Surya Teja Cherukuri

1001565129

Introduction:

We discussed about hadoop and pregel. They are distributed computing frameworks for large scale data analytics. But there is a problem on how the data is stored and accessed. hadoop requires the data to be written to the hdfs between each map reduce job. and pregel is limited in its applications to graph processing though it allows in memory intermediate data.

RDDs provide a more general approach to a wide range of applications that can leverage the distributed memory in a fault tolerant manner.

RDD is a distributed memory abstraction that lets programmer perform in memory computations on large clusters in a fault tolerant manner.

Advantages of in memory computations:
Ability to handle iterative algorithms and interactive data mining tools.

1. “...individual RDDs are immutable...” What does it mean by being “immutable”? What benefits does this property of RDD bring?

Immutable: Content of an RDD cannot be modified.
Benefits: Making RDDs immutable makes it easier to describe lineage graphs.
Each time a different transformation is used on a RDD it adds to the linear graph of the particular RDD. Each different RDD will have a different transformation and hence different lineage graphs. If we make the RDDs immutable then we will create a new RDD for the dataset without adding any new transformations. Ie., the same transformations will be performed for all the RDDs. Hence simplifies the lineage graph.

2. When an RDD is being created (new data are being written into it), can the data in the RDD be read for computing before the RDD is completely created?

No.
An RDD might have some kind of dependency with an another RDD so we should not use the data in RDD until its completely ready.
Also, the computations in RDD are evaluated as stages in a DAG called lazy evaluation. So spark waits for the computations and data to be ready so that it can optimize the steps for better performance.
3. “This allows them to efficiently provide fault tolerance by logging the transformations used to build a dataset (its lineage) rather than the actual data.” “To achieve fault tolerance efficiently, RDDs provide a restricted form of shared memory, based on coarse-grained transformations rather than fine-grained updates to shared state.” Why does using RDD help to provide efficient fault tolerance? or why does coarse-grained transformation help with the efficiency?

Course Grained transformations: the transformation is performed on the entire RDD instead of just small part of RDD.
This feature has an advantage. Since the RDD are immutable, each time a fine grained transformation is made to a part of RDD a new RDD is created. So for every fine grained transformation a new RDD is created which increases the size of the lineage graph and thus increase number of intermediate checkpoints in between RDD transformations. This increases disk IO. Hence course grained transformations are performed.

4. “In addition, programmers can call a persist method to indicate which RDDs they want to reuse in future operations.” What’s the consequence if a user does not explicitly request persistence of an RDD?

Spark by default keeps persistent RDDs in memory or spills it to disk if there is not enough RAM.

5. Explain Figure 1 about a lineage graph.

![Lineage Graph Diagram](image)

**Figure 1:** Lineage graph for the third query in our example.
Boxes represent RDDs and arrows represent transformations.

- lines is an RDD backed by an HDFS file
- The filter operation only reads the lines that start with ERROR and store it in an errors RDD.
- The next filter operation only selects the errors that contain HDFS into HDFS errors.
- The HDFS errors is then split using tab spaces and the 3rd element of each HDFS errors is extracted. Which can then be collected using a collect().