frovedis::rowmajor_matrix_local<T>

NAME

`frovedis::rowmajor_matrix_local<T>` - A two-dimensional dense matrix with elements stored in row-wise order supported by frovedis

SYNOPSIS

```
#include <frovedis/matrix/rowmajor_matrix.hpp>
```

Constructors

rowmajor_matrix_local ();
rowmajor_matrix_local (size_t nrow, size_t ncol);
rowmajor_matrix_local (const `rowmajor_matrix_local<T>`& m);
rowmajor_matrix_local (`rowmajor_matrix_local<T>`&& m);
rowmajor_matrix_local (const `std::vector<T>`& v);
rowmajor_matrix_local (`std::vector<T>`&& v);

Overloaded Operators

`rowmajor_matrix_local<T>`& operator= (const `rowmajor_matrix_local<T>`& m);
`rowmajor_matrix_local<T>`& operator= (`rowmajor_matrix_local<T>`&& m);

Public Member Functions

void set_local_num (size_t nrow, size_t ncol);
void save (const std::string& file);
void savebinary (const std::string& dir);
void debug_print ();
`rowmajor_matrix_local<T>` transpose () const;
`node_local<rowmajor_matrix_local<T>>` broadcast();

Public Data Members

`std::vector<T>` val;
size_t local_num_row;
size_t local_num_col;

DESCRIPTION

`rowmajor_matrix_local<T>` is a template based non-distributed row-major data storage supported by frovedis.

Although it provides a 2D row-major storage view to the user, internally the matrix elements are stored in 1D vector form with additional row and column number information stored separately. The structure of this class is as follows:

```
template <class T>
struct rowmajor_matrix_local {
  std::vector<T> val;     // to contain matrix elements in 1D rowmajor form
  size_t local_num_row;   // number of rows in 2D matrix view
  size_t local_num_col;   // number of columns in 2D matrix view
};
```

Constructor Documentation

rowmajor_matrix_local ()

This is the default constructor which creates an empty rowmajor matrix with local_num_row = local_num_col = 0.

rowmajor_matrix_local (size_t nrow, size_t ncol)

This is the parameterized constructor which creates an empty rowmajor matrix of the given dimension (memory allocation takes place).

rowmajor_matrix_local (const `rowmajor_matrix_local<T>`& m)

This is the copy constructor which creates a new rowmajor matrix by deep-copying the contents of the input rowmajor matrix.

rowmajor_matrix_local (`rowmajor_matrix_local<T>`&& m)

This is the move constructor. Instead of copying the input matrix, it moves the contents of the input rvalue matrix to the newly constructed matrix. Thus it is faster and recommended to use when input matrix will no longer be used in a user program.

rowmajor_matrix_local (const `std::vector<T>`& v)

This is a special constructor for implicit conversion. It converts an input lvalue `std::vector<T>` to `rowmajor_matrix_local<T>` with dimensions Nx1, where N = size of the input vector. It attempts to copy the input vector during the conversion. Thus input vector remains unchanged.

rowmajor_matrix_local (`std::vector<T>`&& v)

This is a special constructor for implicit conversion. It converts an input rvalue `std::vector<T>` to `rowmajor_matrix_local<T>` with dimensions Nx1, where N = size of the input vector. It attempts to move the elements from the input vector during the conversion. Thus input vector will contain unknown values after the conversion.

Overloaded Operator Documentation

rowmajor_matrix_local<T>& operator= (const rowmajor_matrix_local<T>& m)

It deep-copies the input rowmajor matrix into the left-hand side matrix of the assignment operator "=".

rowmajor_matrix_local<T>& operator= (rowmajor_matrix_local<T>&& m)

Instead of copying, it moves the contents of the input rvalue rowmajor matrix into the left-hand side matrix of the assignment operator "=". Thus it is faster and recommended to use when input matrix will no longer be used in a user program.

Public Member Function Documentation

void set_local_num (size_t nrow, size_t ncol)

It sets the matrix information related to number of rows and number of columns as specified by the user. It assumes the user will provide the valid matrix dimension according to the number of elements in it. Thus no validity check is performed on the provided dimension values.

void debug_print ()

It prints the contents and other information related to the matrix on the user terminal. It is mainly useful for debugging purpose.

