基于 Hadoop Map-Reduce 的 word count 程序

1. 设计思路

对于英文环境下的词频统计，首先应以空格分隔，将文章整体切分成多个可能带有标点（符号）的单词。可能的形式有：

（1）单词+标点（如 word,）

（2）标点+单词（如 ”word）

（3）标点+单词+标点（如 ’word’）

（4）单词+标点+单词（如 it’s）

为了能够快速针对上述情况进行处理，则需要在一次 Map-Reduce 中，输出正确的结果。为此，有两种可行的方案：一是在进行单词切分时，以空格和单引号（为了处理形式4）进行分割；二是以空格进行切分，在 Map 的同时，去除“单词”两边的符号。

对比以上两种方案，我们认为若“it’s”被切分成了“it”和“s”，那么则丢失了单词本身的含义（“s”并不是一个单词），“it’s”在英文中可以称作一个单词，所以对形式4不作任何处理，认为其是一个完整的单词。因此选取了方案二来实现 Map-Reduce 程序。

另外，我们认为词频统计中，应该统计原文中真正的单词，诸如 URL 地址等看起来包含多个单词的字符串，不应进行切分，因为那样失去了原文的含义。比如，很多时候可能同一个 URL 会出现多次，或者某一个诸如“Mr.John”的名字会出现多次，我们就应该统计出这类词究竟出现了多少次，而不是将这个实体分成多个部分。

2. 实现方案

程序采用 Java 语言，实现了基于 Hadoop 2.4 的 Map-Reduce 程序。

2.1 准备工作

为了在 Map 阶段处理标点符号，程序首先设置单词两边需要剔除的标点符号，如下表所示：

需要被剔除的标点符号				
$	`	~	!	:
@	#	%	^	;
&	*	(	)	\
[	]	{	}	"

需要被剔除的标点符号				
'	,	.	<	>
?	-	+	=	

在接下来的 Map 阶段，若单词两边出现了相应的标点符号，则 Map 输出的 key 将是剔除单词两边符号的实际单词。

2.2 Map 阶段

在 Map 阶段，程序会按行扫描输入文件，将每行文本按空格切分，剔除符号后，以每一个单词作为 key，其 value 为数字1。

Map 输入：<文本 Object, Text>

Map 输出：<word, 1>

2.3 Reduce 阶段

Reduce 接收 Map 的输出，将相同 key 的 value 进行相加，求出其出现次数。

在 Reduce 之前，还添加了 Combine 阶段，先将多个 Map 中相同 key 的 value 合并，从而减少 Reduce 的输入大小，提升程序的效率。

2.4 输出设计

Hadoop 默认的 outputformat 为 key 和 value 以 tab 分割，题目要求的输出为冒号分隔。因此，在配置 Hadoop 程序时修改程序的输出格式，设置 mapred.textoutputformat.separator 为":"。这样在最终的输出结果中 key 和 value 的输出格式可以满足题目的输出要求。

另外，我们只进行实际的单词词频统计，因此忽略产生的数字，使得输出中只包含实际的由字母组成的单词或实体。

3. 测试结果

本题目的 jar 包为 DataSpark_problem1.jar。具体运行环境及截图见PPT 。

3.1 运行方法

本程序可在 Hadoop 2.4.1 环境下分布式运行。使用方法为：

hadoop jar DataSpark_problem1.jar <input_path> <output_path>

3.2 运行结果

程序选择了Hadoop 2.4.1的 License 文件（$HADOOP_HOME/LICENSE.txt）作为输入，统计该文件中的词频。输出的部分结果如下所示（详见附件）：

a:25

above:5

acceptance:1

accepting:3

act:1

acting:1

acts:1

add:2

addendum:1

additional:5

additions:1

advised:3

against:2

agree:1

agreed:3

agreement:1

algorithm:1

all:5

alleging:1

alone:1

……

……

……

you:27

your:14

yyyy:1