Problem Statement

1) Pen down the limitations of MapReduce.

2) What is RDD? Explain few features of RDD?

3) List down few Spark RDD operations and explain each of them.

1) Pen down the limitations of MapReduce.

- It's based on disk based computing.
- Suitable for single pass computations - not iterative computations.
- Needs a sequence of MR jobs to run iterative tasks.
- Needs integration with several other frameworks/tools to solve bigdata usecases.
    - Apache Storm for stream data processing
    - Apache Mahout for machine learning

2) What is RDD? Explain few features of RDD?

RDD: Resilient Distributed Datasets (RDD) is a fundamental data structure of Spark. It is an immutable distributed collection of objects. Each dataset in RDD is divided into logical partitions, which may be computed on different nodes of the cluster.

Features of RDD

Resilient, i.e. fault-tolerant with the help of RDD lineage graph and so able to recompute missing or damaged partitions due to node failures

Distributed, with data residing on multiple nodes in a cluster.

Dataset, is a collection of partitioned data with primitive values or values of values, e.g. tuples or other objects

In-Memory, i.e. data inside RDD is stored in memory as much (size) and long (time) as possible.

Immutable or Read-Only, i.e. it does not change once created and can only be transformed using transformations to new RDDs.

Lazy evaluated, i.e. the data inside RDD is not available or transformed until an action is executed that triggers the execution.

Cacheable, i.e. you can hold all the data in a persistent "storage" like memory (default and the most preferred) or disk (the least preferred due to access speed).

IParallel , i.e. process data in parallel.

Typed — RDD records have types, e.g. Long in RDD[Long] or (Int, String) in RDD[(Int, String)].
Partitioned, records are partitioned (split into logical partitions) and distributed across nodes in a cluster
.
Location-Stickiness, RDD can define placement preferences to compute partitions (as close to the records as possible).

3) List down few Spark RDD operations and explain each of them.

- RDD Supports two kinds of operations
- Actions - operations that trigger computation and return value
- Transformations - lazy operations that return another RDD

Actions

- Actions are RDD operations that produce non-RDD values. In other words, a RDD operation that returns a value of any type except RDD[T] is an action.

- They trigger execution of RDD transformations to return values. Simply put, an action evaluates the RDD lineage graph.

- You can think of actions as a valve and until action is fired, the data to be processed is not even in the pipes, i.e. transformations. Only actions can materialize the entire processing pipeline with real data.
- Actions are one of two ways to send data from executors to the driver (the other being accumulators).

Transformations

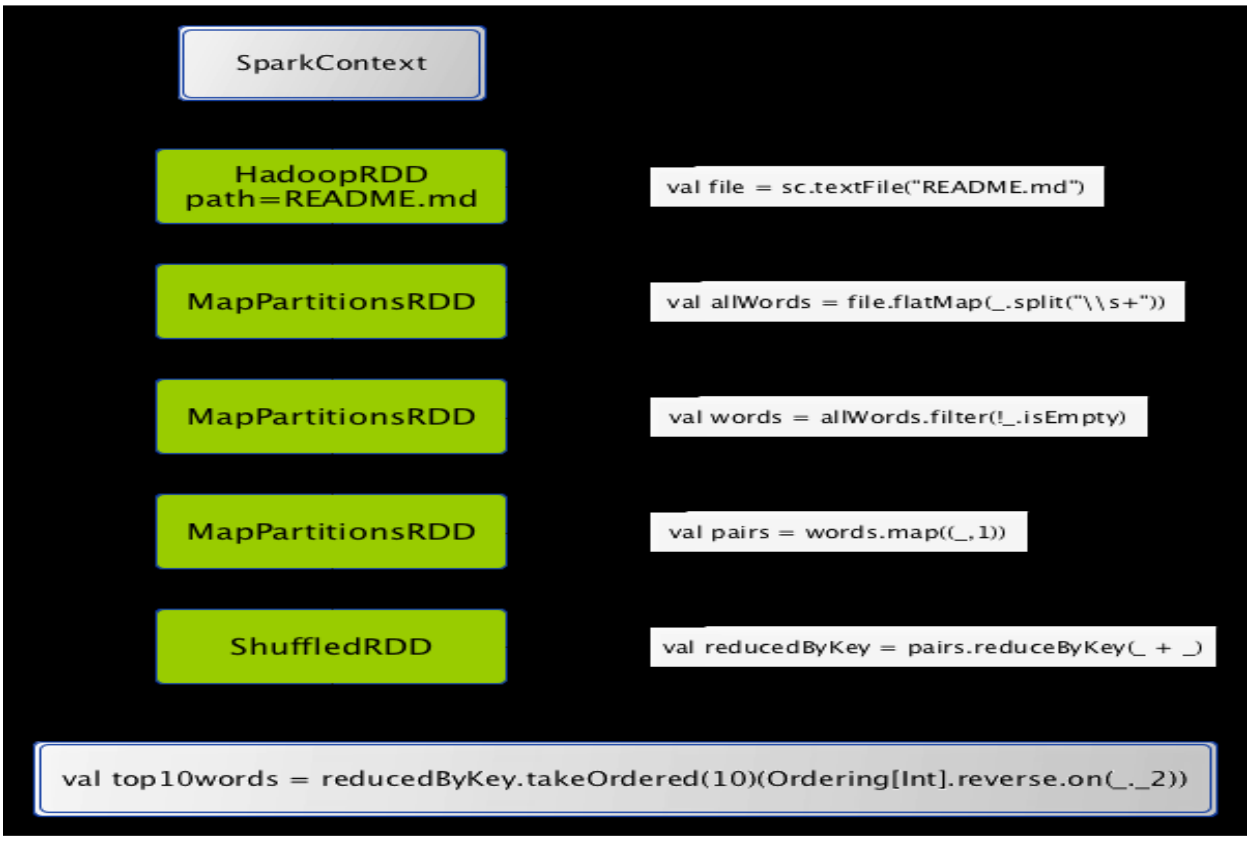
- Transformations are lazy operations on a RDD that create one or many new RDDs,

e.g. map, filter, reduceByKey, join, cogroup, randomSplit.

- They are functions that take a RDD as the input and produce one or many RDDs as the output. They do not change the input RDD (since RDDs are immutable), but always produce one or more new RDDs by applying the computations they represent.

- Transformations are lazy, i.e. are not executed immediately. Only after calling an action are transformations executed.

- After executing a transformation, the result RDD(s) will always be different from their parents and can be smaller (e.g. filter, count, distinct, sample), bigger (e.g. flatMap, union, cartesian) or the same size (e.g. map).



Submitted By:

Shishir Kumar Jha

Accadgild_Session_16_Assignment_16.1