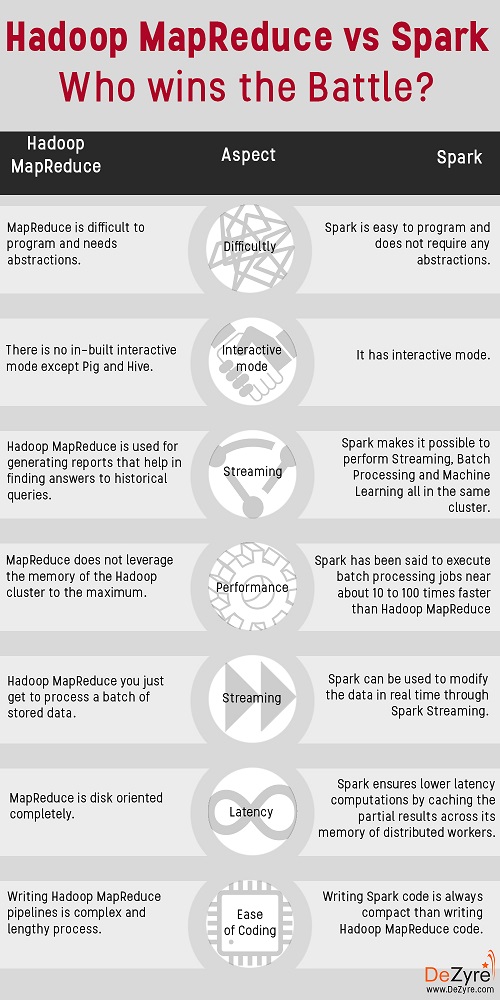
**1) Compare Spark vs Hadoop MapReduce**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Hadoop MapReduce** | **Apache Spark** |
| Memory | Does not leverage the memory of the hadoop cluster to maximum. | Let's save data on memory with the use of RDD's. |
| Disk usage | MapReduce is disk oriented. | Spark caches data in-memory and ensures low latency. |
| Processing | Only batch processing is supported | Supports real-time processing through spark streaming. |
| Installation | Is bound to hadoop. | Is not bound to Hadoop. |
| **Spark vs Hadoop** | | |



Simplicity, Flexibility and Performance are the major advantages of using Spark over Hadoop.

* Spark is 100 times faster than Hadoop for big data processing as it stores the data in-memory, by placing it in Resilient Distributed Databases (RDD).
* Spark is easier to program as it comes with an interactive mode.
* It provides complete recovery using lineage graph whenever something goes wrong.

Refer [Spark vs Hadoop](https://www.dezyre.com/article/hadoop-mapreduce-vs-apache-spark-who-wins-the-battle/83)

**2) What is Shark?**

Most of the data users know only SQL and are not good at programming. Shark is a tool, developed for people who are from a database background - to access Scala MLib capabilities through Hive like SQL interface. Shark tool helps data users run Hive on Spark - offering compatibility with Hive metastore, queries and data.

**3) List some use cases where Spark outperforms Hadoop in processing.**

1. Sensor Data Processing –Apache Spark’s ‘In-memory computing’ works best here, as data is retrieved and combined from different sources.
2. Spark is preferred over Hadoop for real time querying of data
3. Stream Processing – For processing logs and detecting frauds in live streams for alerts, Apache Spark is the best solution.

**4) What is a Sparse Vector?**

A sparse vector has two parallel arrays –one for indices and the other for values. These vectors are used for storing non-zero entries to save space.

**5) What is RDD?**

RDDs (Resilient Distributed Datasets) are basic abstraction in Apache Spark that represent the data coming into the system in object format. RDDs are used for in-memory computations on large clusters, in a fault tolerant manner. RDDs are read-only portioned, collection of records, that are –

* Immutable – RDDs cannot be altered.
* Resilient – If a node holding the partition fails the other node takes the data.

**6) Explain about transformations and actions in the context of RDDs.**

Transformations are functions executed on demand, to produce a new RDD. All transformations are followed by actions. Some examples of transformations include map, filter and reduceByKey.

Actions are the results of RDD computations or transformations. After an action is performed, the data from RDD moves back to the local machine. Some examples of actions include reduce, collect, first, and take.

**7) What are the languages supported by Apache Spark for developing big data applications?**

Scala, Java, Python, R and Clojure

**8) Can you use Spark to access and analyse data stored in Cassandra databases?**

Yes, it is possible if you use Spark Cassandra Connector.

