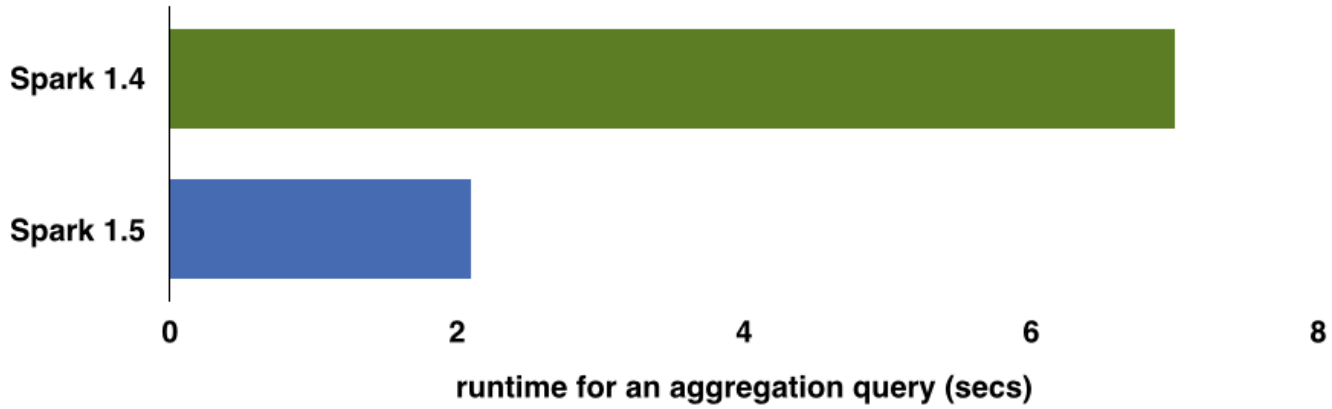
Apache Spark 1.5新特性介绍

Apache Spark社区刚刚发布了1.5版本，大家一定想知道这个版本的主要变化，这篇文章告诉你答案。

DataFrame执行后端优化（Tungsten第一阶段）

DataFrame可以说是整个Spark项目最核心的部分，在1.5这个开发周期内最大的变化就是Tungsten项目的第一阶段已经完成。主要的变化是由Spark自己来管理内存而不是使用JVM，这样可以避免JVM GC带来的性能损失。内存中的Java对象被存储成Spark自己的二进制格式，计算直接发生在二进制格式上，省去了序列化和反序列化时间。同时这种格式也更加紧凑，节省内存空间，而且能更好的估计数据量大小和内存使用情况。如果大家对这部分的代码感兴趣，可以在源代码里面搜索那些Unsafe开头的类即可。在1.4版本只提供UnsafeShuffleManager等少数功能，剩下的大部分都是1.5版本新加入的功能。
其他优化还包括默认使用code generation; cache-aware算法对join, aggregation, shuffle, sorting的增强；window function性能的提高等。

那么性能到底能提升多少呢？可以参考DataBricks给出的这个例子。这是一个16 million行的记录，有1 million的组合键的aggregation查询分别使用Spark 1.4和1.5版本的性能对比，在这个测试中都是使用的默认配置。



那么如果我们想自己测试下Tungsten第一阶段的性能改如何测试呢？Spark 1.4以前的版本中spark.sql.codegen, spark.sql.unsafe.enabled等几个参数在1.5版本里面合并成spark.sql.tungsten.enabled并默认为true，只需要修改这一个参数就可以配置是否开启tungsten优化（默认是开启的）。

DataFrame/SQL/Hive

在DataFrame API方面，实现了新的聚合函数接口AggregateFunction2以及7个相应的build-in的聚合函数，同时基于新接口实现了相应的UDAF接口。新的聚合函数接口把一个聚合函数拆解为三个动作: initialize/update/merge，然后用户只需要定义其中的逻辑既可以实现不同的聚合函数功能。Spark的这个新的聚合函数实现方法和Impala里面非常类似。

Spark内置的expression function
得到了很大的增强，实现了100多个这样的常用函数，例如string, math, unix_timestamp, from_unixtime, to_date等。同时在处理NaN值的一些特性也在增强，例如 NaN = Nan 返回true；NaN大于任何其他值等约定都越来越符合SQL界的规则了。

用户可以在执行join操作的时候指定把左边的表或者右边的表broadcast出去，因为基于cardinality的估计并不是每次都是很准的，如果用户对数据了解可以直接指定哪个表更小从而被broadcast出去。
Hive模块最大的变化是支持连接Hive 1.2版本的metastore，同时支持metastore partition pruning（通过spark.sql.hive.metastorePartitionPruning=true开启，默认为false）。因为很多公司的Hive集群都升级到了1.2以上，那么这个改进对于需要访问Hive元数据的Spark集群来说非常重要。Spark 1.5支持可以连接Hive 0.13, 0.14, 1.0/0.14.1, 1.1, 1.2的metastore。
在External Data
Source方面，Parquet的支持有了很大的加强。Parquet的版本升级到1.7；更快的metadata discovery和schema merging；同时能够读取其他工具或者库生成的非标准合法的parquet文件；以及更快更鲁棒的动态分区插入。

