Apache Spark常见的三大误解

最近几年关于Apache
Spark框架的声音是越来越多，而且慢慢地成为大数据领域的主流系统。最近几年Apache
Spark和Apache Hadoop的Google趋势可以证明这一点：

如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

上图已经明显展示出最近五年，Apache Spark越来越受开发者们的欢迎，大家通过Google搜索更多关于Spark的信息。然而很多人对Apache
Spark的认识存在误解，在这篇文章中，将介绍我们对Apache
Spark的几个主要的误解，以便给那些想将Apache
Spark应用到其系统中的人作为参考。这里主要包括以下几个方面：

- Spark是一种内存技术；
- Spark要比Hadoop快 10x-100x；
- Spark在数据处理方面引入了全新的技术

误解一：Spark是一种内存技术

大家对Spark最大的误解就是其是一种内存技术（in-memory technology）。其实不是这样的！没有一个Spark开发者正式说明这个，这是对Spark计算过程的误解。

我们从头开始说明。什么样的技术才能称得上是内存技术？在我看来，就是允许你将数据持久化（persist）在RAM中并有效处理的技术。然而Spark并不具备将数据数据存储在RAM的选项，虽然我们都知道可以将数据存储在HDFS, Tachyon, HBase,

Cassandra等系统中，但是不管是将数据存储在磁盘还是内存，都没有内置的持久化代码（native persistence code）。它所能做的事就是缓存（cache）数据，而这个并不是数据持久化（persist）。已经缓存的数据可以很容易地被删除，并且在后期需要时重新计算。

但是即使有这些信息，仍然有些人还是会认为Spark就是一种基于内存的技术，因为Spark是在内存中处理数据的。这当然是对的，因为我们无法使用其他方式来处理数据。操作系统中的API都只能让你把数据从块设备加载到内存，然后计算完的结果再存储到块设备中。我们无法直接在HDD设备上计算；所以现代系统中的所有处理基本上都是在内存中进行的。

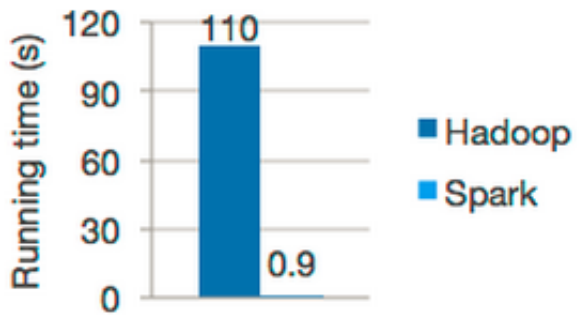
虽然Spark允许我们使用内存缓存以及LRU替换规则，但是你想想现在的RDBMS系统，比如Oracle 和 PostgreSQL，你认为它们是如何处理数据的？它们使用共享内存段（shared memory segment）作为table pages的存储池，所有的数据读取以及写入都是通过这个池的，这个存储池同样支持LRU替换规则；所有现代的数据库同样可以通过LRU策略来满足大多数需求。但是为什么我们并没有把Oracle 和 PostgreSQL称作是基于内存的解决方案呢？你再想想Linux IO，你知道吗？所有的IO操作也是会用到LRU缓存技术的。

你现在还认为Spark在内存中处理所有的操作吗？你可能要失望了。比如Spark的核心：shuffle，其就是将数据写入到磁盘的。如果你再SparkSQL中使用到group by语句，或者你将RDD转换成PairRDD并且在其之上进行一些聚合操作，这时候你强制让Spark根据key的哈希值将数据分发到所有的分区中。shuffle的处理包括两个阶段：map 和 reduce。Map操作仅仅根据key计算其哈希值，并将数据存放到本地文件系统的不同文件中，文件的个数通常是reduce端分区的个数；Reduce端会从 Map端拉取数据，并将这些数据合并到新的分区中。所有如果你的RDD有M个分区，然后你将其转换成N个分区的PairRDD，那么在shuffle阶段将会创建 M*N 个文件！虽然目前有些优化策略可以减少创建文件的个数，但这仍然无法改变每次进行shuffle操作的时候你需要将数据先写入到磁盘的事实！

所以结论是：Spark并不是基于内存的技术！它其实是一种可以有效地使用内存LRU策略的技术。

误解二：Spark要比Hadoop快 10x-100x

相信大家在Spark的官网肯定看到了如下所示的图片



Logistic regression in Hadoop and Spark

如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

这个图片是分别使用 Spark 和 Hadoop 运行逻辑回归（Logistic Regression）机器学习算法的运行时间比较，从上图可以看出Spark的运行速度明显比Hadoop快上百倍！但是实际上是这样的吗？大多数机器学习算法的核心部分是什么？其实就是对同一份数据集进行相同的迭代计算，而这个地方正是Spark的LRU算法所骄傲的地方。当你多次扫描相同的数据集时，你只需要在首次访问时加载它到内存，后面的访问直接从内存中获取即可。这个功能非常的棒！但是很遗憾的是，官方在使用Hadoop运行逻辑回归的时候很大可能没有使用到HDFS的缓存功能，而是采用极端的情况。如果在Hadoop中运行逻辑回归的时候采用到HDFS缓存功能，其表现很可能只会比Spark差3x-4x，而不是上图所展示的一样。

根据经验，企业所做出的基准测试报告一般都是不可信的！一般独立的第三方基准测试报告是比较可信的，比如：TPC-
H。他们的基准测试报告一般会覆盖绝大部分场景，以便真实地展示结果。

一般来说，Spark比MapReduce运行速度快的原因主要有以下几点：

- task启动时间比较快，Spark是fork出线程；而MR是启动一个新的进程；
- 更快的shuffles，Spark只有在shuffle的时候才会将数据放在磁盘，而MR却不是。
- 更快的工作流：典型的MR工作流是由很多MR作业组成的，他们之间的数据交互需要把数据持久化到磁盘才可以；而Spark支持DAG以及pipelining，在没有遇到shuffle完全可以不把数据缓存到磁盘。
- 缓存：虽然目前HDFS也支持缓存，但是一般来说，Spark的缓存功能更加高效，特别是在SparkSQL中，我们可以将数据以列式的形式储存在内存中。

所有的这些原因才使得Spark相比Hadoop拥有更好的性能表现；在比较短的作业确实能快上100倍，但是在真实的生产环境下，一般只会快 2.5x ~ 3x！

误解三：Spark在数据处理方面引入了全新的技术

事实上，Spark并没有引入任何革命性的新技术！其擅长的LRU缓存策略和数据的pipelining处理其实在MPP数据库中早就存在！Spark做出重要的一步是使用开源的方式来实现它！并且企业可以免费地使用它。大部分企业势必会选择开源的Spark技术，而不是付费的MPP技术。