**9) Is it possible to run Apache Spark on Apache Mesos?**

Yes, Apache Spark can be run on the hardware clusters managed by Mesos.

**10) Explain about the different cluster managers in Apache Spark**

The 3 different clusters managers supported in Apache Spark are:

* YARN
* Apache Mesos -Has rich resource scheduling capabilities and is well suited to run Spark along with other applications. It is advantageous when several users run interactive shells because it scales down the CPU allocation between commands.
* Standalone deployments – Well suited for new deployments which only run and are easy to set up.

**11) How can Spark be connected to Apache Mesos?**

To connect Spark with Mesos-

* Configure the spark driver program to connect to Mesos. Spark binary package should be in a location accessible by Mesos. (or)
* Install Apache Spark in the same location as that of Apache Mesos and configure the property ‘spark.mesos.executor.home’ to point to the location where it is installed.

**12) How can you minimize data transfers when working with Spark?**

Minimizing data transfers and avoiding shuffling helps write spark programs that run in a fast and reliable manner. The various ways in which data transfers can be minimized when working with Apache Spark are:

1. Using Broadcast Variable- Broadcast variable enhances the efficiency of joins between small and large RDDs.
2. Using Accumulators – Accumulators help update the values of variables in parallel while executing.
3. The most common way is to avoid operations ByKey, repartition or any other operations which trigger shuffles.

**13)  Why is there a need for broadcast variables when working with Apache Spark?**

These are read only variables, present in-memory cache on every machine. When working with Spark, usage of broadcast variables eliminates the necessity to ship copies of a variable for every task, so data can be processed faster. Broadcast variables help in storing a lookup table inside the memory which enhances the retrieval efficiency when compared to an RDD lookup ().

**14)  Is it possible to run Spark and Mesos along with Hadoop?**

Yes, it is possible to run Spark and Mesos with Hadoop by launching each of these as a separate service on the machines. Mesos acts as a unified scheduler that assigns tasks to either Spark or Hadoop.

**15)  What is lineage graph?**

The RDDs in Spark, depend on one or more other RDDs. The representation of dependencies in between RDDs is known as the lineage graph. Lineage graph information is used to compute each RDD on demand, so that whenever a part of persistent RDD is lost, the data that is lost can be recovered using the lineage graph information.

**16) How can you trigger automatic clean-ups in Spark to handle accumulated metadata?**

You can trigger the clean-ups by setting the parameter ‘spark.cleaner.ttl’ or by dividing the long running jobs into different batches and writing the intermediary results to the disk.

**17) Explain about the major libraries that constitute the Spark Ecosystem**

* **Spark MLib**- Machine learning library in Spark for commonly used learning algorithms like clustering, regression, classification, etc.
* **Spark Streaming**– This library is used to process real time streaming data.
* **Spark GraphX**– Spark API for graph parallel computations with basic operators like joinVertices, subgraph, aggregateMessages, etc.
* **Spark SQL**– Helps execute SQL like queries on Spark data using standard visualization or BI tools.

**18) What are the benefits of using Spark with Apache Mesos?**

It renders scalable partitioning among various Spark instances and dynamic partitioning between Spark and other big data frameworks.

**19) What is the significance of Sliding Window operation?**

Sliding Window controls transmission of data packets between various computer networks. Spark Streaming library provides windowed computations where the transformations on RDDs are applied over a sliding window of data. Whenever the window slides, the RDDs that fall within the particular window are combined and operated upon to produce new RDDs of the windowed DStream.

**20) What is a DStream?**

Discretized Stream is a sequence of Resilient Distributed Databases that represent a stream of data. DStreams can be created from various sources like Apache Kafka, HDFS, and Apache Flume. DStreams have two operations –

* Transformations that produce a new DStream.
* Output operations that write data to an external system.

**21) When running Spark applications, is it necessary to install Spark on all the nodes of YARN cluster?**

Spark need not be installed when running a job under YARN or Mesos because Spark can execute on top of YARN or Mesos clusters without affecting any change to the cluster.

**22) What is Catalyst framework?**

Catalyst framework is a new optimization framework present in Spark SQL. It allows Spark to automatically transform SQL queries by adding new optimizations to build a faster processing system.

