CURSOS DE VERANO 2014

APROXIMACIÓN PRÁCTICA A LA CIENCIA DE DATOS Y BIG DATA: HERRAMIENTAS KNIME, R, HADOOP Y MAHOUT.

Entorno de Procesamiento Hadoop – Caso Práctico 2

Sara Del Río García

Ejemplo: MaxTemperature

- Tenemos un fichero de gran tamaño que contiene datos climatológicos procedentes del Centro Nacional de Datos Climáticos (*National Climatic Data Center*, http://www.ncdc.noaa.gov/).

```
0067011990999991950051507004+68750+023550FM-12+038299999V0203301N00671220001CN9999999N9+00001+99999999999
0043011990999991950051512004+68750+023550FM-12+038299999V0203201N00671220001CN9999999N9+00221+99999999999
0043011990999991950051518004+68750+023550FM-12+038299999V0203201N00261220001CN9999999N9-00111+99999999999
0043012650999991949032412004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+01111+99999999999
0043012650999991949032418004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+00781+99999999999
```

- **OBJETIVO**: ¿Cuál es la temperatura más alta registrada para cada año en el conjunto de datos?

MaxTemperature usando MapReduce

Pseudocódigo:

```
map(key, value):
// key: document ID; value: text of document
        emit(year, temperature);


reduce(key, value-list):
// key: a year; value-list: a list of temperatures
        maxValue = Integer.MIN_VALUE;
        FOR (each count v on value-list)
                maxValue = Max(maxValue , v);
        emit(key, maxValue);
```

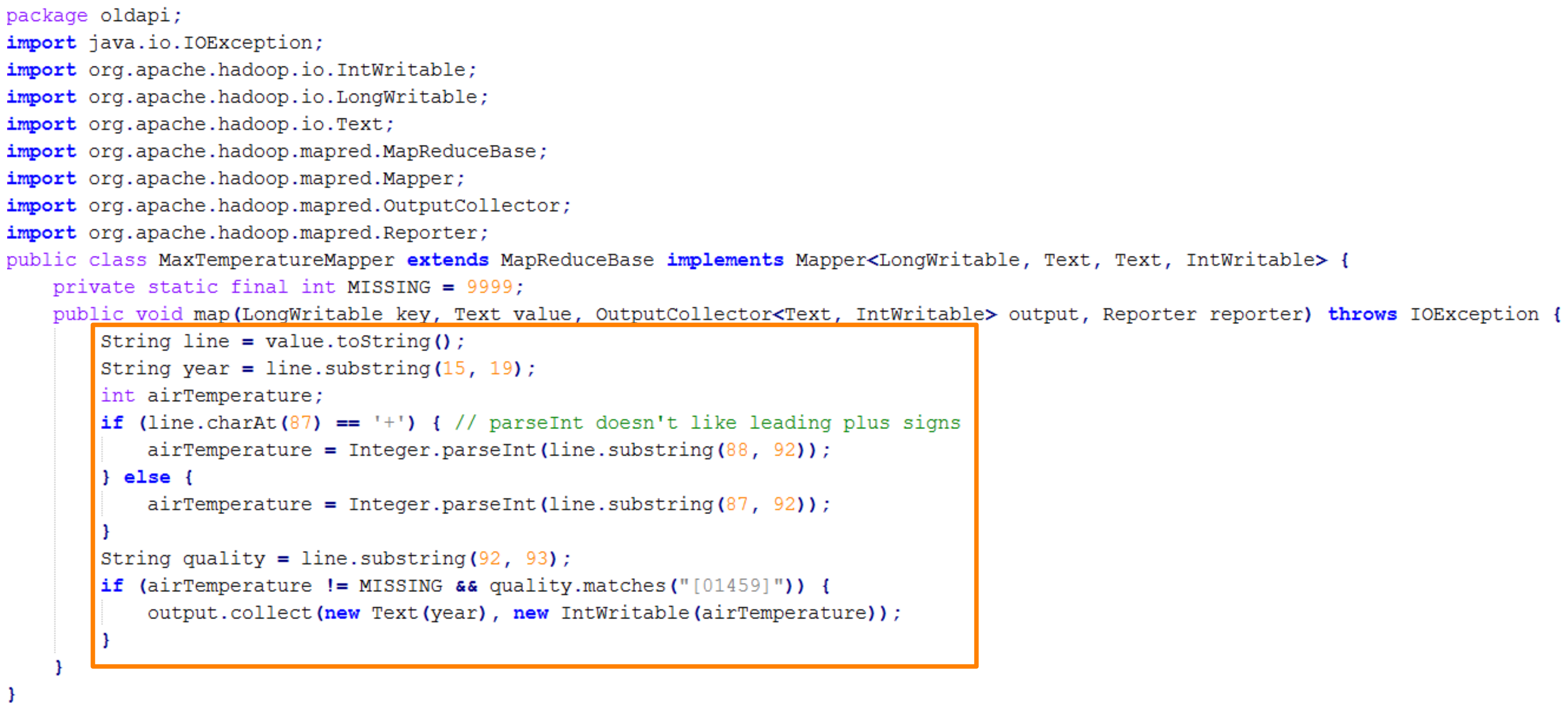
MaxTemperature usando MapReduce

Main()

```
package oldapi;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
public class MaxTemperature {
    public static void main(String[] args) throws IOException {
        if (args.length != 2) {
            System.err.println("Usage: MaxTemperature <input path> <output path>");
            System.exit(-1);
        }
        JobConf conf = new JobConf(MaxTemperature.class);
        conf.setJobName("Max temperature");
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(MaxTemperatureMapper.class);
        conf.setReducerClass(MaxTemperatureReducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        JobClient.runJob(conf);
    }
}
```

