《并行计算》上机报告

姓名：	魏钊	学号：	PB18111699	日期：	2021/6/2
上机题目：	Hadoop 编程实验				

实验环境：
CPU：i7-8750H; 内存: 16GB;操作系统 Ubuntu20.04;软件平台:Visual Studio 2017

一、算法设计与分析：

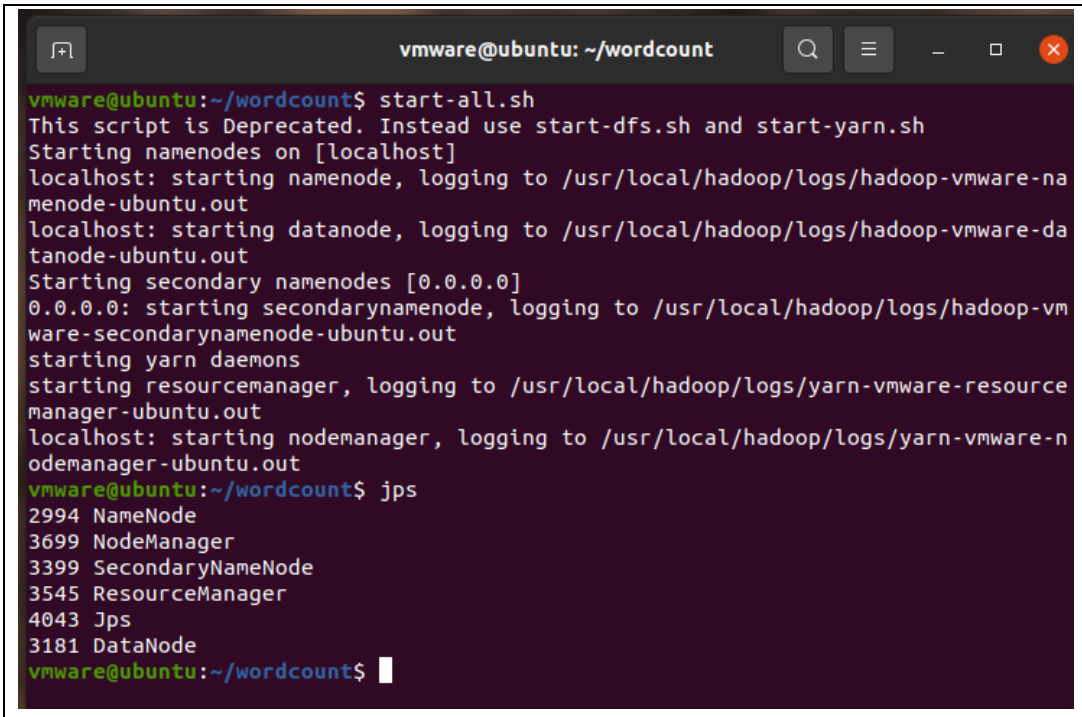
题目一:
按照 Hadoop 安装运行说明文档中的指导自己搭建伪分布式 Hadoop 环境，熟悉 HDFS 的常用操作（参考 Hdoop 实战 第 31-36 页），运行 WordCount 程序，得到统计结果。
题目二：
实现一个统计输入文件中各个长度的单词出现频次的程序。

