

# # 并行计算 上机报告

## 上机题目：

1. 按照Hadoop安装运行说明文档中的指导，自己搭建伪分布式Hadoop环境，熟悉HDFS的常用操作(参考 [Hadoop实战](#) 第31-36页)，运行WordCount程序，得到统计结果。请详细写出你每一步的操作，最好有截图，最后的结果部分必须有截图。
2. 实现一个统计输入文件中各个长度的单词出现频次的程序。

姓名：张劲瞰

学号：PB16111485

日期：2019年5月25日

## 实验环境：

CPU: Intel® Core™ i7-6500U CPU @ 2.50GHz × 4

内存: 7.7 GiB

操作系统: Ubuntu 19.04 64bit

### 软件平台：

1. Hadoop 2.7.6

2. openjdk version "1.8.0\_212"

OpenJDK Runtime Environment (build 1.8.0\_212-8u212-b03-0ubuntu1.19.04.2-b03)

OpenJDK 64-Bit Server VM (build 25.212-b03, mixed mode)

## ## 目录

### 并行计算 上机报告

#### 目录

#### 算法设计与分析

##### 题目一

- 创建Hadoop用户并设置密码
- 为Hadoop用户增加管理员权限
- 切换到Hadoop用户下
- 安装SSH、配置SSH无密码登陆
- 安装JAVA环境
- 安装Hadoop
- Hadoop伪分布式配置
- 在HDFS上创建目录上传输入文件
- WordCount.java程序解析
- WordCount.java编译打包
- 在HDFS上执行WordCount.jar
- 获取输出，实验成功

##### 题目二

- 输入生成代码
- 在HDFS上创建目录上传输入文件

LenCount.java程序解析  
LenCount.java编译打包  
在HDFS上执行LenCount.jar  
获取输出，实验成功

总结

---

## ## 算法设计与分析

### ### 题目一

按照Hadoop安装运行说明文档中的指导，自己搭建伪分布式Hadoop环境，熟悉HDFS的常用操作(参考 [Hadoop实战](#) 第31-36页)，运行WordCount程序，得到统计结果。请详细写出你每一步的操作，最好有截图，最后的结果部分必须有截图。

实验步骤：

#### #### 创建Hadoop用户并设置密码

```
''' bash
1  $sudo useradd -m hadoop -s /bin/bash
2  $sudo passwd hadoop
```

#### #### 为Hadoop用户增加管理员权限

```
''' bash
1  $sudo adduser hadoop sudo
```

#### #### 切换到Hadoop用户下

#### #### 安装SSH、配置SSH无密码登陆

```
''' bash
1  $sudo apt-get update
2  $sudo apt-get install openssh-server
3  cd ~/.ssh/ # 若没有该目录，请先执行一次ssh localhost
4  ssh-keygen -t rsa # 会有提示，都按回车就可以
5  cat ./id_rsa.pub >> ./authorized_keys #
6  ssh localhost
```

```
hadoop@zjt-HP-Pavilion-Notebook:~$ ssh localhost
Welcome to Ubuntu 19.04 (GNU/Linux 5.0.0-15-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Ubuntu's Kubernetes 1.14 distributions can bypass Docker and use containerd
   directly, see https://bit.ly/ubuntu-containerd or try it now with

   snap install microk8s --classic

4 updates can be installed immediately.
4 of these updates are security updates.

Last login: Wed May 22 16:25:03 2019 from 127.0.0.1
hadoop@zjt-HP-Pavilion-Notebook:~$
```

#### #### 安装JAVA环境

```
''' bash
1  $ $JAVA_HOME
2  bash: /usr/lib/jvm/java-8-openjdk-amd64: 是一个目录
3  gedit ~/.bashrc
```

在文件最前面添加如下单独一行

```
''' bash
1  export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

```
WordCount.java  .bashrc  LenCount.java  Settings  settings.json
1  export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
2
```

```
''' bash
1  $ source ~/.bashrc  # 使变量设置生效
```

#### #### 安装Hadoop

1. 下载 [安装包](#)
2. 解压到安装路径

```
''' bash
1  $ sudo tar -zxf ~/下载/hadoop-2.7.6.tar.gz -C /usr/local  # 解压到/usr/local中
2  $ cd /usr/local/
3  $ sudo mv ./hadoop-2.7.4/ ./hadoop  # 将文件夹名改为hadoop
4  $ sudo chown -R hadoop ./hadoop  # 修改文件权限
5  $ cd /usr/local/hadoop  # 进入Hadoop 目录下
6  $ ./bin/hadoop version  # 在该目录下执行该命令
```

```
hadoop@zjt-HP-Pavilion-Notebook:~$ cd /usr/local/hadoop
hadoop@zjt-HP-Pavilion-Notebook:/usr/local/hadoop$ ./bin/hadoop version
Hadoop 2.7.6
Subversion https://shv@git-wip-us.apache.org/repos/asf/hadoop.git -r 085099c66cf28be31604560c376fa282e69282b8
Compiled by kshvachk on 2018-04-18T01:33Z
Compiled with protoc 2.5.0
From source with checksum 71e2695531cb3360ab74598755d036
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-2.7.6.jar
hadoop@zjt-HP-Pavilion-Notebook:/usr/local/hadoop$
```

#### #### Hadoop伪分布式配置

/usr/local/hadoop/etc/hadoop/core-site.xml 设置:

