CSE-411: Advanced Programming Techniques

Yifeng Xu

10/26/2023

Homework 3

1. Background on Gauss

Gauss is considered one of the most important mathematicians in the world and is known as the "Prince of Mathematics". Gauss discovered the prime number distribution theorem and the method of least squares, proved that a 17-sided shape can be constructed using only a ruler, summarized the applications of complex numbers, and rigorously proved that every algebraic equation of order n must have n real or complex solutions.

Gaussian elimination is an algorithm in linear algebraic programming that can be used to solve systems of linear equations. However, the algorithm is very complex and is not commonly used to add and subtract eliminations, to find the rank of a matrix, and to find the inverse matrix of an invertible square matrix.

2. theory of parallel techniques implemented

In this assignment, I used two techniques, a thread pool and a rayon, and I compared Forward Elimination and backward elimination using parallel techniques respectively.

In thread pool techniques, at first the forward elimination phase is done to transform the matrix into an upper triangular form. The outer loop iterates over each row, treating it as a pivot row. And then the pivot row is divided by the pivot value to ensure the diagonal entry is 1. At the thread pool the matrix a and vector b are wrapped in an Arc (Atomic Reference Counting) and a Mutex (Mutual Exclusion) to safely share and modify them across multiple threads. For each row below the pivot row, a thread is created to perform the elimination process. In this method use thread number to control the number of threads. For the backward elimination, it is similar to forward elimination.

In the rayon techniques, at first is still similar to Gaussian Elimination. The first loop works to transform the system into an upper triangular form. Identifies the row with the highest absolute value in the current column from the pivot row downwards and swaps the rows. I use multithreading to optimize the Gaussian elimination. The goal is to make entries below the pivot entry in the pivot column zero. U sing ThreadPoolBuilder to create a thread pool with a specified number of threads. Using pool.instal to executes a closure within the context of the thread pool. using into_par_iter as a parallel iterator.Within the map function, for each row below the pivot row, the changes required to make the entry in the pivot column zero are computed. For the backward elimination, is similar to forward elimination.

3. Implementation and test setup

Use bench to measure the time for each method, set the size of the different matrices a to 256, 512, 1024, 2048, 4096 to bring in the four different methods to calculate the size of the time used for comparison. Then fix the matrix size to 512 and set different number of threads to compare the effect of different threads on the time consumed by the current method.

4. Result

Using cargo bench to measure the time it takes. The result as follow:

size	256	512	1024	2048	4096
Threadpool	**42,202,366**	**247,287,545**	1,811,367,925	10,256,139,221	1,577,925,981,421
rayon	**18,564,554**	**92,587,337**	105,169,879	3,844,208,587	620,138,114,267
Threadpool_back	40,227,591	211,365,487	1,930,446,287	10,023,579,441	1,379,261,475,281
Rayon_back	17,290,368	80,553,413	100,159,692	2,979,854,875	601,685,245,663

Thread num	**4**	**8**	**16**
threadpool	**247,287,545**	**296,418,837**	**341,462,212**
rayon	**92,587,337**	145,090,941	**243,765,258**

5. conclusions

As can be seen from the above tables, as the size of matrix a increases, the time consumed by all four methods increases, and the time consumed by increasing the number of threads instead increases, and it may be that when dealing with smaller sized matrices, an excessive number of threads may instead increase the consumption of the system.