« Big Data » course - CentraleDigitalLab@LaPlateforme – 2023-2024

Tutorial 2: Advanced Map-Reduce algorithms in Spark

Stéphane Vialle & Gianluca Quercini

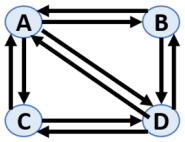
This tutorial deals with RDD problems involving key-value pairs, where the sequence of transformations modifies the keys. Some steps may transform values into keys, or vice versa, or build new keys.

Exercise 3: Search for common friends in a social network

Consider a social network graph encoded in a text file.

- A line in the file is a comma-separated list of identifiers: A, B, C, D, which means that A has friends B, C and D.
- All the lines in the file describe the graph's "has for friend" relationships. Ex:

A, B, C, D
B, A, D
C, A, D
D, A, B, C



- We'll assume symmetrical relations: if we have A, B then we have B, A.

We wish to obtain the list of friends common to pairs of individuals, in the form of an RDD of key-value pairs:

((A, B), [D])
((A, C), [D])
((A, D), [B, C])
((B, C), [A, D])
((B, D), [A])
((C, D), [A])

Note: We only want to show each couple once (as above). If we show the friends of couple (A, B), then we will NOT show couple (B, A).

The following Spark code is to be completed:

```
#Search for new friends ----------------------------------------
def common_friends(theTextFile):
  cf = theTextFile                                  \
       .map(lambda line: line.split(","))           \
       . ..............      # nb of lines is not limited - TO DO TO DO
  return cf

#main code -----------------------------------------------------
input_file_name = ......      #details in the real python source file
output_file_name = ......     #details in the real python source file

sc = SparkContext()
text_file = sc.textFile(input_file_name)

cf = common_friends(text_file)
cf.saveAsTextFile(output_file_name)
```

Question 3.1: production of intermediate key-value pairs

First, we'll try to produce key-value pairs representing ONE friend common to a pair of individuals.

Ex : ((A, D), B) signifiera que B est un ami commun à A et D. La clé est ici un couple.

- Propose a solution in Spark to obtain the set of key-value pairs (Couple, Friend) of the graph from the text file representing the graph.
- Does your solution generate symmetrical pairs?

Question 3.2: production of the complete solution

- Now propose a complete solution in Spark, to obtain the list of common friends of pairs of individuals (if they do indeed have common friends):

  (A, B), [X_1, X_2, … X_n]
  ….
- Do you remain on RDD until the end of treatment?
- What are your *Narrow* and *Wide* stages?

Question 3.3: modeling processing complexity

Let be:

- N_{id}: the nbr of identifiers (nodes) in the graph (i.e. the nbr of individuals in the social network)
- N_{link}: the total nbr of bidirectional links in the graph (i.e. the total nbr of symmetrical relationships between two individuals)
- n_{link}^{id}: the nbr of links leaving the id node.

The number of links n_{link}^{id} varies from node to node. But in this tutorial, we make the simplifying assumption that for each node: $n_{link}^{id} = \overline{n_{link}}$

- Calculate the number of intermediate pairs ((A,B), X) generated at the output of question 1.
- Can we consider that the complexity of the problem is directly linked to the number of pairs generated at the output of question 1 (and therefore at the input of question 2)?

If N_{id} and N_{link} are totally independent values, is this search for common friends more sensitive to the total number of nodes or the total number of links?