Labolatory Exercise no: 3

Sizwe Lekoba (793314)[*], Phiwe Mbongwa (606352)[†], Ndabezinhle Ndlovu (545276)[‡], Tshembani Sibuyi (838214)
School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2050, South Africa
ELEN4020: Data Intensive Computing in Data Science

I. Introduction

This paper describes the implemenation of matrix multiplication using the map reduce framework. This is done using Mrjob, a Python based library.

II. solution

A. Algorithm A

The procedure for algorithm one involves a single step mapping function and a single step redusing function. The mapping function reads in the file and assigns a key value pair to each line in the file excluding the first line. The format of the file is a matrix market format. Two files are read in, for matrix one and two which are to be multiplied. The key of the file line is made of the matrix the value belongs to and the index of that value. the value is simply the actual value in the matrix. The reducer then creates two lists, each containing one of the matrices' values. Each element in the two lists are then multiplied together to give a new value in the key value pair. Let "i,j" be the index of matrix one values and "a,b" the index of matrix two values. The key in the new key value pair of the multiplied elements is then "i,b". Finally all the values with the same keys are summed together to produce the resultant matrix.

Algorithm 1 The mapping function

for each element m_{ij} of M **do**
  produce (key, value) pairs as $[(M, i, j), (m_{ij})]$
  for k = 1,2,3 ... up to the number of columns of N
for each element n_{ij} of N **do**
  produce (key, value) pairs as $[(N, i, j), (n_{ij})]$
  for k = 1,2,3 ... up to the number of rows of N
return Set of (key, value) pairs

Algorithm 2 The reducing function

for each key(i,k) **do**
  sort values begin with M by j in $list_M$
  sort values begin with N by j in $list_N$
  multiply m_{ij} and n_{jk} for j^{th} list of each list
  sum up $\sum_{j=1} m_{ij} * n_{jk}$
return $(i, k), \sum_{j=1} m_{ij} * n_{jk}$

B. Algorithm B

The second procedure implements matrix multiplication as two cascaded MapReduce operation. The first mapReduce involves a Map function that produce the key value pair (j,(M,i,m_{ij})), for the first matrix and a key value pair (j,(N,k,n_{jk})) for the second matrix. Where M and N are matrix names with elements m_{ij} and n_{jk} respectively.i and j are the rows and columns of matrix M and the rows and columns of marix N are represented by j and k.The first mapReduce also involves Reduce function where each key j examine the list of its associated values and produce a key-value pair with key (i,k) and the value being the product of m_{ij} and n_{jk}. The second mapReduce have a reduce function that sum up values of the list for each key (i,k). The result is the key-value pair ((i,k),value).

Algorithm 3 The mapping function

for each element m_{ij} of M **do**
  produce (key, value) pairs as $[j, (M, i, m_{ij})]$
  for k = 1,2,3 ... up to the number of columns of N
for each element n_{ij} of N **do**
  produce (key, value) pairs as $[j, (N, k, n_{jk}]$
  for k = 1,2,3 ... up to the number of rows of N
return Set of (key, value) pairs

Algorithm 4 The reducer1 function

for each key j that **do**
for each value of N and M that **do**
  multiply m_{ij} and n_{jk} for j^{th} list
return$(i, k), m_{ij} * n_{jk}$

Algorithm 5 The reducer2 function

for each key (i,k) **do**
  sum up the list of of value associated with this key
return$(i, k), V$

C. Three path network

The network G is represented as a matrix. Each element in the matrix represents a node. If the element is a one the node is connected to the network, if its a zero then the node is not connected to the network. To determine a length of three in a path the matrix is to be multiplied by itself three times. This is done using one of the previously described algorithms. A count of the remaining non zeros in the resultant matrix is summed. This is the total number of paths of length three in the network.