Fundusze Europejskie Wiedza Edukacja Rozwój
Politechnika Wrocławska
Unia Europejska Europejski Fundusz Społeczny

„ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej”

Big Data Algorithms

dr Maciej Gębala

Selected problems for exercises and laboratories

2019

Contents

1 Introduction

Problem 1 (laboratory)

Find the sources of your favorite book. Save them in the `utf-8` text file format. Implement tasks in the Scala language.

1. Import the `io.Source` library (`import scala.io.Source`).
2. Use the `Source.fromFile(source, "UTF-8")` command to load the book, replace the file with a string (`mkString`) and then divide it into words (`split("\s+")`). You can do it with one command.
3. Remove the stop-words from this list.

   > You can find stop-words at
   > https://sites.google.com/site/kevinbouge/stopwords-lists.
   > Use something like `Book.filterNot(Stop.contains(_))`.

4. Transform the list of words into a collection of pairs `(word,1)` of type `(String,Int)`.
5. Group the list of pairs.

   > Use something like `Filtered.groupBy(x=>x._1)`.

6. Reduce words.

   > Use something like `Groupped.mapValues(x=>x.length)`.

7. Sort by the second parameter.

   > Use e.g. `reduced.toSeq.sortWith((x,y)=>x._2>y._2)`.

8. Save the result to a file.

   > Use the `PrintWriter` object from the `java.io` libraries.

9. Display a few dozen of the first elements. Remove a dozen of initial elements from it and save the list to a text file.
10. Build a word cloud from the list received.

    > Use, for example, the website http://www.wordclouds.com/.

The goal of this task is to generate more or less such a Picture 1 (example for the book ”War and Peace”).

> You can read about `TF-IDF` in the book *Mining of Massive Datasets* by J. Leskovec, A. Rajaraman and J. Ullman, in Chapter 1: *Data Mining* (http://infolab.stanford.edu/~ullman/mmds/ch1.pdf).

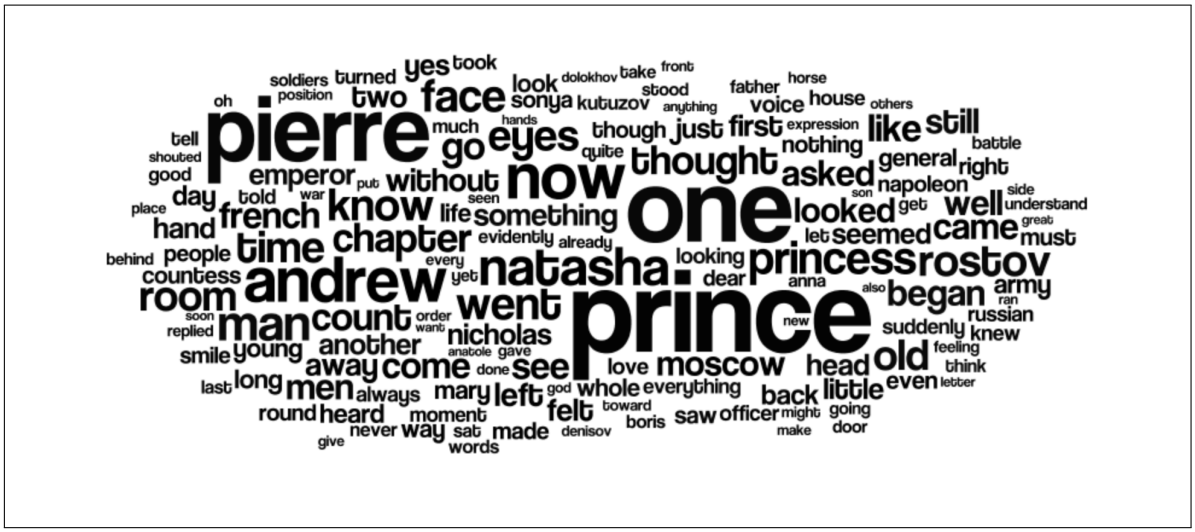

Figure 1: Word cloud for ”War and Peace”.

Problem 2 (laboratory)

This is the continuation of the Problem 1.

