Assesment 17 (sreevidya) 22-12-23

Spark Architecture

Apache Spark is an open-source, distributed computing system that provides fast and general-purpose cluster computing for big data processing. Spark was developed to address the limitations of MapReduce, offering better performance and ease of use. Here's an overview of the Spark architecture:

**Driver Program:**

The driver program is the main control program that coordinates the execution of tasks on a cluster. It contains the user's application code and creates a SparkContext to coordinate the execution of tasks.

**SparkContext:**

SparkContext is the entry point for any Spark functionality. It coordinates the execution of tasks and manages the distributed data across the cluster.

**Cluster Manager:**

Spark can run on various cluster managers like Apache Mesos, Hadoop YARN, or its standalone built-in cluster manager. The cluster manager is responsible for allocating resources and managing the execution of tasks on the cluster.

**Executor:**

Executors are worker nodes responsible for running tasks. The driver program communicates with the executors to schedule tasks and collect results. Executors are launched at the beginning of a Spark application and run throughout the application's lifetime.

**Task:**

A task is the smallest unit of work in Spark and represents a computation that will be run on a single partition of the distributed data. Tasks are scheduled by the SparkContext on the executors.

**Resilient Distributed Dataset (RDD):**

RDD is the fundamental data structure in Spark, representing a fault-tolerant collection of elements that can be processed in parallel. RDDs can be created from Hadoop Distributed File System (HDFS) files, local files, or by transforming existing RDDs through operations like map, filter, and reduce.

**Directed Acyclic Graph (DAG):**

Spark operations on RDDs form a directed acyclic graph (DAG) of stages. A stage is a set of tasks that can be executed in parallel. Spark breaks down transformations and actions into stages for optimization.

**Transformations:**

Transformations are operations that create a new RDD from an existing one. Examples include map, filter, and reduceByKey. Transformations are lazily evaluated, meaning they are not executed immediately but are recorded in the DAG.

**Actions:**

Actions are operations that trigger the execution of the DAG and return a result to the driver program or write data to an external storage system. Examples include count, collect, and save.

**Shuffling:**

Shuffling is the process of redistributing data across the partitions, usually occurring after certain operations like groupByKey or reduceByKey. It involves the exchange of data between executors and can be a performance-intensive operation.

**Broadcast Variables:**

Broadcast variables are read-only variables cached on each machine rather than being sent over the network with tasks. They are useful for efficiently sharing large, read-only data structures among the tasks.

