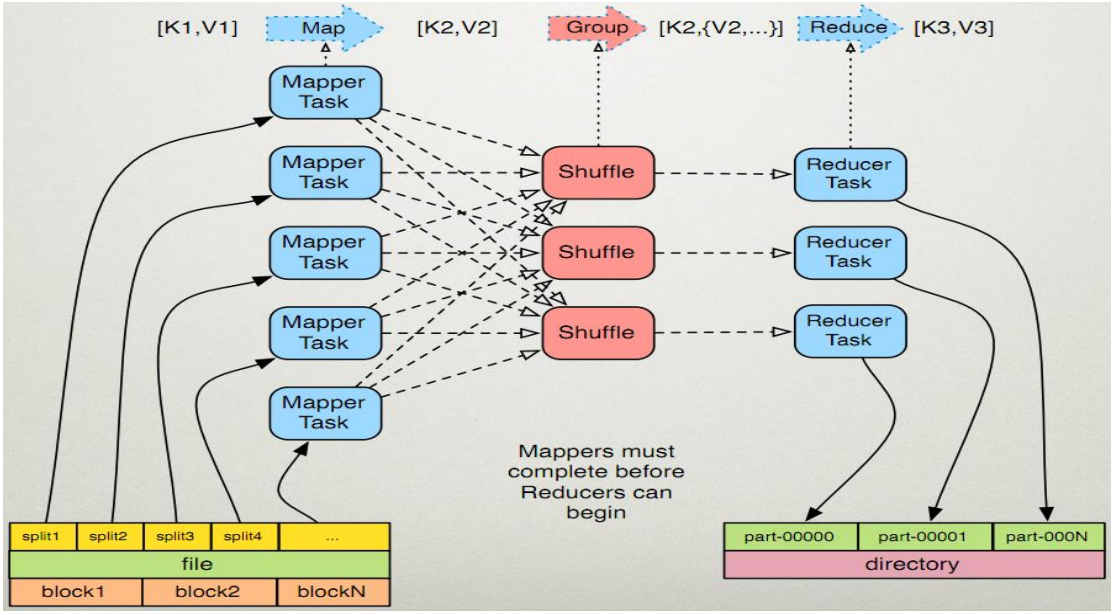
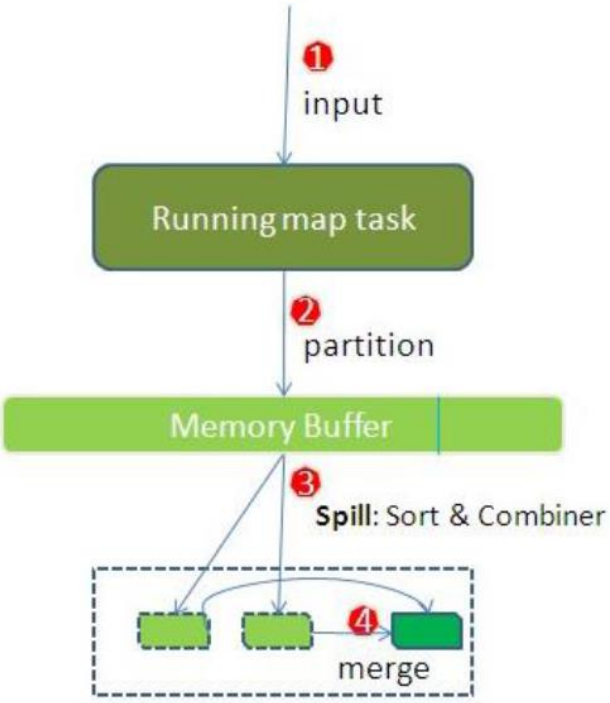
MapTask 运行机制详解以及 Map 任务的并行度





整个 Map 阶段流程大体如上图所示。简单概述：inputFile 通过 split 被逻辑切分为多个 split 文件，通过 Record 按行读取内容给 map（用户自己实现的）进行处理，数据被 map 处理结束之后交给 OutputCollector 收集器，对其结果 key 进行分区（默认使用 hash 分区），然后写入 buffer，每个 map task 都有一个内存缓冲区，存储着 map 的输出结果，当缓冲区快满的时候需要将缓冲区的数据以一个临时文件的方式存放到磁盘，当整个 map task 结束后再对磁盘中这个 map task 产生的所有临时文件做合并，生成最终的正式输出文件，然后等待 reduce task 来拉数据。