```
''' xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3  <!--
4      Licensed under the Apache License, Version 2.0 (the "License");
5      you may not use this file except in compliance with the License.
6      You may obtain a copy of the License at
7
8          http://www.apache.org/licenses/LICENSE-2.0
9
10     Unless required by applicable law or agreed to in writing, software
11     distributed under the License is distributed on an "AS IS" BASIS,
12     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13     See the License for the specific language governing permissions and
14     limitations under the License. See accompanying LICENSE file.
15 -->
16
17 <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20     <property>
21         <name>hadoop.tmp.dir</name>
22         <value>file:/usr/local/hadoop/tmp</value>
23         <description>Abase for other temporary directories.</description>
24     </property>
25     <property>
26         <name>fs.defaultFS</name>
27         <value>hdfs://localhost:9000</value>
28     </property>
29 </configuration>
```

/usr/local/hadoop/etc/hadoop/hdfs-site.xml 设置:

```
''' xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3  <!--
4      Licensed under the Apache License, Version 2.0 (the "License");
5      you may not use this file except in compliance with the License.
6      You may obtain a copy of the License at
7
8          http://www.apache.org/licenses/LICENSE-2.0
9
10     Unless required by applicable law or agreed to in writing, software
11     distributed under the License is distributed on an "AS IS" BASIS,
12     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```

13 See the License for the specific language governing permissions and
14 limitations under the License. See accompanying LICENSE file.
15 -->
16
17 <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20     <property>
21         <name>dfs.replication</name>
22         <value>1</value>
23     </property>
24     <property>
25         <name>dfs.namenode.name.dir</name>
26         <value>file:/usr/local/hadoop/tmp/dfs/name</value>
27     </property>
28     <property>
29         <name>dfs.datanode.data.dir</name>
30         <value>file:/usr/local/hadoop/tmp/dfs/data</value>
31     </property>
32 </configuration>

```

执行 **NameNode** 的格式化:

```

''' bash
1 /usr/local/hadoop$ ./bin/hdfs namenode -format

```

开启 **NameNode** 和 **DataNode** 守护进程

```

''' bash
1 /usr/local/hadoop$ ./sbin/start-dfs.sh

```

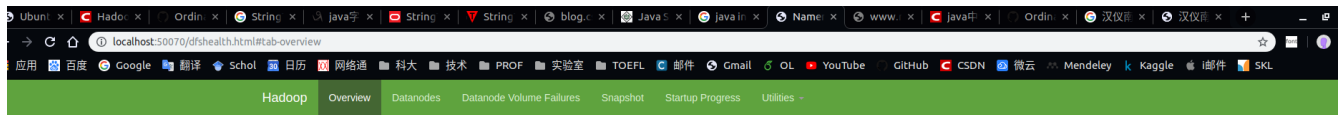
通过命令 **jps** 来判断是否成功启动

```

hadoop@zjt-HP-Pavilion-Notebook: /usr/local/hadoop$ ./sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-namenode-zjt-HP-Pavilion-Notebook.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hadoop-datanode-zjt-HP-Pavilion-Notebook.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-secondarynamenode-zjt-HP-Pavilion-Notebook.out
hadoop@zjt-HP-Pavilion-Notebook: /usr/local/hadoop$ jps
3062 NameNode
2166 org.eclipse.equinox.launcher_1.5.400.v20190514-1658.jar
2248 org.eclipse.equinox.launcher_1.5.400.v20190514-1658.jar
3579 Jps
3469 SecondaryNameNode
3247 DataNode
hadoop@zjt-HP-Pavilion-Notebook: /usr/local/hadoop$ █

```

访问 Web 界面 <http://localhost:50070> 查看 **NameNode** 和 **Datanode** 信息



## Overview 'localhost:9000' (active)

Started:	Wed May 22 16:57:37 CST 2019
Version:	2.7.6, r085099c66cf28be31604560c376fa282e9282b8
Compiled:	2018-04-18T01:33Z by kshvachk from branch-2.7.6
Cluster ID:	CID-cd0501ed-903c-471d-a0e8-608bb7d4f9d5
Block Pool ID:	BP-499000652-127.0.1.1-1558515450605

## Summary

Security is off.  
Safemode is off.

33 files and directories, 21 blocks = 54 total filesystem object(s).

Heap Memory used 83.25 MB of 270 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 47.48 MB of 48.53 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	146.65 GB
DFS Used:	220 KB (0%)
Non DFS Used:	49.27 GB
DFS Remaining:	89.86 GB (61.28%)
Block Pool Used:	220 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)

## #### 在HDFS上创建目录上传输入文件

