LITERATURE REVIEW: A Parallel Jacobi Gauss-Seidel Method with Dynamic Parallelization

Nirav C. Pansuriya
School of Computer Science
Carleton University
Ottawa, Canada K1S 5B6
niravchhaganbhaipan@cmail.carleton.ca

15 October 2021

1 Introduction

Computer architecture has become increasingly parallel in recent years. Modern GPUs, such as the NVIDIA GForce GTX 280 provide enormous parallelism. So, the focus of researchers has moved to parallelism rather than continuous improvements to the single unit speed of computation. We regularly encounter several linear systems in physics, mathematics, and engineering. In many scientific simulations, we have to solve large-scale linear equations. However, large-scale linear equations require a significant amount of time and resources, such as memory. It is always a trending topic among researchers to find an algorithm that can solve large-scale linear equations in less time and with fewer resources. The Jacobi and Gauss-Seidel methods are two famous and well-known iterative methods for solving systems of linear equations. Although the Jacobi approach is extremely simple to implement in a parallel environment, it requires an excessive number of iterations to solve a large system of linear equations. The Gauss-Seidel method is an improved version of the Jacobi method. The GS method is capable of solving a large system of linear equations in a small number of iterations. However, because this method is sequential in nature, it is extremely difficult to implement in a parallel environment. One method is easily implemented in a parallel environment but requires an excessive number of iterations to solve, whereas the other method can solve a large system of linear equations in a very few iterations but cannot be implemented in a parallel environment. To address this issue, the author of the base paper used for this research work introduced the PJG (Parallel Jacobian Gauss-Seidel Method). This method is capable of solving large systems of linear equations in a small number of iterations and is extremely simple to apply in a parallel environment too. The primary objective of this research is to further enhance the performance of the PJG method by implementing the concept of dynamic parallelization.

2 Literature Review

Large linear equation systems are used in many complex scientific simulators. Solving such huge systems requires a significant amount of processing resources, such as memory and CPU. Solving a large system of linear equations is a very time consuming procedure. That

is why developing effective algorithms to solve such big systems in a limited amount of time and with limited resources is always a research interest.

There are two approaches to solving linear equations: a direct approach and an iterative approach. In a direct approach, equations are solved in finite steps, but the time complexity is $O(n^3)$ [5], which is quite high when the system of linear equations is large. This is because direct approaches scale intensively with respect to the size of system of linear equations. The indirect technique has a time complexity of $O(n^2)$. The variables in the linear equation are initially initialized with random values in an indirect way. Following that, we find a close approximation with each iteration. Additionally, indirect approaches consume less memory and resources than direct approaches. That is why the majority of research is conducted using iterative approaches, as they need fewer computational resources than direct methods.

The Jacobi method and the Gauss-Seidel method are two well-known iterative approaches. [3]. Both methods are used to solve linear equations represented in matrix form. Any system of linear equations can be $AX = B$, where A is the coefficent matrix, B is a vector of competitors, and X is a vector of variables. In the Jacobi method, variables (vector X) initialised with random values. In the Jacobi method, variables (vector X) initialised with random values. After that, to solve each diagonal element in matrix A, values of variables (vector X) are plugged into the linear equation. By doing this, the new values of variables are calculated. All of these steps are repeated until convergence occurs. The Gauss-Seidel method is similar to the Jacobi method. The only difference is that variables in the Jacobi approach are updated after each iteration, but variables in the GS method are updated immediately after each diagonal element is solved. That is why, in comparison to the Jacobi method, the GS method has a higher convergence speed. The only disadvantage of the GS method is that it is very hard to implement in parallel systems compared to the Jacobi method, because the GS method is sequential in nature.

There are many parallel algorithms for the GS method. The Red-Black GS algorithm is very popular [4]. This algorithm is based on the ordering of multiple colours. The specific matrix structure require to work this this method, and so it is dependent on the sparsity pattern of a matrix. As a result, we cannot use it with all systems of linear equations.

In the paper [2], the author has come up with a row-based parallel method. This method is derived from the GS method. To solve every diagonal element, we do the multiplication of constants and current variables. For each linear equation, this method performs all of these multiplication operations simultaneously in a parallel environment. After all these multiplication operations are completed for a full row of matrix A, variables get updated. This procedure is repeated until convergence occurs.

In paper [1], the author has come up with a new parallel method to solve linear equations by combining two other algorithms. This new method can achieve a good parallelization and can solve large systems of linear equations in very few iterations. The author has merged the Jacobi method's parallelism with the GS method's rapid convergence. The main advantage of this method is that there is no need to have any special pattern in matrix. And so this method can work with a sparse matrix as well as a dense matrix. In this new approach, linear equations are divided into blocks. Every equation within the same block is solved in a parallel environment at the same time, as the Jacobi method is very easy to implement

in a parallel environment. After that, variables get updated as one would do in the GS method, and equations within the next block will use the updated values of variables. This new approach is called the PJG method. This method is being improved in this research work.

The primary objective of this research is to further enhance the performance of the PJG method by implementing the concept of dynamic parallelization. Another aim is to compare the proposed method (PJG method with dynamic parallelization) with the Jacobi method, the GS method, and the PJG method. I aim to implement all of these algorithms in CUDA and executeÂ them on a GPU in order to compare their performance with the proposed technique.

References

[1] Afshin Ahmadi, Felice Manganiello, Amin Khademi, and Melissa C. Smith. A parallel jacobi-embedded gauss-seidel method. *IEEE Transactions on Parallel and Distributed Systems*, 32(6):1452–1464, 2021.

[2] Hadrien Courtecuisse and Jérémie Allard. Parallel dense gauss-seidel algorithm on many-core processors. In *2009 11th IEEE International Conference on High Performance Computing and Communications*, pages 139–147, 2009.

[3] L.A. Hageman. *Applied Iterative Methods*. Elsevier Science, 2016.

[4] Kawai, Masatoshi, Iwashita, Takeshi, Nakashima, Hiroshi, and Osni Marques. Parallel smoother based on block red-black ordering for multigrid poisson solver. *Lecture Notes in Computer Science High Performance Computing for Computational Science - VECPAR 2012*, page 292–299, 2013.

[5] A. Quarteroni, R. Sacco, , and F. Saleri. *Numerical Mathematics*, volume 37. Springer, 2010.