Spark性能优化：资源调优篇

Spark　2016-05-05　7287　6评论　下载为PDF　为什么不允许复制

《Spark性能优化：开发调优篇》
《Spark性能优化：资源调优篇》
《Spark性能优化：数据倾斜调优》
《Spark性能优化：shuffle调优》

在开发完Spark作业之后，就该为作业配置合适的资源了。Spark的资源参数，基本都可以在spark-submit命令中作为参数设置。很多Spark初学者，通常不知道该设置哪些必要的参数，以及如何设置这些参数，最后就只能胡乱设置，甚至压根儿不设置。资源参数设置的不合理，可能会导致没有充分利用集群资源，作业运行会极其缓慢；或者设置的资源过大，队列没有足够的资源来提供，进而导致各种异常。总之，无论是哪种情况，都会导致Spark作业的运行效率低下，甚至根本无法运行。因此我们必须对Spark作业的资源使用原理有一个清晰的认识，并知道在Spark作业运行过程中，有哪些资源参数是可以设置的，以及如何设置合适的参数值。

文章目录

1 Spark作业基本运行原理
2 资源参数调优
3 资源参数参考示例

Spark作业基本运行原理

详细原理见上图。我们使用spark-submit提交一个Spark作业之后，这个作业就会启动一个对应的Driver进程。根据你使用的部署模式（deploy-mode）不同，Driver进程可能在本地启动，也可能在集群中某个工作节点上启动。Driver进程本身会根据我们设置的参数，占有一定数量的内存和CPU core。而Driver进程要做的第一件事情，就是向集群管理器（可以是Spark Standalone集群，也可以是其他的资源管理集群，美团•大众点评使用的是YARN作为资源管理集群）申请运行Spark作业需要使用的资源，这里的资源指的就是Executor进程。YARN集群管理器会根据我们为Spark作业设置的资源参数，在各个工作节点上，启动一定数量的Executor进程，每个Executor进程都占有一定数量的内存和CPU core。

在申请到了作业执行所需的资源之后，Driver进程就会开始调度和执行我们编写的作业代码了。Driver进程会将我们编写的Spark作业代码分拆为多个stage，每个stage执行一部分代码片段，并为每个stage创建一批task，然后将这些task分配到各个Executor进程中执行。task是最小的计算单元，负责执行一模一样的计算逻辑（也就是我们自己编写的某个代码片段），只是每个task处理的数据不同而已。一个stage的所有task都执行完毕之后，会在各个节点本地的磁盘文件中写入计算中间结果，然后Driver就会调度运行下一个stage。下一个stage的task的输入数据就是上一个stage输出的中间结果。如此循环往复，直到将我们自己编写的代码逻辑全部执行完，并且计算完所有的数据，得到我们想要的结果为止。

Spark是根据shuffle类算子来进行stage的划分。如果我们的代码中执行了某个shuffle类算子（比如reduceByKey、join等），那么就会在该算子处，划分出一个stage界限来。可以大致理解为，shuffle算子执行之前的代码会被划分为一个stage，shuffle算子执行以及之后的代码会被划分为下一个stage。因此一个stage刚开始执行的时候，它的每个task可能都会从上一个stage的task所在的节点，去通过网络传输拉取需要自己处理的所有key，然后对拉取到的所有相同的key使用我们自己编写的算子函数执行聚合操作（比如reduceByKey()算子接收的函数）。这个过程就是shuffle。

当我们在代码中执行了cache/persist等持久化操作时，根据我们选择的持久化级别的不同，每个task计算出来的数据也会保存到Executor进程的内存或者所在节点的磁盘文件中。

因此Executor的内存主要分为三块：第一块是让task执行我们自己编写的代码时使用，默认是占Executor总内存的20%；第二块是让task通过shuffle过程拉取了上一个stage的task的输出后，进行聚合等操作时使用，默认也是占Executor总内存的20%；第三块是让RDD持久化时使用，默认占Executor总内存的60%。

task的执行速度是跟每个Executor进程的CPU core数量有直接关系的。一个CPU core同一时间只能执行一个线程。而每个Executor进程上分配到的多个task，都是以每个task一条线程的方式，多线程并发运行的。如果CPU core数量比较充足，而且分配到的task数量比较合理，那么通常来说，可以比较快速和高效地执行完这些task线程。

以上就是Spark作业的基本运行原理的说明，大家可以结合上图来理解。理解作业基本原理，是我们进行资源参数调优的基本前提。

资源参数调优

了解完了Spark作业运行的基本原理之后，对资源相关的参数就容易理解了。所谓的Spark资源参数调优，其实主要就是对Spark运行过程中各个使用资源的地方，通过调节各种参数，来优化资源使用的效率，从而提升Spark作业的执行性能。以下参数就是Spark中主要的资源参数，每个参数都对应着作业运行原理中的某个部分，我们同时也给出了一个调优的参考值。

num-executors

参数说明：该参数用于设置Spark作业总共要用多少个Executor进程来执行。Driver在向YARN集群管理器申请资源时，YARN集群管理器会尽可能按照你的设置来在集群的各个工作节点上，启动相应数量的Executor进程。这个参数非常之重要，如果不设置的话，默认只会给你启动少量的Executor进程，此时你的Spark作业的运行速度是非常慢的。

