## MapReduce的优化[Hadoop技术学习](http://mp.weixin.qq.com/s/9k0Zq9fWuDEMXawHXcxV0w" \l "#)

随着企业要处理的数据量越来越大，MapReduce思想越来越受到重视。Hadoop是MapReduce的一个开源实现，由于其良好的扩展性和容错性，已得到越来越广泛的应用。Hadoop作为一个基础数据处理平台，虽然其应用价值已得到大家认可，但仍存在很多问题；为了提高其数据性能，很多人开始优化Hadoop。总结看来，对于Hadoop，当前主要有几个优化思路：

**一．操作系统调优**

Ø  增大打开文件数据和网络连接上限，调整内核参数net.core.somaxconn，提高读写速度和网络带宽使用率

Ø  适当调整epoll的文件描述符上限，提高Hadoop RPC并发

Ø  关闭swap。如果进程内存不足，系统会将内存中的部分数据暂时写入磁盘，当需要时再将磁盘上的数据动态换置到内存中，这样会降低进程执行效率

Ø  增加预读缓存区大小。预读可以减少磁盘寻道次数和I/O等待时间

Ø  设置openfile

**二．HDFS参数调优**

1.     core-site.xml

Ø hadoop.tmp.dir：默认值： /tmp

说明： 尽量手动配置这个选项，否则的话都默认存在了里系统的默认临时文件/tmp里。并且手动配置的时候，如果服务器是多磁盘的，每个磁盘都设置一个临时文件目录，这样便于mapreduce或者hdfs等使用的时候提高磁盘IO效率。

Ø fs.trash.interval：默认值： 0

说明： 这个是开启hdfs文件删除自动转移到垃圾箱的选项，值为垃圾箱文件清除时间（分钟）。一般开启这个会比较好，以防错误删除重要文件。

Ø io.file.buffer.size：默认值：4096

说明：SequenceFiles在读写中可以使用的缓存大小，可减少 I/O 次数。在大型的 Hadoop cluster，建议可设定为 65536 到 131072。

2.     hdfs-site.xml

Ø dfs.blocksize：默认值：134217728

说明： 这个就是hdfs里一个文件块的大小了，CDH5中默认128M。太大的话会有较少map同时计算，太小的话也浪费可用map个数资源，而且文件太小namenode就浪费内存多。根据需要进行设置。

Ø dfs.namenode.handler.count：默认值：10

说明：设定 namenode server threads 的数量，这些 threads 會用 RPC 跟其他的 datanodes 沟通。当 datanodes 数量太多时会发現很容易出現 RPC timeout，解決方法是提升网络速度或提高这个值，但要注意的是 thread 数量多也表示 namenode 消耗的内存也随着增加

**三．MapReduce参数调优**

Ø mapred.reduce.tasks（mapreduce.job.reduces）：默认值：1

说明：默认启动的reduce数。通过该参数可以手动修改reduce的个数。

Ø mapreduce.task.io.sort.factor：默认值：10

说明：Reduce Task中合并小文件时，一次合并的文件数据，每次合并的时候选择最小的前10进行合并。

Ø mapreduce.task.io.sort.mb：默认值：100

说明： Map Task缓冲区所占内存大小。

Ø mapred.child.Java.opts：默认值：-Xmx200m

说明：jvm启动的子线程可以使用的最大内存。建议值-XX:-UseGCOverheadLimit-Xms512m -Xmx2048m -verbose:gc -Xloggc:/tmp/@taskid@.gc

Ø mapreduce.jobtracker.handler.count：默认值：10

说明：JobTracker可以启动的线程数，一般为tasktracker节点的4%。

Ø mapreduce.reduce.shuffle.parallelcopies：默认值：5

说明：reuduce shuffle阶段并行传输数据的数量。这里改为10。集群大可以增大。

Ø mapreduce.tasktracker.http.threads：默认值：40

说明：map和reduce是通过http进行数据传输的，这个是设置传输的并行线程数。

Ø mapreduce.map.output.compress：默认值：false

说明： map输出是否进行压缩，如果压缩就会多耗cpu，但是减少传输时间，如果不压缩，就需要较多的传输带宽。配合 mapreduce.map.output.compress.codec使用，默认是org.apache.hadoop.io.compress.DefaultCodec，可以根据需要设定数据压缩方式(org.apache.hadoop.io.compress.SnappyCodec)。

Ø mapreduce.reduce.shuffle.merge.percent：默认值： 0.66

说明：reduce归并接收map的输出数据可占用的内存配置百分比。类似mapreduce.reduce.shuffle.input.buffer.percent属性。

Ø mapreduce.reduce.shuffle.memory.limit.percent：默认值： 0.25

说明：一个单一的shuffle的最大内存使用限制。

Ø mapreduce.jobtracker.handler.count：默认值： 10

说明：可并发处理来自tasktracker的RPC请求数，默认值10。

Ø mapred.job.reuse.jvm.num.tasks（mapreduce.job.jvm.numtasks）：默认值： 1

说明：一个jvm可连续启动多个同类型任务，默认值1，若为-1表示不受限制。

Ø mapreduce.tasktracker.tasks.reduce.maximum：默认值： 2

说明：一个tasktracker并发执行的reduce数，建议为cpu核数

**四．系统优化**

1）避免排序

对于一些不需要排序的应用，比如hash join或者limit n，可以将排序变为可选环节，这样可以带来一些好处：

Ø 在Map Collect阶段，不再需要同时比较partition和key，只需要比较partition，并可以使用更快的计数排序（O(n)）代替快速排序（O(NlgN)）

Ø 在Map Combine阶段，不再需要进行归并排序，只需要按照字节合并数据块即可。

Ø 去掉排序之后，Shuffle和Reduce可同时进行，这样就消除了Reduce Task的屏障（所有数据拷贝完成之后才能执行reduce()函数）。

2）Shuffle阶段内部优化

Ø Map端--用Netty代替Jetty

Ø Reduce端--批拷贝

Ø 将Shuffle阶段从ReduceTask中独立出来

**五、经常调整参数:**

在运行mapreduce任务中，经常调整的参数有：

Ø mapred.reduce.tasks：手动设置reduce个数

Ø mapreduce.map.output.compress：map输出结果是否压缩

Ø mapreduce.map.output.compress.codec：压缩格式

Ø mapreduce.output.fileoutputformat.compress：job输出结果是否压缩

Ø mapreduce.output.fileoutputformat.compress.type：默认RECORD

Ø mapreduce.output.fileoutputformat.compress.codec：压缩格式