UNIVERSITY OF WITWATERSRAND

ELEN4020

DATA INTENSIVE COMPUTING IN DATA SCIENCE

Laboratory Exercise No 3

Authors:
KISHAN NAROTAM - 717 931
JESAL CHANA - 603 177

Authors:
SYED HUSSAIN - 600 524
YUSUF ALLY - 604 973

5th April 2018

I. Problem Description

Two matrices, *A* and *B* of different dimensions and sizes must be multiplied together to produce the product matrix, *C*, i.e. $C = A \times B$. In order for A and B to be multiplied, the rules of matrix multiplication must be met. These matrices are read in from text files, where the respective matrices are stored in the *Matrix-Market* format. For example, if matrix A is a 3×4 matrix, represented as:

$$A[3][4] = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix}$$

This matrix will be read in from the file as follows:

3	4	
0	0	a
0	1	b
2	3	l
0	3	d
1	0	e
2	0	i
1	1	f
1	2	g
2	1	j
0	2	c
2	2	k
1	3	h

The first line of the text file is the dimensions of the matrix, and following that, the first two elements of each line is the position, A_{ij} of the third element in the line which is the value at that respective position within the matrix. The matrix could be an integer which is 4 bytes or a short floating point which too is 4 bytes. The elements of the matrix must be generated using the formula presented in equation 1:

$$A\langle i, j\rangle = i * N_j + j \tag{1}$$

II. Background

A. MapReduce

MapReduce is said to be the heart of the *Apache Hadoop* software framework. This framework is written and built in order to process large amounts of data in parallel on large computer clusters, i.e. multiple nodes, with each node of the cluster having its own storage. There are two essential tasks that the program performs. The first of which is the `map` function which is said to take a set of data and convert it into another set. This *mapper* filters and divides the work to the various nodes on the cluster (map). The individual elements are reduced to tuples, i.e. (*key, value pairs*) [1], [2], [3].

The second task is the `reduce` function, which is responsible for taking the output from the `map`, which in turn becomes the input, and combines the data tuples into smaller set of tuples. The *reducer* organizes and reduces the results from each node into a cohesive answer to a query. As the name suggests, the `reduce` function will always happen after the `map` function [1], [2], [3].

An example of how the *MapReduce* can be used could be as follows. Consider 5 files, containing two columns each. In terms of the framework, each file has a key and a value. Within each file, the first column is the name of a city and the second column would be the temperature on a random day. From these files, the goal is to find the maximum temperature for each city recorded. Using the *MapReduce* framework, five mappers will work on its own file. The mapper will return the maximum temperature for each city. The reducer will then take the results from each mapper, and produce one output of each city and their maximum temperature recorded [2].

III. Function Description

Using the __ framework, the algorithm was written in __ programming language.

A. Algorithm A: Repetitive use of MapReduce

Given two matrices, *M* and *N*, with the respective elements m_{ij} and n_{jk}, the product, *P*, will be $P = MN$, with the elements p_{ik}, where

$$p_{ik} = \textstyle\sum_j m_{ij} n_{jk}$$

The matrix will three key attributes, the first two would be regarded as the key, i.e. row number and column number, with the third being the value at that position. Matrix M has the relation $M(I, J, V)$ with the corresponding tuples (i, j, m_{ij}). Matrix N would have a similar relation as $N(J, K, W)$ with the tuples (j, k, n_{jk}). Since *MapReduce* was created in order to handle larget sets of data, many large matrices are sparse, i.e. most elements are 0, and thus these tuples can be omitted. i, j and k could be implicit in the position of an element rather than written explicitly with the element itself. In a situation like this, the `map` function must be designed to construct I, J, and K from the tuples from the position of the data [4].

B. Algorithm B: Using MapReduce once

IV. Pseudo Code

V. Final Code and Output

VI. Conclusion

The conclusion goes here.

References

[1] Bigelow, SJ; Chu-Carroll, MC; *What is MapReduce? - Definition from WhatIs.com*; WhatisMapReduce?-DefinitionfromWhatIs.com; Last Accessed: 04/04/2018

[2] IBM Analytics; *What is MapReduce? — IBM Analytics*; https://www.ibm.com/analytics/hadoop/mapreduce; Last Accessed: 04/04/2018

[3] Hadoop; *MapReduce Tutorial*; https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html; Last Accessed: 04/04/2018

[4] Leskovec, J; Rajaraman, A; Ullman, JD; *Mining of Massive Datasets - Chapter 2*; Stanford University; Palo Alto, CA; March 2014