package com.hadoop.hadoop;

import java.io.IOException;

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.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WJob

{

public static void main(String[] args) throws IOException,

ClassNotFoundException, InterruptedException

{

// TODO Auto-generated method stub

/\*

\* if(args.length !=2){

\* System.err.println("Usage: MaxTemperature <input path> <output path>"

\* ); System.exit(-1); }

\*/

long startTime = System.currentTimeMillis();

@SuppressWarnings("deprecation")

Job job = new Job();

job.setJarByClass(WJob.class);

job.setJobName("hadoop\_of\_video");

FileInputFormat.addInputPath(job, new Path(

"hdfs://master:9000/WHadoop/input/output"));

FileOutputFormat.setOutputPath(job, new Path(

"hdfs://master:9000/WHadoop/output"));

job.setMapperClass(WMap.class);

System.out.println(job.getMapperClass());

job.setReducerClass(WReduce.class);

System.out.println(job.getReducerClass());

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

System.out.println("Map " + job.getMapOutputKeyClass()

+ job.getMapOutputValueClass());

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

System.out.println("Put " + job.getOutputKeyClass()

+ job.getOutputValueClass());

// System.out.println(job.waitForCompletion(true));

System.out.println(job.waitForCompletion(true) ? true : false);

long endTime = System.currentTimeMillis();

System.out.println(((endTime - startTime) / 1000));

}

}

package com.hadoop.hadoop;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Map;

import java.util.StringTokenizer;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import com.hadoop.util.Count;

public class WMap extends Mapper<LongWritable, Text, Text, Text>

{

private static Text line = null;

private Text url = null;

private static Map<String,double[]> instantMap = null;

public void map(LongWritable key, Text value, Context context)

throws IOException, InterruptedException

{

// TODO Auto-generated method stub

ArrayList<ArrayList<Object>> mapArray = new ArrayList<ArrayList<Object>>();

ArrayList<Object> al = new ArrayList<Object>();

StringTokenizer itr = new StringTokenizer(value.toString(), "+");

StringTokenizer shotItr = null;

// count()

// al:url;id;length;startTime;endTime;speed

while (itr.hasMoreTokens())

{

al = new ArrayList<Object>();

shotItr = new StringTokenizer(itr.nextToken().toString(), "-");

while(shotItr.hasMoreTokens())

{

al.add(shotItr.nextToken());

}

mapArray.add(al);

}

instantMap = Count.newInstanse().count(mapArray);

for(Map.Entry<String, double[]> entryMap : instantMap.entrySet())

{

url = new Text(entryMap.getKey());

String s = new String();

s+="[";

for(int i = 0; i < entryMap.getValue().length; i++)

{

s+=String.format("%.2f", entryMap.getValue()[i]);

if(i < entryMap.getValue().length-1)

{

s+=",";

}

}

s+="]\r\n";

line = new Text(s);

//context.write(url, line);

}

}

}

package com.hadoop.hadoop;

import java.io.IOException;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class WReduce extends Reducer<Text, Text, Text, Text>

{

/\*

\*(non-Javadoc)

\*

\* @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN,

\* java.lang.Iterable, org.apache.hadoop.mapreduce.Reducer.Context) reduce

\*/

private Text result = null;

public void reduce(Text key, Text values, Context context)

throws IOException, InterruptedException

{

result = new Text();

result = values;

context.write(key, result);

}

}

package com.hadoop.util;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.HashMap;

import java.util.Map;

import com.hadoop.global.ICount;

public class Count implements ICount

{

public static Map<String, double[]> longMap = new HashMap<String, double[]>();

public static Count newInstanse()

{

return new Count();

}

public Map<String, double[]> count(ArrayList<ArrayList<Object>> list)

{

// TODO Auto-generated method stub

// need check

double[] line = null;

// count:startTime to endTime puls 1/speed

// al:url;id;length;startTime;endTime;speed

for (ArrayList<Object> al : list)

{

// init array line

line = new double[Integer.parseInt(String.valueOf(al.get(2)))];

// put the line

if (line != null)

{

if (longMap.entrySet() != null)

{

Arrays.fill(line, 1);

for (int i = Integer.parseInt(String.valueOf(al.get(3))); i <= Integer.parseInt(String.valueOf(al.get(4))); i++)

{

line[i] += (1 / Double.parseDouble(String.valueOf(al

.get(5))));

}

longMap.put(String.valueOf(al.get(0)), line);

}

else

{

for (Map.Entry<String, double[]> shotMap : longMap

.entrySet())

{

if (shotMap.getKey().equals(String.valueOf(al.get(0))))

{

for (int i = Integer.parseInt(String.valueOf(al

.get(3))); i <= Integer.parseInt(String

.valueOf(al.get(4))); i++)

{

shotMap.getValue()[i] += (1 / Double

.parseDouble(String.valueOf(al.get(5))));

longMap.put(shotMap.getKey(),

shotMap.getValue());

}

}

else

{

// function Arrays.fill: for(int

// j=0;j<line.length;j++){line[j]=1;}

Arrays.fill(line, 1);

for (int i = (Integer) al.get(3); i <= (Integer) al

.get(4); i++)

{

line[i] += (1 / Double.parseDouble(String

.valueOf(al.get(5))));

}

longMap.put(String.valueOf(al.get(0)), line);

}

}

}

}

}

return longMap;

}

}

package com.hadoop.hadoop;

import java.io.IOException;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class WReduce extends Reducer<Text, Text, Text, Text>

{

/\*

\*(non-Javadoc)

\* @see org.apache.hadoop.mapreduce.Reducer#reduce(KEYIN,

\* java.lang.Iterable, org.apache.hadoop.mapreduce.Reducer.Context) reduce

\*/

private Text result = null;

public void reduce(Text key, Text values, Context context)

throws IOException, InterruptedException

{

result = new Text();

result = values;

context.write(key, result);

}

}

package com.hadoop.global;

import java.util.ArrayList;

import java.util.Map;

public interface ICount

{

public Map count(ArrayList<ArrayList<Object>> list);

}

package com.hadoop.global;

import com.hadoop.service.impl.Video;

import com.hadoop.service.impl.VideoBehavior;

public interface IHandle

{

boolean handle(Video video, VideoBehavior behavior);

}