THE UNIVERSITY OF SYDNEY

School of Information Technologies
A/Prof Uwe Roehm

COMP5349: Cloud Computing **1.Sem./2017**

Assignment 2: Apache Spark/Flink Programming

Group Work: 15% **15.05.2017**

Introduction

In this assignment, you have to write a series of Apache Spark or Apache Flink programs to analyze a *flow cytometry* data set that was collected to study the immune reaction of an organism to a virus infection. Note that this is a different data set than we have used in assignment 1. The analysis is structured into three separate tasks. You will practice dataflow programming techniques and also observe the ease of use of the Apache Spark and Flink frameworks.

Input Data Set Description

The input data can be found in the HDFS directory `/share/cytometry` in our cluster. There are several csv files all with names like `measurements_arcsin200_p`X`.csv`. All of them have the same comma-delimited format, with each line representing the measurements for a single cell:

```
sample, FSC-A, SSC-A, CD48, Ly6G, CD117, SCA1, CD11b, CD150, CD11c, B220, Ly6C,...
```

Each sample consists of a number of individual cells that have been analysed (such as a blood sample). Each row in the measurement files corresponds to one cell. Note that there is a header row that just gives the meaning of each column:

each row - one cell!!!

- The `sample` column specifies the specific sample to which a measurement belongs to.
- `FSC-A` and `SSC-A` are quality metrics about the measurements that reflect how much laser light was scattered from a measured cell (FSC: "forward-scatter"; SSC: "side-scattered").
- The remaining 14 columns contain the measurements for various cell markers. The values have been already normalised by us into the same value space (using arcsin). Each marker corresponds to a specific antigen on the cell surface that was investigated in an experiment. These antigens correspond to different functions or development stages of a cell, and the more an antigen is expressed by a cell, the higher is the measured value. For the purpose of this assignment, think of them as coordinates in a 14-dimensional (normalised) value space where each dimension has a name that corresponds to one of the antigens.

The meta-data for the experiments are stored in the csv file `/share/cytometry/experiments.csv` Each line of this file contains a record with the following `comma-delimited` information:

```
sample, date, experiment, day, subject, kind, instrument, researchers
```

with mutiple values need to flatmap

The `sample` column specifies to which specific flow cytometry sample a measurement belongs to. The `date` column specifies when an experiment was conducted. In our use case, the different samples are from an investigation of the immune reaction of an organism to an infection with the

West Nile Virus (WNV). `Day` gives the number of days since the infection until a sample was taken and analysed. A value of 0 corresponds to a healthy cell before the infection, a value of, eg., day 3 would be a sample taken at the third day of an infection. The samples are taken from different `subjects` and are of a certain `kind`. In our case, the samples are taken from the bone marrow of different individual mice. The last column lists one or multiple researchers who were conducting the different experiments, with the names of the researchers separated by semicolon.

Analysis Task Descriptions

You shall implement three different analysis tasks of the data set in either Apache Flink or Spark (as assigned by your tutor):

1. **Task 1: Number of (valid) measurements conducted *per researcher*.**
   The first task is an explorative analysis of the flow cytometry data similar to our assignment 1. You shall write a Flink or Spark program that finds the number of valid measurements done per individual researcher. We consider a measurement valid if its FSC-A and SSC-A values are in the range of 1 to 150,000. Note that there might be more than one researcher involved in analysing a certain sample . In this case, you can count all measurements from the same sample to each researcher who was involved.

   The output file should have the following format, ordered by number of measurements in descending order (researchers with most measurements first); researchers with the same number of measurements should be listed alphabetically:

   ```
   researcher \t numberOfMeasurements
   ```

2. **Task 2: k-means clustering of the measurements.**
   In the second task, you shall implement the iterative k-means algorithm and then use this algorithm to cluster all valid cell measurements into k clusters (k as input parameter). You can restrict your clustering to three dimensions, with the default: Ly6C, CD11b, and SCA1

   Filter out all measurements with FSC-A or SSC-A values outside the range 1 to 150,000. Then start with k random centroids and conduct k-means clustering for a number of iterations, with a default value of 10 iterations (configurable at runtime). To determine the distance between a point p and a cluster c use the standard Euclidean distance:
   $distance(p, c) = \sqrt{(c.x - p.x)^2 + (c.y - p.y)^2 + (c.z - p.z)^2}$

   The output file should have the following tab-delimited format (ordered by cluster ID):

   ```
   clusterID \t number_of_measurements \t Ly6C \t CD11b \t SCA1
   ```
   centroid

   The cluster ID is an integer value between 1 and k . For each cluster, give the number of valid measurements which have been associated to this cluster (i.e. the size of the cluster). The last three values specify the centroid of the cluster for the three dimensions which we consider (default: Ly6C, CD11b and SCA1).

3. **Task 3: Outlier removal and reclustering.**
   In the third task, you shall identify and remove outliers from the dataset: For each cluster, remove the 10% of measurements with the highest residual error, where residual error is defined as the Euclidean distance from a measurement to its assigned cluster centroid. Create a new dataset with these identified outliers removed. Then use this new dataset to produce a new k-means clustering with the same values for k and the same number of iterations. The output should have the same format than for Task 2.

Special Coding Requirements

1. You will solve this assignment with either Apache Flink or Apache Spark. The tutors will assign each team to one of these two systems so that each systems is used by half the teams. Note that whatever system you are not using now, you will then be using for assignment 3 so that by the end of the semester, each team will have worked with both systems.

2. The purpose of this assignment is to experience iterative programming in Spark/Flink. You are hence not allowed to use any pre-defined machine learning libraries for this assignment, but rather you should implement the k-means algorithm yourself.

3. If you use any code fragments or code cliches from third-party sources, you need to reference those properly. Include a clear statement on which parts of your submission are from yourself.

4. Always test your code using a small data set (eg. just day 0) before clustering a large one.

5. Do not copy the entire input data to your HDFS home directory or Linux home directory

6. We will provide a Python script to visualise the clustering results for debugging purposes.

Deliverable

There are three deliverables: **source code**, a brief **program design documentation** (up to 3 pages), and a **self-reflection survey**. All are due on Week 12 Monday 29th of May. The marking rubric will be available on eLearning. Please submit the source code and a soft copy of design documentation as a zip or tar file in eLearning. Your document should contain the following:

- **Job Design Documentation**
  In your document, describe the Flink/Spark jobs you use to implement Tasks 1 to 3. For each job, briefly describe the different transformation functions. If you use a user-defined functions, classes or operators, please describe those too.

- Include as appendix the HDFS location of your final output files from various executions.

Self-Reflection Survey: As part of the submission, you are also asked to fill in a self-reflection survey, one per each team member. This survey will ask you to assess your experience with your chosen implementation systems (Flink or Spark).

Demo: A few points of the marking scheme will be given to any submission which can be demoed successfully on either our own cluster, or when running on Windows Azure.

Group member participation

If members of your group do not contribute sufficiently you should alert your tutor as soon as possible. The tutor has the discretion to scale the group's mark for each member as follows, based on the outcome of the group's demo in either Week 11 or 12:

Level of contribution	Proportion of final grade received
No participation.	0%
Passive member, but full understanding of the submitted work.	50%
Minor contributor to the group's submission.	75%
Major contributor to the group's submission.	100%