**23) Name a few companies that use Apache Spark in production.**

Pinterest, Conviva, Shopify, Open Table

**24) Which spark library allows reliable file sharing at memory speed across different cluster frameworks?**

Tachyon

**25) Why is BlinkDB used?**

BlinkDB is a query engine for executing interactive SQL queries on huge volumes of data and renders query results marked with meaningful error bars. BlinkDB helps users balance ‘query accuracy’ with response time.

**26) How can you compare Hadoop and Spark in terms of ease of use?**

Hadoop MapReduce requires programming in Java which is difficult, though Pig and Hive make it considerably easier. Learning Pig and Hive syntax takes time. Spark has interactive APIs for different languages like Java, Python or Scala and also includes Shark i.e. Spark SQL for SQL lovers - making it comparatively easier to use than Hadoop.

**27) What are the common mistakes developers make when running Spark applications?**

Developers often make the mistake of-

* Hitting the web service several times by using multiple clusters.
* Run everything on the local node instead of distributing it.

Developers need to be careful with this, as Spark makes use of memory for processing.

**28) What is the advantage of a Parquet file?**

Parquet file is a columnar format file that helps –

* Limit I/O operations
* Consumes less space
* Fetches only required columns.

**29) What are the various data sources available in SparkSQL?**

* Parquet file
* JSON Datasets
* Hive tables

**30) How Spark uses Hadoop?**

Spark has its own cluster management computation and mainly uses Hadoop for storage.

For the complete list of big data companies and their salaries- [**CLICK HERE**](https://docs.google.com/forms/d/1LFuWEKQKCLR231qR9WE5PZakJj77fTDIW6ox5328HFM/viewform)

**31) What are the key features of Apache Spark that you like?**

* Spark provides advanced analytic options like graph algorithms, machine learning, streaming data, etc
* It has built-in APIs in multiple languages like Java, Scala, Python and R
* It has good performance gains, as it helps run an application in the Hadoop cluster ten times faster on disk and 100 times faster in memory.

**32) What do you understand by Pair RDD?**

Special operations can be performed on RDDs in Spark using key/value pairs and such RDDs are referred to as Pair RDDs. Pair RDDs allow users to access each key in parallel. They have a reduceByKey () method that collects data based on each key and a join () method that combines different RDDs together, based on the elements having the same key.

**33) Which one will you choose for a project –Hadoop MapReduce or Apache Spark?**

The answer to this question depends on the given project scenario - as it is known that Spark makes use of memory instead of network and disk I/O. However, Spark uses large amount of RAM and requires dedicated machine to produce effective results. So the decision to use Hadoop or Spark varies dynamically with the requirements of the project and budget of the organization.

**34) Explain about the different types of transformations on DStreams?**

* Stateless Transformations- Processing of the batch does not depend on the output of the previous batch. Examples – map (), reduceByKey (), filter ().
* Stateful Transformations- Processing of the batch depends on the intermediary results of the previous batch. Examples –Transformations that depend on sliding windows.

**35) Explain about the popular use cases of Apache Spark**

Apache Spark is mainly used for

* Iterative machine learning.
* Interactive data analytics and processing.
* Stream processing
* Sensor data processing

**36) Is Apache Spark a good fit for Reinforcement learning?**

No. Apache Spark works well only for simple machine learning algorithms like clustering, regression, classification.

**37) What is Spark Core?**

It has all the basic functionalities of Spark, like - memory management, fault recovery, interacting with storage systems, scheduling tasks, etc.

**38) How can you remove the elements with a key present in any other RDD?**

Use the subtractByKey () function

**39) What is the difference between persist() and cache()**

persist () allows the user to specify the storage level whereas cache () uses the default storage level.

**40) What are the various levels of persistence in Apache Spark?**

Apache Spark automatically persists the intermediary data from various shuffle operations, however it is often suggested that users call persist () method on the RDD in case they plan to reuse it. Spark has various persistence levels to store the RDDs on disk or in memory or as a combination of both with different replication levels.

The various storage/persistence levels in Spark are -

* MEMORY\_ONLY
* MEMORY\_ONLY\_SER
* MEMORY\_AND\_DISK
* MEMORY\_AND\_DISK\_SER, DISK\_ONLY
* OFF\_HEAP

**41) How Spark handles monitoring and logging in Standalone mode?**

Spark has a web based user interface for monitoring the cluster in standalone mode that shows the cluster and job statistics. The log output for each job is written to the work directory of the slave nodes.