```
''' bash
1 /usr/local/hadoop$ ./bin/hdfs dfs -mkdir input
2 /usr/local/hadoop$ ./bin/hdfs dfs -put ./etc/hadoop/*.xml input
3 /usr/local/hadoop$ ./bin/hdfs dfs -ls input
4 Found 8 items
5 -rw-r--r-- 1 hadoop supergroup 4436 2019-05-22 17:01 input/capacity-scheduler.xml
6 -rw-r--r-- 1 hadoop supergroup 1116 2019-05-22 17:01 input/core-site.xml
7 -rw-r--r-- 1 hadoop supergroup 9683 2019-05-22 17:01 input/hadoop-policy.xml
8 -rw-r--r-- 1 hadoop supergroup 1188 2019-05-22 17:01 input/hdfs-site.xml
9 -rw-r--r-- 1 hadoop supergroup 620 2019-05-22 17:01 input/httpfs-site.xml
10 -rw-r--r-- 1 hadoop supergroup 3518 2019-05-22 17:01 input/kms-acls.xml
11 -rw-r--r-- 1 hadoop supergroup 5540 2019-05-22 17:01 input/kms-site.xml
12 -rw-r--r-- 1 hadoop supergroup 690 2019-05-22 17:01 input/yarn-site.xml
13 /usr/local/hadoop$ ./bin/hadoop fs -ls -R
14 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:01 input
15 -rw-r--r-- 1 hadoop supergroup 4436 2019-05-22 17:01 input/capacity-scheduler.xml
16 -rw-r--r-- 1 hadoop supergroup 1116 2019-05-22 17:01 input/core-site.xml
17 -rw-r--r-- 1 hadoop supergroup 9683 2019-05-22 17:01 input/hadoop-policy.xml
18 -rw-r--r-- 1 hadoop supergroup 1188 2019-05-22 17:01 input/hdfs-site.xml
19 -rw-r--r-- 1 hadoop supergroup 620 2019-05-22 17:01 input/httpfs-site.xml
20 -rw-r--r-- 1 hadoop supergroup 3518 2019-05-22 17:01 input/kms-acls.xml
21 -rw-r--r-- 1 hadoop supergroup 5540 2019-05-22 17:01 input/kms-site.xml
22 -rw-r--r-- 1 hadoop supergroup 690 2019-05-22 17:01 input/yarn-site.xml
23 /usr/local/hadoop$ ./bin/hadoop fs -mkdir wordcount/input
24 /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input3.txt
wordcount/input
25 /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input2.txt
wordcount/input
26 /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input1.txt
wordcount/input
27 /usr/local/hadoop$ ./bin/hadoop fs -ls -R
28 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:35 input
29 -rw-r--r-- 1 hadoop supergroup 4436 2019-05-22 17:01 input/capacity-scheduler.xml
```

```

30 -rw-r--r-- 1 hadoop supergroup 1116 2019-05-22 17:01 input/core-site.xml
31 -rw-r--r-- 1 hadoop supergroup 9683 2019-05-22 17:01 input/hadoop-policy.xml
32 -rw-r--r-- 1 hadoop supergroup 1188 2019-05-22 17:01 input/hdfs-site.xml
33 -rw-r--r-- 1 hadoop supergroup 620 2019-05-22 17:01 input/httpfs-site.xml
34 -rw-r--r-- 1 hadoop supergroup 3518 2019-05-22 17:01 input/kms-acls.xml
35 -rw-r--r-- 1 hadoop supergroup 5540 2019-05-22 17:01 input/kms-site.xml
36 -rw-r--r-- 1 hadoop supergroup 690 2019-05-22 17:01 input/yarn-site.xml
37 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:36 wordcount
38 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:36 wordcount/input
39 -rw-r--r-- 1 hadoop supergroup 464 2019-05-22 17:36 wordcount/input/input1.txt
40 -rw-r--r-- 1 hadoop supergroup 511 2019-05-22 17:36 wordcount/input/input2.txt
41 -rw-r--r-- 1 hadoop supergroup 643 2019-05-22 17:36 wordcount/input/input3.txt

```

### #### WordCount.java程序解析

```

``` java
1  import org.apache.hadoop.fs.FileSystem;```java
2  import org.apache.hadoop.fs.FileSystem;
3  import org.apache.hadoop.conf.Configuration;
4  import org.apache.hadoop.fs.Path;
5  import org.apache.hadoop.io.IntWritable;
6  import org.apache.hadoop.io.Text;
7  import org.apache.hadoop.mapreduce.Job;
8  import org.apache.hadoop.mapreduce.Mapper;
9  import org.apache.hadoop.mapreduce.Reducer;
10 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
11 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
12 import org.apache.hadoop.util.GenericOptionsParser;
13
14 import java.io.IOException;
15 import java.util.StringTokenizer;
16
17 public class WordCount {
18     /*****
19      * MapReduceBase类:实现了Mapper和Reducer接口的基类（其中的方法只是实现接口，而未作任何事情）
20      * Mapper接口：
21      * WritableComparable接口：实现WritableComparable的类可以相互比较。所有被用作key的类应该实现此接口。
22      * Reporter 则可用于报告整个应用的运行进度，本例中未使用。
23      *****/
24     public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable>{
25         /*****
26          * LongWritable, IntWritable, Text 均是 Hadoop 中实现的用于封装 Java 数据类型的类，这些类实现了
27          * WritableComparable接口，
28          * 都能够被串行化从而便于在分布式环境中进行数据交换，你可以将它们分别视为long,int,String 的替代
29          * 品。
30          *****/
31         private final static IntWritable one = new IntWritable(1);
32         private Text word = new Text();//Text 实现了BinaryComparable类可以作为key值
33         /*****
34          * Mapper接口中的map方法：
35          * void map(K1 key, V1 value, OutputCollector<K2,V2> output, Reporter reporter)
36          * 映射一个单独的输入k/v对一个中间的k/v对
37          * 输出对不需要和输入对是相同的类型，输入对可以映射到0个或多个输出对。
38          * OutputCollector接口：收集Mapper和Reducer输出的<k,v>对。
39          * OutputCollector接口的collect(k, v)方法：增加一个(k,v)对到output
40          *****/

