Final Project INF 583

Angelos-Christos Anadiotis Oana Balalau Pierre Bourhis

Organization

Your project should be ready before TO BE ANNOUNCED. You will be able to upload it on Moodle. The presentations will be on TO BE ANNOUNCED: you will have 15-20 minutes per presentation and the entire exam will take 45 minutes. You should expect questions based on both the project and the lectures. Each team must have two members. Specific requests for teams of three should be made to the instructors with proper justification. Note that the duration of the exam and the expectations for the project will rise proportionally with the team size.

A. Lists of integers

Design MapReduce algorithms that take as input a file(s) containing lists of integers (we have provided such a file on Moodle) and produce as output:

1. The largest integer.
2. The average of all the integers.
3. The same set of integers, but with each integer appearing only once.
4. The count of the number of distinct integers in the input.

You are requested to:

1. Implement exercises 1-4 in Apache Hadoop or Spark
2. Implement exercises 1, 2 and 4 in Spark Streaming (in addition to Hadoop/Spark)

Hint: For the implementation of 4 in Spark Streaming, use the Flajolet–Martin algorithm[1]. Note that this is a difficult exercise, so it is recommended to complete it after Section B. *Fun fact*: Philippe Flajolet was a student of Ecole Polytechnique.

B. Ranking Wikipedia Web pages with a centrality measure

Eigenvector centrality is a measure of the importance of a node in a graph. The centrality score is computed iteratively and its intuition is that, at each step the importance of a node is sent to its neighbours. Therefore, a high score means that a node is connected to many nodes, which have high scores themselves.

You are given a graph G where nodes represent Wikipedia Web pages, while edges represent hyperlinks. The graph is stored as a list of Wikipedia pages (*idslabels.txt*) and a set of links (*edgelist.txt*) where every line is of the kind:

$P\ P1\ P2 \dots Pn$

where P is the *id* of a web page, $P1, \dots, Pn$ are the pages P links to. In the file *idslabels.txt* every line contains an identifier and the page associated with the identifier.

There are $n = 64375$ pages. The graph is represented as an adjacency matrix A with n rows and n columns, i.e., $A(i, j) = 1$ if and only if there is a link between node i and j, otherwise $A(i, j) = 0$. Please

[1] `https://en.wikipedia.org/wiki/Flajolet-Martin_algorithm`

note this is a sparse matrix (the majority of elements of the matrix are 0), and we do not represent the elements that are 0. The eigenvector centrality score for each node can be computed as follows:

1. Let r^0 be the uniform vector, i.e. each element is equal to $1/n$. This is the eigenvector of the pages (i.e. it contains one score for each page) at step 0. The vector has to be created and stored in a file.
2. Let r^t be the eigenvector at step t. Iterate for a number of iterations given in input:

$r^{t+1} := Ar^t$.

$r^{t+1} := r^{t+1}/||r^{t+1}||$, where $||r||$ is the L_2 norm of the vector r.

Questions:

1. Implement the eigenvector centrality in Apache Hadoop, Apache Spark and using only threads and compare the performance of the three implementations. For threads, implement matrix multiplication using blocks[2].

2. What is the most important page in Wikipedia? Answer using your previous implementations, in Apache Hadoop or Apache Spark.

3. Implement matrix multiplication using two mapreduce steps and using just one mapreduce step. Is there a difference in performance?

4. Many applications are bottlenecked by transfering the output of map tasks to the corresponding reduce tasks, instead of the actual computations that are done by the map and reduce tasks themselves. Assume that the cost of a task is the size of the input to the task, that is the number of tuples it receives. Please note that the reducer does not receive in input from the mapper the tuple $(key, [v1, v2 \ldots])$, but the tuples $(key, v1), (key, v2) \ldots$. Let the cost of an algorithm be the sum of the costs of all the tasks implementing that algorithm. What is the cost of computing the eigenvector centrality? You can use n and m, the number of nodes and edges in graph, and k the number of iteration of the algorithm, to express this cost. Is there a difference according to the number of mapreduce steps used for the matrix multiplication?

Hint: A map function can output more than one (key, value) pair. The value can be a tuple, and you can use this tuple to know if an element is part of the matrix or of the vector.

[2] `https://ximera.osu.edu/la/LinearAlgebra/MAT-M-0023/main`