MaxTemperature usando MapReduce

Map()



```java
package oldapi;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class MaxTemperatureMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
    private static final int MISSING = 9999;
    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
        String line = value.toString();
        String year = line.substring(15, 19);
        int airTemperature;
        if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs
            airTemperature = Integer.parseInt(line.substring(88, 92));
        } else {
            airTemperature = Integer.parseInt(line.substring(87, 92));
        }
        String quality = line.substring(92, 93);
        if (airTemperature != MISSING && quality.matches("[01459]")) {
            output.collect(new Text(year), new IntWritable(airTemperature));
        }
    }
}
```

MaxTemperature usando MapReduce

Reduce()

```
package oldapi;
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
public class MaxTemperatureReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
        int maxValue = Integer.MIN_VALUE;
        while (values.hasNext()) {
            maxValue = Math.max(maxValue, values.next().get());
        }
    output.collect(key, new IntWritable(maxValue));
    }
}
```

MaxTemperature usando MapReduce

Uso:

1. **Crear el directorio de entrada "/user/<user>/maxtemperature/input" en HDFS:**

   ```
   $ hadoop fs -mkdir /user/<user>/maxtemperature
   /user/<user>/maxtemperature/input
   ```

   Donde <user> es el nombre de usuario Linux del usuario

2. **Copiar el fichero de texto de ejemplo al directorio creado previamente en HDFS:**

   ```
   $ hadoop fs -put sample.txt
   /user/<user>/maxtemperature/input
   ```

MaxTemperature usando MapReduce

3. **Compilar “MaxTemperature.java”, “MaxTemperatureMapper.java” y “MaxTemperatureReducer.java”:**

```
$ mkdir maxtemperature_classes
```

```
$ javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-0.20-mapreduce/* -
d maxtemperature_classes MaxTemperature.java
MaxTemperatureMapper.java MaxTemperatureReducer.java
```

4. **Crear el JAR:**

```
$ jar -cvf maxTemperature.jar -C maxtemperature_classes / .
```

5. **Ejecutar la aplicación:**

```
$ hadoop jar maxTemperature.jar oldapi.MaxTemperature
/user/<user>/maxtemperature/input
/user/<user>/maxtemperature/output
```

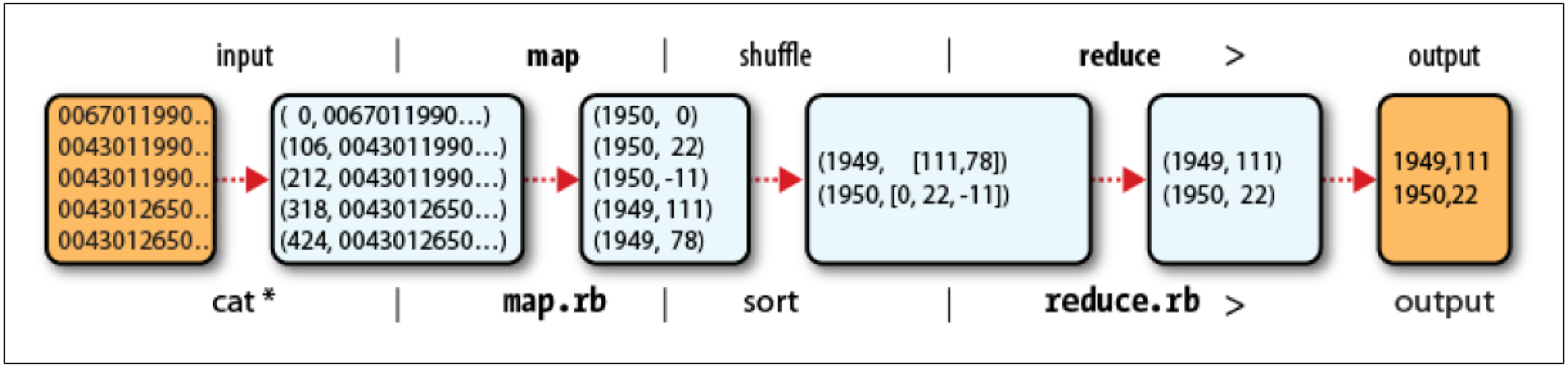
MaxTemperature usando MapReduce

6. **Comprobar la salida:**

```
$ hadoop fs -cat maxtemperature/output/part-00000

1949    111
1950    22
```

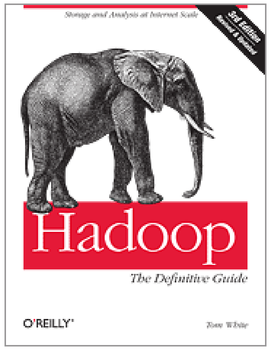
MaxTemperature usando MapReduce



Referencias

- T. White, Hadoop, The Denitive Guide,

  O'Reilly Media, Inc., 2012.



- Example source code accompanying O'Reilly's "Hadoop: The Definitive Guide" by Tom White:
  https://github.com/tomwhite/hadoop-book/