二、Spark 算子和 RDD

作者：shiweichn

原文链接：https://ld246.com/article/1534818062072

来源网站：链滴

二、Spark 算子和RDD

Spark 的算子分为两类：

Transformation：转换 。Spark所有Transformation

Action　　：　动作 。Spark所有Action

Transformation延迟执行，Transformation会记录元数据信息，当计算任务触发Action时，才会被正计算。

创建 RDD 的两种方式

1.通过HDFS支持的文件系统创建RDD，创建后RDD里并没有真正要计算的数据(懒加载)，只记录了数据(数据路径)

2. 通过Scala集合或数组以并行化的方式创建RDD。　sc.parallelize(Array(1,2,3,4,5))

RDD.partitions.length 可以查看RDD的分区数量

RDD 的特点

- 具有一个分区列表，　一个分区肯定在某个机器上，但一个机器可能有多个分区
- 函数会作用于每一个分区上，同一个操作会在每个分区上都执行
- RDD 之间有一系列的依赖，算子是链式调用，每个算子都生成一个新的RDD。　后一个RDD会记前一个RDD的某些信息，所有当计算过程中前面的数据丢失，后面的RDD根据依赖关系重新恢复丢掉数据。
- 如果是Key-Value类型，会有一个分区器(Partitioner)。默认是hash-Partitioner。可以自定义分区。
- 记录RDD中每个分区位置的列表，为了移动计算，而不移动数据。例如，HDFS文件的块位置。

- Internally, each RDD is characterized by five main properties:
  - A list of partitions
  - A function for computing each split
  - A list of dependencies on other RDDs
  - Optionally, a Partitioner for key-value RDDs (e.g. to say that the RDD is hash-partitioned)
  - Optionally, a list of preferred locations to compute each split on (e.g. block locations for n HDFS file)

提交任务到集群

还是WordCount的例子。这次不使用spark-shell。在本地写好后，打出jar包，传到spark集群，然执行。

```
import org.apache.spark.{SparkConf, SparkContext}

/**
  * Created by Sweeney on 2017/11/7.
  */
object WordCount {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf().setAppName("WC")
    val sc = new SparkContext(conf) // 创建一个SparkContext
    sc.textFile(args(0))            // 从路径中读取数据
      .flatMap(_.split(" "))
      .map((_, 1))
      .reduceByKey(_ + _)
      .sortBy(_._2, false)
      .saveAsTextFile(args(1))      // 将结果报存到一个目录

    sc.stop()
  }
}
```

打出jar包后传到 Master

```
shiwei@ubuntu1:~/bigdata/jars$ pwd
/home/shiwei/bigdata/jars
shiwei@ubuntu1:~/bigdata/jars$ ls
WC-assembly-1.0.jar
```

通过spark-submit 来提交任务。

```
./bin/spark-submit --class WordCount --master spark://ubuntu1:7077 /home/shiwei/bigdata/
ars/WC-assembly-1.0.jar ./README.md ~/bigdata/results/WC.out
```

最终结果保存在 worker 机器上的 ~/bigdata/results/WC.out 目录下。而在Master上是不会有的因为真正执行操作的是 worker机器。所以这里使用 HDFS要好一点，可以指定输出结果到哪个节点。

踩的坑

第一个坑是我在WordCount项目中引入了 2.11 的 Spark-core，但是项目Scala SDK使用的是 2.12 所以在上传到集群提交执行的时候报错了。最后将SDK版本调整为 2.11 才解决问题！！！

报错如下：

```
Caused by: java.lang.NoSuchMethodError: scala.Predef$.refArrayOps([Ljava/lang/Object;)[Ljav
/lang/Object;
    at WordCount$.$anonfun$main$1(WordCount.scala:10)
    at WordCount$.$anonfun$main$1$adapted(WordCount.scala:10)
```

```
at scala.collection.Iterator$$anon$12.nextCur(Iterator.scala:434)
at scala.collection.Iterator$$anon$12.hasNext(Iterator.scala:440)
at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:408)
at org.apache.spark.util.collection.ExternalSorter.insertAll(ExternalSorter.scala:191)
at org.apache.spark.shuffle.sort.SortShuffleWriter.write(SortShuffleWriter.scala:63)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:96)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:53)
at org.apache.spark.scheduler.Task.run(Task.scala:108)
at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:335)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
at java.lang.Thread.run(Thread.java:748)
```

第二个坑是提交执行的时候，给输出结果指定目录。指定的是~/bigdata/results/WC.out，成功运行毕后再 Master上这个目录下找不到输出结果，重试了几次，鬼斧神差的去worker上这个目录看了下果然每个worker的这个目录都有结果。唯独执行提交的master上没有。后来想了想，这样应该是正的，因为任务都是在worker上执行的。