Introduction to Spark - Practical Work

Vincent Leroy

2017

This exercise is part of the evaluation of the Data Management class and can be done in pairs of students. The deadline for this work is the 15th of December. Send a zip or tgz containing your names, code and comments at vincent.leroy@univ-grenoble-alpes.fr including **[M2R]** in the email subject.

1 Getting started

The setup for this exercise is exactly the same as for Twitter sentiment analysis. You need to use `sbt` to generate a project file and open it in Scala-IDE. You may have to fix the Scala version in the project properties. The data file we use in this exercise is `flickrSample.txt`. This file is part of a real dataset released by the Flickr website. Each line contains the meta-data of a picture hosted on Flickr. Data is organized in columns separated by tabulations. You will find all the information necessary in the `flickrSpecs.txt` file.

2 Computing tag usage per country

The file `FlickrExercise.scala` contains the beginning of a program loading the Flickr data using a RDD. You are also given a class `Picture.scala` to help you parse the picture meta-data. The `Picture` class will automatically use the `Country` class that contains a very basic GSP to country database. You can have a look at these classes to better understand them, but you should not need to modify them.

1. Display the 5 lines of the RDD and display the number of elements in the RDD.
2. Transform the `RDD[String]` in `RDD[Picture]` using the Picture class. Only keep interesting pictures having a valid country and tags. To check your program, display 5 elements.
3. Now group these images by country (`groupBy`). Print the list of images corresponding to the first country. What is the type of this RDD?
4. We now wish to process a RDD containing pairs in which the first element is a country, and the second element is the list of tags used on pictures taken in this country. When a tag is used on multiple pictures, it should appear multiple times in the list.
5. We wish to avoid repetitions in the list of tags, and would rather like to have each tag associated to its frequency. Hence, we want to build a RDD of type `RDD[(Country, Map[String, Int])]`.

6. There are often several ways to obtain a result. The method we used to compute the frequency of tags in each country quickly reaches a state in which the size of the RDD is the number of countries. This can limit the parallelism of the execution as the number of countries is often quite small. Can you propose another way to reach the same result without reducing the size of the RDD until the very end?