Leipzig University
Department of Computer Science

PPRL on Flink — Conceptual Design

Authors:
Martin Franke
Thomas Hornoff

Supervisor:
Ziad Sehili

August 10, 2016

1 Introduction

In order to fulfill the first assignment of the *Big Data Praktikum* three main topics have to be implemented. Firstly, the system has to be set up, i.e. Apache Flink has to run on a computer system. Secondly, the planned architecture for the implementation of PPRL with Flink has to be outlined. Thirdly, an implementation plan has to be described for future work. In the following sections all three issues are discussed.

2 Installation

In this section we will shortly outline setting up the Flink on a machine in order to test it and getting a first impression of the framework.

2.1 Setting up & testing Flink

Apache Flink[1] is downloadable from the Apache website `https://flink.apache.org`. The installation is straight forward and therefore does not need any explanation here, since there is a very good documentation on the *Apache Flink* website.

In case of testing the Bloom Filter we decided to use the Bloom Filter by *Baqend*[**?**]. Furthermore, we used *Debatty*[**?**] as a framwork for LSH. Both functions will be implemented by ourself, since our purposes will be different.

2.2 Dataset

The *North Carolina Voter Registratin Databse*[2] supports our puposes to test simple implementations of the Bloom-Filter and the LSH. This Dataset is one of the few which includes real names, addresses, as well as the age of these people. Since we want to encrypt this data it is highly important that we use such data for PPRL with Flink.

Furthermore, the use of a *DataCorruptor* constructing subsets of the dataset as well as modifying data is preferable. Implementing a corruptor as described in [3], and thus transforming it into Java is another issue of our goal on PPRL with Flink.

3 Architecture

The data owner will represent our data, i.e. the North Carolina Voter Registration and the given csv-files given by Sehili. We will use the first name, last name, address, and date of birth. Both, the length of the Bloom-Filter input and its token length must be flexible. That is, the data is encoded via the Bloom-Filter which then solves as input to the Lnkage Unit. The Linkage Unit uses LSH-MinHashes in order to block the data. The LSH-MinHashes will use

the Hamming-Distance to claculate the LSH. After blocking the data, the Linkage algorithm calculates the desired outcome, which is sent to the data owner. That concludes the cycle of PPRL with Flink.

4 Implementation Plan

In order to implement our architecture we will first have to read the Voter-files and extract the relevant data, i.e. first name, last name, address, and date of birth. We then have to generate N-Grams. Therefore, we have to use the FlatMap function in order to transfer the concatinated QIDs into N-Grams. N-Grams of the same record will then be combined using a Reduce-function. Having reduced the N-Grams of one record, Bloom-Filter will be calculated subsequently.

The Linkage Unit will then block these records and create blocks of Candidate-Record-Pairs using a LSH-MinHash-function using subkeys. That means that the FlatMap and Reduce functions will be used in the order 'FlatMap-¿FlatMap-¿Reduce'. In the first Linkage in the Linkage Unit all Candidates-Record-Pairs of one block are compared with the help of the FlatMap function. The second linkage will compare all Candidates-Record-Pairs of a block using the FlatMap function as well. The final and third linkage will then compare all Candidates-Record-Pairs of a block using the Reduce-function. Finally, the results are stored for further examitnation by the data owners.

References

[1] The Apache Software Foundation. Apache Flink. https://flink.apache.org, 2014-2015. [Online; accessed 02-June-2016].

[2] North Carolina Government. North Carolina Voter Registration Database. http://dl.ncsbe.gov, 2016. [Online; accessed 02-June-2016].

[3] Dinusha Vatsalan Peter Christen. Flexible and extensible generation and corruption of personal data. http://dl.acm.org/citation.cfm?id=2507815, 2013. [Online; accessed 02-June-2016].