1. Share your book into chapters.
2. Treat each chapter as a separate document.
3. Divide documents into words. Find `TF-IDF` indexes of all words in all chapters.
4. Build word clouds for all chapters and one cloud for the entire document.

Problem 3 (laboratory)

This is the continuation of the Problem 2.

1. For each chapter, identify 20 words with the highest `TF-IDF` coefficients.
2. Write a function that performs the following task: for the word being entered returns the list of the most matching chapters (that is, sorts the chapters according to the `TF-IDF` parameter).

Problem 4 (exercise)

Let’s assume that we have access to the customers’ purchase database in the chemical warehouse wholesale network from the previous year. During a year, 10^7 customers visit it 10 times and each time buys an average of 10 different types of products from the pool of 200 available product types. Let’s assume that we found two customers in this database who bought at least once the same basket of products. Is this a pure coincidence?

2 Similarity of Objects

> You can read about similarity of objects in the book *Mining of Massive Datasets* by J. Leskovec, A. Rajaraman and J. Ullman, in Chapter 3: *Finding Similar Items* (http://infolab.stanford.edu/~ullman/mmds/ch3.pdf).

Problem 5 (exercise)
Show that the function $d(A, B) = |A \triangle B|$ is a metric over the space of non-empty finite subsets of a fixed set X.

Problem 6 (exercise)
Let $f : [0, \infty) \to [0, \infty$ be a growing and concave function.

1. Show that for $a, b \geq 0$ we have $f(a + b) \leq f(a) + f(b)$.

   > Note that we can assume that $a + b > 0$, then notice that $a = (a + b)\frac{a}{a+b}$ and $b = (a + b)\frac{b}{a+b}$ and use Jensen's inequality for concave functions.

2. Suppose in addition that $f(0) = 0$. Let d be a metric on the set X. Show that the function $\rho(x, y) = f(d(x, y))$ is also a metric on the set X.

3. Show that if $\varepsilon \in (0, 1)$ and d is a metric on the set X, then the function $\rho(x, y) = d(x, y)^{\varepsilon}$ is also a metric on the set X.

4. Show that if d is a metric on the set X, then the function $\rho(x, y) = \frac{d(x,y)}{1+d(x,y)}$ is also a metric on the set X.

Problem 7 (exercise)
Let's choose two random m-element subsets A and B from the n-element set X. What is the expected value of Jaccard similarity $J(A, B)$?

Problem 8 (exercise, laboratory)
Use the **Prime Numbers Theorem** to estimate the number of prime numbers from the range $[2^{64}, 2^{64} + 1000]$ and then determine these numbers.

Problem 9 (exercise)

Theorem 2.1 (Steinhaus theorem) *Let d be a metric on the set X. Let's fix the element $a \in X$ and define the function*

$$\rho(x, y) = \frac{d(x, y)}{d(x, a) + d(y, a) + d(x, y)}$$

Then ρ is a metric on the set X.

Prove Steinhaus theorem.

1. Show first that if $0 < p \leq q$ and $r \geq 0$ then $\frac{p}{q} \leq \frac{p+r}{q+r}$.

2. Mark $p = d(x, y)$, $q = d(x, y)+d(x, a)+d(y, a)$ and $r = d(x, z)+d(y, z)-d(x, y)$. Use observation from the previous point to show the triangle inequality for the ρ function.

Problem 10 (exercise)
Apply the Steinhaus Theorem to the metric defined by $d(X, Y) = |X \triangle Y|$ on a set of finite subsets of the set Ω to show that the function $d_J(X, Y) = 1 - J(X, Y)$ (Jaccard distance) is a metric on the set Ω.

Problem 11 (exercise)
Assume that $\mathcal{H} = \{h_1, \ldots, h_n\}$ is a family of k-independent hash functions from the set D into a finite set R, i.e. for any $x_1, \ldots, x_k \in D$ and $y_1, \ldots, y_k \in R$ we have

$$\Pr_{h \leftarrow H}[h(x_1) = y \wedge \ldots \wedge h(x_k) = y_k] = \frac{1}{|R|^k}.$$

Show that for any $m \leq k$ family $\mathcal{H}$ is also m-independent.

