Apache Spark

SNEHAL CHEMBURKAR[1] AND RAHUL RAGHATATE[1]

[1]School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*]Corresponding authors: snehchem@iu.edu, rraghtate@iu.edu

project-001, March 7, 2017

Apache Spark, developed at UC Berkeley AMPLAB, is a high performance framework for analyzing large datasets [1]. The main idea behind the development of Spark was to create a generalized framework that could process diverse and distributed data as opposed to MapReduce which only support batch processing of data. Spark has multiple libraries built on top of its core computational engine which help process diverse data. This paper will discuss the spark runtime architecture, its core and libraries.

Keywords: Spark, RDDs, DAG, Driver, Cluster, Worker, I524

https://github.com/snehalvartak/sp17-i524/paper1/S17-IR-2006/report.pdf

INTRODUCTION

Spark is an open source distributed cluster computing engine for processing the different types of data available these days. The distribution, scheduling and monitoring of clusters is done by the Spark core. The high level components required for processing the diverse workloads such as structured or streaming data are powered by the spark core. "These components are designed to inter-operate closely letting you combine them like libraries in a software project [2]."

The Spark core and the higher level libraries on top of the core are tightly integrated meaning when updates or improvements are implemented in the spark core help improve the spark libraries as well. Tight integration also makes it easier to write applications combining different workloads. This is explained nicely in the following example. One can build an application using machine learning libraries to process real time data from streaming sources and analysts can simultaneously access the data using SQL also in real time. In this example three different workloads namely SQL, streaming data and machine learning algorithms can be implemented in a single system which is a requirement in today's age of big data.

SPARK COMPONENTS

Figure 1 depicts the various building blocks of the spark stack. The Spark Core is computational engine which performs the task scheduling, distribution, and cluster monitoring tasks. Resilient Distributed Datasets(RDD) [3] and Directed Acyclic Graphs(DAG) are two important concepts in Spark. The libraries or packages supporting the diverse workloads are built on top of the Spark core. These packages include Spark SQL, Spark Streaming, MLlib (machine learning library) and GraphX.

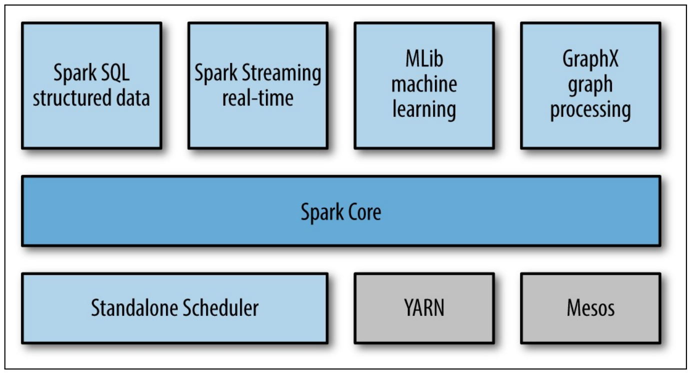

Fig. 1. Spark Components [2]

Spark Core

Spark Core is the foundation framework that provides basic I/O functionality, distributed task scheduling and dispatching. more [1]."

Resilient Distributed Datasets(RDD)

Resilient Distributed Datasets(RDD) [3] are Spark's primary abstraction, which are a fault-tolerant collection of elements that can be operated in parallel. RDDs are immutable once they are created but they can be transformed or actions can be performed on them [1]. Users can create RDDs through external sources or by transforming another RDD. Transformations and Actions are the two types of operations supported by RDDs.

1. Transformations: Since RDDs are immutable the transformations return a new RDD and not a single value." Transformations are lazily evaluated, i.e. they are not computed immediately. They are executed only when an action runs on it. Some of the Transformation functions are map, filter, ReduceByKey, FlatMap and GroupByKey [1]."

2. Actions are operations that result in a return value after computation or triggers a task in response to some operation.Some Action operations are first, take, reduce, collect, count, foreach and CountByKey [1].

As such, RDDs are ephemeral disk, which means they do not persist data, however, users can explicitly persist RDDs to ease data reuse. Traditional distributed computing systems provide fault tolerance through checkpoint or data replication. "RDDs provide fault tolerance by logging the transformations used to build a data set through (its lineage) rather than actual data [3]." If one of the RDD fails, it has enough information about of its lineage so as to recreate the dataset from other RDDs, thus saving cost and time.

Directed Acyclic Graph(DAG)

Directed Acyclic Graph(DAG), which supports a cyclic data flow, "consists of finitely many vertices and edges, with each edge directed from one vertex to another, such that there is no way to start at any vertex v and follow a consistently-directed sequence of edges that eventually loops back to v again [4]." When we run any application in spark, the driver program converts the transformations and actions to logical directed acyclic graphs(DAG). The DAGs are then converted to physical execution plans with a set of stages which are distributed and bundled into tasks. These tasks are distributed among the different worker nodes for execution.

Spark SQL

Spark SQL[2] is a library built on top of the Spark Core to support querying structured data using SQL or Hive Query Language. It allows users to perform ETL (Extract, Transform and Load) operations on data from various sources such as JSON, Hive Tables and Parquet. Developers can "intermix SQL queries with programmatic data manipulations supported by RDDs in Python, Java, and Scala, all within a single application [2]."

