金融大数据 实验一 实验报告

151220006 陈安宇

一、实验需求

需求 1：针对股票新闻数据集中的新闻标题，编写 WordCount 程序，统计所有除 Stop-word（如“的”，“得”，“在”等）出现次数 k 次以上的单词计数，最后的结果按照词频从高到低排序输出。
需求 2：针对股票新闻数据集，以新闻标题中的词组为 key，编写带 URL 属性的文档倒排索引程序，将结果输出到指定文件。

二、实验设计说明

实验 1：对中文新闻标题进行分词并计数，实现出现频率大于某一值的所有词的从高到低的输出。

设计思路：程序框架：该实验的主要部分是对词频计数，因此采用 WordCount2 算法的框架。
中文进行分词程序：编写一个简单的分词程序。首先把预先准备好的词典中的词读入一个字符串类型的链表中。并记录最长的字符串的长度。维护一个最终分词结果的链表。对读入的段落按照取指定的最大长度的文本去词典里面匹配，如果匹配不到，则长度减一继续匹配，若匹配成功，把该词加入分词结果列表，并从待分词文本中去除已经分词的文本。直到待分词文本为空，分词结束。
停词：读取停词的文件，将停词放入一个链表中。从取出的中文标题中逐一地去除停词。
词频从高到低输出：调用本身有的比较器 Comparator，在前面加上负号实现从词频从高到低的排序。

程序流程：从文本中获取中文标题→去除停词→分词→去除长度为 1 的词→计数→与 K 比较→结果写入临时文件→排序→写入结果文件

设计说明：分词类 **WordSeg**: List<String> DIC 存储词典
List<String> result 存储分词结果
词频计数 **WordCount3**: Set<String> patternsToSkip 存储停词
String[] itr = line.split("\t");用于截取中文文本
List<String> lis 存储切分好的中文单词

程序运行： **bin/hadoop jar ./share/hadoop/mapreduce/WordCount3.jar WordCount3 -Dwordcount.case.sensitive=true inputdata output1 100 -skip /user/hadoop/stopwords/patterns.txt**

运行结果截图：

词频较高部分：

28275	公告
8449	提示
7183	板块
7107	股东
6609	业绩
6292	涨停
5036	晚间
4178	投资
3533	资金
3499	控股
3038	行业
3022	减持
2890	今日
2886	增持
2877	年报
2806	科技股份
2726	中国
2668	年度
2625	交易
2622	增长
2550	净利
2317	集团股份
2299	资产
2260	股权
2133	中报
2093	股价
2072	利好
2031	股票
1831	报告
1779	拉升
1729	次新股
1705	市场
1701	突破
1647	披露
1619	银行
1549	龙头
1508	发展
1498	项目

词频较低部分：

120	绿色
119	处理
119	变动
118	账户
118	陕西
117	民营
116	宣布
115	黑马
114	募投
114	支持
113	期货
113	高速公路
112	幸福
111	投入
110	创意
110	特钢
110	违法
109	阅兵
109	暂时
109	贷款
109	精华
108	合肥
108	说明书
108	人事
108	格局
107	长期
107	电源
106	新低
106	交通
106	存货
106	行政处罚
105	机场
104	拓展
103	移动
102	营业部
102	亮点
101	债务
101	具备

实验不足之处：1.用 StringTokenizer 比 String.split()效率高。整个程序在处理大数据集 fulldat 时效率不高，运行较慢。

2.在实现通过词频排序时，没有考虑当两个词词频相同时的排序情况。

实验 2：实现带 URL 属性的文档倒排索引程序。

设计思路：该实验的主要目的是文档的倒排索引，借鉴带词频属性的文档倒排索引程序的框架。

中文分词程序与停词：同实验 1。

Map 类：计算每一行的词与该行每个词出现的次数。每个词作为 KEY，该行的 URL+@+该词在该行的词频作为 VALUE。

Combiner 类：通过@符号抽取出 URL 和词频，合并 URL 和词频，合并本地 mapper

函数的输出

Reducer:合并 URL 和词频，合并 mapper 函数的输出,将结果写入输出文件。

程序流程：从文本中获取中文标题→去除停词→分词→去除长度为 1 的词→计数并记录该词的 URL→合并 URL 和词频→写入结果文件

设计说明：分词类和停词类同实验 1

在 map 中维护一个 hashmap()，统计一行出现的词，以及这一行出现该词的次数。

Text fileName_frequence 记录 URL 与词频

String UrlList 记录合并的 URL。URL 与 URL 之间用******隔开

程序运行：

```
bin/hadoop jar ./share/hadoop/mapreduce/InvertedIndexer.jar InvertedIndexer
-Dwordcount.case.sensitive=true inputdata output2
-skip /user/hadoop/stopwords/patterns.txt
```

运行结果截图：

实验不足之处：1.用 StringTokenizer 比 String.split()效率高。整个程序在处理大数据集 fulldat 时效率不高，运行较慢。

2.对于关键字的筛选做的不够细致。