Problem 12 (exercise)
Consider the hash function given by the formula $h(x) = x \mod 21$. We apply it to numbers divisible by a certain constant c. For which constants c is the proper hash function, i.e for which constants c it can be expected that the distribution of bucket loading $\{0, \ldots, 20\}$ will be uniform?

Problem 13 (exercise)
Find the formula for the order of element $k \in \{0, \ldots, n-1\}$ in group $C_n = (\{0, \ldots, n-1\}, \oplus_n)$? What is the relationship between this problem and the previous one?

Problem 14 (exercise, laboratory)
We have n buckets. We throw k balls to them with a uniform distribution.

1. Estimate k so that there is a high probability of 3-collisions, that is, there exists a bucket with three balls.

2. Experimentally check this result.

3. Generalize the task at the m-collision.

4. Assign the expected value of the number of empty buckets in a similar way. When (for which m) this number becomes less than 1.

Problem 15 (laboratory)

Two students have a jug with a capacity 8 liters filled with 8 liters of drink. They also have a jug with a capacity of 5 liters and a jug with a capacity of 3 liters. They want to share the drink equally. How can they do it?

This can be considered as a rewriting system with an initial state $(8, 0, 0)$. We can try to solve this problem in this way: randomly select two different numbers $i, j \in \{1, 2, 3\}$ and try to transfer the drink from the i-th container to the j-th container; we iterate this random walk until we reach the state $(4, 4, 0)$. However, this is a poor solution - this algorithm often falls into loops. Use the tracking technique to avoid loops. Implement this problem in Scala.

Problem 16 (exercise)

Suppose that S is similarity of objects from the set Ω and there is a family $\mathcal{H}$ of hash functions with probability on the $\mathcal{H}$ family that for any two objects $A, B \in \Omega$ we have $\Pr[h(A) = h(B)] = S(A, B)$.

Show that then the function $d(A, B) = 1 - S(A, B)$ is a metric on the set Ω.

Problem 17 (exercise)

Complete the details of the proof that if $\Omega = \{\omega_i : 1 \leq i \leq N\}$, π is the random permutation of the set $\{1, \ldots, N\}$ (chosen according to the uniform distribution), and $h_\pi(X) = \min\{k : \omega_{\pi(k)} \in X\}$ for $X \subseteq \Omega$, then

$$\Pr_\pi[h_\pi(A) = h_\pi(B)] = J(A, B).$$

Problem 18 (laboratory)

Write the procedure with the interface

```
jaccard(f1:String,f2:String,k:Integer):Double
```

which for the files named `f1` and `f2` determines their k-shingles and then calculates their Jaccard distance. Before determining k-shingles, the files should be cleaned (the minimum is to delete new line characters, tabs and double spaces).

1. Apply this procedure to several variants of your program file (use 4-shingles).
2. Use this procedure to compare subsequent chapters of the book analyzed in Problem 2 (use 7-shingles).

Problem 19 (laboratory)

Apply the min-hash method to the previous problem (Problem 18). Your procedure should depend on the H parameter, which determines the number of hash functions used to build the signature.

Test this procedure on the data from the previous problem for $H \in \{50, 100, 250\}$ and compare the Jaccard distance approximation with its exact values. Remember to generate a shared family of hash functions for all analyzed texts.

3 Streaming

> You can read about data streaming in the book *Mining of Massive Datasets* by J. Leskovec, A. Rajaraman and J. Ullman, in Chapter 4: *Mining Data Streams* (http://infolab.stanford.edu/~ullman/mmds/ch4.pdf).

Problem 20 (exercise, laboratory)
Let C_n be the value of the Morris counter after calling the INCREMENT procedure n times. Let $L_n = 2^{C_n}$.

1. Find the variance of the variable L_n and calculate $\frac{\sigma(L_n)}{E[L_n]}$.
2. Investigate the accuracy of the set of four Morris counters experimentally. As an estimator of the number n, take $2^{(C_1(n)+C_2(n)+C_3(n)+C_4(n))/4}$.
3. For which n we have $4\log_2(\log_2(n)) < \log_2(n)$?

> For details of algorithm, see *Approximate Counting: a Detailed Analysis* by P. Flajolet (http://algo.inria.fr/flajolet/Publications/Flajolet85c.pdf).