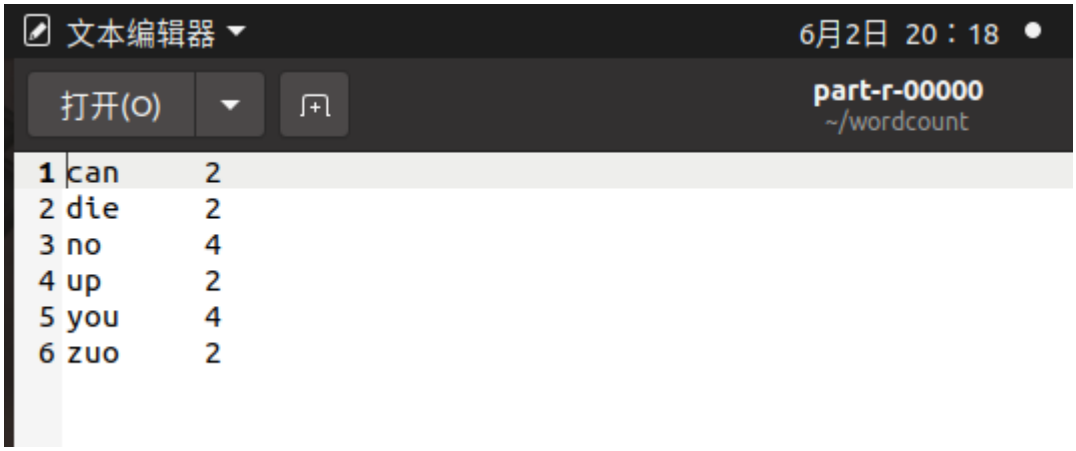
二、核心代码：

题目一:
略
题目二：
修改 MAP 即可，将映射由单词内容改为长度

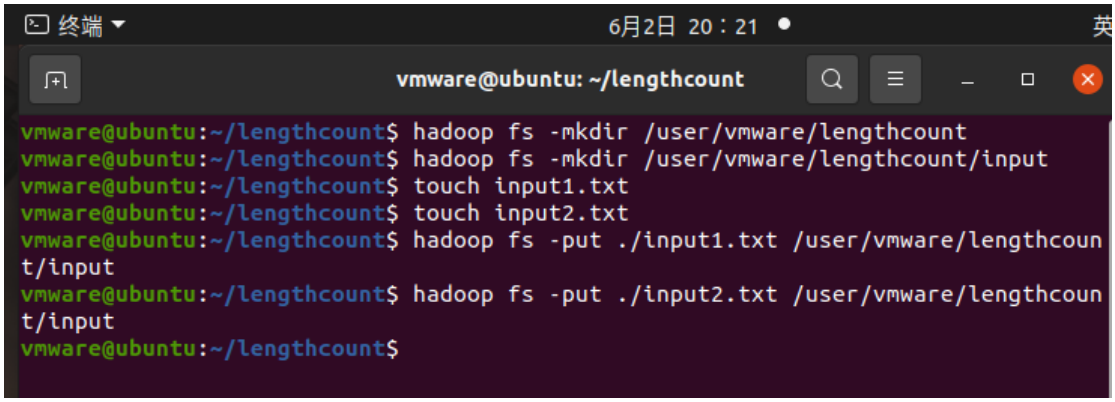
```
  public void map(Object key, Text value, Context context
                  ) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
      word.set(itr.nextToken());
      Text truth = new Text(Integer.toString(word.getLength()));
      context.write(truth, one);
    }
  }
}
```

