Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Отчёт по практической работе № 1

По дисциплине: «Распределенная обработка информации»

Тема: «MapReduce: подсчет частоты слов»

Выполнил:
студент гр. МГ-101
Лукошкин В.Ю.

Проверил:
Профессор Кафедры ВС
Курносов М.Г.

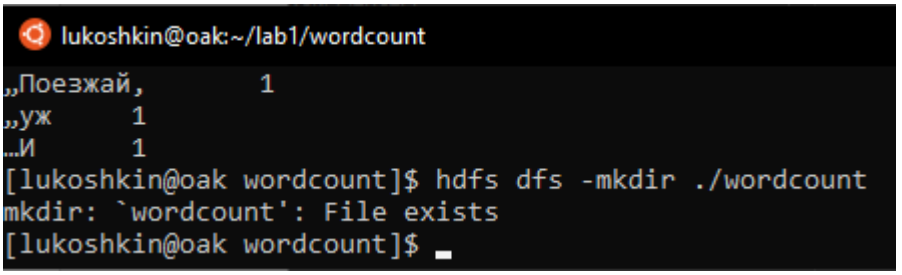
Новосибирск 2021

Задание.

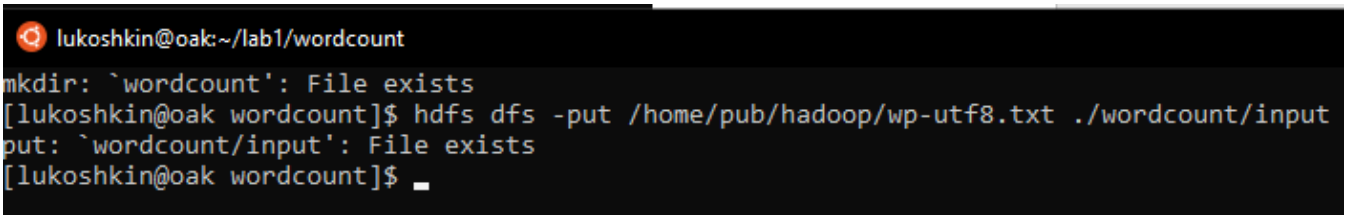
С помощью Hadoop MapReduce подсчитать количество вхождений каждого слова в заданном входном наборе.

Ход работы.

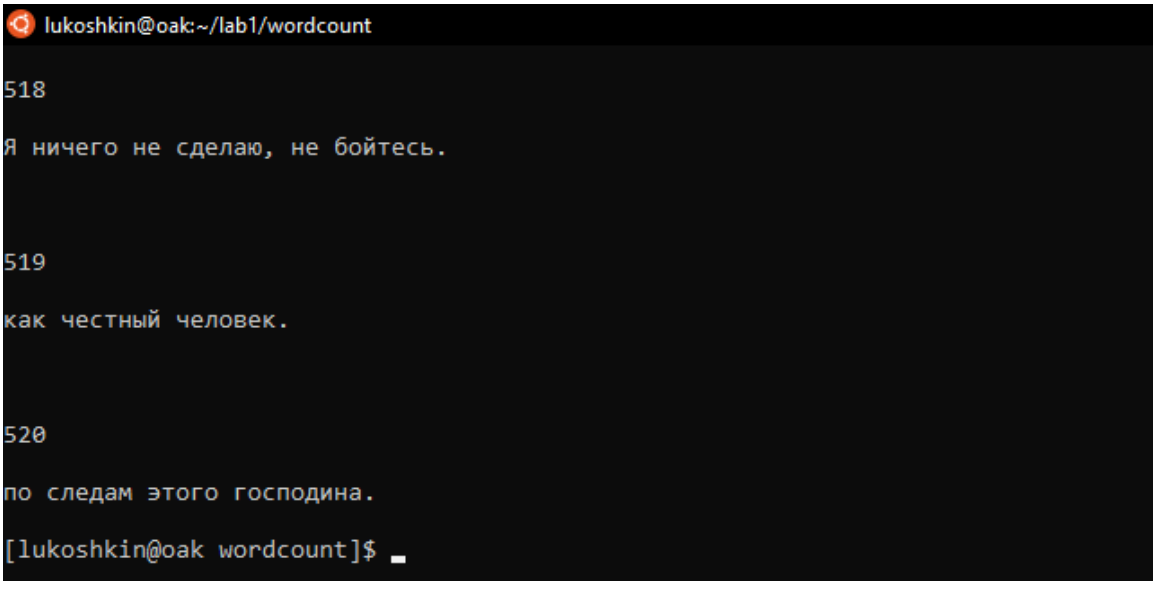
Создаем каталог в HDFS:



