AMD project report: Pagerank

Alfredo Funicello

March 20, 2023

1 Introduction

This report shows an implementation of the PageRank index calculation using pySpark on the 'Amazon US Customer Reviews' dataset. The ranked nodes are products, which are linked if ever reviewed by at least one same customer.

2 Methodology

2.1 Data

The dataset is available on Kaggle under the name 'Amazon US Customer Reviews Dataset'. A sampling of the initial 10gb dataset has been chosen for this work; the data utilized is composed by the Baby products, Digital videogames and Digital movies products datasets. The total amount of rows in the sampled dataset is approximately 3.5 mln.

The rows are composed by multiple attributes but the only relevant ones, which have been selected, for this work are:

- `customer_id`: Random identifier that can be used to aggregate reviews written by a single author.
- `product_parent`: Random identifier that can be used to aggregate reviews for the same product (referred as `product_id` from now on).

2.2 Shaping the data

The initial shape of the data is `(customer_id - product_id)`.

The first reshaping operation is a join of the data on itself on the attribute `customer_id` which creates rows of the shape `(customer_id, (product_id1,product_id2))` where `product_id1` and `product_id2` are both products who have been bought by the `customer_id` (*cell 5*).

After the join, through a `filter` operation, rows in which the `product_id` is duplicated are removed (*cell 6*).

The next step is removing the `product_id` index to create an edge list of the form `(product_id1,product_id2)` (*cell 7*).

Through a `reduceByKey` operation the edge list is reduced to an adjacency list of the form `(product_id1,[product_id2,product_id3,...])` (*cell 8*).

2.3 Pagerank computation

The first operation needed for the Pagerank computation is the creation of a transition matrix.

The transition matrix of the form `(product_id1,product_id2,prob)`, where `prob` is the probability of moving from node `product_id1` to node `product_id2`, has been initialized uniformly (*cell 9*).

The Pagerank calculation is roofed by 300 steps or an Euclidean distance from the previous step lower than 10^{-10}. A taxation parameter β of 0.85 has been chosen to adjust for spidertraps and dead ends in the graph. The Pagerank vector is kept in main memory.

The calculation step is modelled as a vector matrix multiplication where the Pagerank x_i, which is the probability of a surfer being at node i in the next step, is defined by:

$$x_i = \beta \sum_j m_{ij} v_j + (1-\beta) 1/n \tag{1}$$

where v_j is the probability that the surfer was at node j in the previous step and m_{ij} is the probability that the the surfer will move from the node j to the node i. n is the total number of nodes in the graph.
This operation can be modelled on pySpark with the MapReduce framework through a Map and Reduce operation (*cell 15*).
The `map` function executes the multiplications between the node current Pagerank and the Pagerank of the linked nodes. The calculated values are then summed up by a `reduce` function which will come up with the pagerank value x_i from the next step for the current node.

2.4 Topic aware Pagerank

A topic aware Pagerank calculation has been explored. The videogames category has been chosen as a selected topic (*cell 18*). An array has been used to recognize which items are part of the selected set. The items which belong to the selected set get a boost in their pagerank value given by the teleportation segment of the summation. The summation for a selected set item is defined as:

$$x_i = \beta \sum_j m_{ij} v_j + (1-\beta) 1/s \tag{2}$$

where s is the order of the selected set rather than the total number of nodes. The summation for a non selected item is instead:

$$x_i = \beta \sum_j m_{ij} v_j \tag{3}$$

3 Conclusions

3.1 Results

The average found Pagerank in the basic version of the computation is 0.001135444161043827. The results are dramatically different in the Topic aware computation, which showcases the possibility of steering the Pagerank values towards a certain topic; infact the average pagerank found for products in the Videogame category, in the second computation, is 0.007904045707260303, much higher than the previous found total value. The average Pagerank for items in categories Baby products and Music, in the second computation, is 0.0005901374609417463; Pagerank value has been driven away from them and moved to Videogames.

3.2 Possible improvements

The pagerank array has been stored in the main memory, as well as the array used to identify Videogames items. These two arrays, while small enough to fit in main memory in the context of this work, could become unmaneageable in an actual Big Data scenario. Some additional work and design could be done to move them to pySpark in order to be able to distribute them between workers.