```

```

39     public void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
40         // 用StringTokenizer作为分词器, 对value进行分词
41         StringTokenizer itr = new StringTokenizer(value.toString());
42         // 遍历分词后结果
43         while (itr.hasMoreTokens()) {
44             // 将String设置入Text类型word
45             word.set(itr.nextToken());
46             // 将(word,1), 即(Text,IntWritable)写入上下文context, 供后续Reduce阶段使用
47             context.write(word, one);
48         }
49     }
50 }
51
52 // IntSumReducer作为Reduce阶段, 需要继承Reducer, 并重写reduce()函数
53 public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable> {
54     private IntWritable result = new IntWritable();
55
56     public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
InterruptedException {
57         int sum = 0;
58         // 遍历map阶段输出结果中的values中每个val, 累加至sum
59         for (IntWritable val : values) {
60             sum += val.get();
61         }
62         // 将sum设置入IntWritable类型result
63         result.set(sum);
64         // 通过上下文context的write()方法, 输出结果(key, result), 即(Text,IntWritable)
65         context.write(key, result);
66     }
67 }
68
69 public static void main(String[] args) throws Exception {
70     Configuration conf = new Configuration();
71     String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
72     if (otherArgs.length < 2) {
73         System.err.println("Usage: wordcount <in> [<in>...] <out>");
74         System.exit(2);
75     }
76     // 构造一个Job实例job, 并命名为"word count"
77     Job job = Job.getInstance(conf, "word count");
78     // 设置jar
79     job.setJarByClass(WordCount.class);
80     job.setMapperClass(TokenizerMapper.class); // 为job设置Mapper类
81     job.setCombinerClass(IntSumReducer.class); // 为job设置Combiner类
82     job.setReducerClass(IntSumReducer.class); // 为job设置Reduce类
83
84     job.setOutputKeyClass(Text.class);
85     job.setOutputValueClass(IntWritable.class);
86
87     for (int i = 0; i < otherArgs.length - 1; ++i) {
88         FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
89     }
90     FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length - 1]));
91     // 等待作业job运行完成并退出
92     System.exit(job.waitForCompletion(true) ? 0 : 1);
93 }
94 }
95
96 ...

```



```

97 import org.apache.hadoop.conf.Configuration;
98 import org.apache.hadoop.fs.Path;
99 import org.apache.hadoop.io.IntWritable;
100 import org.apache.hadoop.io.Text;
101 import org.apache.hadoop.mapreduce.Job;
102 import org.apache.hadoop.mapreduce.Mapper;
103 import org.apache.hadoop.mapreduce.Reducer;
104 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
105 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
106 import org.apache.hadoop.util.GenericOptionsParser;
107
108 import java.io.IOException;
109 import java.util.StringTokenizer;
110
111 public class WordCount {
112     /*****
113      * MapReduceBase类:实现了Mapper和Reducer接口的基类（其中的方法只是实现接口，而未作任何事情）
114      * Mapper接口：
115      * WritableComparable接口：实现WritableComparable的类可以相互比较。所有被用作key的类应该实现此接口。
116      * Reporter 则可用于报告整个应用的运行进度，本例中未使用。
117      *****/
118     public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable>{
119         /*****
120          * LongWritable, IntWritable, Text 均是 Hadoop 中实现的用于封装 Java 数据类型的类，这些类实现了
121          WritableComparable接口，
122          * 都能够被串行化从而便于在分布式环境中进行数据交换，你可以将它们分别视为long,int,String 的替代
123          品。
124          *****/
125         private final static IntWritable one = new IntWritable(1);
126         private Text word = new Text(); //Text 实现了BinaryComparable类可以作为key值
127         /*****
128          * Mapper接口中的map方法：
129          * void map(K1 key, V1 value, OutputCollector<K2,V2> output, Reporter reporter)
130          * 映射一个单独的输入k/v对一个中间的k/v对
131          * 输出对不需要和输入对是相同的类型，输入对可以映射到0个或多个输出对。
132          * OutputCollector接口：收集Mapper和Reducer输出的<k,v>对。
133          * OutputCollector接口的collect(k, v)方法：增加一个(k,v)对到output
134          *****/
135         public void map(Object key, Text value, Context context) throws IOException,
136             InterruptedException {
137             // 用StringTokenizer作为分词器，对value进行分词
138             StringTokenizer itr = new StringTokenizer(value.toString());
139             // 遍历分词后结果
140             while (itr.hasMoreTokens()) {
141                 // 将String设置入Text类型word
142                 word.set(itr.nextToken());
143                 // 将(word,1)，即(Text,IntWritable)写入上下文context，供后续Reduce阶段使用
144                 context.write(word, one);
145             }
146         }
147     }
148
149     // IntSumReducer作为Reduce阶段，需要继承Reducer，并重写reduce()函数
150     public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable> {
151         private IntWritable result = new IntWritable();
152
153         public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
154             InterruptedException {
155             int sum = 0;
156             // 遍历map阶段输出结果中的values中每个val，累加至sum

