frovedis::colmajor_matrix_local<T>

NAME

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

SYNOPSIS

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

Constructors

colmajor_matrix_local ();
colmajor_matrix_local (size_t nrow, size_t ncol);
colmajor_matrix_local (const `colmajor_matrix_local<T>`& m);
colmajor_matrix_local (`colmajor_matrix_local<T>`&& m);
colmajor_matrix_local (const `rowmajor_matrix_local<T>`& m);

Overloaded Operators

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

Public Member Functions

`rowmajor_matrix_local<T>` to_rowmajor();
`rowmajor_matrix_local<T>` moveto_rowmajor();
`colmajor_matrix_local<T>` transpose () const;
`node_local<colmajor_matrix_local<T>>` broadcast();
void debug_print ();

Public Data Members

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

DESCRIPTION

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

Although it provides a 2D column-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 colmajor_matrix_local {
  std::vector<T> val;     // to contain matrix elements in 1D colmajor 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
};
```

A colmajor_matrix_local can be created from a rowmajor_matrix_local object and it can be converted back to the rowmajor_matrix_local object. Thus loading from file, saving into file etc. interfaces are not provided for colmajor_matrix_local structure. User may like to perform the conversion from/to rowmajor_matrix_local structure for the same.

Constructor Documentation

colmajor_matrix_local ()

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

colmajor_matrix_local (size_t nrow, size_t ncol)

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

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

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

colmajor_matrix_local (`colmajor_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.

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

It accepts a rowmajor_matrix_local object and constructs an equivalent colmajor_matrix_local object by simply changing the storage order of the elements in input matrix. Number of rows and number of columns will be same in both the input matrix and constructed colmajor matrix.

Overloaded Operator Documentation

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

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

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

Instead of copying, it moves the contents of the input rvalue colmajor 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 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 = {1,3,2,4}; //desired storage
colmajor_matrix_local<int> m;
m.val.swap(v);
m.local_num_row = 2;
m.local_num_col = 2;
m.debug_print();
```

The above program will output:

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

colmajor_matrix_local<T> transpose ()

It returns the transposed colmajor_matrix_local of the source matrix object.

For example,

```
std::vector<int> v = {1,3,2,4};
colmajor_matrix_local<int> m;
m.val.swap(v);
m.local_num_row = 2;
m.local_num_col = 2;
m.transpose().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> to_rowmajor();

It converts the colmajor storage of the target matrix to a rowmajor storage and returns the output rowmajor_matrix_local<T> after successful conversion. The target colmajor storage remains unchanged after the conversion.

rowmajor_matrix_local<T> moveto_rowmajor();

If the target matrix has only a single column, then rowmajor storage and column major storage both will be the same. Thus instead of any conversion overhead, elements in target matrix can simply be moved while creating the rowmajor_matrix_local object. It is faster and recommended, only when the target matrix is no longer be needed in a user program.

node_local<colmajor_matrix_local<T>> broadcast();

It broadcasts the source colmajor_matrix_local<T> to all the participating worker nodes. After successful broadcasting, it returns a node_local<colmajor_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 = {1,3,2,4};
colmajor_matrix_local<int> m;
m.val.swap(v);
m.local_num_row = 2;
m.local_num_col = 2;
auto bm1 = m.broadcast(); // faster
auto bm2 = frovedis::broadcast(m); // slower (serialization overhead)
```

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

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

Public Data Member Documentation

val

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

SEE ALSO

rowmajor_matrix_local, colmajor_matrix