Distributed Systems,

Project3-Assignment1

CSE 5306, Fall 2021

Contributors:

PAVAN KUMAR CHAPARLA – 1001865675, pxc5675@mavs.uta.edu

POLA MANOHAR – 1001980283, mxp0284@mavs.uta.edu

05th December 2021.

Assignment-1

1. What are the key differences between Hadoop and Spark, and their respective advantages?

Key Differences between Hadoop and Spark.

HADOOP	SPARK
1.) Hadoop is slow compared to Spark as it performs operations on the disk and not directly on the ram like Spark and thus cannot be suitable for real-time analytics most of the time.	1.) Spark runs 100 x faster in-memory, and 10x faster on disk compared to Hadoop. If Spark runs on YARN with other resources demanding services, there could be major degradation.
2.) Hadoop works with disk sizes, and which is very much less in cost compared to the rams and thus it is less expensive than Apache Spark.	2.) Spark is very expensive compared to Hadoop because it deals with the ram size, when needed to be scaled it increases the cost and cluster size every time.
3.) Hadoop's fault-tolerance is very high because it uses a mechanism where it replicates the data of every file that it works with in various nodes. Each file is split into blocks and replicated numerous times across many machines.	3.) Spark uses RDD's (Resilient Distributed Datasets) which are collection of elements that can be used for fault-tolerance but, they as are already runs in the ram they are very fast compared to the Hadoop.
4.) In Hadoop data processing is done in the form of batches in a sequential order one by one and taken from the cluster and after operation done returned to the cluster.	4.) Just like Hadoop, Spark also performs batch processing of data, but along with that it also performs graph processing and real-time processing of data.
5.) Hadoop doesn't have a interactive model of Map-Reducing concept and is very complex compared to Spark.	5.) Spark has a interactive mode and also supports a lot of user-friendly API's for the necessity.
6.) Hadoop has been developed from in Java language and the Map-Reducing concept of Hadoop can be implemented in C, C++, or Python.	6.) Spark has been developed in Scala language and can be implemented in Java, Python, R etc.
7.) Hadoop has authentication supported by Kerberos, etc. and the traditional way of giving permissions to files accesses.	7.) Spark provides security via password mechanism. Also when Spark runs on Yarn it can also use the file-level authorizations.

8.) Hadoop uses Mahout, etc. for data processing and building models.	8.) Spark has inbuilt libraries such as machine learning that can be used for data regression and classifications.

ADVANTAGES OF HADOOP:-

High Availability:

The Hadoop File System is highly available file system because data is replicated across various nodes in a cluster by generating and having various replicas of the blocks on the other slaves present in the Hadoop File System cluster. So, when a user wants to access his/her data, they can still access their data from the nodes which contains its blocks and which is available on the nearest node in the cluster. When a node fails suddenly, a user can access their data from other nodes which already have the replica of the data the failed node has and thus eliminating loss of data.

Data Reliability:

The Hadoop File System provides reliable storage for the data as it can store large amounts of data ranging around 100's of petabytes with having replicas for every block of data present at various nodes and thus if one or more fails still has the replica that can provide with the data.

Scalability:

The Hadoop file System deals with a great amount of data as mentioned earlier and thus has a lot of space available for the scalability when required. It has two mechanisms when it comes to the scalability. Horizontally and vertically.

Fault Tolerance:

The Hadoop File System provides high fault-tolerance as the data in the File System is replicated across various machines in the cluster in the form of blocks. Even if any machine in a cluster which has a block of data fails or goes down the data will still be available for the user as the Hadoop Filesystem has the block of data replicated in another machine and that can be used to get back the data.

OpenSource:

The Hadoop framework is an open-source framework and thus it can be used and implemented in any required way as per the requirements.

Cost-Efficient:

Since the commodity hardware that is used in Hadoop framework is very economical, the cost for using new nodes to the framework is very cheap and thus makes it cost-efficient.

Performance:

Its concept of distribution and processing helps in processing the large amounts of data very quickly and thus has a high-performance rate. All the processing happens parallelly and so the performance efficiency is achieved due to this. It distributes the jobs to various nodes for execution on a high note and thus will compute the parallel processing.

Compatibility:

A lot of technologies in the industry currently use Hadoop as their storage system to store and process huge amounts of data quickly. Hadoop gives the flexibility for a lot of software's to work efficiently with it like Spark, Flink etc.