详细步骤：

1、 首先，读取数据组件InputFormat（默认TextInputFormat）会通过getSplits方法对输入目录中文件进行逻辑切片规划得到splits，有多少个split就对应启动多少个MapTask。split与block的对应关系默认是一对一。

2、 将输入文件切分为splits之后，由RecordReader对象（默认LineRecordReader）进行读取，以\n作为分隔符，读取一行数据，返回<key，value>。Key表示每行首字符偏移值，value表示这一行文本内容。

3、 读取split返回<key,value>，进入用户自己继承的Mapper类中，执行用户重写的map函数。RecordReader读取一行这里调用一次。

4、 map逻辑完之后，将map的每条结果通过context.write进行collect数据收集。在collect中，会先对其进行分区处理，默认使用HashPartitioner。

MapReduce 提供 Partitioner 接口，它的作用就是根据 key 或 value 及 reduce 的数量来决定当前的这对输出数据最终应该交由哪个 reduce task 处理。默认对 key

hash 后再以 reduce task 数量取模。默认的取模方式只是为了平均 reduce 的处理能力，如果用户自己对 Partitioner 有需求，可以订制并设置到 job 上。

5、接下来，会将数据写入内存，内存中这片区域叫做环形缓冲区，缓冲区的作用是批量收集 map 结果，减少磁盘 IO 的影响。我们的 key/value 对以及 Partition 的结果都会被写入缓冲区。当然写入之前，key 与 value 值都会被序列化成字节数组。

环形缓冲区其实是一个数组，数组中存放着key、value的序列化数据和key、value的元数据信息，包括partition、key的起始位置、value的起始位置以及value的长度。环形结构是一个抽象概念。

缓冲区是有大小限制，默认是100MB。当map task的输出结果很多时，就可能会撑爆内存，所以需要在一定条件下将缓冲区中的数据临时写入磁盘，然后重新利用这块缓冲区。这个从内存往磁盘写数据的过程被称为Spill，中文可译为溢写。这个溢写是由单独线程来完成，不影响往缓冲区写map结果的线程。溢写线程启动时不应该阻止map的结果输出，所以整个缓冲区有个溢写的比例spill.percent。这个比例默认是0.8，也就是当缓冲区的数据已经达到阈值（buffer size * spill percent = 100MB * 0.8 = 80MB），溢写线程启动，锁定这80MB的内存，执行溢写过程。Map task的输出结果还可以往剩下的20MB内存中写，互不影响。

