Matrix Inversion Algorithm Streamlining With Intel TBB® and The Terasic DE2i-150-FPGA Board

Ruger Stellberger

Overview

- Cpp program to compute Cholesky and LU decomposition
- Random matrix generation with input rows and columns
- Sequential Implementation
- TBB Implementation
- Comparison of computation times
- Display output matrices and confirm with MATLAB script.

Cholesky Decomposition

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T = \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} L_{11} & L_{21} & L_{31} \\ 0 & L_{22} & L_{32} \\ 0 & 0 & L_{33} \end{pmatrix}$$

$$= \begin{pmatrix} L_{11}^2 & & \text{(symmetric)} \\ L_{21}L_{11} & L_{21}^2 + L_{22}^2 & \\ L_{31}L_{11} & L_{31}L_{21} + L_{32}L_{22} & L_{31}^2 + L_{32}^2 + L_{33}^2 \end{pmatrix},$$

- Cholesky decomposition is a technique used to break down a specific type of matrix into simpler components.
- It works specifically for matrices that are symmetric (or Hermitian) and positive-definite.
- The original matrix is split into the product of two easier matrices: one lower triangular matrix and its transpose.
- It finds applications in fields like statistics, optimization, and simulations.

LU Decomposition

$$A = LU,\ \text{where } A \text{ is a } 2\times 2 \text{ square matrix}$$

$$(1)\ A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{bmatrix} = LU,\ \text{where } A \text{ is a } 2\times 2 \text{ square matrix}$$

$$(2) A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} = LU,$$

$$\text{when } A \text{ is } 3\times 3 \text{ square matrix}$$

- Given a square matrix A, the LU decomposition seeks to express A as the product of two matrices: A = LU.
- L is a lower triangular matrix, and U is an upper triangular matrix.
- LU decomposition is commonly used in numerical algorithms for solving linear systems of equations, as it simplifies the process of finding the solution.

TBB Approach

```
void choleskyDecompositionTBB(double *matrix, int n) {
    tbb::parallel_for(tbb::blocked_range<int>(0, n), [=](const tbb::blocked_range<int>& r) {
        for (int k = r.begin(); k < r.end(); ++k) {
            matrix[k * n + k] = std::sqrt(matrix[k * n + k]);
            for (int i = k + 1; i < n; ++i) {
                matrix[i * n + k] /= matrix[k * n + k];
                for (int j = k + 1; j <= i; ++j) {
                    matrix[i * n + j] -= matrix[i * n + k] * matrix[j * n + k];
                }
            }
        }
    });
}
```

```
void luDecompositionTBB(double *matrix, int rows, int cols) {
    tbb::parallel_for(tbb::blocked_range<int>(0, std::min(rows, cols)), [=](const tbb::blocked_range<int>& r) {
        for (int k = r.begin(); k < r.end(); ++k) {
            for (int i = k + 1; i < rows; ++i) {
                matrix[i * cols + k] /= matrix[k * cols + k];
                for (int j = k + 1; j < cols; ++j) {
                    matrix[i * cols + j] -= matrix[i * cols + k] * matrix[k * cols + j];
                }
            }
        }
    });
}
```

- The program makes use of TBB’s parallel_for
- It seemed the most practical for a very calculation intense algorithm with many iterations and traversing
- Simple lambda function with respective decomposition algorithms.