```

```

153         for (IntWritable val : values) {
154             sum += val.get();
155         }
156         // 将sum设置入IntWritable类型result
157         result.set(sum);
158         // 通过上下文context的write()方法, 输出结果(key, result), 即(Text,IntWritable)
159         context.write(key, result);
160     }
161 }
162
163 public static void main(String[] args) throws Exception {
164     Configuration conf = new Configuration();
165     String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
166     if (otherArgs.length < 2) {
167         System.err.println("Usage: wordcount <in> [<in>...] <out>");
168         System.exit(2);
169     }
170     // 构造一个Job实例job, 并命名为"word count"
171     Job job = Job.getInstance(conf, "word count");
172     // 设置jar
173     job.setJarByClass(WordCount.class);
174     job.setMapperClass(TokenizerMapper.class); // 为job设置Mapper类
175     job.setCombinerClass(IntSumReducer.class); // 为job设置Combiner类
176     job.setReducerClass(IntSumReducer.class); // 为job设置Reduce类
177
178     job.setOutputKeyClass(Text.class);
179     job.setOutputValueClass(IntWritable.class);
180
181     for (int i = 0; i < otherArgs.length - 1; ++i) {
182         FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
183     }
184     FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length - 1]));
185     // 等待作业job运行完成并退出
186     System.exit(job.waitForCompletion(true) ? 0 : 1);
187 }
188 }
189

```

#### #### WordCount.java编译打包

```

''' bash
1  $ cd wordcount_hadoop/
2  $ ls
3  WordCount.java
4  $ mkdir ./classes
5  $ javac -classpath /usr/local/hadoop/share/hadoop/common/hadoop-common-
2.7.6.jar:/usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-core-
2.7.6.jar:/usr/local/hadoop/share/hadoop/common/lib/commons-cli-1.2.jar -d ./classes/ WordCount.java
6  $ jar -cvf ./WordCount.jar -C ./classes .
7  已添加清单
8  正在添加: WordCount$TokenizerMapper.class(输入 = 1736) (输出 = 754)(压缩了 56%)
9  正在添加: WordCount$IntSumReducer.class(输入 = 1739) (输出 = 737)(压缩了 57%)
10 正在添加: WordCount$TokenizerMapper$CountersEnum.class(输入 = 1021) (输出 = 507)(压缩了 50%)
11 正在添加: WordCount.class(输入 = 1907) (输出 = 1038)(压缩了 45%)

```

#### #### 在HDFS上执行WordCount.jar

```

''' bash
1 /usr/local/hadoop$ ./bin/hadoop jar ~/ParallelComputingAlgorithm/MapReduce/wordcount_hadoop/WordCount.jar
  WordCount wordcount/input wordcount/output
2 /usr/local/hadoop$ ./bin/hadoop fs -ls -R
3 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:35 input
4 -rw-r--r-- 1 hadoop supergroup 4436 2019-05-22 17:01 input/capacity-scheduler.xml
5 -rw-r--r-- 1 hadoop supergroup 1116 2019-05-22 17:01 input/core-site.xml
6 -rw-r--r-- 1 hadoop supergroup 9683 2019-05-22 17:01 input/hadoop-policy.xml
7 -rw-r--r-- 1 hadoop supergroup 1188 2019-05-22 17:01 input/hdfs-site.xml
8 -rw-r--r-- 1 hadoop supergroup 620 2019-05-22 17:01 input/httpfs-site.xml
9 -rw-r--r-- 1 hadoop supergroup 3518 2019-05-22 17:01 input/kms-acls.xml
10 -rw-r--r-- 1 hadoop supergroup 5540 2019-05-22 17:01 input/kms-site.xml
11 -rw-r--r-- 1 hadoop supergroup 690 2019-05-22 17:01 input/yarn-site.xml
12 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:45 wordcount
13 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:36 wordcount/input
14 -rw-r--r-- 1 hadoop supergroup 464 2019-05-22 17:36 wordcount/input/input1.txt
15 -rw-r--r-- 1 hadoop supergroup 511 2019-05-22 17:36 wordcount/input/input2.txt
16 -rw-r--r-- 1 hadoop supergroup 643 2019-05-22 17:36 wordcount/input/input3.txt
17 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:45 wordcount/output
18 -rw-r--r-- 1 hadoop supergroup 0 2019-05-22 17:45 wordcount/output/_SUCCESS
19 -rw-r--r-- 1 hadoop supergroup 1274 2019-05-22 17:45 wordcount/output/part-r-00000
'''

```

#### 获取输出，实验成功

```

''' bash
1 /usr/local/hadoop$ ./bin/hadoop fs -get wordcount/output/part-r-00000
''' ~/ParallelComputingAlgorithm/MapReduce/

