Ceph RGW Cache Prefetching for Batch Jobs

Xun Lin
Yang Qiao
Tianyi Tang
Gang Wei
Zhangyu Wan

Outline

- Directed acyclic graph(DAG) interpretation
  - RDD
  - Dataframe
- Integrate Spark and Ceph
  - S3a filesystem client
  - Configuration try

RDD(Resilient Distributed Dataset)

It is an immutable (read only) collection of partitioned data distributed on different machines. RDD enables in-memory computation on large clusters to speed up big data processing in a **fault tolerant** manner.

RDD uses **DAG (Directed Acyclic Graph)**. The vertices and edges in DAG represent the RDD and the operation to be applied on that RDD respectively.

DAG

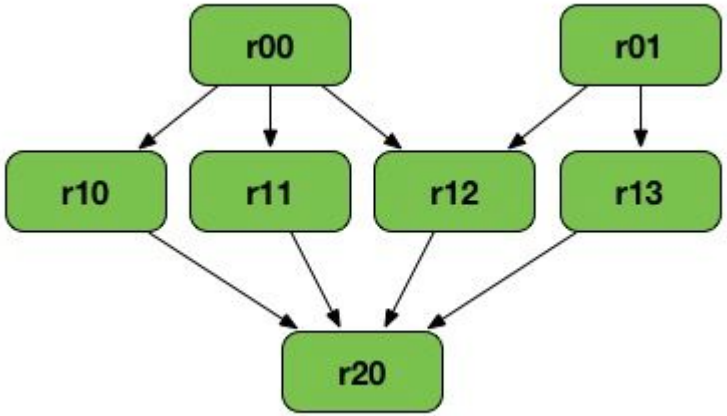
At high level, when any action is called on the RDD, Spark creates the DAG and submits it to the DAG scheduler.

The DAG scheduler divides operators into stages of tasks. A stage is comprised of tasks based on partitions of the input data. The DAG scheduler pipelines operators together. The final result of a DAG scheduler is a set of stages.

The Stages are passed on to the Task Scheduler. The task scheduler launches tasks via cluster manager (Spark Standalone/Yarn/Mesos).