参数调优建议：每个Spark作业的运行一般设置50~100个左右的Executor进程比较合适，设置太少或太多的Executor进程都不好。设置的太少，无法充分利用集群资源；设置的太多的话，大部分队列可能无法给予充分的资源。

executor-memory

参数说明：该参数用于设置每个Executor进程的内存。Executor内存的大小，很多时候直接决定了Spark作业的性能，而且跟常见的JVM OOM异常，也有直接的关联。

参数调优建议：每个Executor进程的内存设置4G~8G较为合适。但是这只是一个参考值，具体的设置还是得根据不同部门的资源队列来定。可以看看自己团队的资源队列的最大内存限制是多少，num-executors乘以executor-memory，就代表了你的Spark作业申请到的总内存量（也就是所有Executor进程的内存总和），这个量是不能超过队列的最大内存量的。此外，如果你是跟团队里其他人共享这个资源队列，那么申请的总内存量最好不要超过资源队列最大总内存的1/3~1/2，避免你自己的Spark作业占用了队列所有的资源，导致别的同学的作业无法运行。

executor-cores

参数说明：该参数用于设置每个Executor进程的CPU core数量。这个参数决定了每个Executor进程并行执行task线程的能力。因为每个CPU core同一时间只能执行一个task线程，因此每个Executor进程的CPU core数量越多，越能够快速地执行完分配给自己的所有task线程。

参数调优建议：Executor的CPU core数量设置为2~4个较为合适。同样得根据不同部门的资源队列来定，可以看看自己的资源队列的最大CPU core限制是多少，再依据设置的Executor数量，来决定每个Executor进程可以分配到几个CPU core。同样建议，如果是跟他人共享这个队列，那么num-executors * executor-cores不要超过队列总CPU core的1/3~1/2左右比较合适，也是避免影响其他同学的作业运行。

driver-memory

参数说明：该参数用于设置Driver进程的内存。

参数调优建议：Driver的内存通常来说不设置，或者设置1G左右应该就够了。唯一需要注意的一点是，如果需要使用collect算子将RDD的数据全部拉取到Driver上进行处理，那么必须确保Driver的内存足够大，否则会出现OOM内存溢出的问题。

spark.default.parallelism

参数说明：该参数用于设置每个stage的默认task数量。这个参数极为重要，如果不设置可能会直接影响你的Spark作业性能。

参数调优建议：Spark作业的默认task数量为500~1000个较为合适。很多同学常犯的一个错误就是不去设置这个参数，那么此时就会导致Spark自己根据底层HDFS的block数量来设置task的数量，默认是一个HDFS block对应一个task。通常来说，Spark默认设置的数量是偏少的（比如就几十个task），如果task数量偏少的话，就会导致你前面设置好的Executor的参数都前功尽弃。试想一下，无论你的Executor进程有多少个，内存和CPU有多大，但是task只有1个或者10个，那么90%的Executor进程可能根本就没有task执行，也就是白白浪费了资源！因此Spark官网建议的设置原则是，设置该参数为num-executors * executor-cores的2~3倍较为合适，比如Executor的总CPU core数量为300个，那么设置1000个task是可以的，此时可以充分地利用Spark集群的资源。

spark.storage.memoryFraction

参数说明：该参数用于设置RDD持久化数据在Executor内存中能占的比例，默认是0.6。也就是说，默认Executor 60%的内存，可以用来保存持久化的RDD数据。根据你选择的不同的持久化策略，如果内存不够时，可能数据就不会持久化，或者数据会写入磁盘。

参数调优建议：如果Spark作业中，有较多的RDD持久化操作，该参数的值可以适当提高一些，保证持久化的数据能够容纳在内存中。避免内存不够缓存所有的数据，导致数据只能写入磁盘中，降低了性能。但是如果Spark作业中的shuffle类操作比较多，而持久化操作比较少，那么这个参数的值适当降低一些比较合适。此外，如果发现作业由于频繁的gc导致运行缓慢（通过spark web ui可以观察到作业的gc耗时），意味着task执行用户代码的内存不够用，那么同样建议调低这个参数的值。

spark.shuffle.memoryFraction

参数说明：该参数用于设置shuffle过程中一个task拉取到上个stage的task的输出后，进行聚合操作时能够使用的Executor内存的比例，默认是0.2。也就是说，Executor默认只有20%的内存用来进行该操作。shuffle操作在进行聚合时，如果发现使用的内存超出了这个20%的限制，那么多余的数据就会溢写到磁盘文件中去，此时就会极大地降低性能。