```

```
19/05/22 17:45:25 INFO mapred.LocalJobRunner: Finishing task: attempt_local613246123_0001_r_000000_0
19/05/22 17:45:25 INFO mapred.LocalJobRunner: reduce task executor complete.
19/05/22 17:45:26 INFO mapreduce.Job: Job job_local613246123_0001 running in uber mode : false
19/05/22 17:45:26 INFO mapreduce.Job: map 100% reduce 100%
19/05/22 17:45:26 INFO mapreduce.Job: Job job_local613246123_0001 completed successfully
19/05/22 17:45:26 INFO mapreduce.Job: Counters: 35
    File System Counters
        FILE: Number of bytes read=24714
        FILE: Number of bytes written=1208957
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=5033
        HDFS: Number of bytes written=1274
        HDFS: Number of read operations=33
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=6
    Map-Reduce Framework
        Map input records=3
        Map output records=255
        Map output bytes=2629
        Map output materialized bytes=2344
        Input split bytes=375
        Combine input records=255
        Combine output records=181
        Reduce input groups=140
        Reduce shuffle bytes=2344
        Reduce input records=181
        Reduce output records=140
        Spilled Records=362
        Shuffled Maps =3
        Failed Shuffles=0
        Merged Map outputs=3
        GC time elapsed (ms)=3
        Total committed heap usage (bytes)=1559756800
```

```
''' bash
1 /usr/local/hadoop$ cat ~/ParallelComputingAlgorithm/MapReduce/part-r-00000
2 Although 1
3 As 1
4 First 1
5 In 1
6 Maybe 2
7 Or 1
8 Since 1
9 Studies 1
10 The 1
11 Therefore 1
12 Therefore, 1
13 a 2
14 abilities 1
15 about 1
16 actively 1
17 ages. 1
18 all, 1
19 an 1
20 and 2
21 are 3
22 babies 2
23 be 5
24 because 1
25 being 4
26 better 2
```

27 birth 2  
28 both 1  
29 brothers 1  
30 busy 1  
31 care 1  
32 cause 1  
33 certain 1  
34 changes 1  
35 children 2  
36 cognitive 1  
37 compared 1  
38 cortisol 3  
39 could 3  
40 danger 1  
41 different 2  
42 direct 1  
43 effects 2  
44 emotional 1  
45 excited 2  
46 excitement 2  
47 expecting 1  
48 explain 1  
49 explanation 1  
50 exposed 1  
51 feel 1  
52 first-time 2  
53 firstborn 4  
54 firstborn. 2  
55 for 2  
56 from 2  
57 geared 1  
58 genes 2  
59 genes, 1  
60 genetically 2  
61 happen 1  
62 have 1  
63 high 2  
64 higher 2  
65 human 1  
66 identical 1  
67 in 1  
68 infants 3  
69 infants, 1  
70 infants. 1  
71 intense 1  
72 is 1  
73 it 1  
74 larger 1  
75 lead 1  
76 level 5  
77 levels 2  
78 may 2  
79 monkey 1  
80 monkeys 1  
81 more 4  
82 mothers 2  
83 mothers, 1  
84 necessary 1  
85 nervous 1  
86 nervous. 1

```
87 not 2
88 of 19
89 older 2
90 or 5
91 order 1
92 order. 1
93 out 1
94 parent. 1
95 possible 1
96 potential 1
97 pregnant 2
98 random 1
99 rather 1
100 relatively 2
101 released. 1
102 releasing 1
103 responded 1
104 result 2
105 returning 1
106 sample 1
107 samples 1
108 sensitive 1
109 share 1
110 siblings 2
111 similar 1
112 simply 1
113 sisters. 1
114 situations. 1
115 size 1
116 small, 1
117 stimulating 1
118 stimulation 3
119 stimulation. 2
120 stress 1
121 studies 1
122 taking 1
123 tease 1
124 terms 1
125 than 1
126 that 2
127 the 19
128 their 4
129 they 2
130 to 9
131 too 1
132 towards 1
133 unfamiliar 1
134 usually 1
135 was 1
136 we 1
137 were 4
138 when 1
139 which 1
140 with 3
141 younger 2
```

### ### 题目二

实现一个统计输入文件中各个长度的单词出现频次的程序。

#### #### 输入生成代码

```
''' python
1  import numpy.random as random
2
3  def generate_random_str(randomlength=16):
4      """
5      生成一个指定长度的随机字符串
6      """
7      random_str = ''
8      base_str = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
9      length = len(base_str) - 1
10     for i in range(randomlength):
11         random_str += base_str[random.randint(0, length)]
12     return random_str
13
14     output = open("./input.txt", "w")
15     count = random.randint(20, 50)
16     for i in range(count):
17         output.write(generate_random_str(random.randint(0, 10)) + " ")
18
19     output.close()
```