For example,

```
std::vector<int> v = {1,2,3,4};
rowmajor_matrix_local<int> m;
m.val.swap(v);
m.set_local_num(2,2); // nrow: 2, ncol:2
m.debug_print();
```

The above program will output:

```
node = 0, local_num_row = 2, local_num_col = 2, val = 1 2 3 4
```

rowmajor_matrix_local<T> transpose ()

It returns the transposed rowmajor_matrix_local of the source matrix object.

For example,

```
std::vector<int> v = {1,2,3,4};
rowmajor_matrix_local<int> m;
m.val.swap(v);
m.set_local_num(2,2); // nrow: 2, ncol:2
std::cout << m.transpose(); // a rowmajor matrix can be printed on user terminal
```

It will output like:

```
1 3
2 4
```

void save (const std::string& file)

It writes the elements of a rowmajor matrix to the specified file in rowmajor format with text data.

void savebinary (const std::string& dir)

It writes the elements of a rowmajor matrix to the specified directory in rowmajor format with binary data.

The output directory will contain two files, named "nums" and "val" respectively. "nums" is a text file containing the number of rows and number of columns information in first two lines of the file. And "val" is a binary file containing the matrix elements stored in little-endian form.

`node_local<rowmajor_matrix_local<T>> broadcast();`

It broadcasts the source `rowmajor_matrix_local<T>` to all the participating worker nodes. After successful broadcasting, it returns a `node_local<rowmajor_matrix_local<T>>` object representing the broadcasted matrices at each worker nodes.

It is equivalent to broadcasting the matrix using frovedis global function "frovedis::broadcast()" (explained in node_local manual). But from performance point of view this is efficient as it avoids the internal serialization overhead of the vector elements.

For example,

```
std::vector<int> v = {1,2,3,4};
rowmajor_matrix_local<int> m;
m.val.swap(v);
m.set_local_num(2,2); // nrow: 2, ncol:2
auto bm1 = m.broadcast(); // faster
auto bm2 = frovedis::broadcast(m); // slower (serialization overhead)
```

```
master                            worker0                    worker1
------                            -------                    -------
m: rowmajor_matrix_local<int>
   1 2
   3 4

bm1: node_local<
     rowmajor_matrix_local<int>>  rowmajor_matrix_local<int>  rowmajor_matrix_local<int>
                                     1 2                        1 2
                                     3 4                        3 4
bm2: node_local<
     rowmajor_matrix_local<int>>  rowmajor_matrix_local<int>  rowmajor_matrix_local<int>
                                     1 2                        1 2
                                     3 4                        3 4
```

Public Data Member Documentation

val

An instance of `std::vector<T>` type to contain the elements of the matrix in 1D row-major form.

local__num__row

A size_t attribute to contain the number of rows in the 2D matrix view.

local__num__col

A size_t attribute to contain the number of columns in the 2D matrix view.

Public Global Function Documentation

`rowmajor_matrix_local<T>` make__rowmajor__matrix__local__load(filename)

Parameters
filename: A string object containing the name of the text file having the data to be loaded.

Purpose
This function loads the text data from the specified file and creates a `rowmajor_matrix_local<T>` object filling the data loaded.
It assumes that there is no empty lines in the input file. The desired type of the matrix (e.g., int, float, double etc.) is to be explicitly specified when loading the matrix data from reading a file.

For example, considering "./data" is a text file having the data to be loaded,

```
auto m1 = make_rowmajor_matrix_local_load<int>("./data");
auto m2 = make_rowmajor_matrix_local_load<float>("./data");
```

"m1" will be a `rowmajor_matrix_local<int>`, whereas "m2" will be a `rowmajor_matrix_local<float>`.

Return Value
On success, it returns the created matrix of the type `rowmajor_matrix_local<T>`. Otherwise, it throws an exception.

`rowmajor_matrix_local<T>` make__rowmajor__matrix__local__loadbinary(dirname)

Parameters
dirname: A string object containing the name of the directory having the data to be loaded. It expects two files "nums" and "val" to be presented in the input directory, where "nums" is the text file containing number of rows and number of columns information (new line separated) and "val" is the little-endian binary data to be loaded.

