Unique Word Count with MapReduce*

Montasser AKERMI

Last Update: February, 2025

*The course material is hosted at `https://akermi.org/c2/`.

Learning objectives

Write a MapReduce application that counts the number unique words in a corpus of text of one or multiple files.

Input data

```
my dreams my dreams
what has become of their sweetness
what indeed has become of my youth
```

Output data

```
Unique words  10
```

Solution

Listing 1: UniqueWordCount Class

```
1  import java.io.IOException;
2  import java.util.StringTokenizer;
3
4  import org.apache.hadoop.conf.Configuration;
5  import org.apache.hadoop.fs.Path;
6  import org.apache.hadoop.io.IntWritable;
7  import org.apache.hadoop.io.Text;
8  import org.apache.hadoop.mapreduce.Job;
9  import org.apache.hadoop.mapreduce.Mapper;
10 import org.apache.hadoop.mapreduce.Reducer;
11 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
12 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
13
14 public class UniqueWordCount {
15
16   public static class TokenizerMapper
17       extends Mapper<Object, Text, Text, IntWritable> {
18
19     private final static IntWritable one = new IntWritable(1);
20     private Text word = new Text();
21
22     public void map(Object key, Text value, Context context
23     ) throws IOException, InterruptedException {
24       StringTokenizer itr = new StringTokenizer(value.toString());
25       while (itr.hasMoreTokens()) {
26         word.set(itr.nextToken());
27         context.write(word, one);
28       }
29     }
30   }
31
32   public static class LineMapper
33       extends Mapper<Object, Text, Text, IntWritable> {
34     private final static IntWritable one = new IntWritable(1);
35     private final static Text word = new Text("Unique words");
```

```
36
37      public void map(Object key, Text value, Context context
38      ) throws IOException, InterruptedException {
39        context.write(word, one);
40      }
41    }
42
43    public static class IntSumReducer
44        extends Reducer<Text, IntWritable, Text, IntWritable> {
45      private IntWritable result = new IntWritable();
46
47      public void reduce(Text key, Iterable<IntWritable> values,
48                         Context context
49      ) throws IOException, InterruptedException {
50        int sum = 0;
51        for (IntWritable val : values) {
52          sum += val.get();
53        }
54        result.set(sum);
55        context.write(key, result);
56      }
57    }
58
59    public static void main(String[] args) throws Exception {
60      Configuration conf1 = new Configuration();
61      Job job1 = Job.getInstance(conf1, "Distinct word count - step 1");
62      job1.setJarByClass(UniqueWordCount.class);
63      job1.setMapperClass(TokenizerMapper.class);
64      job1.setCombinerClass(IntSumReducer.class);
65      job1.setReducerClass(IntSumReducer.class);
66      job1.setOutputKeyClass(Text.class);
67      job1.setOutputValueClass(IntWritable.class);
68      FileInputFormat.addInputPath(job1, new Path(args[0]));
69      FileOutputFormat.setOutputPath(job1, new Path(args[1]));
70      job1.waitForCompletion(true);
71
72      Configuration conf2 = new Configuration();
73      Job job2 = Job.getInstance(conf2, "Distinct word count - step 2");
74      job2.setJarByClass(UniqueWordCount.class);
75      job2.setMapperClass(LineMapper.class);
76      job2.setCombinerClass(IntSumReducer.class);
77      job2.setReducerClass(IntSumReducer.class);
78      job2.setOutputKeyClass(Text.class);
79      job2.setOutputValueClass(IntWritable.class);
80      FileInputFormat.addInputPath(job2, new Path(args[1]));
81      FileOutputFormat.setOutputPath(job2, new Path(args[2]));
82      System.exit(job2.waitForCompletion(true) ? 0 : 1);
83    }
84  }
```

Note: It is highly recommended to manually write the code instead of copy/pasting.