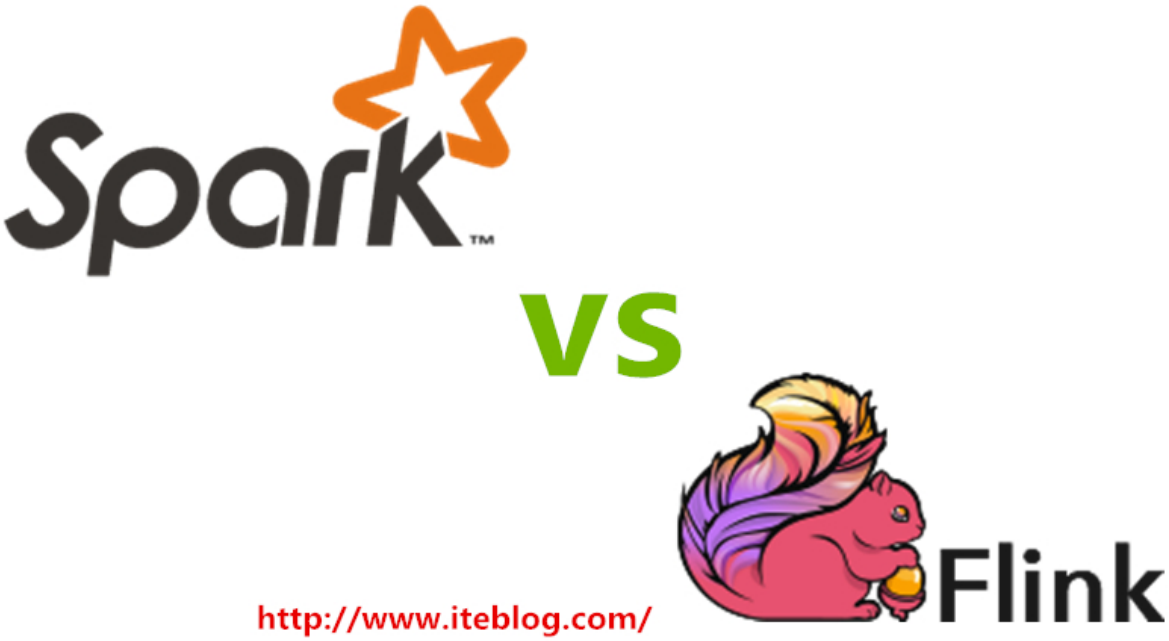
Apache Flink vs Apache Spark

我们是否还需要另外一个新的数据处理引擎？当我第一次听到Flink的时候这是我是非常怀疑的。在大数据领域，现在已经不缺少数据处理框架了，但是没有一个框架能够完全满足不同的处理需求。自从Apache Spark出现后，貌似已经成为当今把大部分的问题解决得最好的框架了，所以我对另外一款解决类似问题的框架持有很强烈的怀疑态度。

不过因为好奇，我花费了数个星期在尝试了解Flink。一开始仔细看了Flink的几个例子，感觉和Spark非常类似，心理就倾向于认为Flink又是一个模仿Spark的框架。但是随着了解的深入，这些API体现了一些Flink的新奇的思路，这些思路还是和Spark有着比较明显的区别的。我对这些思路有些着迷了，所以花费了更多的时间在这上面。

Flink中的很多思路，例如内存管理，dataset API都已经出现在Spark中并且已经证明 这些思路是非常靠谱的。所以，深入了解Flink也许可以帮助我们分布式数据处理的未来之路是怎样的。

在后面的文章里，我会把自己作为一个Spark开发者对Flink的第一感受写出来。因为我已经在Spark上干了2年多了，但是只在Flink上接触了2到3周，所以必然存在一些bias，所以大家也带着怀疑和批判的角度来看这篇文章吧。



Apache Flink是什么

Flink是一款新的大数据处理引擎，目标是统一不同来源的数据处理。这个目标看起来和Spark和类似。没错，Flink也在尝试解决Spark在解决的问题。这两套系统都在尝试建立一个统一的平

台可以运行批量，流式，交互式，图处理，机器学习等应用。所以，Flink和Spark的目标差别并不大，他们最主要的区别在于实现的细节，后面我会重点从不同的角度对比这两者。

Apache Spark vs Apache Flink

1、抽象 Abstraction

Spark中，对于批处理我们有RDD,对于流式，我们有DStream，不过内部实际还是RDD.所以所有的数据表示本质上还是RDD抽象。后面我会重点从不同的角度对比这两者。在Flink中，对于批处理有DataSet，对于流式我们有DataStreams。看起来和Spark类似，他们的不同点在于：

（一）DataSet在运行时是表现为运行计划(runtime plans)的

在Spark中，RDD在运行时是表现为java objects的。通过引入Tungsten，这块有了些许的改变。但是在Flink中是被表现为logical plan(逻辑计划)的，听起来很熟悉？没错，就是类似于Spark中的dataframes。所以在Flink中你使用的类Dataframe api是被作为第一优先级来优化的。但是相对来说在Spark RDD中就没有了这块的优化了。

Flink中的Dataset，对标Spark中的Dataframe，在运行前会经过优化。在Spark 1.6，dataset API已经被引入Spark了，也许最终会取代RDD 抽象。

(二）Dataset和DataStream是独立的API

在Spark中，所有不同的API，例如DStream，Dataframe都是基于RDD抽象的。但是在Flink中，Dataset和DataStream是同一个公用的引擎之上两个独立的抽象。所以你不能把这两者的行为合并在一起操作，当然，Flink社区目前在朝这个方向努力(https://issues.apache.org/jira/browse/Flink-2320)，但是目前还不能轻易断言最后的结果。

