Algorithms

M/R Algorithms

Basic Algorithms

Addition

Addition of multiple matrices

- https://issues.apache.org/jira/browse/HAMA-154

Multiplication

- Iterative Approach

```
For i = 0 step 1 until N -1
  Job: Computes the ith row of C = Matrix-Vector multiplication

Iterative job:

- A map task receives a row n of B as a key, and vector of row as its value
 - Multiplying by all columns of ith row of A
- Reduce task find and add the ith product

1st
+             +   +             +
| a11 a12 a13 |   | a11 a21 a31 |
| ... ... ... | X | a12 a22 a32 |
| ... ... ... |   | a13 a23 a33 |
+             +   +             +

2nd
+             +   +             +
| ... ... ... |   | a11 a21 a31 |
| a21 a22 a23 | X | a12 a22 a32 |
| ... ... ... |   | a13 a23 a33 |
+             +   +             +
....
```

- Blocking Algorithm Approach

To mutliply two dense matrices A and B, We collect the blocks to 'collectionTable' firstly using map/reduce. Rows are named as c(i, j) with sequential number ((N^2 * i) + ((j * N) + k) to avoid duplicated records.

```
CollectionTable:

                            matrix A         matrix B
------------------------+-------------------------------
block(0, 0)-0               block(0, 0)      block(0, 0)
block(0, 0)-1               block(0, 1)      block(1, 0)
block(0, 0)-2               block(0, 2)      block(2, 0)
...         N               ...
block(N-1, n-1)-(N^3-1)     block(N-1, N-1)  block(N-1, N-1)
```

Each row has a two sub matrices of a(i, k) and b(k, j) so that minimized data movement and network cost.

```
Blocking jobs:

Collect the blocks to 'collectionTable' from A and B.

- A map task receives a row n as a key, and vector of each row as its value
 - emit (blockID, sub-vector) pairs
- Reduce task merges block structures based on the information of blockID

Multiplication job:

- A map task receives a blockID n as a key, and two sub-matrices of A and B as its value
 - Multiply two sub-matrices: a[i][j] * b[j][k]
- Reduce task computes sum of blocks
 - c[i][k] += multiplied blocks
```

Matrix Norm

- Find the maximum absolute row sum of matrix

Matrix.Norm.One is that find the maximum absolute row sum of matrix. Comparatively, it's a good fit with MapReduce model because doesn't need iterative jobs or table/file JOIN operations.

```

                                          j=n
The maximum absolute row sum =   max   ( sum | a_{i,j} | )
                               1<=i<=n   j=1


- A map task receives a row n as a key, and vector of each row as its value
 - emit (row, the sum of the absolute value of each entries)
- Reduce task select the maximum one
```

NOTE: Matrix.infinity, Matrix.Maxvalue and Matrix.Frobenius are almost same with this.

Compute the transpose of matrix

The transpose of a matrix is another matrix in which the rows and columns have been reversed. The matrix must be square for this work.

```
+              +    +              +
| a11 a12 a13 |    | a11 a21 a31 |
| a21 a22 a23 | => | a12 a22 a32 |
| a31 a32 a33 |    | a13 a23 a33 |
+              +    +              +

- A map task receives a row n as a key, and vector of each row as its value
 - emit (Reversed index, the entry with the given index)
- Reduce task sets the reversed values
```

Compute the determinant of square matrix

- http://issues.apache.org/jira/browse/HAMA-66

Decomposition Algorithms

Cholesky Decomposition

- http://issues.apache.org/jira/browse/HAMA-94

Singular Value Decompostion

- http://issues.apache.org/jira/browse/HAMA-176