Proceedings of the 5th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/14/20230753

Research and analysis of matrix multiplication in distributed learning algorithms

Xuze Zhou

Hangzhou Dianzi University, Hangzhou, Zhejiang, CHN

20322332@hdu.edu.cn

Abstract. Matrix multiplication have become increasingly important nowadays, which is applied in many kinds of fields. In this case, the need to improve the speed and efficiency of matrix multiplication is increasing. In this paper, the author analyzes some relevant theories about matrix multiplication as well as the advantage and disadvantage of some applications that based on matrix multiplication. It turns out that matrix multiplication has a lot of room for development in the future cause the current method still has many defects and is not perfect. For example, it will still take plenty of time to finish the process of matrix multiplication when the matrices are large in size. However, it's gratifying that some improvements have been achieved, which can help to optimize the efficiency. In addition, some advanced methods start to appear. For instance, the method of combining matrix multiplication with AI provides a new direction for future research and development. Consequently, it is predictable that a significant achievement to optimize matrix multiplication will be made in the future.

Keywords: matrix multiplication, theory, accelerator.

1. Introduction

Matrix multiplication has become increasingly popular nowadays since it is a fundamental and useful operation to analyze data, which can be applied in many types of fields. Firstly, it can play an important role in machine learning. Many workloads are examined in large-scale distributed cloud computing settings as input dataset sizes grow. Consequently, in order to conduct machine learning tasks in the cloud, it is important to understand the features of a distributed matrix multiplication work [1]. Matrix multiplication is also a primitive method that can be found in a variety of systems, including neural networks and scientific computing procedures. Machine learning-based algorithm discovery offers the potential to go above human intuition and outperform the greatest human-designed algorithms currently available [2].

In addition, it is also helpful to apply matrix multiplication in other fields like scientific computing [3]. Therefore, many researchers have become interested in developing matrix computation tasks on distributed systems.

Moreover, matrix multiplication is helpful in both computer science and mathematics. In addition to the fact that many computational issues in linear algebra can be solved by computing the product of two matrices, matrix multiplication's complexity also arises as an obstruction in a number of other computational activities [4]. In addition, on high-performance computers, matrix multiplication can serve as portable, effective building blocks for linear algebra algorithms. For instance, BLAS3 operation algorithms that are asymptotically quicker than the standard ones. These methods are based on the fast matrix multiplication approach developed by Strassen, which is widely acknowledged to be a useful method for real-world applications once matrix dimensions rise above around 100 [5]. Finally, matrix multiplication is also important in deep learning. For example, TensorFlow, Caffe, Torch, and other open-source deep learning frameworks are widely used around the world and their acceleration is critical. Convolution takes a significant amount of computing time on various systems. However, this can be greatly improved by changing the matrix multiplication [6].

Moreover, Most forms of DNNs (deep neural networks) use generalized matrix multiplication (GEMM) as the main kernel for both training and inference [7]. However, there exist some bottlenecks that can impede people when they try to study this field. Bottleneck caused by the unpredictable latency while waiting for the slowest nodes, which is called stragglers, is one of the main difficulties that people need to face with [3]. The fact that stragglers act as bottlenecks when the calculation is split over several workers is one of the main challenges encountered. Computation replication between workers is a crude method of straggler mitigation. This is obviously a waste of resources, and adding redundancy to computational systems requires a methodical approach [8]. In this case, though there are already some applications of matrix multiplication that have been developed, people are still trying to optimize some problems. In general, it is obvious that matrix multiplication is an important method that can be used in many advanced fields. In this paper, the author considers some fundamental theorem and provide some applications that based on matrix multiplication as well as example of optimization based on it.