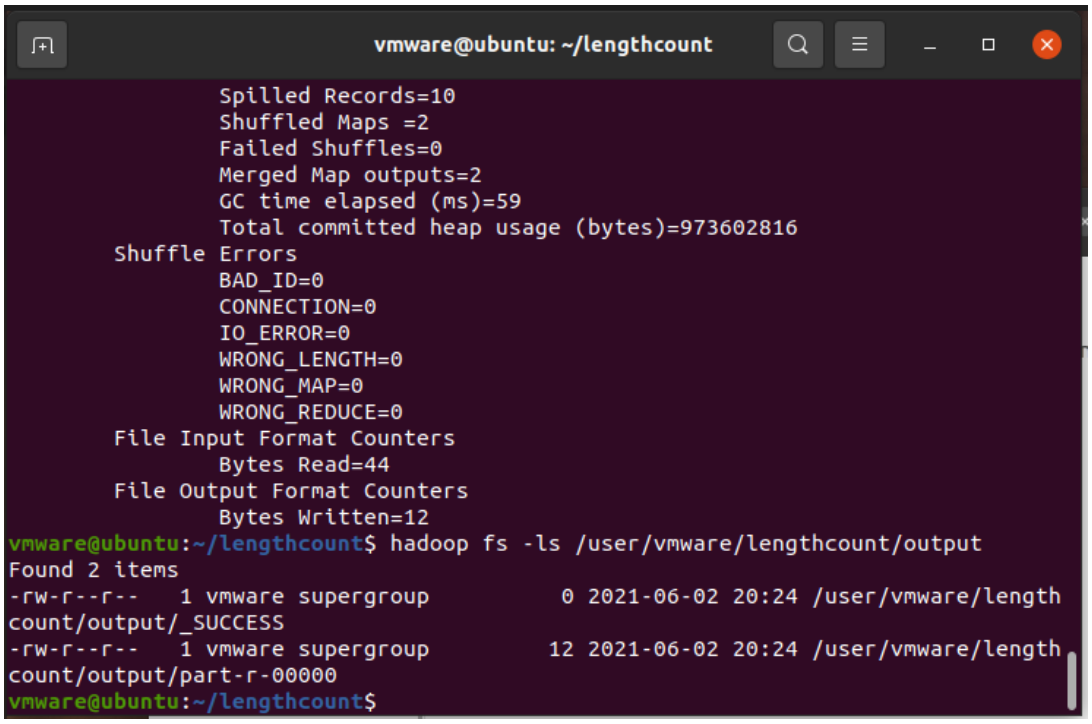
三、结果与分析：
题目一:
按照说明配置完成：

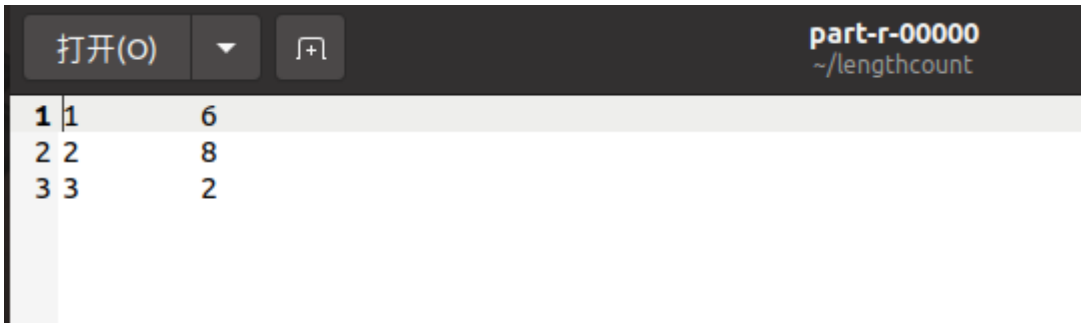

vmware@ubuntu: ~/wordcount
vmware@ubuntu:~/wordcount$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-vmware-namenode-ubuntu.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-vmware-datanode-ubuntu.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-vmware-secondarynamenode-ubuntu.out
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-vmware-resourcemanager-ubuntu.out
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-vmware-nodemanager-ubuntu.out
vmware@ubuntu:~/wordcount$ jps
2994 NameNode
3699 NodeManager
3399 SecondaryNameNode
3545 ResourceManager
4043 Jps
3181 DataNode
vmware@ubuntu:~/wordcount$


文本编辑器
6月2日 20：18
打开(O)
part-r-00000
~/wordcount
1 can 2
2 die 2
3 no 4
4 up 2
5 you 4
6 zuo 2

题目二：


终端
6月2日 20：21
英
vmware@ubuntu: ~/lengthcount
vmware@ubuntu:~/lengthcount$ hadoop fs -mkdir /user/vmware/lengthcount
vmware@ubuntu:~/lengthcount$ hadoop fs -mkdir /user/vmware/lengthcount/input
vmware@ubuntu:~/lengthcount$ touch input1.txt
vmware@ubuntu:~/lengthcount$ touch input2.txt
vmware@ubuntu:~/lengthcount$ hadoop fs -put ./input1.txt /user/vmware/lengthcount/input
vmware@ubuntu:~/lengthcount$ hadoop fs -put ./input2.txt /user/vmware/lengthcount/input
vmware@ubuntu:~/lengthcount$





四、备注（* 可选）：

有可能影响结论的因素：

总结：

通过分布式系统实现并行

附录（源代码）	算法源代码（C/C++/JAVA 描述） import java.io.IOException; import java.util.StringTokenizer; import org.apache.hadoop.conf.Configuration; import org.apache.hadoop.fs.Path; import org.apache.hadoop.io.IntWritable; import org.apache.hadoop.io.Text; import org.apache.hadoop.mapreduce.Job; import org.apache.hadoop.mapreduce.Mapper; import org.apache.hadoop.mapreduce.Reducer; import org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

```
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        Text truth = new Text(Integer.toString(word.getLength()));
        context.write(truth, one);
      }
    }
  }

  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs = new GenericOptionsParser(conf,
args).getRemainingArgs();
    if (otherArgs.length != 2) {
      System.err.println("Usage: wordcount <in> <out>");
      System.exit(2);
```

```
    }
    Job job = new Job(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```