#### #### 在HDFS上创建目录上传输入文件

```
''' bash
1  /usr/local/hadoop$ ./bin/hadoop fs -mkdir lencount
2  /usr/local/hadoop$ ./bin/hadoop fs -mkdir lencount/input
3  /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input4.txt lencount/input
4  /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input5.txt lencount/input
5  /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input6.txt lencount/input
6  /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input7.txt lencount/input
7  /usr/local/hadoop$ ./bin/hadoop fs -put ~/ParallelComputingAlgorithm/MapReduce/input8.txt lencount/input
8  /usr/local/hadoop$ ./bin/hadoop fs -ls -R
9  drwxr-xr-x   - hadoop supergroup          0 2019-05-22 17:35 input
10 -rw-r--r--   1 hadoop supergroup    4436 2019-05-22 17:01 input/capacity-scheduler.xml
11 -rw-r--r--   1 hadoop supergroup    1116 2019-05-22 17:01 input/core-site.xml
12 -rw-r--r--   1 hadoop supergroup    9683 2019-05-22 17:01 input/hadoop-policy.xml
13 -rw-r--r--   1 hadoop supergroup    1188 2019-05-22 17:01 input/hdfs-site.xml
14 -rw-r--r--   1 hadoop supergroup     620 2019-05-22 17:01 input/httpfs-site.xml
15 -rw-r--r--   1 hadoop supergroup    3518 2019-05-22 17:01 input/kms-acls.xml
16 -rw-r--r--   1 hadoop supergroup    5540 2019-05-22 17:01 input/kms-site.xml
17 -rw-r--r--   1 hadoop supergroup     690 2019-05-22 17:01 input/yarn-site.xml
18 drwxr-xr-x   - hadoop supergroup          0 2019-05-22 20:47 lencount
19 drwxr-xr-x   - hadoop supergroup          0 2019-05-22 20:48 lencount/input
20 -rw-r--r--   1 hadoop supergroup      58 2019-05-22 20:47 lencount/input/input4.txt
21 -rw-r--r--   1 hadoop supergroup     107 2019-05-22 20:48 lencount/input/input5.txt
22 -rw-r--r--   1 hadoop supergroup      77 2019-05-22 20:48 lencount/input/input6.txt
23 -rw-r--r--   1 hadoop supergroup     116 2019-05-22 20:48 lencount/input/input7.txt
24 -rw-r--r--   1 hadoop supergroup     127 2019-05-22 20:48 lencount/input/input8.txt
25 drwxr-xr-x   - hadoop supergroup          0 2019-05-22 17:45 wordcount
26 drwxr-xr-x   - hadoop supergroup          0 2019-05-22 17:36 wordcount/input
27 -rw-r--r--   1 hadoop supergroup     464 2019-05-22 17:36 wordcount/input/input1.txt
28 -rw-r--r--   1 hadoop supergroup     511 2019-05-22 17:36 wordcount/input/input2.txt
29 -rw-r--r--   1 hadoop supergroup     643 2019-05-22 17:36 wordcount/input/input3.txt
30 drwxr-xr-x   - hadoop supergroup          0 2019-05-22 17:45 wordcount/output
31 -rw-r--r--   1 hadoop supergroup          0 2019-05-22 17:45 wordcount/output/_SUCCESS
32 -rw-r--r--   1 hadoop supergroup    1274 2019-05-22 17:45 wordcount/output/part-r-00000
```

#### #### LenCount.java程序解析

```
``` java
1  import org.apache.hadoop.fs.FileSystem;
2  import org.apache.hadoop.conf.Configuration;
3  import org.apache.hadoop.fs.Path;
4  import org.apache.hadoop.io.IntWritable;
5  import org.apache.hadoop.io.Text;
6  import org.apache.hadoop.mapreduce.Job;
7  import org.apache.hadoop.mapreduce.Mapper;
8  import org.apache.hadoop.mapreduce.Reducer;
9  import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
10 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
11 import org.apache.hadoop.util.GenericOptionsParser;
12
13 import java.io.IOException;
14 import java.util.StringTokenizer;
15
16 public class LenCount {
17     public static class CounterMapper extends Mapper<Object, Text, Text, IntWritable>{
18         private final static IntWritable one = new IntWritable(1);
19         private Text word_len = new Text();
20         public void map(Object key, Text value, Context context) throws IOException, InterruptedException
21     {
22         // 用StringTokenizer作为分词器，对value进行分词
23         StringTokenizer itr = new StringTokenizer(value.toString());
24         // 遍历分词后结果
25         while (itr.hasMoreTokens()) {
26             // 将String设置入Text类型word
27             word_len.set(Integer.toString(itr.nextToken().length()));
28             // 将(word,1)，即(Text,IntWritable)写入上下文context，供后续Reduce阶段使用
29             context.write(word_len, one);
30         }
31     }
32     public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable> {
33         private IntWritable result = new IntWritable();
34
35         public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
36         InterruptedException {
37             int sum = 0;
38             // 遍历map阶段输出结果中的values中每个val，累加至sum
39             for (IntWritable val : values) {
40                 sum += val.get();
41             }
42             // 将sum设置入IntWritable类型result
43             result.set(sum);
44             // 通过上下文context的write()方法，输出结果(key, result)，即(Text,IntWritable)
45             context.write(key, result);
46         }
47
48         public static void main(String[] args) throws Exception {
49             Configuration conf = new Configuration();
50             String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
51             if (otherArgs.length < 2) {
52                 System.err.println("Usage: wordlen <in> [<in>...] <out>");
53                 System.exit(2);
54             }
55         }
56     }
57 }
```



```

54     }
55     // 构造一个Job实例job, 并命名为"wordlen count"
56     Job job = Job.getInstance(conf, "wordlen count");
57     // 设置jar
58     job.setJarByClass(LenCount.class);
59     job.setMapperClass(CounterMapper.class); // 为job设置Mapper类
60     job.setCombinerClass(IntSumReducer.class); // 为job设置Combiner类
61     job.setReducerClass(IntSumReducer.class); // 为job设置Reduce类
62
63     job.setOutputKeyClass(Text.class);
64     job.setOutputValueClass(IntWritable.class);
65
66     for (int i = 0; i < otherArgs.length - 1; ++i) {
67         FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
68     }
69     FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length - 1]));
70     // 等待作业job运行完成并退出
71     System.exit(job.waitForCompletion(true) ? 0 : 1);
72 }
73 }
74

```

#### #### LenCount.java编译打包

