaharia et al. Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling. In EuroSys 10, 2010.
11. G. Malewicz et al. Pregel: a system for large-scale graph processing. In SIGMOD, 2010.