Problem 21 (laboratory, exercise)
Implement the Boyer-Moore Majority algorithm in Scala.

1. Write the function whose parameter is the list of strings (`List[String]`).
2. Design an object with two methods `add(x:String):Unit` and `get():String` that implements this algorithm.
3. Estimate the computational and memory complexity of this algorithm.

> For details of algorithm, see *MJRTY - A Fast Majority Vote Algorithm* by R.S. Boyer and J.S. Moore (https://link.springer.com/content/pdf/10.1007/978-94-011-3488-0_5.pdf).

Problem 22 (laboratory, exercise)
Implement the Misra-Gries algorithm in Scala.

1. Write the function whose parameters are the list of strings (`List[String]`) and the number k specifying the maximum number of objects to be tracked. Use the collection `scala.collection.mutable.Map`.
2. Design an object with two methods `add(x:String):Unit` and `get():String` that implements this algorithm. You need one parameter k to create this object.
3. Estimate the computational and memory complexity of this algorithm.

> For details of algorithm, see *Finding Repeated Elements* by J. Misra and D. Gries (http://www.cs.utexas.edu/users/misra/scannedPdf.dir/FindRepeatedElements.pdf).

Problem 23 (exercise)

The HyperLogLog algorithm uses the value $h(x) = (b_0 b_1 b_2 b_3 \ldots)$ to determine the number of the incremented counter ($i = (b_0 \ldots b_{k-1})_2 + 1$) and to determine from the rest of the sequence bits $(b_k b_{k+1} \ldots)$ to increase the value of the counter.

Assume that $h(x)$ is of type `Int` or `Long` and that $h(x) \geq 0$.

1. How can you determine with $h(x)$ a string $(b_0 \ldots b_{k-1})$ using bit operations?
2. How can you determine with $h(x)$ a string $(b_k b_{k+1} \ldots)$ using bit operations?
3. Assume that $n \geq 0$. What the $n\&(-n)$ operation does. How can this operation be used to increment the counter.

> For details of algorithm, see *HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm* by P. Flajolet, É. Fusy, O. Gandouet and F. Meunier (http://algo.inria.fr/flajolet/Publications/FlFuGaMe07.pdf).

Problem 24 (laboratory)

Implement the HyperLogLog algorithm in Scala.

1. Download from ftp://ita.ee.lbl.gov/html/contrib/LBL-PKT.html file `lbl-point-4` and extract the `lbl-point-4.tcp` file from it. Here is the data format: timestamp, (renumbered) source host, (renumbered) destination host, source TCP port, destination TCP port, number of data bytes (zero for ”pure-ack” packets).
2. Apply HyperLogLog to determine the number of different source hosts, the number of different destination hosts and the number of different pairs (source, destination).

Problem 25 (laboratory)

Implement the *Geometric Histrogram Streaming Window* algorithm in Scala. Estimate the computational and memory complexity of this algorithm.

> For details of algorithm, see *Maintaining Stream Statistic Over Sliding Windows* by M. Datar, A. Gionis, P. Indyk and R. Motwani (http://www-cs-students.stanford.edu/~datar/papers/sicomp_streams.pdf).

4 Model Map-Reduce

> You can read about similarity of objects in the book *Mining of Massive Datasets* by J. Leskovec, A. Rajaraman and J. Ullman, in Chapter 2: *Map-Reduce and the New Software Stack* (http://infolab.stanford.edu/~ullman/mmds/ch2.pdf).

Problem 26 (laboratory)
Rewrite Problem 1 algorithms to programs running in the Map-Reduce model.

Problem 27 (exercise)
Show that the following operations in $\mathbb{R}$ are commutative and associative

1. $\min(x, y)$,
2. $\max(x, y)$,
3. $x \oplus y = x + y + 1$,
4. $x \otimes y = xy + x + y$.

Problem 28 (exercise)
Give a few examples of operations that are not commutative.
Give a few examples of operations that are not associative.
Is the $s(x, y) = \frac{x+y}{2}$ associative?

Problem 29 (exercise, laboratory)
Design a Map-Reduce algorithm that gets a very large set of integers and outputs at the same time:

1. The largest and the smallest number.
2. The average of all numbers.
3. The same set but without repetition of elements.
4. The number of different elements without repetition.