```

''' bash
1  $ cd ../lencount/
2  $ mkdir ./classes
3  $ javac -classpath /usr/local/hadoop/share/hadoop/common/hadoop-common-
  2.7.6.jar:/usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-core-
  2.7.6.jar:/usr/local/hadoop/share/hadoop/common/lib/commons-cli-1.2.jar -d ./classes/ LenCount.java
4  $ jar -cvf ./LenCount.jar -C ./classes .
5  已添加清单
6  正在添加: LenCount.class(输入 = 1902) (输出 = 1034)(压缩了 45%)
7  正在添加: LenCount$CounterMapper.class(输入 = 1843) (输出 = 805)(压缩了 56%)
8  正在添加: LenCount$IntSumReducer.class(输入 = 1736) (输出 = 739)(压缩了 57%)

```

#### #### 在HDFS上执行LenCount.jar

```

''' bash
1  /usr/local/hadoop$ ./bin/hadoop jar ~/ParallelComputingAlgorithm/MapReduce/lencount/LenCount.jar LenCount
  lencount/input lencount/output
2  ...
3  /usr/local/hadoop$ ./bin/hadoop fs -ls -R
4  drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:35 input
5  -rw-r--r-- 1 hadoop supergroup 4436 2019-05-22 17:01 input/capacity-scheduler.xml
6  -rw-r--r-- 1 hadoop supergroup 1116 2019-05-22 17:01 input/core-site.xml
7  -rw-r--r-- 1 hadoop supergroup 9683 2019-05-22 17:01 input/hadoop-policy.xml
8  -rw-r--r-- 1 hadoop supergroup 1188 2019-05-22 17:01 input/hdfs-site.xml
9  -rw-r--r-- 1 hadoop supergroup 620 2019-05-22 17:01 input/httpfs-site.xml
10 -rw-r--r-- 1 hadoop supergroup 3518 2019-05-22 17:01 input/kms-acls.xml
11 -rw-r--r-- 1 hadoop supergroup 5540 2019-05-22 17:01 input/kms-site.xml
12 -rw-r--r-- 1 hadoop supergroup 690 2019-05-22 17:01 input/yarn-site.xml
13 drwxr-xr-x - hadoop supergroup 0 2019-05-22 20:49 lencount
14 drwxr-xr-x - hadoop supergroup 0 2019-05-22 20:48 lencount/input
15 -rw-r--r-- 1 hadoop supergroup 58 2019-05-22 20:47 lencount/input/input4.txt
16 -rw-r--r-- 1 hadoop supergroup 107 2019-05-22 20:48 lencount/input/input5.txt
17 -rw-r--r-- 1 hadoop supergroup 77 2019-05-22 20:48 lencount/input/input6.txt

```

```

18 -rw-r--r-- 1 hadoop supergroup 116 2019-05-22 20:48 lencount/input/input7.txt
19 -rw-r--r-- 1 hadoop supergroup 127 2019-05-22 20:48 lencount/input/input8.txt
20 drwxr-xr-x - hadoop supergroup 0 2019-05-22 20:49 lencount/output
21 -rw-r--r-- 1 hadoop supergroup 0 2019-05-22 20:49 lencount/output/_SUCCESS
22 -rw-r--r-- 1 hadoop supergroup 40 2019-05-22 20:49 lencount/output/part-r-00000
23 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:45 wordcount
24 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:36 wordcount/input
25 -rw-r--r-- 1 hadoop supergroup 464 2019-05-22 17:36 wordcount/input/input1.txt
26 -rw-r--r-- 1 hadoop supergroup 511 2019-05-22 17:36 wordcount/input/input2.txt
27 -rw-r--r-- 1 hadoop supergroup 643 2019-05-22 17:36 wordcount/input/input3.txt
28 drwxr-xr-x - hadoop supergroup 0 2019-05-22 17:45 wordcount/output
29 -rw-r--r-- 1 hadoop supergroup 0 2019-05-22 17:45 wordcount/output/_SUCCESS
30 -rw-r--r-- 1 hadoop supergroup 1274 2019-05-22 17:45 wordcount/output/part-r-00000

```

#### #### 获取输出，实验成功

```

''' bash
1 /usr/local/hadoop$ ./bin/hadoop fs -get lencount/output/part-r-00000
~/ParallelComputingAlgorithm/MapReduce/lencount/
2 /usr/local/hadoop$ cat ~/ParallelComputingAlgorithm/MapReduce/lencount/part-r-00000

```

```

hadoop@zjt-HP-Pavilion-Notebook:/usr/local/hadoop$ cat ~/ParallelComputingAlgorithm/MapReduce/lencount/part-r-00000
1      13
2      9
3      11
4      11
5      10
6      6
7      9
8      8
9      8
hadoop@zjt-HP-Pavilion-Notebook:/usr/local/hadoop$ 

```

## ## 总结

通过算法实现锻炼了并行思维，熟悉了MapReduce分布式并行环境的使用。