Паралелно и дистрибуирано процесирање

*Лабараториска вежба 5*

*Петар Атанасовски - 216052*

За да го решиме овој проблем користејќи ја парадигмата MapReduce на платформата Hadoop, треба да ги креираме следните компоненти:

1. **Mapper**
   * Ја чита секоја линија од log датотеката.
   * Ја вади IP адресата, временскиот печат, типот на барањето, датотеката и други релевантни информации.
   * Емитира парови клуч-вредност каде клучот е комбинација од месецот и датотеката, а вредноста е IP адресата.

import java.io.IOException;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Mapper;  
  
public class LogMapper extends Mapper<LongWritable, Text, Text, Text> {  
 private final Pattern logPattern = Pattern.*compile*("\\[([^\\s:/]+\\S\*)\\S\*\\] \"(?:[A-Z]+)\\s+([^\"]\*)\"");  
  
 @Override  
 protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {  
 String line = value.toString();  
 Matcher matcher = logPattern.matcher(line);  
  
 if (matcher.find()) {  
 String month = matcher.group(1);  
 String file = matcher.group(2).trim();  
 String ip = line.split("\\s")[0];  
  
 context.write(new Text(month + "\t" + file), new Text(ip));  
 }  
 }  
}

1. **Reducer**
   * Добива парови клуч-вредност од mapper.
   * Ги собира IP адресите за секоја единствена комбинација на месец и датотека.

import java.io.IOException;  
import java.util.HashSet;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Reducer;  
  
public class LogReducer extends Reducer<Text, Text, Text, Text> {  
 @Override  
 protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {  
 HashSet<String> uniqueIPs = new HashSet<>();  
  
 for (Text value : values) {  
 uniqueIPs.add(value.toString());  
 }  
  
 context.write(key, new Text(uniqueIPs.toString()));  
 }  
}

1. **Driver Class**
   * Конфигурирање и извршување на задачата MapReduce со поставување на различни параметри
   * Специфицирање на влезни и излезни патеки
   * Дефинирање на класите на mapper и reducer.

import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;  
  
public class LogAnalysisDriver {  
 public static void main(String[] args) throws Exception {  
 Job job = Job.getInstance();  
 job.setJarByClass(LogAnalysisDriver.class);  
 job.setJobName("LogAnalysis");  
  
 job.setInputFormatClass(TextInputFormat.class);  
 TextInputFormat.addInputPath(job, new Path(args[0]));  
  
 job.setOutputFormatClass(TextOutputFormat.class);  
 TextOutputFormat.setOutputPath(job, new Path(args[1]));  
  
 job.setMapperClass(LogMapper.class);  
 job.setReducerClass(LogReducer.class);  
  
 job.setOutputKeyClass(Text.class);  
 job.setOutputValueClass(Text.class);  
  
 System.exit(job.waitForCompletion(true) ? 0 : 1);  
 }  
}

1. **Build and Run**
   * Пакување на Java-кодот во JAR-датотека.
   * Користење на командата Hadoop за да поднесување на job-от.

hadoop jar JarFileName.jar LogAnalysisDriver <input\_path> <output\_path>