2、内存管理

一直到1.5版本，Spark都是试用java的内存管理来做数据缓存，明显很容易导致OOM或者gc。所以从1.5开始，Spark开始转向精确的控制内存的使用，这就是tungsten项目了。

而Flink从第一天开始就坚持自己控制内存试用。这个也是启发了Spark走这条路的原因之一。Flink除了把数据存在自己管理的内存以外，还直接操作二进制数据。在Spark中，从1.5开始，所有的dataframe操作都是直接作用在tungsten的二进制数据上。

3、语言实现

Spark是用scala来实现的，它提供了Java，Python和R的编程接口。Flink是java实现的，当然同样提供了Scala API
所以从语言的角度来看，Spark要更丰富一些。因为我已经转移到scala很久了，所以不太清楚这两者的java api实现情况。

4、API

Spark和Flink都在模仿scala的collection API.所以从表面看起来，两者都很类似。下面是分别用RDD和DataSet API实现的word count

```
// Spark wordcount
object WordCount {

  def main(args: Array[String]) {
    val env = new SparkContext("local","wordCount")
    val data = List("hi","how are you","hi")
    val dataSet = env.parallelize(data)
    val words = dataSet.flatMap(value => value.split("₩₩s+"))
    val mappedWords = words.map(value => (value,1))
    val sum = mappedWords.reduceByKey(_+_)
    println(sum.collect())
  }
}

// Flink wordcount
object WordCount {

def main(args: Array[String]) {
      val env = ExecutionEnvironment.getExecutionEnvironment
      val data = List("hi","how are you","hi")
      val dataSet = env.fromCollection(data)
      val words = dataSet.flatMap(value => value.split("₩₩s+"))
      val mappedWords = words.map(value => (value,1))
      val grouped = mappedWords.groupBy(0)
      val sum = grouped.sum(1)
      println(sum.collect())
    }
}
```

不知道是偶然还是故意的，API都长得很像，这样很方便开发者从一个引擎切换到另外一个引擎。我感觉以后这种Collection API会成为写data pipeline的标配。

5、Steaming

Spark把streaming看成是更快的批处理，而Flink把批处理看成streaming的special case。这里面的思路决定了各自的方向，其中两者的差异点有如下这些：

实时 vs 近实时的角度

Flink提供了基于每个事件的流式处理机制，所以可以被认为是一个真正的流式计算。它非常像storm的model。
而Spark，不是基于事件的粒度，而是用小批量来模拟流式，也就是多个事件的集合。所以Spark被认为是近实时的处理系统。

Spark streaming 是更快的批处理，而Flink Batch是有限数据的流式计算。
虽然大部分应用对准实时是可以接受的，但是也还是有很多应用需要event level的流式计算。这些应用更愿意选择storm而非Spark streaming，现在，Flink也许是一个更好的选择。

流式计算和批处理计算的表示

Spark对于批处理和流式计算，都是用的相同的抽象：RDD，这样很方便这两种计算合并起来表示。而Flink这两者分为了DataSet和DataStream，相比Spark，这个设计算是一个糟糕的设计。

对 windowing 的支持

因为Spark的小批量机制，Spark对于windowing的支持非常有限。只能基于process time，且只能对batches来做window。而Flink对window的支持非常到位，且Flink对windowing API的支持是相当给力的，允许基于process time,data time,record 来做windowing。我不太确定Spark是否能引入这些API，不过到目前为止，Flink的windowing支持是要比Spark好的。Steaming这部分Flink胜

6、SQL interface

目前Spark-sql是Spark里面最活跃的组件之一，Spark提供了类似Hive的sql和Dataframe这种DSL来查询结构化数据，API很成熟，在流式计算中使用很广，预计在流式计算中也会发展得很快。至于Flink，到目前为止，Flink Table API只支持类似DataFrame这种DSL，并且还是处于beta状态，社区有计划增加SQL的interface，但是目前还不确定什么时候才能在框架中用上。所以这个部分，Spark胜出。

7、外部数据源的整合

Spark的数据源 API是整个框架中最好的，支持的数据源包括NoSql db,parquet,ORC等，并且支持一些高级的操作，例如predicate push down。Flink目前还依赖map/reduce InputFormat来做数据源聚合。这一场Spark胜

8、Iterative processing

Spark对机器学习的支持较好，因为可以在Spark中利用内存cache来加速机器学习算法。但是大部分机器学习算法其实是一个有环的数据流，但是在Spark中，实际是用无环图来表示的，一般的分布式处理引擎都是不鼓励试用有环图的。但是Flink这里又有点不一样，Flink支持在runtime中的有环数据流，这样表示机器学习算法更有效而且更有效率。这一点Flink胜出。

9、Stream as platform vs Batch as Platform

Spark诞生在Map/Reduce的时代，数据都是以文件的形式保存在磁盘中，这样非常方便做容错处理。Flink把纯流式数据计算引入大数据时代，无疑给业界带来了一股清新的空气。这个idea非常类似akka-streams这种。成熟度目前的确有一部分吃螃蟹的用户已经在生产环境中使用Flink了，不过从我的眼光来看，Flink还在发展中，还需要时间来成熟。

结论

目前Spark相比Flink是一个更为成熟的计算框架，但是Flink的很多思路很不错，Spark社区也意识到了这一点，并且逐渐在采用Flink中的好的设计思路，所以学习一下Flink能让你了解一下Streaming这方面的更迷人的思路。

作者：billen pan
链接：https://www.zhihu.com/question/30151872/answer/82554774
来源：知乎