Purpose
This function loads the binary data from the specified directory and creates a `rowmajor_matrix_local<T>` object filling the data loaded. The desired type of the matrix (e.g., int, float, double etc.) is to be explicitly specified when loading the matrix data from reading a file.

For example, considering "./bin" is a binary file having the data to be loaded,

```
auto m1 = make_rowmajor_matrix_local_loadbinary<int>("./bin");
auto m2 = make_rowmajor_matrix_local_loadbinary<float>("./bin");
```

"m1" will be a `rowmajor_matrix_local<int>`, whereas "m2" will be a `rowmajor_matrix_local<float>`.

Return Value
On success, it returns the created matrix of the type `rowmajor_matrix_local<T>`. Otherwise, it throws an exception.

std::ostream& operator<<(str, mat)

Parameters
str: A std::ostream& object representing the output stream buffer.
mat: A const& object of the type `rowmajor_matrix_local<T>` containing the matrix to be handled.

Purpose
This function writes the contents of the matrix in 2D row-major matrix form in the given output stream. Thus a rowmajor matrix can simply be printed on the user terminal as "std::cout << mat", where "mat" is the input matrix.

Return Value
On success, it returns a reference to the output stream.

std::istream& operator>>(str, mat)

Parameters
str: A std::istream& object representing the input stream buffer.
mat: A const& object of the type `rowmajor_matrix_local<T>` to be filled.

Purpose
This function reads the data from the input stream and writes the same in the given matrix. Each new-line character in the given stream is considered as a new row. The number of columns is automatically calculated based on the read elements count in each line of the input stream (it assumes that all the lines contain same number of elements).

Here the matrix "mat" is overwritten with the data read from the input stream. Thus any prior data in the matrix "mat" would be lost. Thus a rowmajor matrix can simply be read from standard input terminal as "std::cin >> mat", where "mat" is the matrix to be filled with data read from "std::cin".

Return Value
On success, it returns a reference to the input stream.

`rowmajor_matrix_local<T>` operator*(m1,m2)

Parameters
m1: A const& object of the type `rowmajor_matrix_local<T>`.
m2: Another const& object of the type `rowmajor_matrix_local<T>`.

Purpose
This function performs matrix multiplication between two input rowmajor_matrix_local objects of the same type.

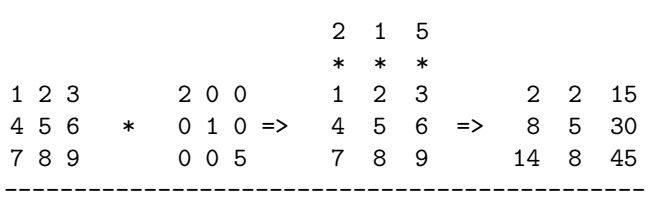
Return Value
If the input matrix conforms matrix multiplication rule (number of columns in m1 matches with the number of rows in m2), then it returns the resultant rowmajor matrix of the type `rowmajor_matrix_local<T>`. Otherwise, it throws an exception.

`rowmajor_matrix_local<T>` operator*(m1,m2)

Parameters
m1: A const& object of the type `rowmajor_matrix_local<T>`.
m2: A const& object of the type `diag_matrix_local<T>`.

Purpose
When multiplying a rowmajor matrix with a diagonal matrix (e.g., unit matrix etc.), actually every column of

the input rowmajor matrix is multiplied by every diagnonal element of the input diagonal matrix, as depicted below.



Thus frovedis provides an efficient overloaded operator*() to handle such situation. In case of diagnonal matrix, it only stores the diagonal elements (e.g., 2, 3, 5) in a data structure called `diag_matrix_local<T>` (see diag_matrix_local manual) and the overloaded operator*() simply multiplies each column of the input rowmajor matrix with each diagonal element.

Return Value
If number of columns in the input rowmajor matrix equals to the number of diagonal elements in the input diagonal matrix, it returns the resultant rowmajor matrix of the type `rowmajor_matrix_local<T>`. Otherwise, it throws an exception.

SEE ALSO

diag_matrix_local, colmajor_matrix_local, rowmajor_matrix