Spark开发中常见的错误

**错误1：**

org.apache.spark.rpc.RpcTimeoutException: Futures timed out after [10 seconds]. This timeout is controlled by spark.executor.heartbeatInterval

网上有提出该问题是由于资源不足导致的。在提交任务的时候添加以下配置：

--conf spark.driver.extraJavaOptions="-XX:PermSize=2g -XX:MaxPermSize=2g"

其他博客提出了以下的解决方法：

1、spark.executor.heartbeatInterval配置

a、可以在spark-submit指定

--conf spark.executor.heartbeatInterval=10000s

b、也可以在spark-defaults.conf中添加

spark.executor.heartbeatInterval 10000s

c、还可以在代码中配置

val spark = SparkSession.builder

.config("spark.executor.heartbeatInterval", "10000s")

.getOrCreate()

2、spark.network.timeout配置

由网络或者gc引起,worker或executor没有接收到executor或task的心跳反馈，设置

spark.network.timeout=600

**错误2：**

java.lang.OutOfMemoryError: GC overhead limit exceeded

java.lang.OutOfMemoryError: Java heap space

以上两种错误都是因为数据量过大，driver内存不足（在不指定给driver分配内存时，默认分配的是512M，在这种情况下，如果处理的数据或者加载的数据很大，driver就可能会爆内存，出现OOM错误。），需增加driver内存，在提交任务的时候添加以下配置，如果已经添加了配置，则需要增加driver内存设置的值。

**方法1：**在spark-submit中指定 --driver-memory memSize参数来设定driver的jvm内存大小，可以通过spark-submit --help查看其他可以设置的参数。

--driver-memory 2g

**方法2**：在spark\_home/conf/目录中，将spark-defaults.conf.template模板文件拷贝一份到/spark\_home/conf目录下，命名为spark-defaults.conf，然后在里面设置spark.driver.memory  memSize属性来改变driver内存大小。

eg:

spark.master spark://master:7077

spark.default.parallelism 10

spark.driver.memory 2g

spark.serializer org.apache.spark.serializer.KryoSerializer

spark.sql.shuffle.partitions 50

针对OOM（内存溢出，也称为爆内存）的一些认知：有driver OOM和executor OOM两种。

(1) driver OOM

一般是使用了collect操作将所有executor的数据聚合到driver导致。尽量不要使用collect操作即可。

(2) executor OOM

* 可以按下面的内存优化的方法增加code使用内存空间
* 增加executor内存总量，即增加spark.executor.memory的值
* 增加任务并行度(大任务就被分成小任务了)，参考下面优化并行度的方法

**OOM优化方法**

1.从“内存”的角度优化

如果任务的shuffle量特别大，同时rdd缓存比较少可以更改下面的参数进一步提高任务运行速度。如果任务运行缓慢，jvm进行频繁gc或者内存空间不足，可以降低下面两个参数。

* spark.storage.memoryFraction - 分配给rdd缓存的比例，默认为0.6(60%)，如果缓存的数据较少可以降低该值。
* spark.shuffle.memoryFraction - 分配给shuffle数据的内存比例，默认为0.2(20%)剩下的20%内存空间则是分配给代码生成对象等。

注意：如果数据只使用一次，不要采用cache操作，因为并不会提高运行速度，还会造成内存浪费。

2.从“并行度”的角度优化

* 增加参数spark.default.parallelism的值

注意：在standalone模式下的shuffle的并行度默认为core的个数，也可手动调整，数量设置太大会造成很多小任务，增加启动任务的开销，太小可能导致运行大数据量的任务时速度缓慢。

* 如果是由于sql聚合操作引起的OOM，可以增加spark.sql.shuffle.partitions，默认为200,如果任务运行缓慢增加这个值。

**错误3：**

org.apache.spark.shuffle.FetchFailedException

报错提示：

(1) missing output location

org.apache.spark.shuffle.MetadataFetchFailedException: Missing an output location for shuffle 0

(2) shuffle fetch faild

org.apache.spark.shuffle.FetchFailedException: Failed to connect to spark047215/192.168.47.215:50268

这种问题一般发生在有大量shuffle操作的时候，task不断的failed，然后又重执行，一直循环下去，非常的耗时。一般遇到这种问题提高executor内存，同时增加每个executor的cpu即可，采用如下方式在spark-submit时配置：

--conf spark.executor.memory 15G

--conf spark.executor.cores 3

**错误4：**

Executor&Task Lost

报错提示：

(1) executor lost

WARN TaskSetManager: Lost task 1.0 in stage 0.0 (TID 1, aa.local): ExecutorLostFailure (executor lost)

(2) task lost

WARN TaskSetManager: Lost task 69.2 in stage 7.0 (TID 1145, 192.168.47.217): java.io.IOException: Connection from /192.168.47.217:55483 closed

(3) 各种timeout

* java.util.concurrent.TimeoutException: Futures timed out after [120 second
* ERROR TransportChannelHandler: Connection to /192.168.47.212:35409 has been quiet for 120000 ms while there are outstanding requests. Assuming connection is dead; please adjust spark.network.timeout if this is wrong

以上错误的解决方法是：提高 spark.network.timeout 的值，根据情况改成300(5min)或更高，默认为 120(120s)，配置所有网络传输的延时，如果没有主动设置以下参数，默认覆盖其属性

* spark.core.connection.ack.wait.timeout
* spark.akka.timeout
* spark.storage.blockManagerSlaveTimeoutMs
* spark.shuffle.io.connectionTimeout
* spark.rpc.askTimeout or spark.rpc.lookupTimeout

**错误5：**

java.util.concurrent.RejectedExecutionException: Task scala.concurrent.impl.CallbackRunnable@3f9bf2d0 rejected from java.util.concurrent.ThreadPoolExecutor

以上错误是由于线程池满了，仍有外部的请求，但是连接已经关闭。这往往是由于partition num（task）与分配的资源（executor num、executor cores）不匹配有关系。增加executor num、executor cores，减少并行task的数量，或者增加spark.shuffle.io.numConnectionsPerPeer的值，该值默认为1。

**错误6：**

java.lang.OutOfMemoryError: PermGen space

java.lang.OutOfMemoryError: Java heap space

由网络或者gc引起，worker或executor没有接收到executor或task的心跳反馈，导致 Executor&Task Lost，这时要提高 spark.network.timeout 的值，根据情况改成300(5min)或更高。

另一种解决方法：在spark-submit时添加如下设置：

--conf spark.driver.extraJavaOptions="-Xms256m -Xmx512m -XX:PermSize=256m -XX:MaxPermSize=256M"

参考：

https://blog.csdn.net/gaokao2011/java/article/details/51707163

https://www.cnblogs.com/double-kill/p/9012383.html

https://www.jianshu.com/p/7c74b8690322

https://www.cnblogs.com/nucdy/p/6728550.html