Для подсчета количества слов будет использоваться файл *wp-utf8.txt* из директории /home/pub/hadoop/. Копируем файл в HDFS:



Проверим содержимое файла:



Копируем пример, который находится в /home/pub/hadoop/lab1/wordcount, в домашнюю директорию, компилируем WordCount.java через команду ./build.sh и запускаем задание ./start-job.sh:

```
[lukoshkin@oak wordcount]$ ./start-job.sh
Deleted wordcount/output
2021-05-31 02:59:17,721 INFO client.RMProxy: Connecting to ResourceManager at /192.168.2.254:8032
2021-05-31 02:59:18,211 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/lukoshkin/.staging/job_1612430908623_1854
2021-05-31 02:59:18,338 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-05-31 02:59:18,503 INFO input.FileInputFormat: Total input files to process : 1
2021-05-31 02:59:18,568 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-05-31 02:59:18,651 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-05-31 02:59:18,676 INFO mapreduce.JobSubmitter: number of splits:1
2021-05-31 02:59:18,730 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
2021-05-31 02:59:18,835 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2021-05-31 02:59:18,851 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1612430908623_1854
2021-05-31 02:59:18,851 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-05-31 02:59:19,040 INFO conf.Configuration: resource-types.xml not found
2021-05-31 02:59:19,040 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-05-31 02:59:19,109 INFO impl.YarnClientImpl: Submitted application application_1612430908623_1854
2021-05-31 02:59:19,151 INFO mapreduce.Job: The url to track the job: http://oak:8088/proxy/application_1612430908623_1854/
2021-05-31 02:59:19,151 INFO mapreduce.Job: Running job: job_1612430908623_1854
2021-05-31 02:59:26,241 INFO mapreduce.Job: Job job_1612430908623_1854 running in uber mode : false
2021-05-31 02:59:26,242 INFO mapreduce.Job:  map 0% reduce 0%
2021-05-31 02:59:32,309 INFO mapreduce.Job:  map 100% reduce 0%
2021-05-31 02:59:38,344 INFO mapreduce.Job:  map 100% reduce 100%
2021-05-31 02:59:39,358 INFO mapreduce.Job: Job job_1612430908623_1854 completed successfully
2021-05-31 02:59:39,463 INFO mapreduce.Job: Counters: 55
```

После выполнения будет создан каталог ./wordcount/output в HDFS. Проверим содержимое файлов командой hdfs dfs -cat ./wordcount/output/part*:

```
– Хоть  3
– Христос       1
– Целуй 1
– Целый 1
– Час   1
– Чего  2
– Чего? –       1
– Чег’т 2
– Через 2
– Честное       1
– Честное,      1
– Чистое        1
– Что?  6
– Что?..        1
– Чудо  1
– Чудо! –       1
– Шшшш! –       1
– Э,    3
– Эй!   1
– Эк    2
– Экая  1
– Эта?  1
– Этого 5
– Этою  1
– Эх,   3
– Я     168
– Я,    1
– Я? –  2
– Я…    3
–       2
„Поезжай,       1
„уж     1
…И      1
[lukoshkin@oak wordcount]$ hdfs dfs -cat ./wordcount/output/part*
```

В результате получаем подсчитанное значение частоты каждого слова из исходного файла.

Исходный код программы

WordCount.java

```
package ddpcourse.mapred;

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());
        context.write(word, 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();
        conf.set("mapreduce.framework.name", "yarn");
        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);
  }
}
```