**42) Does Apache Spark provide check pointing?**

Lineage graphs are always useful to recover RDDs from a failure but this is generally time consuming if the RDDs have long lineage chains. Spark has an API for check pointing i.e. a REPLICATE flag to persist. However, the decision on which data to checkpoint - is decided by the user. Checkpoints are useful when the lineage graphs are long and have wide dependencies.

**43) How can you launch Spark jobs inside Hadoop MapReduce?**

Using SIMR (Spark in MapReduce) users can run any spark job inside MapReduce without requiring any admin rights.

**44) How Spark uses Akka?**

Spark uses Akka basically for scheduling. All the workers request for a task to master after registering. The master just assigns the task. Here Spark uses Akka for messaging between the workers and masters.

**45) How can you achieve high availability in Apache Spark?**

* Implementing single node recovery with local file system
* Using StandBy Masters with Apache ZooKeeper.

**46) Hadoop uses replication to achieve fault tolerance. How is this achieved in Apache Spark?**

Data storage model in Apache Spark is based on RDDs. RDDs help achieve fault tolerance through lineage. RDD always has the information on how to build from other datasets. If any partition of a RDD is lost due to failure, lineage helps build only that particular lost partition.

**47) Explain about the core components of a distributed Spark application.**

* Driver- The process that runs the main () method of the program to create RDDs and perform transformations and actions on them.
* Executor –The worker processes that run the individual tasks of a Spark job.
* Cluster Manager-A pluggable component in Spark, to launch Executors and Drivers. The cluster manager allows Spark to run on top of other external managers like Apache Mesos or YARN.

**48) What do you understand by Lazy Evaluation?**

Spark is intellectual in the manner in which it operates on data. When you tell Spark to operate on a given dataset, it heeds the instructions and makes a note of it, so that it does not forget - but it does nothing, unless asked for the final result. When a transformation like map () is called on a RDD-the operation is not performed immediately. Transformations in Spark are not evaluated till you perform an action. This helps optimize the overall data processing workflow.

**49)  Define a worker node.**

A node that can run the Spark application code in a cluster can be called as a worker node. A worker node can have more than one worker which is configured by setting the SPARK\_ WORKER\_INSTANCES property in the spark-env.sh file. Only one worker is started if the SPARK\_ WORKER\_INSTANCES property is not defined.

**50) What do you understand by SchemaRDD?**

An RDD that consists of row objects (wrappers around basic string or integer arrays) with schema information about the type of data in each column.

**51) What are the disadvantages of using Apache Spark over Hadoop MapReduce?**

Apache spark does not scale well for compute intensive jobs and consumes large number of system resources. Apache Spark’s in-memory capability at times comes a major roadblock for cost efficient processing of big data. Also, Spark does have its own file management system and hence needs to be integrated with other cloud based data platforms or apache hadoop.

**52) Is it necessary to install spark on all the nodes of a YARN cluster  while running Apache Spark on YARN ?**

No , it is not necessary because Apache Spark runs on top of YARN.

**53) What do you understand by Executor Memory in a Spark application?**

Every spark application has same fixed heap size and fixed number of cores for a spark executor. The heap size is what referred to as the Spark executor memory which is controlled with the spark.executor.memory property of the –executor-memory flag. Every spark application will have one executor on each worker node. The executor memory is basically a measure on how much memory of the worker node will the application utilize.

**54) What does the Spark Engine do?**

Spark engine schedules, distributes and monitors the data application across the spark cluster.

**55) What makes Apache Spark good at low-latency workloads like graph processing and machine learning?**

Apache Spark stores data in-memory for faster model building and training. Machine learning algorithms require multiple iterations to generate a resulting optimal model and similarly graph algorithms traverse all the nodes and edges.These low latency workloads that need multiple iterations can lead to increased performance. Less disk access and  controlled network traffic make a huge difference when there is lots of data to be processed.

**Spark Streaming Interview Questions**

1) Name some sources from where Spark streaming component can process real-time data.

Apache Flume, Apache Kafka, Amazon Kinesis

2) Name some companies that are already using Spark Streaming.

Uber, Netflix, Pinterest.

3) What is the bottom layer of abstraction in the Spark Streaming API ?

DStream.