Problem 30 (exercise)
Design the Map-Reduce algorithm, which determines joining of two relations defined by the $R(A, B, C)$ and $S(X, Y, Z)$ schemes, according to the $B = X$ and $C = Y$ connection, that is, calculates

$$\{ (A, Z) \ : \ \exists_{B,C} \ R(A, B, C) \wedge S(B, C, Z) \}.$$

Problem 31 (laboratory)
Reversal graph: A graph is given as a lists of neighbors $[w_i, [w_{i,1}, w_{i,2}, \ldots, w_{i,n}]]$, and stored in a text file, e.g.

```
[
  [1, [3,4,5]],
  [2, [1,3]],
  [3, [4,5]],
  [4, [1,2]],
  [5, [4,5]]
]
```

Apply Map-Reduce technology to build a graph with inverted edges.

Problem 32 (exercise)
Let $F : (\mathbb{N}^+ \times \mathbb{R})^2 \to (\mathbb{N}^+ \times \mathbb{R})$ be a function specified by formula

$$F([c_1, x_1], [c_2, x_2]) = \left[c_1 + c_2, \frac{c_1 x_1 + c_2 x_2}{c_1 + c_2}\right]$$

1. Show that F is associative and commutative.
2. Let us denote $x \oslash y = F(x, y)$. Find a compact formula for
$$[c_1, x_1] \oslash [c_2, x_2] \oslash \ldots \oslash [c_n, x_n].$$
3. Use this property of the function F to design a combiner for determining the mean and variance.

Problem 33 (exercise)
Use the Map-Reduce method to determine the geometric and harmonic mean.

Problem 34 (exercise)
Use the Map-Reduce method to designate all anagrams that appear in a text file.

Problem 35 (laboratory)
Apply Map-Reduce twice to the book you used in Problem 1. Make a list of the five most-related words with the each word. By related words, we mean words that appear side by side (after removing stop-words). Of course, you have to program this task in the Map-Reduce model.
Try to apply the received list to generate a random paragraph from your book.

Problem 36 (laboratory)
Let $G = (V, E)$ be a directed graph. For $v \in V$, we denote

$$inDeg(v) = |\{\, u \in V \;:\; (u, v) \in E \,\}|$$

and

$$outDeg(v) = |\{\, u \in V \;:\; (v, u) \in E \,\}|.$$

Design procedures for determining in Map-Reduce

$$\{ (v, inDeg(v), outDeg(v)) \ : \ v \in V \}.$$

Write also the procedure for determining the average value of degrees $\frac{1}{|V|}\sum_{v\in V} inDeg(v)$ and $\frac{1}{|V|}\sum_{v\in V} outDeg(v)$.

Apply the developed algorithms to determine the average degrees for the *”Stanford web graph”* that you can find at http://snap.stanford.edu/data/web-Stanford.html.

Problem 37 (laboratory)

Let $G = (V, E)$ be a undirected graph. The clustering coefficient of the node $v \in V$ is the number

$$c(v) = \frac{2|\{ (i,j) \in E \ : \ i,j \in N(v) \}|}{Deg(v)(Deg(v)-1)}$$

where $N(v) = \{ u \ : \ (v,u) \in E \}$ and $Deg(v) = |N(v)|$.

Design procedures in Map-Reduce for determining the set $\{ (v, c(v), Deg(v)) \ : \ v \in V \}$. Also write the procedure for determining the average value of $\frac{1}{|V|}\sum_{v\in V} c(v)$.

Apply the developed algorithms to determine the average levels for the *”Stanford web graph”* (you must convert this graph into an undirected graph).

Problem 38 (exercise)

We have n servers (numeric values check for $n = 3000$). The likelihood of a single server being corrupted during a Map-Reduce task by a single server at time T is $1/p$ (assume for numerical calculations that $p = 3000$). Assume that server failure events in subsequent time slots are independent.

1. Compute the probability of the correct completion of the entire task.
2. Now share servers for 3 groups of 1000 servers. Perform each task on three servers (one from each group). The time to complete the task is now 3T. What is the probability of correctly completing the task now?