How to program a MapReduce cluster

TF-IDF step by step

Ján Súkeník

xsukenik@is.stuba.sk
sukenik08@student.fiit.stuba.sk

TF-IDF

- potrebujeme
  - pre každý dokument
    - počet slov
    - frekvenciu každého slova
  - pre každé slovo
    - počet dokumentov, v ktorých sa nachádza

- v MapReduce rozdelíme na 3 fázy

Fáza 1

Map

(dokument, [slová])

↓

({slovo, dokument}, 1)

Reduce

({slovo, dokument}, [1, 1, …])

↓

({slovo, dokument}, počet-výskytov-slova n)

Mapper

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{

}
```

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {



    }
}
```

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String filename = ((FileSplit) context.getInputSplit()).getPath().getName();









    }
}
```

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String filename = ((FileSplit) context.getInputSplit()).getPath().getName();
        Matcher m = Pattern.compile("\\w+").matcher(lineContent.toString());

        while (m.find())
        {









        }
    }
}
```

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String filename = ((FileSplit) context.getInputSplit()).getPath().getName();
        Matcher m = Pattern.compile("\\w+").matcher(lineContent.toString());

        while (m.find())
        {
            String word = m.group().toLowerCase();
            if (shouldSkipWord(word)) continue;





        }
    }
}
```

```java
public class FirstMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String filename = ((FileSplit) context.getInputSplit()).getPath().getName();
        Matcher m = Pattern.compile("\\w+").matcher(lineContent.toString());

        while (m.find())
        {
            String word = m.group().toLowerCase();
            if (shouldSkipWord(word)) continue;

            context.write(
                new Text(word + "@" + filename),
                new IntWritable(1)
            );
        }
    }
}
```

Reducer

```java
public class FirstReducer extends Reducer<Text, IntWritable, Text, IntWritable>
{

}
```

```java
public class FirstReducer extends Reducer<Text, IntWritable, Text, IntWritable>
{
    public void reduce(Text wordAtDocument, Iterable<IntWritable> ones, Context context)
    {









    }
}
```

```java
public class FirstReducer extends Reducer<Text, IntWritable, Text, IntWritable>
{
    public void reduce(Text wordAtDocument, Iterable<IntWritable> ones, Context context)
    {
        int sum = 0;
        for (IntWritable one : ones) sum++;





    }
}
```

```java
public class FirstReducer extends Reducer<Text, IntWritable, Text, IntWritable>
{
    public void reduce(Text wordAtDocument, Iterable<IntWritable> ones, Context context)
    {
        int sum = 0;
        for (IntWritable one : ones) sum++;

        context.write(
            wordAtDocument,
            new IntWritable(sum)
        );
    }
}
```

Driver

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {

    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 1");









    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 1");

        job.setJarByClass(FirstDriver.class);
        job.setMapperClass(FirstMapper.class);
        job.setReducerClass(FirstReducer.class);









    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 1");

        job.setJarByClass(FirstDriver.class);
        job.setMapperClass(FirstMapper.class);
        job.setReducerClass(FirstReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);





    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 1");

        job.setJarByClass(FirstDriver.class);
        job.setMapperClass(FirstMapper.class);
        job.setReducerClass(FirstReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));


    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

```java
public class FirstDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 1");

        job.setJarByClass(FirstDriver.class);
        job.setMapperClass(FirstMapper.class);
        job.setReducerClass(FirstReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        return job.waitForCompletion(true) ? 0 : 1;
    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(
            new Configuration(), new FirstDriver(), args
        ));
    }
}
```

Fáza 2

Map

({slovo, dokument}, n)

↓

(dokument, {slovo, n})

Reduce

(dokument, [{$slovo_1$, n_1}, {$slovo_2$, n_2}, …])

↓

({slovo, dokument}, {n, počet-slov-v-dokumente N})

Mapper

```java
public class SecondMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {

    }
}
```

```java
public class SecondMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String[] line = lineContent.toString().split("\t");
        String[] key = line[0].split("@");

        String word = key[0];
        String document = key[1];
        String count = line[1];





    }
}
```

```java
public class SecondMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String[] line = lineContent.toString().split("\t");
        String[] key = line[0].split("@");

        String word = key[0];
        String document = key[1];
        String count = line[1];

        context.write(
            new Text(document),
            new Text(word + "=" + count)
        );
    }
}
```

Reducer

```java
public class SecondReducer extends Reducer<Text, Text, Text, Text>
{
    public void reduce(Text document, Iterable<Text> wordCounts, Context context)
    {