6、当溢写线程启动后，需要对这80MB空间内的key做排序(Sort)。排序是MapReduce模型默认的行为，这里的排序也是对序列化的字节做的排序。

如果job设置过Combiner，那么现在就是使用Combiner的时候了。将有相同key

的key/value对的value加起来，减少溢写到磁盘的数据量。Combiner会优化MapReduce的中间结果，所以它在整个模型中会多次使用。

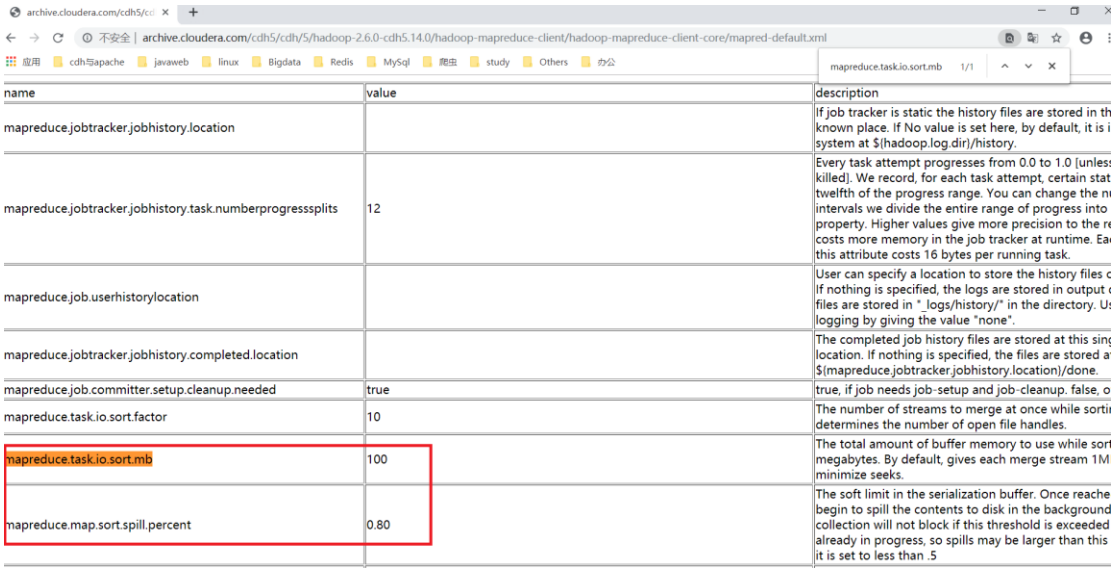
那哪些场景才能使用Combiner呢？从这里分析，Combiner的输出是Reducer的输入，Combiner绝不能改变最终的计算结果。Combiner只应该用于那种Reduce的输入key/value与输出key/value类型完全一致，且不影响最终结果的场景。比如累加，最大值等。Combiner的使用一定得慎重，如果用好，它对job执行效率有帮助，反之会影响reduce的最终结果。

7、合并溢写文件：　每次溢写会在磁盘上生成一个临时文件（写之前判断是否有combiner），如果map的输出结果真的很大，有多次这样的溢写发生，磁盘上相应的就会有多个临时文件存在。当整个数据处理结束之后开始对磁盘中的临时文件进行merge合并，因为最终的文件只有一个，写入磁盘，并且为这个文件提供了一个索引文件，以记录每个reduce对应数据的偏移量。

至此 map 整个阶段结束。

mapTask 的一些基础设置配置（mapred-site.xml 当中）：

http://archive.cloudera.com/cdh5/cdh/5/hadoop-2.6.0-cdh5.14.0/hadoop-mapreduce-client/hadoop-mapreduce-client-core/mapred-default.xml



name	value	description
mapreduce.jobtracker.jobhistory.location		If job tracker is static the history files are stored in th known place. If No value is set here, by default, it is i system at ${hadoop.log.dir}/history.
mapreduce.jobtracker.jobhistory.task.numberprogresssplits	12	Every task attempt progresses from 0.0 to 1.0 [unles killed]. We record, for each task attempt, certain stat twelfth of the progress range. You can change the nu intervals we divide the entire range of progress into property. Higher values give more precision to the re costs more memory in the job tracker at runtime. Ea this attribute costs 16 bytes per running task.
mapreduce.job.userhistorylocation		User can specify a location to store the history files c If nothing is specified, the logs are stored in output d files are stored in "_logs/history/" in the directory. Us logging by giving the value "none".
mapreduce.jobtracker.jobhistory.completed.location		The completed job history files are stored at this sing location. If nothing is specified, the files are stored a ${mapreduce.jobtracker.jobhistory.location}/done.
mapreduce.job.committer.setup.cleanup.needed	true	true, if job needs job-setup and job-cleanup. false, o
mapreduce.task.io.sort.factor	10	The number of streams to merge at once while sorti determines the number of open file handles.
mapreduce.task.io.sort.mb	100	The total amount of buffer memory to use while sort megabytes. By default, gives each merge stream 1M minimize seeks.
mapreduce.map.sort.spill.percent	0.80	The soft limit in the serialization buffer. Once reache begin to spill the contents to disk in the background collection will not block if this threshold is exceeded already in progress, so spills may be larger than this it is set to less than .5

设置一：设置环型缓冲区的内存值大小（默认设置如下）

mapreduce.task.io.sort.mb	**100**

设置二：设置溢写百分比（默认设置如下）

mapreduce.map.sort.spill.percent	0.80

设置三：设置溢写数据目录（默认设置）

mapreduce.cluster.local.dir	${hadoop.tmp.dir}/mapred/local

设置四：设置一次最多合并多少个溢写文件（默认设置如下）

mapreduce.task.io.sort.factor	10