Project #3 - LU Decomposition using MPI

SWE3021 Multicore Computing - Fall 2019

Due date: November 22 (Fri) 17:00pm

1 Parallel LU Decomposition

LU Decomposition is a classical method for transforming an N x N matrix A into the product of a lower-triangular matrix L and an upper-triangular matrix U,

A = LU.

You will use a process known as Gaussian elimination to create an LU decomposition of a square matrix. By performing elementary row operations, Gaussian elimination transforms a square matrix A into an equivalent upper-triangular matrix U. The lower-triangular matrix L consists of the row multipliers used in Gaussian elimination.

In this assignment, you will develop an MPI implementation of LU decomposition that use Gaussian elimination to factor a dense N x N matrix into an upper-triangular one and a lower-triangular one. In matrix computations, pivoting involves finding the largest magnitude value in a row, column, or both and then interchanging rows and/or columns in the matrix for the next step in the algorithm. The purpose of pivoting is to reduce round-off error, which enhances numerical stability. However, in this assignment, you will not use row pivoting and you may assume the input does not have 0 on its diagonal. Below is pseudocode for a sequential implementation of LU decomposition.

```
inputs: a(n,n)
outputs: π(n), l(n,n), and u(n,n)

initialize π as a vector of length n
initialize u as an n x n matrix with 0s below the diagonal
initialize l as an n x n matrix with 1s on the diagonal and 0s
           above the diagonal
for k = 1 to n
  u(k,k) = a(k,k)
  for i = k+1 to n
    l(i,k) = a(i,k)/u(k,k)
    u(k,i) = a(k,i)
  for i = k+1 to n
    for j = k+1 to n
      a(i,j) = a(i,j) - l(i,k)*u(k,j)
```

Your LU decomposition implementation should accept two arguments: n - the size of a matrix followed by s - the seed number for `srand(s)`. For example, you should run the following command if you want to run 64 MPI processes for 1024x1024 matrix and the seed number 1.

```
$ time mpiexec --mca btl openib,self,sm --mca btl_openib_cpc_include rdmacm
--machinefile hosts.txt -n 64 ./a.out 1024 1
```

Your programs will allocate an n x n matrix A of double precision (64-bit) floating point variables. You should initialize the matrix with uniform random numbers computed using a suitable random number generator rand(). That is, you should initialize A[][] as follows.

```
for(int i=0;i<n;i++)
    for(int j=0;j<n;j++)
        A[i][j] = (double) rand();
```

Note: You must NOT generate random numbers in parallel. If you do, your input matrix will be different.

Apply LU decomposition without partial pivoting to factor the matrix into an upper-triangular one and a lower-triangular one. To check your answer, you should compute A - LU and print the sum of Euclidean norm of the columns of (A - LU) as the following. I.e.,

$$\sum_{i=0}^{n-1} \|\mathbf{col_i}\| = \sum_{i=0}^{n-1} \sqrt{\sum_{j=0}^{n-1} col_i(j) \times col_i(j)}$$

This number should be very small and this is the only output your program should print out. **Note that you also need to parallelize this verification step for better performance.** The time command will print out the execution time of your program.

2 Grading

Similar to the previous projects, your 3rd project score will be given as follows.

1. If you do not submit, you will get 0 point.
2. If your code does not compile, you will get only 1 point.
3. If your code compiles and runs MPI processes but outputs are incorrect, you will get only 2 points.
4. The remaining 8 points are the performance score, which will be prorated by the following equation.

$$\frac{(the\ slowest\ code's\ execution\ time) - (your\ code's\ execution\ time)}{(the\ slowest\ code's\ execution\ time) - (the\ fastest\ code's\ execution\ time)}$$

3 How to Submit

To submit your project, you must rename your code to project2.cpp and run `multicore_submit` command in your project directory in skin.skku.edu server.

```
$ multicore_submit project3 project3.cpp
```

For any questions, please post them in Piazza so that we can share your questions and answers with other students. Please feel free to raise any issues and post any questions. Also, if you can answer other students' questions, please don't hesitate to post your answer. You would get some credits for posting questions and answers in Piazza.