参数调优建议：如果Spark作业中的RDD持久化操作较少，shuffle操作较多时，建议降低持久化操作的内存占比，提高shuffle操作的内存占比比例，避免shuffle过程中数据过多时内存不够用，必须溢写到磁盘上，降低了性能。此外，如果发现作业由于频繁的gc导致运行缓慢，意味着task执行用户代码的内存不够用，那么同样建议调低这个参数的值。
资源参数的调优，没有一个固定的值，需要同学们根据自己的实际情况（包括Spark作业中的shuffle操

作数量、RDD持久化操作数量以及spark web ui中显示的作业gc情况），同时参考本篇文章中给出的原理以及调优建议，合理地设置上述参数。

资源参数参考示例

以下是一份spark-submit命令的示例，大家可以参考一下，并根据自己的实际情况进行调节：

```
./bin/spark-submit \
  --master yarn-cluster \
  --num-executors 100 \
  --executor-memory 6G \
  --executor-cores 4 \
  --driver-memory 1G \
  --conf spark.default.parallelism=1000 \
  --conf spark.storage.memoryFraction=0.5 \
  --conf spark.shuffle.memoryFraction=0.3 \
```

本文转载自：http://tech.meituan.com/spark-tuning-basic.html

Spark

« Spark性能优化：开发调优篇

Scala模式匹配和函数组合 »

[电子书]Machine Learning with Spark Second Edition PDF下载	Spark sql解析异常 java.lang.StackOverflowErr 处理	解决Spark shell模式下初始化Job出现的异常	[电子书]Mastering Spark for Data Science PDF下载

[电子书]Machine Learning with Spark	Spark sql解析异常 java.lang.StackOverflowEr	解决Spark shell模式下初始化Job出现的异常	[电子书]Mastering Spark for Data Science PDF下

Apache Spark常见的三大误解	Apache Hivemall:可运行在Hive, Spark 和 Pig 上的可扩展机器学习库	Spark Structured Streaming入门编程指南	Spark 2.1.0与CarbonData 1.0.0集群模式部署及使用入门指南

Apache Spark常见的三大误解	Apache Hivemall:可运行在Hive, Spark 和 Pig 上	Spark Structured Streaming入门编程指南	Spark 2.1.0与CarbonData 1.0.0集群模

下面文章您可能感兴趣

- SQL Joins可视化解释
- 炼数成金-Spark大数据平台视频百度网盘免费下载
- Akka学习笔记：Actors介绍
- Apache Flume: Distributed Log Collection for Hadoop
- Akka学习笔记：Actor消息传递(1)
- Hadoop源码编译与调试
- 上海Spark Meetup第六次聚会
- Hadoop作业JVM堆大小设置优化
- Akka学习笔记：子Actor和Actor路径
- 通过可视化途径理解你的Spark应用程序
- Spark 1.0.1发布了
- Akka学习笔记：Actor消息传递(2)
- 在Hive中使用Avro
- Hadoop Backup and Recovery Solutions
- Hadoop 2.2.0编译hadoop-eclipse-plugin插件
- 运行Jar包文件签名不合法的问题
- 给Hadoop集群中添加Snappy解压缩库
- Mahout项目已经实现的算法
- Spark北京Meetup第五次活动(Streaming专题)
- Ubuntu安装依赖文件

发表我的评论

写点什么吧

写点什么吧...

表情 本博客评论系统带有自动识别垃圾评论功能，请写一些有意义的评论，谢谢！ 提交评论

(6)个小伙伴在吐槽

大神，请教一下，我用的kafka，现在是一个kafka分区对应一个task，有比要去设置spark.default.parallelism这个参数吗?

YUZJANG 2016-06-05 07:22 回复

要设置。

w397090770 2016-06-05 09:41 回复

我设置了--conf spark.default.parallelism=32，但是后来发现每个stage里面还是只有8个task在跑（我的kafka设置了8个分区）。
这是我的启动命令：./spark-submit
--class driver.DemoDirectYarn
--name direct-optimized
--master yarn-cluster
--num-executors 4
--executor-memory 2g
--executor-cores 2
--conf spark.default.parallelism=32
/home/jaremo_yarn-assembly-1.0.0.SNAPSHOP.jar
30 20000

YUZJANG 2016-06-05 14:07 回复

这个是正常的，你是不是用了direct api? 如果这样的话那就是每个Kafka分区对应RDD每个partition，所有会起8个task

w397090770 2016-06-07 08:00 回复

是的，我用的direct api

YUZJANG 2016-06-15 14:07

看到spark.default.parallelism这个参数时，想起你之前有篇文章是说放个文件到集群上运行，因为没有切片导致并行度很低要三个小时完成（spark on yarn 提高cpu利用率），有点疑惑请教下大神，设置这个参数是不是可以实现spark 读取文件时拆分多个task,提高并行度?

淼淼 2016-05-13 10:55 回复