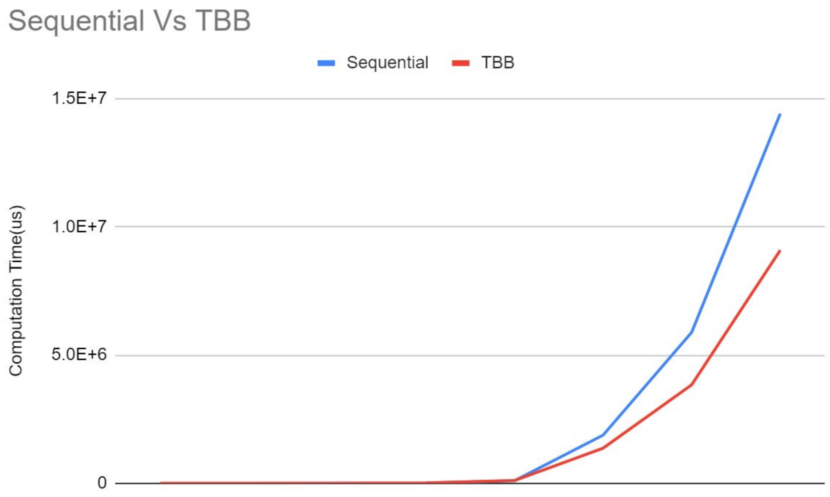
Results

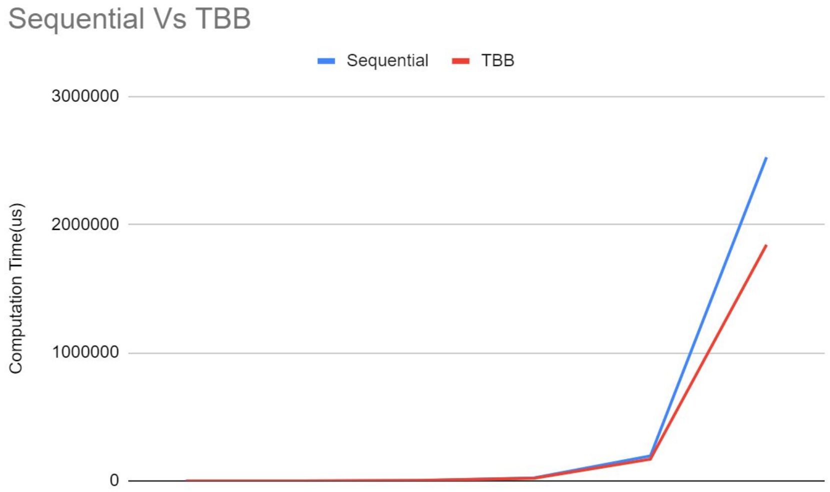
Computation times averages over 10 runs

			Cholesky decomposition(Positive difference is TBB improvment)					
Matrix Size -->	5x5	10x10	50x50	100x100	200x200	500x500	750x750	1000x1000
Sequential(us)	41.3	49	1675.6	12117.6	104451	1879731	5885838.6	14406541.6
TBB(us)	3686.5	3509.7	5035.1	15728	107175.2	1371099.8	3840201	9089741.6
Difference	-3645.2	-3460.7	-3359.5	-3610.4	-2724.2	508631.2	2045637.6	5316800

		LU decompostion(Positive difference is TBB improvment)				
Matrix Size	5	10	50	100	200	500
5	-26.6	-23	-55.3	-187.4	-154.3	18.7
10	-48	-53.6	-140.7	-51.3	-410.4	89
50	-44.7	-50	-406.8	1443.3	3650.6	4161.6
100	-77.7	-65.7	1296.7	3122.6	5986.2	24546.3
200	-24.7	77.7	2654	6838.8	25436.6	74089.6
500	-73.5	142.6	9732.4	16380.7	47813.5	682935.5

		Computation Times(us) measured over average of 10 runs				
	5x5	10x10	50x50	100x100	200x200	500x500
Sequential(us)	3.4	23.6	2755.2	23505.6	194946.8	2525344.5
TBB(us)	30	77.3	3162	20383	169510.2	1842409
Difference	-26.6	-53.7	-406.8	3122.6	25436.6	682935.5





Results (continued)

```
Enter the number of rows: 3
Enter the number of columns: 3

Original Cholesky Matrix:
      12        44         1
      10        72        80
      61        45        67

Original LU Matrix:
      12        44         1
      10        72        80
      61        45        67

Sequential Cholesky Decomposition Time: 5 microseconds

Parallel Cholesky Decomposition with TBB Time: 749 microseconds

Cholesky Decomposed Matrix:
   3.464        44         1
   2.887     7.979        80
   17.61   -0.7311      -nan

Sequential LU Decomposition Time: 1 microseconds

Parallel LU Decomposition with TBB Time: 1 microseconds

LU Decomposed Matrix with TBB:
      12        44         1
  0.8333     35.33     79.17
   5.083    -5.057     462.2
```

```
·772 ▸ Final Project
Command Window
New to MATLAB? See resources for Getting Started.

>> MatrixInversionAlgotithms
LU Decomposition - L:
    1.0000         0         0
    0.8333    1.0000         0
    5.0833   -5.0566    1.0000

LU Decomposition - U:
   12.0000   44.0000    1.0000
         0   35.3333   79.1667
         0         0  462.2311

Cholesky Decomposition:
   3.4641 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   2.8868 + 0.0000i   7.9791 + 0.0000i   0.0000 + 0.0000i
  17.6092 + 0.0000i  -0.7311 + 0.0000i   0.0000 +15.6083i

fx >>
```

Thank You

Demo Time!