DAG of RDDs

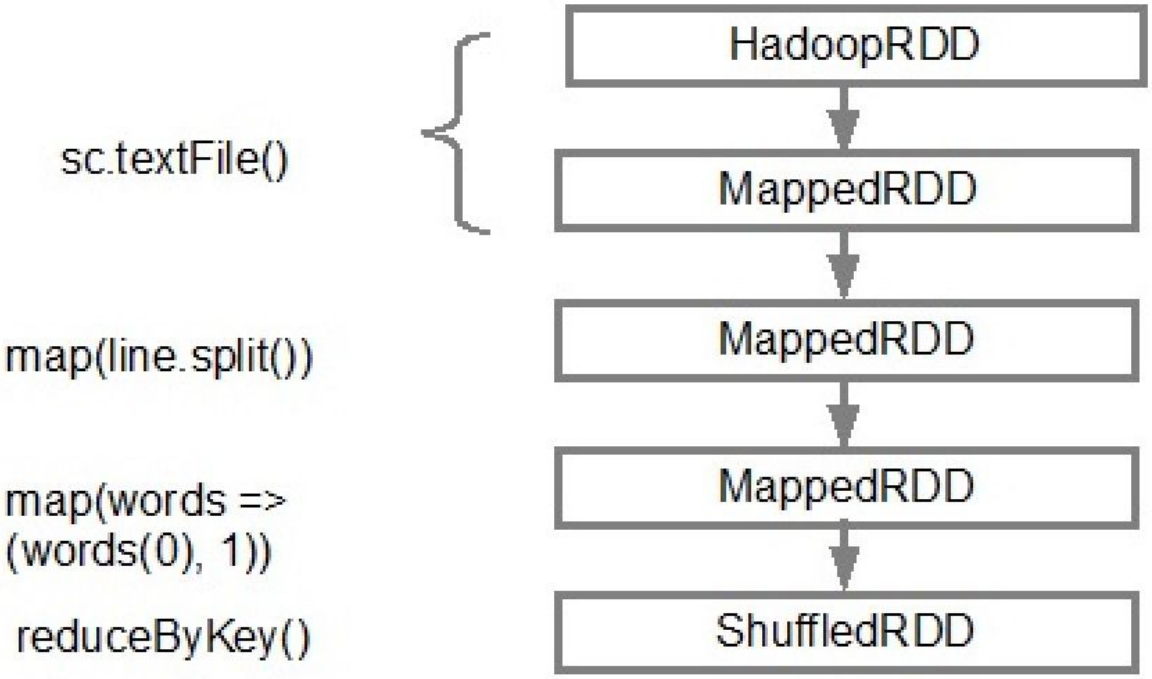
RDD Lineage - Logic execution plan



toDebugString method

```
(2) ShuffledRDD[6] at reduceByKey at <console>:25 []
    +-(2) MapPartitionsRDD[5] at map at <console>:24 []
    |  MapPartitionsRDD[4] at map at <console>:23 []
    |  log.txt MapPartitionsRDD[1] at textFile at <console>:21 []
    |  log.txt HadoopRDD[0] at textFile at <console>:21 []
```

Interpretation



Problem:

Lack of useful information

Dataframe

The DataFrame APIs organizes the data into named columns like a table in relational database. DataFrame specifies the operations needs to be performed on the distributed collection of the data. It is **immutable.**

Hive compatible: Compatible with Hive query language.

Schema support: Define a schema or read a schema from a data source

Type safety: Each row in a DataFrame is of object type

Dataframe explain method

scala> records.explain(extended = true)

== Parsed Logical Plan ==

== Analyzed Logical Plan ==

== Optimized Logical Plan ==

== Physical Plan ==

Problem: Get the datasource, not understandable, results are only in console.

S3a filesystem client

- S3n (S3 Native FileSystem): a native filesystem for reading and writing regular files on Amazon S3. (5GB limit)
- S3a: system uses Amazon's libraries to interact with S3 (successor to S3n)
- Users can use the S3 API to interact with Ceph storage infrastructure
- It leverage the might of the Amazon ecosystem, including all the SDKs and tools written to interact with Amazon S3.

Access control

1. Ceph support basic S3 access control
    a. access key
    b. secret key

2. Endpoint
    a. Users can talk to Ceph cluster instead of Amazon S3 by setting the fs.s3a.endpoint in the core-site.xml

Configuration try

Context :

- We haven’t got access key, secret key and endpoint to access Ceph
- Integration between Spark and Ceph is very similar to integration with S3.

SO Try S3 first...

Configuration try

Preparations (setup a role to access S3):

Step 1 : Log into AWS

Step 2 : From the AWS console go to the following options and create a user in for the demo in AWS

Identity and Access Management (IAM) --> Users --> Create New Users

Configuration try

Preparations (setup a role to access S3):

Step 3 : Make note of the credentials

awsAccessKeyId = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';

awsSecretAccessKey = 'yyyyyyyyyyyyyyyyyyyyyyyyyyyy';

Step 4 : Add the User to the Admin Group

Step 5 : Assign the Administration Access Policy to the User (admin)

Configuration try

1. **Load dependencies when running (AWS java sdk is required):**

./spark-shell --master yarn --jars
/Users/wang/Desktop/spark/hadoop-3.2.1-src/jars/hadoop-aws-2.7.3.jar,
/Users/wang/Desktop/spark/hadoop-3.2.1-src/jars/hadoop-auth-2.7.3.jar,
/Users/wang/Desktop/spark/hadoop-3.2.1-src/jars/aws-java-sdk-1.7.4.jar

Result: No error, but fail because SparkUI could not bind on port 4040.

Potential issue: There already have been a Spark cluster running. (but actually no)

Configuration try

2. Modify the spark-defaults.conf

we need to set some parameters in the configuration file for spark which is located at $SPA# spark-defaults.conf

spark.hadoop.fs.s3a.access.key=MY_ACCESS_KEY

spark.hadoop.fs.s3a.secret.key=MY_SECRET_KEY

spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem

Result: java.lang.RuntimeException: java.lang.ClassNotFoundException: Class org.apache.hadoop.fs.s3native.NativeS3FileSystem not found

Configuration try

3. Configuration within code

hadoopConf = sc._jsc.hadoopConfiguration()

hadoopConf.set(“fs.s3a.awsAccessKeyId”, “XXXXX”)

hadoopConf.set(“fs.s3a.awsSecretAccessKey”, “XXXXX”)

hadoopConf.set(“fs.s3a.endpoint”, “XXXXX”)

hadoopConf.set(“fs.s3a.impl”, “org.apache.hadoop.fs.s3a.S3AFileSystem”)

Result: java.lang.RuntimeException: java.lang.ClassNotFoundException: Class org.apache.hadoop.fs.s3native.NativeS3FileSystem not found

Next step

- Integration Spark and Ceph properly
- Interprete DAG to understandable format

Thank you!