Language Support:

Hadoop supports a lot of languages like C, C++, Java, Python etc. which can be implemented accordingly.

ADVANTAGES OF SPARK:

Speed:

Apache Spark provides high speed when it comes with computing the data. Since it uses the memory which is RAM as the computing system it processes the data very quickly and capable of holding huge and huge amounts of data like 100's and 1000's of petabytes of data for processing in a given time.

Advanced Analytics:

Spark provides additional features along with processing the data like, graph processing of data and real time analytics of the data.

Language Support:

Spark can be implemented in different programming languages like Python, R, etc.

Dynamicity Of Nature:

Spark offers various number of high-level operators. With them we can develop the parallel applications very easily.

Open-Source:

Spark is again an open-source framework that can be implemented as per any kind of requirements.

Simplicity:

Spark provides a lot of rich APIs which are provided with good documentation and also very easy to understand. These API's can be easily used by the developers for working with the Spark framework.

Fault Tolerance:

Apache spark provides fault tolerance using RDD (Resilient Distributed Datasets), has a capability of handling if any loss occurs. It can recover the failure itself, here fault refers to failure. If any bug or loss found, RDD can recover the loss on itself.

2. Discuss how to recover a failed task in Hadoop and Spark, respectively.

Fault Tolerance and Task Recovery in Hadoop:

A failed task can be any of the task in the Hadoop framework. It can be the reduce task or the map task.

In case of a particular task failure, Hadoop initiates another computational resource in order to perform failed map or reduce tasks. When it comes to failure of shuffle and sort process, it is basically a failure in the particular node where reducer task has failed and it would be set to run afresh in another resource (btw, reducer phase begin with shuffle and sort process). It doesn't allocate the tasks infinitely if the tasks keep failing again and again. The two properties below will give the information about how many failures or attempts of a task will be accepted.

mapred.map.max.attempts for Map tasks and a property **mapred.reduce.max.attempts** for reduce tasks will be checked every time. By default, if any task fails five times (or whatever you configure in those properties), the whole job would be considered as failed. As per the Hadoop Definitive Guide. In short shuffle and sort being a part of reducer, it would only initiate attempt to rerun reducer task. Map tasks would not be re-run as they are considered as completed.

The tasks can also fail at different phases.

They can fail at the Master Node, the Worker nodes

Worker Failures:

The master pings every worker periodically. If no response is received from a worker in a certain amount of time, the master marks the worker as failed. Any map tasks completed by the worker are reset back to their initial idle state, and therefore become eligible for scheduling on other workers. Similarly, any map task or reduce task in progress on a failed worker is also reset to idle and becomes eligible for rescheduling.

If the Mapper task fails it cannot be resumed from the exit point and will be restarted as they are performed on the disks, and they cannot be accessed. The reduce tasks can be re-executed because their output will be stored on the file system, and it will retrieved and executed. When a map task is executed first by worker A and then later executed by worker B (because A failed), all To appear in OSDI 2004 4 workers executing reduce tasks are notified of the reexecution. Any reduce task that has not already read the data from worker A will read the data from worker B.

Master Failures:

If the master task dies the master node has a checkpoint always maintained and thus when failed the task's new copy will be started from the last checkpointed stage of the master task. In this way it will be recovered using the checkpoints.

Fault Tolerance and Task Recovery in Spark:

Spark framework uses RDD model which helps in recovery. RDDs do not have to follow the checkpoint strategy, as they can be recovered using lineage. And also, only the lost partitions of an RDD need to be recomputed upon failure, and they can be recomputed in parallel on different nodes, without having to roll back the whole program.

If any of the nodes of processing data gets crashed, that results in a fault in a cluster. In other words, RDD is logically partitioned and each node is operating on a partition at any point in time. Operations which are being performed is a series of scala functions. Those operations are being executed on that partition of RDD. This series of operations are merged together and create a Directed Acyclic Graph.. That means DAG keeps track of operations performed.

If any node crashes in the middle of an operation, the cluster manager finds out that node. Then, it tries to assign another node to continue the processing at the same place.

So, when a node fails the driver spawn another executor in some other node and provides it the Data partition on which it was supposed to work and the DAG associated with it in a closure. Now with this information it can recompute the data and materialize it.

In the mean time the cached data in the RDD won't have all the data in memory, the data of the lost nodes it has to fetch from the disk it will take so little more time.

Mesos open source software which is another way of recovering the data can also be used by the Spark framework. This software also deals with large-scale applications and frameworks.