2. Relevant theories and current situation analysis of matrix multiplication

2.1. Theory of matrix multiplication

2.1.1. Theory 1: The theory of algorithmic fault-tolerant matrix multiplication. The ability to convert most calculations into large matrices is a prerequisite for the high performance of many linear algebraic procedures. The cost of moving b × b blocks of operands at the level of the memory hierarchy is proportional to b^2, which must be amortized in O(b^3) [9] calculations. This leads to high performance for multiplication of the matrices themselves. This fact relates to algorithmic problems for linear algebraic functions which spend most of the time on competing matrices. Therefore, it is important to add tolerance to matrix-matrix multiplication while maintaining high performance, so the existence of error-tolerant matrix-multiplication is an important first step to improve fault-tolerant linear algebra libraries. Moreover, crime at the algorithmic level is expected to be important in future computer systems. However, it is known that in real-world applications, the normal implementation of the break-even method (ABFT) cannot protect all elements of floating-point numbers [10].

Obviously, this approach needs to be expanded to remove bit-flip from all assumptions without adding overhead. So, the main goal of this engine is to detect the biggest errors with the least amount of overhead. Best algorithm-based fault tolerance (ABFT) requires adding a small amount of repeated data, a checksum, to the input to ensure that the appropriate number of errors has been properly closed. detected and corrected during or after calculation [10] . This approach (algorithm-based fault tolerance) has many advantages over others, despite its age. The first is a complete theory that can be used to identify faults precisely. Second, it specifically mentions how to distinguish between misguidance and injustice [9]. Finally, it confirms that by adding a tolerance to the multiplication of high-performance matrices, theoretical results can be achieved without sacrificing high performance. Therefore, ABFT overcomes the weakness of the algorithm by adding rows or columns to the matrix which carry statistical information. To use ABFT, transactions must have a checksum protector. Most ABFT studies focus on failure. In this variation, also known as global ABFT, the checksums are stored in a continuous line and sequence for the encoded checksum values [10].

2.1.2. Theory 2: Theory of group-theoretic algorithms for matrix multiplication. One of the most important problems in algorithmic linear algebra is matrix multiplication. This is an important task, and the number of problems that can occur only serves to show how important the task is. Through some modifications, a better bound on the exponential matrix multiplication has been established, which for all ε > 0 is the smallest number that can be multiplied by an n × n matrix in the function O(n^(ω+ε)). It is recognized that ω is equal to 2, but the most correct answer is ω < 2.38 [11-12]. In fact, ω is an important number for understanding algorithmic linear algebra because solving all linear algebra problems, such as finding determinants, solving systems of equations, and inverting matrices, there are equations such as matrix multiplication. It also shows that reducing n × n matrix equations to groups of algebraic equations is supported by many families of non-abelian groups. Specifically, it includes a family of size groups n^(2+O(1)). Although not sufficient, the existence of these families is a prerequisite for the group theory process to show that ω = 2 [11]. Therefore, it is necessary to make a new approach to frontiers, leaving the problem to group theory and representation theory. Several researchers have tried to solve this problem, providing a simple group theory matrix multiplication method with O(n^2.9088) direct processing time [11]. In this case, the newly developed group theoretical algorithms will be easier to present and understand, because they fit the mathematical process clearly and avoid some of the shortcomings of the old algorithms.

2.1.3. Theory 3: Algebraic complexity theory of matrix multiplication. The analysis of computation using algebraic models is known as algebraic complexity theory. The development of techniques to demonstrate lower bounds on the computational complexity of actual problems in algebraic models of computing has been one of this field's major accomplishments [13]. In this case, it is important to know how natural topics in geometry and representation theory are addressed as open questions in algebraic complexity theory, which can help us to have a better understanding of matrix multiplication. One of the typical examples is Strassen's algorithm. In this algorithm, each matrix is partitioned into 4 copies, and then 10 intermediate matrices are created as follows.

$$\begin{aligned} S_1 &= B_{12} - B_{22} \\ S_2 &= A_{11} + A_{12} \\ S_3 &= A_{21} + A_{22} \\ S_4 &= B_{21} - B_{11} \\ S_5 &= A_{11} + A_{22} \\ S_6 &= B_{11} + B_{22} \\ S_7 &= A_{12} - A_{22} \\ S_8 &= B_{21} + B_{22} \\ S_9 &= A_{11} - A_{21} \\ S_{10} &= B_{11} + B_{12} \end{aligned} \tag{1}$$

Then calculate matrix multiplication for 7 times.

