Project Title:

Product of Vectors, Matrixes using Big Data Technologies

Project Statement:

Matrix/Vector multiplication is the one of the most fundamental operations that most of the machine learning algorithms rely on. Knowing the working of matrix multiplication in a distributed system provides important insights on understanding the cost of our algorithms.

Proposed Solution:

Naïve Matrix Multiplication:

Suppose **A** and **B** are two **matrices** with dimensions respectively **(am,an)**and **(bm, bn)**and that you computing A \* B.

1. You can perform the **matrix multiplication** if and only if a**n** **= bm**. In other words, you can only perform if the number of columns of the left matrix is equal to the number of rows of the right matrix.
2. The result of the multiplication is a **matrix** with dimensions **(am, bn)**.
3. Each entry of the new matrix will be the**sum of the product** of the corresponding **row in A** and **column in B.**

we repeat the same process for every row of the left matrix and every column of the right one.

Algorithm pseudocode:

Data: A[am][an], B[bn][bm]

Result: R[][]={0}

If an == bn then

For m=0; m<am,m++ do

For r=0; r<bm;r++ do

For k=0;k<bn;k++ do

Q[m][r] += A[m][k] \* B[k][r];

End

End

End

End

Time Complexity Analysis:

The naive matrix multiplication contains three nested loops. For each iteration of the outer loop, the total number of the runs in the inner loops would be equivalent to the length of the matrix. Here, integer operations take O(1) time. **In general, if the length of the matrix is** N, **the total time complexity would be** O(N3 )

Matrix Multiplication using Map Reduce:

Before working on the code to compute multiplication using map reduce, we need to understand the data representation.

First column represents the matrix name of which the value belongs.

Second column represents the row of matrix in which the value belongs.

Third column represents the column of matrix in which the value belongs

Fourth column represents the value.

Mapper Function:

The mapper will read the matrixes and to differentiate them we keep a count of line number we are reading. We keep a key value, which represents the elements that need to be multiplied and the elements that need to summed.

{0} {1} {2} are the part of key and {3} is the value.

Diagram

Description automatically generated

{0} {1} {2} actually represents the position of element from A or B to A\*B

* {0} is the row position of the element
* {1} is the column position of the element
* {2} is the position of the element in addition. (like 1, 6 are at position 0 in addition and 2,5 are at position 1)

Elements with same key will get multiplied and the elements with same first two numbers of the key are part of sum in same row.

After mapper produces its result, Hadoop will sort the mapper results by key and provide it as an input to Reducer Function.

Mapper Code:

Text

Description automatically generated

Reducer Function:

The Reducer function does the multiplication of the values with same key and sums up the values.

Text

Description automatically generated

Time Complexity Analysis:

Time complexity of Map function: O(1).

Time complexity of Reduce function: O(N2)

Assuming the number of clusters is K. then overall Time complexity is O((N2)/k).

Results and Analysis:

I have implemented two matrix multiplication programs, one in naïve matrix multiplication and other using map reduce to capture the time differences in both methodologies.