    }
}
```

```java
public class SecondReducer extends Reducer<Text, Text, Text, Text>
{
    public void reduce(Text document, Iterable<Text> wordCounts, Context context)
    {



        for (Text wordCountStr : wordCounts)
        {
            String[] wordCount = wordCountStr.toString().split("=");

            String word = wordCount[0];
            int count = Integer.parseInt(wordCount[1]);



        }









    }
}
```

```java
public class SecondReducer extends Reducer<Text, Text, Text, Text>
{
    public void reduce(Text document, Iterable<Text> wordCounts, Context context)
    {
        int sumOfWordsInDocument = 0;


        for (Text wordCountStr : wordCounts)
        {
            String[] wordCount = wordCountStr.toString().split("=");

            String word = wordCount[0];
            int count = Integer.parseInt(wordCount[1]);


            sumOfWordsInDocument += count;
        }









    }
}
```

```java
public class SecondReducer extends Reducer<Text, Text, Text, Text>
{
    public void reduce(Text document, Iterable<Text> wordCounts, Context context)
    {
        int sumOfWordsInDocument = 0;
        Map<String, Integer> tempCounter = new HashMap<String, Integer>();

        for (Text wordCountStr : wordCounts)
        {
            String[] wordCount = wordCountStr.toString().split("=");

            String word = wordCount[0];
            int count = Integer.parseInt(wordCount[1]);

            tempCounter.put(word, count);
            sumOfWordsInDocument += count;
        }









    }
}
```

```java
public class SecondReducer extends Reducer<Text, Text, Text, Text>
{
    public void reduce(Text document, Iterable<Text> wordCounts, Context context)
    {
        int sumOfWordsInDocument = 0;
        Map<String, Integer> tempCounter = new HashMap<String, Integer>();

        for (Text wordCountStr : wordCounts)
        {
            String[] wordCount = wordCountStr.toString().split("=");

            String word = wordCount[0];
            int count = Integer.parseInt(wordCount[1]);

            tempCounter.put(word, count);
            sumOfWordsInDocument += count;
        }

        for (Entry<String, Integer> entry : tempCounter.entrySet())
        {
            context.write(
                new Text(entry.getKey() + "@" + document.toString()),
                new Text(entry.getValue() + "/" + sumOfWordsInDocument)
            );
        }

    }
}
```

Driver

```java
public class SecondDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 2");

        job.setJarByClass(SecondDriver.class);
        job.setMapperClass(SecondMapper.class);
        job.setReducerClass(SecondReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        return job.waitForCompletion(true) ? 0 : 1;
    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(new Configuration(), new SecondDriver(), args));
    }
}
```

Fáza 3

Map

$$(\{slovo, dokument\}, \{n, N\})$$

$$\downarrow$$

$$(slovo, \{dokument, n, N\})$$

Reduce

$$(slovo, [\{d_1, n_1, N_1\}, \{d_2, n_2, N_2\}, \ldots])$$

$$\downarrow$$

$$(\{slovo, dokument\}, tf\text{-}idf)$$

Mapper

```java
public class ThirdMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {



    }
}
```

```java
public class ThirdMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String[] line = lineContent.toString().split("\t");
        String[] key = line[0].split("@");

        String word = key[0];
        String document = key[1];
        String countAndTotal = line[1];





    }
}
```

```java
public class ThirdMapper extends Mapper<LongWritable, Text, Text, Text>
{
    public void map(LongWritable lineNumber, Text lineContent, Context context)
    {
        String[] line = lineContent.toString().split("\t");
        String[] key = line[0].split("@");

        String word = key[0];
        String document = key[1];
        String countAndTotal = line[1];

        context.write(
            new Text(word),
            new Text(document + "=" + countAndTotal)
        );
    }
}
```

Reducer

```java
public class ThirdReducer extends Reducer<Text, Text, Text, Text>
{



    public void reduce(Text word, Iterable<Text> documentCounts, Context context)
    {



    }
}
```

```java
public class ThirdReducer extends Reducer<Text, Text, Text, Text>
{



    public void reduce(Text word, Iterable<Text> documentCounts, Context context)
    {
        int numberOfDocumentsWithWord = 0;


        for (Text documentCountStr : documentCounts)
        {
            String[] documentCount = documentCountStr.toString().split("=");
            String document = documentCount[0];
            String count = documentCount[1];

            numberOfDocumentsWithWord++;