$$\begin{aligned} P_1 &= A_{11} * S_1 \\ P_2 &= S_2 * B_{22} \\ P_3 &= S_3 * B_{11} \\ P_4 &= A_{22} * S_4 \end{aligned} \tag{2}$$

$$P_5 = S_5 * S_6$$

$$P_6 = S_7 * S_8$$

$$P_7 = S_9 * S_{10}$$

Finally, matrix C can be calculated according to these results

$$\begin{aligned} C_{11} &= P_4 + P_5 + P_6 - P_2 \\ C_{12} &= P_1 + P_2 \\ C_{21} &= P_3 + P_4 \\ C_{22} &= P_1 + P_5 - P_3 - P_7 \end{aligned} \tag{3}$$

There is a simple example: suppose that there are two 2 × 2matrices A and B

$$A = \begin{matrix} a_1^1 & a_2^1 \\ a_1^2 & a_2^2 \end{matrix}, \ B = \begin{matrix} b_1^1 & b_2^1 \\ b_1^2 & b_2^2 \end{matrix} \tag{4}$$

According to Strassen's algorithm, it is obvious that the output matrix of C = A*B is

$$C = \begin{matrix} a_1^1 * b_1^1 + a_2^1 * b_1^2 & a_1^1 * b_2^1 + a_2^1 * b_2^2 \\ a_1^2 * b_1^1 + a_2^2 * b_1^2 & a_1^2 * b_2^1 + a_2^2 * b_2^2 \end{matrix} \tag{5}$$

However, the disadvantage of this algorithm is that it uses too much arithmetic operations (+, -), or many multiplications. The asymptotic results are mostly determined by the number of multiplications required, which controls the overall number of arithmetic operations [14].

In order to make an improvement, some researchers use tensors to investigate the complexity of matrix multiplication in a more geometrical manner [13-14], which is important in some advanced techniques like approximate algorithms [14].

2.2. *Current situation analysis*

Matrix multiplication is one of the most basic and universal operations in all of mathematics. Researchers have tried to find an effective way to perform matrix multiplication for many years. Many methods are created successfully like Strassen's algorithm.

Before Strassen's algorithm was created, it was extremely complex to perform matrix multiplication when the matrix is large (like thousands of rows and columns). However, after the invention of Strassen's algorithm, it is possible to multiply a pair of 2 × 2 matrices in 7 steps instead of 8 multiplication steps, at the cost of introducing more addition steps. This algorithm can improve the efficiency of calculation largely when the matrix is large enough cause the elements of a matrix themselves can be matrices. For instance, a 1000 × 1000 matrix can be divided into four 500 × 500 matrices, and so on. Strassen's discovery has encouraged many researchers to study deeper and to find an efficient algorithm for matrix multiplication. In this case, two different research directions have produced.

The first one is how does the number of multiplication steps grow with n in the fastest possible algorithm for multiplying two n × n matrices together so that n tends to infinity? Currently, the best reduction record is $n^{2.3728596}$.

The second one is the least possible number of multiplication steps. It has been show proved that the multiplication steps of a 2 × 2 matrix cannot be less than 7 steps. However, the minimal number of necessary multiplications is still unknown for all other matrix sizes. Quick techniques for tiny matrices may have a significant impact because, when multiplied by matrices of a reasonable size, they may outperform Stratham's approach. Unfortunately, even for 3 × 3 matrices, the number of possible algorithms exceeds the limit of what can be computed.

However, researchers have made progress in this field. Recently, in an article published in Nature, a team from the artificial intelligence company DeepMind demonstrated how to approach this problem from a novel angle. A unique mathematical object called tensor, which is a three-dimensional array of numbers, can be used to illustrate the abstract problem of multiplying two matrices. After that, the researcher can break down this tensor into a sum of its basic parts, where each one stands for a distinct stage of the associated matrix multiplication process. In this way, the researchers discovered new algorithms, which can help to improve the efficiency.

The strategy that DeepMind use is based on deep learning algorithm. Neural networks are the foundation of all deep learning techniques: networks of artificial neurons are divided into layers, and the strength of the connections can vary, representing the degree of influence each layer has on the neurons in the next layer. The strength of these connections is adjusted over multiple iterations of the training process, in which the neural network learns to convert each input it receives into an output that helps the algorithm achieve its overall goal.

