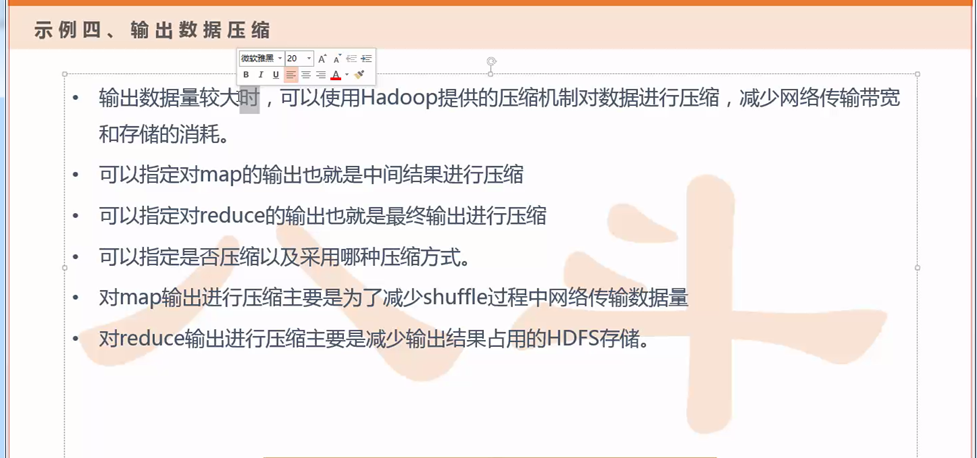
回顾上节课我们搞了三个分发实践：

1. -file 从本地往集群上每一个节点做分发文件，当然小文件的时候比较适合这样的方式去做

如果文件比较大可以采用下面两种方式：

1. -cachefile 以文件的形式去做分发，当然这个文件可能数据比较大，得提前放到HDFS集群上的某一个位置，然后通过一个#号加别名的方式进行一个映射
2. -cachearchive 把多个文件或者是有目录有文件的这样一个可能有层级关系的进行一个整理，然后打一个压缩包，所以这种方式包含于cachefile，相当于是cachefile的高级用法。所以注意一下，有这种需求的时候，看看需要用哪种方式更合适。

图1：



我们上节课说了数据是可以压缩的，压缩的形式可以不经过你的split，你前面有几个压缩文件，如果是作为一个数据源输入的话，你后面这个MapReduce的到底启多少个Map并发的个数是根据你这个压缩文件的个数来分配的。比如当你有100个的压缩文件，那你后面读这个100个压缩文件的时候，那么你就是100个map来读。

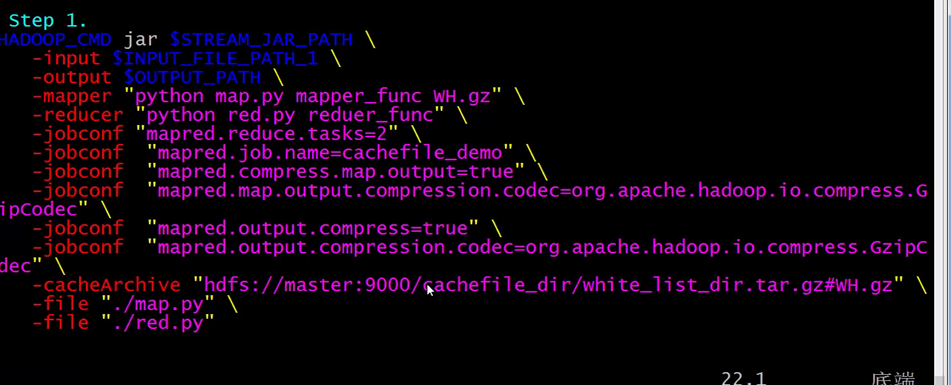
压缩细节有两个：

1. 可以在中间的时候进行压缩，那么在中间的时候进行压缩是我们的Map和Reduce相当于是可以说是两个不同节点，然后map产生的一些数据是需要通过shuffle拷贝到Reduce这个机器上，所以这里面牵扯到一些数据的远程拷贝远程传输这么一个过程，然后如果说数据量比较大的话，它必然会导致它传输的过程中对于带宽是有一定的压力，所以相当于如果要是想把这个map到reduce这个中间结果进行压缩的话，它可以大大的减少你这个网络传输整体的数据量数据规模，并且它也不会影响你的最终结果，所以这里面可以通过一个参数去控制map到reduce这个过程中的中间环节是可以压缩还是不压缩，是由我们通过配置就是提交任务的时候可以指定，这个是可以做到的，当然了你在中间环节，就是临时文件压缩的话可能是对我们用户来说是透明的，但是真正在reduce的输出它是实实在在落地到了你的HDFS上，需要我们用户后续去进一步的使用，所以最后reduce输出的这个数据是对于我们来说是更敏感的，所以也可以通过另外一个参数来控制它最终输出是否压缩以及它的压缩格式。
2. 可以在整个任务结束以后再进行压缩，就是最终的输出你也可以指定说压缩还是不压缩。

所以对数据的压缩主要包含两个部分，一个是中间阶段的压缩，一个是最终输出的压缩。

重点：压缩是需要代码的开发，只仅仅需要参数的控制（不需要开发mapreduce，仅仅是要你提交任务的时候通过参数去指定）（如下图2所示）：

图2：

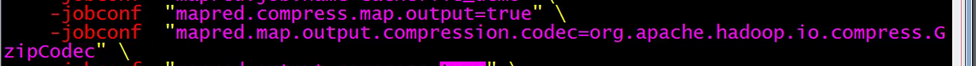


相比于我们上次课学到的一些配置参数，这张图2里面多加了四个带compress的参数。

然后上面那两个compress参数是一组，下面compress参数是一组。

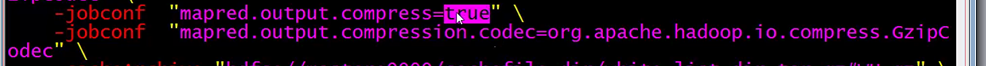
上面compress的参数是一组的意思是，从这个字面就可以看的出来，这个map.output=true说明从map阶段产生的这个数据是否压缩这个开关，如果说true就是相当于这个开关打开了，如果是把这个开关打开之后，就必须要配置一个具体压缩这个数据的格式是怎样的，所以后面紧跟着一个参数来指定你通过什么样子的压缩方法去对这个数据进行压缩，那么这里面默认比较常用的就是：GzipCodec这种压缩算法。（如下图3所示）

图3：



然后下面那两个compress参数是最终的一个mapreduce的output最终的一个输出，也是一个开关默认是falue，那么需要用的话就需要把这参数加上并且设置为true（如下图4所示）

图4:



如果你这个数据最后输出的时候你想压缩的话，也要指定压缩格式。

所以在这个mapReduce配置压缩的这个事情就需要配置参数就可以了。