        }









    }
}
```

```java
public class ThirdReducer extends Reducer<Text, Text, Text, Text>
{



    public void reduce(Text word, Iterable<Text> documentCounts, Context context)
    {
        int numberOfDocumentsWithWord = 0;
        Map<String, String> tempFrequencies = new HashMap<String, String>();

        for (Text documentCountStr : documentCounts)
        {
            String[] documentCount = documentCountStr.toString().split("=");
            String document = documentCount[0];
            String count = documentCount[1];

            numberOfDocumentsWithWord++;
            tempFrequencies.put(document, count);
        }

        for (Entry<String, String> entry : tempFrequencies.entrySet())
        {
            String[] countTotal = entry.getValue().split("/");
            int count = Integer.parseInt(countTotal[0]);
            int total = Integer.parseInt(countTotal[1]);









        }
    }
}
```

```java
public class ThirdReducer extends Reducer<Text, Text, Text, Text>
{

    private static final int NUMBER_OF_DOCS = 37;

    public void reduce(Text word, Iterable<Text> documentCounts, Context context)
    {
        int numberOfDocumentsWithWord = 0;
        Map<String, String> tempFrequencies = new HashMap<String, String>();

        for (Text documentCountStr : documentCounts)
        {
            String[] documentCount = documentCountStr.toString().split("=");
            String document = documentCount[0];
            String count = documentCount[1];

            numberOfDocumentsWithWord++;
            tempFrequencies.put(document, count);
        }

        for (Entry<String, String> entry : tempFrequencies.entrySet())
        {
            String[] countTotal = entry.getValue().split("/");
            int count = Integer.parseInt(countTotal[0]);
            int total = Integer.parseInt(countTotal[1]);

            double tf = (double) count / (double) total;
            double idf = Math.log10((double) NUMBER_OF_DOCS / (double) numberOfDocumentsWithWord);
            double tfidf = tf * idf;





        }
    }
}
```

```java
public class ThirdReducer extends Reducer<Text, Text, Text, Text>
{
    private static final DecimalFormat DF = new DecimalFormat("###.########");
    private static final int NUMBER_OF_DOCS = 37;

    public void reduce(Text word, Iterable<Text> documentCounts, Context context)
    {
        int numberOfDocumentsWithWord = 0;
        Map<String, String> tempFrequencies = new HashMap<String, String>();

        for (Text documentCountStr : documentCounts)
        {
            String[] documentCount = documentCountStr.toString().split("=");
            String document = documentCount[0];
            String count = documentCount[1];

            numberOfDocumentsWithWord++;
            tempFrequencies.put(document, count);
        }

        for (Entry<String, String> entry : tempFrequencies.entrySet())
        {
            String[] countTotal = entry.getValue().split("/");
            int count = Integer.parseInt(countTotal[0]);
            int total = Integer.parseInt(countTotal[1]);

            double tf = (double) count / (double) total;
            double idf = Math.log10((double) NUMBER_OF_DOCS / (double) numberOfDocumentsWithWord);
            double tfidf = tf * idf;

            context.write(
                new Text(word + "@" + entry.getKey()),
                new Text(DF.format(tfidf))
            );
        }
    }
}
```

Driver

```java
public class ThirdDriver extends Configured implements Tool
{
    public int run(String[] args) throws Exception
    {
        Configuration conf = getConf();
        Job job = new Job(conf, "TF-IDF phase 3");

        job.setJarByClass(ThirdDriver.class);
        job.setMapperClass(ThirdMapper.class);
        job.setReducerClass(ThirdReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        return job.waitForCompletion(true) ? 0 : 1;
    }

    public static void main(String[] args) throws Exception
    {
        System.exit(ToolRunner.run(new Configuration(), new ThirdDriver(), args));
    }
}
```

Hotovo

Ako to spustím?

Príprava vstupov na HDFS

- Vytvorenie HDFS adresára

```
$ hadoop dfs -mkdir /pewe
```

- Nakopírovanie súborov na HDFS

```
$ hadoop dfs -put ~/input /pewe
```

Spustenie

- Vytvoriť jar so skompilovanými súbormi