Consequently, a new algorithm named AlphaTensor have been created by DeepMind. Like all neural networks, AlphaTensor requires a large amount of data for training. After training, AlphaTensor rediscover Stratham's algorithm. Then, for each matrix size, it has discovered up to hundreds of new algorithms, although the number of multiplication steps is the same. In a few instances, AlphaTensor even set new records. The algorithm's most unexpected finding is that it discovers a new method for multiplying 4 × 4 matrices in 47 steps rather than the 49 steps needed for Stratham's algorithm. Therefore, it shows that the new algorithm (AlphaTensor) has a lot of room for improvement in the future.

3. Applications

3.1. Example 1: Photonic matrix multiplication – photonic accelerator

The increasing global demand for artificial intelligence has led to a very high demand for computing power and memory. In addition, it is increasingly difficult to rely on semiconductor technology to improve computing performance and energy efficiency. However, it is recognized that optical devices can have very large bandwidths and low power consumption because the quantum state of light has many frequencies and many degrees of freedom, making it a broad area of research. In recent years, photonic matrix multiplication has developed rapidly and is widely used in photonic acceleration fields such as optical signal processing, artificial intelligence and photonic neural networks. The use of matrix multiplication based refers to the great potential of the photonic accelerator and various applications. In this regard, some researchers are increasingly focusing on photonic matrix research, and it turns out that this method has a bright future.

Most of the computational overhead in generating flow signals and intelligence algorithms is aided by matrix computation, an important form of information creation in science and technology. To meet the increasing demand for cost-effectiveness and capacity, photonic accelerators are designed to increase the objective to evolve, and artificial intelligence can benefit from the superior capabilities of photonic matrix multiplication. Recent advances in photonic matrix research will make it possible to develop applications that are not currently possible for conventional electronic computers [15].

The great potential for optical data processing and the speed of intelligence is demonstrated by photonic matrix multiplication. Power consumption and signal delay can be reduced. Future photonic matrix nuclei will be more complete and more capable. In conclusion, photonic matrix multiplication has been used in many fields, such as AI accelerators and optical signal processing in optical communications.

3.2. Example 2: Sparse-sparse matrix multiplication accelerator

A sparse matrix is a matrix that has a large percentage of zeros. Various formats have been developed to efficiently remove matrices from memory by eliminating zero points. Compressed Sparse Row (CSR) or Com-pressed Sparse Column (CSC) is one of the most commonly used six-mats [16]. Sparse matrix-matrix multiplication (SpGEMM) is an important computational basis in several important applications, such as graph analytics, machine learning, and scientific computing. More specifically, SpGEMM is part of many graph algorithms. SpGEMM calculations have additional hardware components that are provided to support different functions. Hardware accelerators are used specifically to solve this problem with energy efficiency. However, there are some challenges that scientists must overcome when building a new accelerator. In addition, since the number of non-zero elements in the output of SpGEMM is not known in advance, synchronization is necessary to show the corresponding data for different rows and columns computed simultaneously [17].

Recently, several researchers have proposed a new SpGEMM-based accelerator called MatRaptor, which has the best power and performance compared to the newest accelerator OuterSPACE [17]. This was achieved by the development of a new small data file called a row-sparse cyclic channel (C^2SR), which can provide multiple channels for similar access to large memory.

Several other researchers have developed another accelerator called InnerSP, which is also based on SpGEMM. The construction of InnerSP is based on extensive analysis of 755 sparse matrices, which shows how big the problem is in memory. After that, scientists came up with a fix to reverse this problem. Consequently, the product-InnerSP-does not require more memory than the two input and one output matrices [16]. According to this study, it is better to achieve unit efficiency when using the space in the equation matrix equation and not using the chips involved in the previous message. It turns out that the memory-efficient acceleration of domestic products can be a good substitute for external production.

