FrovedisBlockcyclicMatrix

NAME

FrovedisBlockcyclicMatrix - A data structure used in modeling the in-memory blockcyclic matrix data of frovedis server side at client spark side.

SYNOPSIS

import com.nec.frovedis.matrix.FrovedisBlockcyclicMatrix

Constructors

FrovedisBlockcyclicMatrix (`RDD[Vector]` data)

Public Member Functions

Unit load (`RDD[Vector]` data)
Unit load (String path)
Unit loadbinary (String path)
Unit save (String path)
Unit savebinary (String path)
RowMatrix to_spark_RowMatrix (SparkContext sc)
Vector to_spark_Vector ()
Matrix to_spark_Matrix ()
Unit debug_print()
Unit release()

DESCRIPTION

FrovedisBlockcyclicMatrix is a pseudo matrix structure at client spark side which aims to model the frovedis server side `blockcyclic_matrix<double>` (see manual of frovedis blockcyclic_matrix for details).

Note that the actual matrix data is created at frovedis server side only. Spark side FrovedisBlockcyclicMatrix contains a proxy handle of the in-memory matrix data created at frovedis server, along with number of rows and number of columns information.

Constructor Documentation

FrovedisBlockcyclicMatrix (RDD[Vector] data)

It accepts a spark-side RDD[Vector] and converts it into the frovedis server side blockcyclic matrix data whose proxy along with number of rows and number of columns information are stored in the constructed FrovedisBlockcyclicMatrix object.

For example,

```
// sample input matrix file with elements in a row separated by whitespace
val data = sc.textFile(input)
// parsedData: RDD[Vector]
val parsedData = data.map(s => Vectors.dense(s.split(' ').map(_.toDouble)))
// conversion of spark data to frovedis blockcyclic matrix
val fdata = new FrovedisBlockcyclicMatrix(parsedData)
```

Pubic Member Function Documentation

Unit load (RDD[Vector] data)

This function can be used to load a spark side dense data to a frovedis server side blockcyclic matrix. It accepts a spark RDD[Vector] object and converts it into the frovedis server side blockcyclic matrix whose proxy along with number of rows and number of columns information are stored in the target FrovedisBlockcyclicMatrix object.

For example,

```
// sample input matrix file with elements in a row separated by whitespace
val data = sc.textFile(input)
// parsedData: RDD[Vector]
val parsedData = data.map(s => Vectors.dense(s.split(' ').map(_.toDouble)))

val fdata = new FrovedisBlockcyclicMatrix() // an empty object
// conversion of spark data to frovedis blockcyclic matrix
fdata.load(parsedData)
```

Unit load (String path)

This function is used to load the text data from the given file in the target server side matrix instance. Note that the file must be placed at server side at the given path.

Unit loadbinary (String path)

This function is used to load the little-endian binary data from the given file in the target server side matrix instance. Note that the file must be placed at server side at the given path.

Unit save (String path)

This function is used to save the target matrix as text file with the filename at the given path. Note that the file will be saved at server side at the given path.

Unit savebinary (String path)

This function is used to save the target matrix as little-endian binary file with the filename at the given path. Note that the file will be saved at server side at the given path.

RowMatrix to_spark_RowMatrix (SparkContext sc)

This function is used to convert the target matrix into spark RowMatrix. Note that this function will request frovedis server to send back the distributed data at server side blockcyclic matrix in the rowmajor-form and the spark client will then convert the distributed chunks received from frovedis server to spark distributed RowMatrix.

The SparkContext object "sc" will be required while converting the frovedis data to spark distributed RowMatrix.

Vector to_spark_Vector ()

This function is used to convert the target matrix into spark Vector form. Note that this function will request frovedis server to send back the distributed data at server side blockcyclic matrix in the rowmajor-form and the spark client will then convert the received rowmajor data from frovedis server into spark non-distributed Vector object.

Matrix to_spark_Matrix ()

This function is used to convert the target matrix into spark Matrix form. Note that this function will request frovedis server to send back the distributed data at server side blockcyclic matrix in the column-major form and the spark client will then convert the received column-major data from frovedis server into spark Matrix object.

Unit debug_print()

It prints the contents of the server side distributed matrix data on the server side user terminal. It is mainly useful for debugging purpose.

Unit release()

This function can be used to release the existing in-memory data at frovedis server side.