Spark Streaming中空batches处理的两种方法

Spark Streaming是近实时(near real time)的小批处理系统。对给定的时间间隔(interval)，Spark Streaming生成新的batch并对它进行一些处理。每个batch中的数据都代表一个RDD，但是如果一些batch中没有数据会发生什么事情呢？Spark Streaming将会产生EmptyRDD的RDD，它的定义如下：

```
package org.apache.spark.rdd

import scala.reflect.ClassTag

import org.apache.spark.{Partition, SparkContext, TaskContext}

/**
 * An RDD that has no partitions and no elements.
 */
private[spark] class EmptyRDD[T: ClassTag](sc: SparkContext) extends RDD[T](sc, Nil) {

  override def getPartitions: Array[Partition] = Array.empty

  override def compute(split: Partition, context: TaskContext): Iterator[T] = {
    throw new UnsupportedOperationException("empty RDD")
  }
}
```

可以看到这个RDD并不对任何父RDD有依赖关系，我们不能调用compute方法计算每个分区的数据。EmptyRDD的存在是为了保证Spark Streaming中多个batch的处理是一致的。但是存在EmptyRDD有时候会产生一些问题，比如：如果你想将接收到的Streaming数据写入HDFS中：

```
val ssc = new StreamingContext(args(0),"iteblog",Seconds(10))
val socketStream = ssc.socketTextStream("www.iteblog.com",8888)
val outputDir = args(1)

socketStream.foreachRDD(rdd => {
  rdd.saveAsTextFile(outputDir)
})
```

当你调用foreachRDD的时候如果当前rdd是EmptyRDD，这样会导致在HDFS上生成大量的空文件！这肯定不是我们想要的，我们只想在存在数据的时候才写HDFS，我们可以通过以下的两种方法来避免这种情况：

```
socketStream.foreachRDD(rdd => {
  if(rdd.count() != 0){
   rdd.saveAsTextFile(outputDir)
  }
})
```

EmptyRDD的count肯定是0，所以这样可以避免写空文件，或者我们也可以用下面方法解决：

```
socketStream.foreachRDD(rdd => {
  if(!rdd.partitions.isEmpty){
   rdd.saveAsTextFile(outputDir)
  }
})
```

EmptyRDD是没有分区的，所以调用partitions.isEmpty是true。这样也可以解决上述问题。

虽然上面两种方法都可以解决这个问题，但是推荐使用第二种方法。因为第一种方法调用了RDD的count函数，这是一个Action，会触发一次Job的计算，当你的数据量比较大的时候，这可能会带来性能方面的一些影响；而partitions.isEmpty是不需要触发Job的。

不过如果你使用的是Sprk 1.3.0，你可以调用isEmpty函数来判断一个RDD是否为空，这个函数是在SPARK-5270引入的。