Spark Streaming

Spark Streaming [2] library enables Spark to process real time data. Examples of streaming data are messages being published to a queue for real time flight status update or the log files for a production server. "Spark Streaming provides an API for manipulating data streams that closely matches the Spark Core's RDD API, making it easy for programmers to learn the project and move between applications that manipulate data stored in memory, on disk, or arriving in real time." Spark Streaming is designed such that it provides the same level of fault tolerance, throughput and scalability as the Spark Core.

MLlib

MLlib [2] is rich library of machine learning algorithms for Spark which can be accessed from Java, Scala as well as Python. It provides Spark with machine learning algorithms such as "classification, regression, clustering, and collaborative filtering". It also provides machine learning functionality such as "model evaluation and data import". The common machine learning algorithms include K-means, navie bayes, logistic regression, Principal component analysis and so on.

GraphX

GraphX introduces the Resilient Distributed Property Graph, which is directed multi-graph having properties attached to each edge and vertex. GraphX includes a set of operators like aggregateMessages, subgraph and joinVertices, and optimized variant of Pregel API. It also includes builders and graph algorithms to simplify graph analytics tasks [1].

RUNTIME ARCHITECTURE

The runtime architecture of Spark, illustrated in Figure 2, consists of a driver program, a cluster manager, workers or executors and the HDFS (Hadoop Distributed File System) [1]. Spark uses a master/slave architecture in which the driver program is the master and worker nodes or executors are the slaves. The driver runs the main() method of the user program which creates the SparkContext, the RDDs and performs transformations and actions [2].

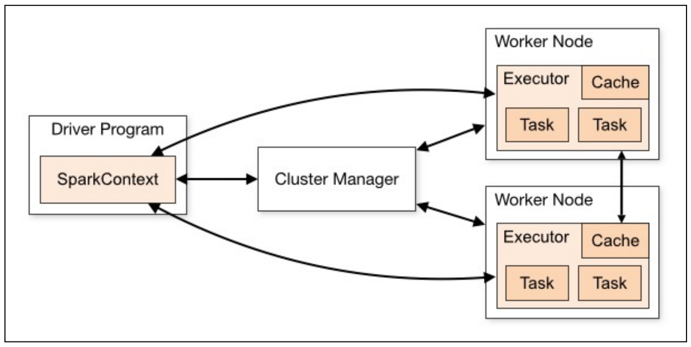

Fig. 2. Spark Architecture [5]

When we launch an application using the Spark Shell it creates a driver program which in turn initializes the SparkContext. Each spark application has its own SparkContext object which is responsible for the entire execution of the job. The SparkContext object then connects to cluster manager to request resources for its workers. The cluster manager provide executers to worker nodes, which are used to run the logic and also store the application data. The driver will send the tasks to the executors based on the data placement. The executors register themselves with the driver, which helps the driver keep tabs on the executors. Driver can also schedule future tasks by caching or persisting data. The following Cluster Managers are used in Spark based on the requirement-

- Standalone cluster manager is a simple cluster manager built into Spark to manage is own clusters [5].

- Apache Mesos is a dedicated cluster manager that provides Spark with rich resource scheduling capabilities [5].

- YARN is the only cluster manager in Spark that provides security support. “It allows dynamic sharing and central configuration of the same pool of cluster resources between various frameworks that run on YARN [6]”.

Educational Resources

The Apache Spark website has a detailed documentation on the how to get started with spark [7]. It explains the concepts and shows examples to help us familiarize with Spark.

ACKNOWLEDGEMENTS

This paper is written as part of the I524: Big Data and Open Source Software Projects coursework at Indiana University. We would like to thank our Prof. Gregor von Laszewski, Prof. Gregory Fox and the AIs for their help and support

REFERENCES

[1] A. Bansod, “Efficient big data analysis with apache spark in hdfs,” *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 4, no. 6, pp. 313–316, aug 2015.
[2] H. Karau, A. Konwinski, P. Wendell, and M. Zaharia, *Learning Spark: Lightning-Fast Big Data Analytics*, 1st ed. O’Reilly Media, Inc., feb 2015. [Online]. Available: https://www.safaribooksonline.com/library/view/learning-spark/9781449359034/
[3] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, and I. Stoica, “Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing,” in *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012*. Berkeley, CA, USA: USENIX Association, 2012, pp. 15–28. [Online]. Available: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/zaharia
[4] “Directed acyclic graph - wikipedia,” Article, accessed: 02-26-2017. [Online]. Available: https://en.wikipedia.org/wiki/Directed_acyclic_graph
[5] A. S. Foundation, “Cluster mode overview - spark 2.1.0 documentation,” Article, accessed: 02-22-2017. [Online]. Available: http://spark.apache.org/docs/latest/cluster-overview.html
[6] “Apache spark ecosystem and spark components,” Feb. 2016, apache Spark Ecosystem and Spark Components. [Online]. Available: https://www.dezyre.com/article/apache-spark-ecosystem-and-spark-components/219
[7] A. S. Foundation, “Spark programming guide - spark 2.1.0 documentation,” Article, accessed: 02-22-2017. [Online]. Available: http://spark.apache.org/docs/latest/programming-guide.html