```
$ hadoop jar tfidf.jar package.FirstDriver
$   hadoop dfs –ls /pewe
     /pewe/input
     /pewe/first-output
$ hadoop jar tfidf.jar package.SecondDriver
$   hadoop dfs –ls /pewe
     /pewe/input
     /pewe/first-output
     /pewe/second-output
$ hadoop jar tfidf.jar package.ThirdDriver
$   hadoop dfs –ls /pewe
     /pewe/input
     /pewe/first-output
     /pewe/second-output
     /pewe/third-output
```

Vstup

Down, down, down. There was nothing else to do, so Alice soon began talking again. 'Dinah'll miss me very much to-night, I should think!' (Dinah was the cat.) 'I hope they'll remember her saucer of milk at tea-time. Dinah my dear! I wish you were down here with me! There are no mice in the air, I'm afraid, but you might catch a bat, and that's very like a mouse, you know. But do cats eat bats, I wonder?' And here Alice began to get rather sleepy, and went on saying to herself, in a dreamy sort of way, 'Do cats eat bats? Do cats eat bats?' and sometimes, 'Do bats eat cats?' for, you see, as she couldn't answer either question, it didn't much matter which way she put it. She felt that she was dozing off, and had just begun to dream that she was walking hand in hand with Dinah, and saying to her very earnestly, 'Now, Dinah, tell me the truth: did you ever eat a bat?' when suddenly, thump! thump! down she came upon a heap of sticks and dry leaves, and the fall was over.

Presently she began again. 'I wonder if I shall fall right THROUGH the earth! How funny it'll seem to come out among the people that walk with

Výstup po fáze 1

```
yard@pg1322.txt 10
yases@pg2265.txt        1
yawning@pg135.txt       14
yevgeney@pg1399.txt     1
yo@pg2000.txt   2077
yonder@pg1112.txt       6
yours@pg1502.txt        4
yourself@pg236.txt      1
youssouf@pg35830.txt    4
youthful@pg14591.txt    3
yuliya@pg35830.txt      14
zabljak@pg35830.txt     2
zalec@pg35830.txt       2
zip@pg236.txt   1
zlatibor@pg35830.txt    2
zrnovci@pg35830.txt     2
zuzemberk@pg35830.txt   2
```

Výstup po fáze 2

```
directing@pg2701.txt    3/153640
olassen@pg2701.txt      1/153640
sharp@pg2701.txt        42/153640
shark@pg2701.txt        30/153640
peremptorily@pg2701.txt 4/153640
inventor@pg2701.txt     1/153640
women@pg2701.txt        11/153640
godly@pg2701.txt        3/153640
compendious@pg2701.txt  1/153640
unforseen@pg2701.txt    1/153640
hannibal@pg2701.txt     1/153640
massive@pg2701.txt      5/153640
popping@pg2701.txt      1/153640
districts@pg2701.txt    3/153640
claimed@pg2701.txt      2/153640
malignantly@pg2701.txt  1/153640
crippled@pg2701.txt     3/153640
```

Výstup po fáze 3

```
have@pg1080.txt 0
have@pg1101.txt 0
have@pg1112.txt 0
have@pg11.txt   0
have@pg1232.txt 0
have@pg1257.txt 0
...
ver@pg2000.txt  0.00118238
italy@pg25717.txt       0.00142321
sperm@pg2701.txt        0.00173278
scallops@pg20776.txt    0.00196741
clifford@pg1101.txt     0.0019727
constantius@pg25717.txt 0.00247444
kotick@pg236.txt        0.00315428
toomai@pg236.txt        0.00343735
shere@pg236.txt 0.00351823
clifford@pg1502.txt     0.00417993
```

Iné Hadoop-related projekty

... alebo dá sa to ešte ľahšie?

Mahout

- knižnica na strojové učenie a data mining
- Hadoop verzie známych algoritmov
  - classification
  - clustering
  - pattern mining
  - regression
  - dimension reduction
  - evolutionary algorithms
  - collaborative filtering
  - vector similarity
- stále vo vývoji

Hive a Pig

- jazyky
- abstrahovanie od Hadoop MapReduce v Jave
- Hive
  - SQL-like jazyk
  - vznikol na Facebook hackatrone
- Pig
  - procedurálny jazyk
  - vznikol v Yahoo! Research
- napríklad nad HDFS súbormi

HBase a Cassandra

- distribuované databázové systémy
- open-source verzie Google BigTable
- HBase
  - vytvorené špeciálne pre Hadoop a HDFS
- Cassandra
  - vytvorené vo Facebooku
  - nemala s Hadoop nič spoločné
  - dnes už sú kamaráti
- dá sa použiť Hive a Pig

Ďakujem

Otázky?

Zdroje

1. http://hadoop.apache.org/common/docs/r0.17.2/
2. http://cassandra.apache.org/
3. http://hbase.apache.org/
4. http://hive.apache.org/
5. http://pig.apache.org/
6. http://mahout.apache.org/
7. http://marcellodesales.wordpress.com/2009/12/31/tf-idf-in-hadoop-part-1-word-frequency-in-doc/
8. http://www.gutenberg.org/