4. Conclusion

In general, matrix multiplication is an advanced method that can be applied in many fields with high efficiency, despite that there are still some problems to be solved. It seems that matrix multiplication can become much more important and powerful in the future, with the combination of AI. So, it's obvious that matrix multiplication has a wide range of development prospects. In this paper, the author firstly discusses the background and significance of the study of matrix multiplication, like its contribution to machine learning. Then, the author reveals some theories about matrix multiplication, which contains fault-tolerant matrix-matrix multiplication, group-theoretic matrix multiplication and algebraic complexity theory of matrix multiplication. In addition, current status of matrix multiplication is also discussed. Finally, some applications that based on matrix multiplication are listed and discussed, like photonic accelerator and Sparse-Sparse Matrix Multiplication Accelerator, which might play an important role in the future. However, there are still some researches that people need to do. For instance, how to accelerate complex matrix multiplication faster and more efficiently still needs to be studied in depth by researchers, although some progresses have been achieved. Moreover, boosting the speed of matrix multiplication can be guaranteed by combining matrix multiplication with AI, which can be used to solve a broader range of mathematical and computational tasks consequently.

References

[1] Son, Myungjun, and Kyungyong Lee. "Distributed matrix multiplication performance estimator for machine learning jobs in cloud computing." 2018 IEEE 11th International Conference on Cloud Computing (CLOUD). IEEE(2018).

[2] Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., ... & Kohli, P. Discovering faster matrix multiplication algorithms with reinforcement learning. Nature, 610(7930), 47-53(2022).

[3] Yu, Qian, Mohammad Ali Maddah-Ali, and A. Salman Avestimehr. "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding." IEEE Transactions on Information Theory 66.3, 1920-1933(2020).

[4] Le Gall, François. "Faster algorithms for rectangular matrix multiplication." 2012 IEEE 53rd annual symposium on foundations of computer science. IEEE(2012).

[5] Higham, Nicholas J. "Exploiting fast matrix multiplication within the level 3 BLAS." ACM Transactions on Mathematical Software (TOMS) 16.4, 352-368(1990).

[6] Osawa, Kazuki, et al. "Accelerating matrix multiplication in deep learning by using low-rank approximation." 2017 International Conference on High Performance Computing &

Simulation (HPCS). IEEE(2017).

[7] San Juan, Pablo, et al. "Low precision matrix multiplication for efficient deep learning in NVIDIA Carmel processors." The Journal of Supercomputing 77, 11257-11269(2021).

[8] Muralee Krishnan, Nikhil Krishnan, Seyederfan Hosseini, and Ashish Khisti. "Coded sequential matrix multiplication for straggler mitigation." Advances in Neural Information Processing Systems 33, 16060-16069(2020).

[9] Gunnels, J. A., Katz, D. S., Quintana-Orti, E. S., & Van de Gejin, R. A. (2001, July). Fault-tolerant high-performance matrix multiplication: Theory and practice. International Conference on Dependable Systems and Networks (pp. 47-56). IEEE(2001).

[10] Moldaschl, Michael, Karl E. Prikopa, and Wilfried N. Gansterer. "Fault tolerant communication-optimal 2.5 D matrix multiplication." Journal of Parallel and Distributed Computing 104, 179-190(2017).

[11] Cohn, Henry, et al. "Group-theoretic algorithms for matrix multiplication." 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05). IEEE(2005).

[12] Cohn, Henry, and Christopher Umans. "A group-theoretic approach to fast matrix multiplication." 44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings. IEEE(2003).

[13] Le Gall, François. "Algebraic complexity theory and matrix multiplication." ISSAC(2014).

[14] Landsberg, J. "Geometry and the complexity of matrix multiplication." Bulletin of the American Mathematical Society 45.2, 247-284(2008).

[15] Hailong Zhou, et al. "Photonic matrix multiplication lights up photonic accelerator and beyond." Light: Science & Applications 11.1 (2022): 30.

[16] Baek, Daehyeon, et al. "InnerSP: A memory efficient sparse matrix multiplication accelerator with locality-aware inner product processing." 2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT). IEEE(2021).

[17] Srivastava, Nitish, et al. "Matraptor: A sparse-sparse matrix multiplication accelerator based on row-wise product." 2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). IEEE(2020).