Spark 1.2.0发布啦

Spark 1.2.0于美国时间2014年12月18日发布，Spark 1.2.0兼容Spark 1.0.0和1.1.0，也就是说不需要修改代码即可用，很多默认的配置在Spark 1.2发生了变化

1、spark.shuffle.blockTransferService由nio改成netty
2、spark.shuffle.manager由hash改成sort
3、在PySpark中，默认的batch size改成0了，
4、Spark SQL方面做的修改：
spark.sql.parquet.cacheMetadata: false -> true
spark.sql.parquet.compression.codec: snappy -> gzip
spark.sql.hive.convertMetastoreParquet: false -> true
spark.sql.inMemoryColumnarStorage.compressed: false -> true
spark.sql.inMemoryColumnarStorage.batchSize: 1000 -> 10000
spark.sql.autoBroadcastJoinThreshold: 10000 -> 10485760 (10 MB)

详情：

我很高兴地公布Spark 1.2.0已经发布了。Spark 1.2.0是1.X线上的第三个版本。目前这个版本是最新的(废话，今天发布肯定是最新的)。有172位开发这参与。

该版本的core模块在操作和表现等方面带来了很大的提升。其中包括了特地为传输非常大的shuffle而设计的network传输子系统。Spark SQL模块引进了操作外部数据源的API，对Hive 0.13的支持，动态分区的支持以及固定精度的decimal类型。MLlib模块添加了新的pipeline-oriented package (spark.ml) for composing multiple algorithms。Spark Streaming添加了Python API,为容错而开发的写前日志。最后，GraphX 从alpha
版升级到稳定版，并且在性能方面有所提升。

Spark 1.2.0 is the third release on the 1.X line. This release brings performance and usability improvements in Spark's core engine, a major new API for MLlib, expanded ML support in Python, a fully H/A mode in Spark Streaming, and much more. GraphX has seen major performance and API improvements and graduates from an alpha component. Spark 1.2 represents the work of 172 contributors from more than 60 institutions in more than 1000 individual patches.

To download Spark 1.2 visit the downloads page.

Spark Core

In 1.2 Spark core upgrades to major subsystems to improve the performance and stability of very large scale shuffles. The first is Spark's communication manager used during bulk transfers, which upgrades to a netty-based implementation. The second is Spark's shuffle mechanism, which upgrades to the "sort based" shuffle initially released in Spark 1.1. These

both improve the performance and stability of very large scale shuffles. Spark also adds an elastic scaling mechanism designed to improve cluster utilization during long running ETL-style jobs. This is currently supported on YARN and will make its way to other cluster managers in future versions. Finally, Spark 1.2 adds support for Scala 2.11. For instructions on building for Scala 2.11 see the build documentation.

Spark Streaming
This release includes two major feature additions to Spark's streaming library, a Python API and a write ahead log for full driver H/A. The Python API covers almost all the DStream transformations and output operations. Input sources based on text files and text over sockets are currently supported. Support for Kafka and Flume input streams in Python will be added in the next release. Second, Spark streaming now features H/A driver support through a write ahead log (WAL). In Spark 1.1 and earlier, some buffered (received but not yet processed) data can be lost during driver restarts. To prevent this Spark 1.2 adds an optional WAL, which buffers received data into a fault-tolerant file system (e.g. HDFS). See the streaming programming guide for more details.

MLLib
Spark 1.2 previews a new set of machine learning API's in a package called spark.ml that supports learning pipelines, where multiple algorithms are run in sequence with varying parameters. This type of pipeline is common in practical machine learning deployments. The new ML package uses Spark's SchemaRDD to represent ML datasets, providing directly interoperability with Spark SQL. In addition to the new API, Spark 1.2 extends decision trees with two tree ensemble methods: random forests and gradient-boosted trees, among the most successful tree-based models for classification and regression. Finally, MLlib's Python implementation receives a major update in 1.2 to simplify the process of adding Python APIs, along with better Python API coverage.

Spark SQL
In this release Spark SQL adds a new API for external data sources. This API supports mounting external data sources as temporary tables, with support for optimizations such as predicate pushdown. Spark's Parquet and JSON bindings have been re-written to use this API and we expect a variety of community projects to integrate with other systems and formats during the 1.2 lifecycle.

Hive integration has been improved with support for the fixed-precision decimal type and Hive 0.13. Spark SQL also adds dynamically partitioned inserts, a popular Hive feature. An internal re-architecting around caching improves the performance and semantics of caching SchemaRDD instances and adds support for statistics-based partition pruning for cached data.

GraphX
In 1.2 GraphX graduates from an alpha component and adds a stable API. This means applications written against GraphX are guaranteed to work with future Spark versions without code changes. In 1.2 GraphX graduates from an alpha component and adds a stable API. This means applications written against GraphX are guaranteed to work with future Spark versions without code changes. A new core API, aggregateMessages, is introduced to replace

the now deprecated mapReduceTriplet API. The new aggregateMessages API features a more imperative programming model and improves performance. Some early test users found 20% – 1X performance improvement by switching to the new API.

In addition, Spark now supports graph checkpointing and lineage truncation which are necessary to support large numbers of iterations in production jobs. Finally, a handful of performance improvements have been added for PageRank and graph loading.

Other Notes
PySpark's sort operator now supports external spilling for large datasets.
PySpark now supports broadcast variables larger than 2GB and performs external spilling during sorts.
Spark adds a job-level progress page in the Spark UI, a stable API for progress reporting, and dynamic updating of output metrics as jobs complete.
Spark now has support for reading binary files for images and other binary formats.
Upgrading to Spark 1.2
Spark 1.2 is binary compatible with Spark 1.0 and 1.1, so no code changes are necessary. This excludes API's marked explicitly as unstable. Spark changes default configuration in a handful of cases for improved performance. Users who want to preserve identical configurations to Spark 1.1 can roll back these changes.

spark.shuffle.blockTransferService has been changed from nio to netty
spark.shuffle.manager has been changed from hash to sort
In PySpark, the default batch size has been changed to 0, which means the batch size is chosen based on the size of object. Pre-1.2 behavior can be restored using SparkContext([... args...], batchSize=1024).
Spark SQL has changed the following defaults:
spark.sql.parquet.cacheMetadata: false -> true
spark.sql.parquet.compression.codec: snappy -> gzip
spark.sql.hive.convertMetastoreParquet: false -> true
spark.sql.inMemoryColumnarStorage.compressed: false -> true
spark.sql.inMemoryColumnarStorage.batchSize: 1000 -> 10000
spark.sql.autoBroadcastJoinThreshold: 10000 -> 10485760 (10 MB)
Known Issues

A few smaller bugs did not make the release window. They will be fixed in Spark 1.2.1:

Netty shuffle does not respect secured port configuration. Work around – revert to nio shuffle: SPARK-4837
java.io.FileNotFound exceptions when creating EXTERNAL hive tables. Work around – set hive.stats.autogather = false. SPARK-4892.
Exception PySpark zip function on textfile inputs: SPARK-4841
MetricsServlet not properly initialized: SPARK-4595

本文链接: 【】（）