由于Parquet升级到1.7，原来的一个重要bug被修复，所以Spark SQL的Filter Pushdown默认改为开启状态（spark.sql.parquet.filterPushdown=true），能够帮助查询过滤掉不必要的IO。

Spark 1.5可以通过指定spark.sql.parquet.output.committer.class参数选择不同的output committer类，默认是org.apache.parquet.hadoop.ParquetOutputCommitter，用户可以继承这个类实现自己的output committer。由于HDFS和S3这两种文件存储系统的区别，如果需要向S3里面写入数据，可以使用DirectParquetOutputCommitter，能够有效提高写效率，从而加快Job执行速度。

另外还有一些改动，包括：StructType支持排序功能；TimestampType的精度减小到1us；Spark现在的checkpoint是基于HDFS的，从1.5版本开始支持基于memory和local disk的checkpoint。这种类型的checkpoint性能更快，虽然不如基于HDFS的可靠，但是对于迭代型机器学习运算还是很有帮助的。

机器学习MLlib

MLlib最大的变化就是从一个机器学习的library开始转向构建一个机器学习工作流的系统，这些变化发生在ML包里面。MLlib模块下现在有两个包：MLlib和ML。ML把整个机器学习的过程抽象成Pipeline，一个Pipeline是由多个Stage组成，每个Stage是Transformer或者Estimator。

以前机器学习工程师要花费大量时间在training model之前的feature的抽取、转换等准备工作。ML提供了多个Transformer，极大提高了这些工作的效率。在1.5版本之后，已经有了25+个feature transformer，其中CountVectorizer, Discrete Cosine Transformation, MinMaxScaler,

NGram, PCA, RFormula, StopWordsRemover, and VectorSlicer这些feature transformer都是1.5版本新添加的，做机器学习的朋友可以看看哪些满足你的需求。
这里面的一个亮点就是RFormula的支持，目标是使用户可以把原来用R写的机器学习程序（目前只支持GLM算法）不用修改直接搬到Spark平台上来执行。不过目前只支持集中简单的R公式(包括'.', '~', '+'和 '-')，社区在接下来的版本中会增强这项功能。

另外越来越多的算法也作为Estimator搬到了ML下面，在1.5版本中新搬过来的有Naive Bayes, K-means, Isotonic Regression等。大家不要以为只是简单的在ML下面提供一个调用相应算法的API，这里面变换还是挺多的。例如Naive Bayes原来的模型分别用Array[Double]和Array[Array[Double]]来存储pi和theta，而在ML下面新的API里面使用的是Vector和Matrix来存储。从这也可以看出，新的ML框架下所有的数据源都是基于DataFrame，所有的模型也尽量都基于Spark的数据类型表示。在ML里面的public
API下基本上看不到对RDD的直接操作了，这也与Tungsten项目的设计目标是一致的。

除了这些既有的算法在ML API下的实现，ML里面也增加了几个新算法：
1、MultilayerPerceptronClassifier(MLPC) 这是一个基于前馈神经网络的分类器，它是一种在输入层与输出层之间含有一层或多层隐含结点的具有正向传播机制的神经网络模型，中间的节点使用sigmoid (logistic)函数，输出层的节点使用softmax函数。输出层的节点的数目表示分类器有几类。MLPC学习过程中使用BP算法，优化问题抽象成logistic loss function并使用L-BFGS进行优化。
2、MLlib包里面增加了一个频繁项挖掘算法PrefixSpan，AssociationRules能够把FreqItemset生成关联式规则。
3、在MLlib的统计包里面实现了Kolmogorov–Smirnov检验，用以检验两个经验分布是否不同或一个经验分布与另一个理想分布是否不同。

另外还有一些现有算法的增强：LDA算法，决策树和ensemble算法，GMM算法：
1、ML里面的多个分类模型现在都支持预测结果的概率而不像过去只支持预测结果，像LogisticRegressionModel, NaiveBayesModel, DecisionTreeClassificationModel, RandomForestClassificationModel, GBTClassificationModel等，分别使用predictRaw, predictProbability, predict分别可以得到原始预测、概率预测和最后的分类预测。同时这些分类模型也支持通过设置thresholds指定各个类的阈值。
2、RandomForestClassificationModel和RandomForestRegressionModel模型都支持输出feature importance
3、GMM EM算法实现了当feature维度或者cluster数目比较大的时候的分布式矩阵求逆计算。实验表明当feature维度>30，cluster数目>10的时候，这个优化性能提升明显。

对于LinearRegressionModel和LogisticRegressionModel实现了LinearRegressionTrainingSummary和LogisticRegressionTrainingSummary用来记录模型训练过程中的一些统计指标。
1.5版本的Python API也在不断加强，越来越多的算法和功能的Python API基本上与Scala API对等了。此外在tuning和evaluator上也有增强。

其他

从1.5开始，Standalone, YARN和Mesos三种部署方式全部支持了动态资源分配。
SparkR支持运行在YARN集群上，同时DataFrame的函数也提供了一些R风格的别名，可以降

低熟悉R的用户的迁移成本。

在Streaming和Graphx方面也有非常大的改进，在这里不在一一赘述，详细可以参考release note。

本文转载自：http://weibo.com/p/1001603885229470280066