EasyChair Preprint

№ 13964

An Asynchronous Matrix Multiplication Accelerator

Lingzhuang Zhang, Rongqing Hu, Hongrui Zhang and Anping He

July 13, 2024

An Asynchronous Matrix Multiplication Accelerator

Lingzhuang Zhang, Rongqing Hu, Hongrui Zhang, and Anping He*

Lanzhou University, Lanzhou, 730000, China
{zhanglzh22,hurq2023,hrzhang21,heap}@lzu.edu.cn

Abstract. Matrix multiplication plays an important role in various territories. The input data density leads to low computation efficiency, and synchronous circuits fail to meet the low-power requirement of specific fields. Therefore, an asynchronous matrix multiplication accelerator is proposed, which selects the appropriate calculation method by sensing the data density. For SpGEMM, a two-way condensation technique is adopted to solve the problem of spoiling the right matrix input reuse. An "one-to-one" merge strategy to reduce the uncertainty of the merge process is further proposed. Finally, the area of the accelerator is 17.3 mm^2, and the power demand is only 0.00468W in the UMC 110nm process. This research evaluates the accelerator on the SuiteSparse set and random matrix set, achieving 3.4× and 3.1× speed-up and 86× and 304× energy saving over MKL and cuSPARSE, respectively. It also achieves 62.3× and 11.25× energy saving over OuterSPACE and SpArch, respectively.

Keywords: Matrix Multiplication · Asynchronous circuit · Low-power Domain Architecture

1 Introduction

Matrix multiplication is indispensable in almost all large scientific computing domains, which is widely used in various algorithms (Deep Neural Networks [1], etc.). Depending on the data density, matrix multiplication can be divided into dense matrix-matrix multiplication and sparse matrix-matrix multiplication(SpGEMM). Running on general-purpose computing platforms (e.g., CPUs and GPUs) will lower the performance of SpGEMM, due to the irregular memory access of low-density matrices. Therefore, there is an inevitable requirement to study specific architectures for different data densities. Most recent hardware architectures have focused on the inner product (e.g., [2, 3]) or outer product (e.g., [4, 5]). The inner product has good output reuse and is more suitable for denser matrices. The outer product is more suitable for highly sparse matrices, and provides input reuse to get a low density partial matrix in the multiplication phase. So, matrix multiplication requires a different strategy for different data density to improve the processing advantage.

Furthermore, all the above hardware architectures use the synchronous circuit design method. As the size of the circuits increases, the timing of the circuits becomes more strict. Clock jitter, clock skew and other issues pose substantial design

challenges. The final implemented accelerator is difficult to break through performance bottleneck. Secondly, in certain battery-powered scenarios, synchronous accelerators cannot meet the low-power requirement. We exploit the clockless asynchronous circuit method and design an asynchronous architecture. Eventually, the internal circuitry of the accelerator should start working only when the conditions are met.

In the present work, an asynchronous matrix multiplication accelerator for low-power domains is proposed. The main contributions are as follows: The architecture chooses the inner/outer product method to complete matrix multiplication according to the input matrix data density; A new sparse matrix condensation technique is used: this avoids breaking data reuse and reduces the difficulty of data-matching; An "one-to-one" merging strategy is adopted to reduce the uncertainty associated with the merging process; An asynchronous design approach is used to implement the matrix multiplication accelerator that avoids the problems caused by clocks.

2 Architecture

The system architecture consists of five main parts: CPU, NOC(Networks On-Chip), SPI, Memory, and Matrix Calculation(MC), as shown in Fig. 1. The CPU first sends the matrix loading instruction to the SPI, which informs the external input of dense or sparse matrices. The sparse matrix is condensed and stored in COO format in the Memory, then SPI sends an Ack and the CPU sends the matrix multiplication instruction to the MC through NOC.

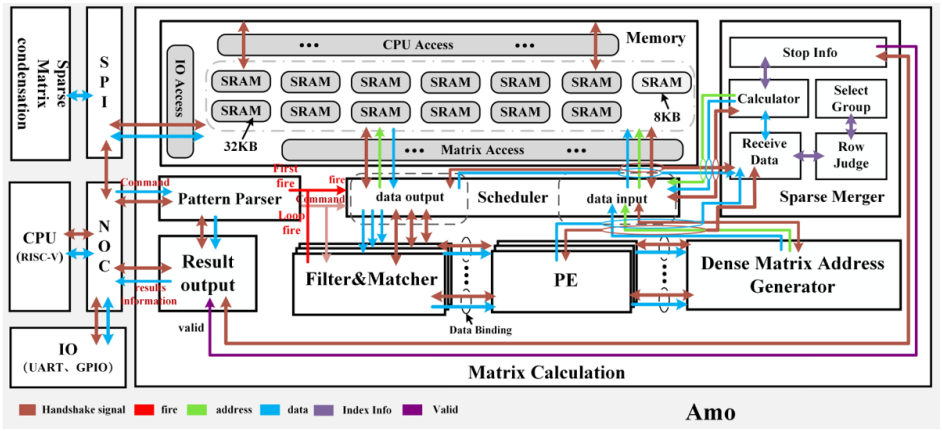

Fig. 1. System architecture of asynchronus matrix accelerator

The Memory uses the fire of Click to trigger the corresponding SRAM. An IO Access, a Matrix Access and 13 SRAMs together form the Memory module. The

data stored in SRAMs are condensed, following the rule: two-way condensation of sparse matrices. Take the left matrix as an example, if an element has a 0 upper neighbor, it moves up to the 0 position and repeats the above operation. The advantages of using this method are ensuring data reuse for input matrices and reducing the amount of data for the partial matrices.

In the MC, the Pattern Parser decides the way to accomplish matrix multiplication. The Scheduler continuously generates the correct data address according to the result of the Pattern Parser, and obtains the initial or partial data from the Memory and outputs them to the Filter/Matcher/Merger modules. The Filter avoids the participation of 0s which remain in the condensed matrix. Due to the difference between the inner and outer products, a general Matcher which can support different matching processes is designed. The matched data must be input to the PE to complete the calculation. In dense mode, PE outputs the result to the Address Generator. In sparse matrix mode, PE outputs the result to the Merger which adopts an "one-to-one" sparse merging strategy in Fig. 1. The storage address is calculated based on the element index. The same index means the same storage location, and the elements with the same location need to complete the merge operation. Finally, the Result Output module analyses the final calculation result flag and outputs the resulting information.

3 System Analysis

A C++ simulator is employed to obtain a set of synthetic matrix datasets and regular and irregularly structured sparse matrices in the SuiteSparse dataset are selected. The performance of accelerator on the two datasets is evaluated. Multiple computing platforms including OuterSPACE, SpArch, CPU and GPU are used. An Intel® MKL is used on an eight-core AMD Ryzen 7 5800H CPU. For the GPU, cuSPARSE is used on an NVIDIA GeForce RTX3050 Ti Laptop GPU. Fig.2 shows the average performance comparison after the scale. Our accelerator slightly under performs for the dense matrix computation compared to MKL and cuSPARSE. When calculating highly sparse matrices, it can maintain stable performance, achieving 2.3×, 3.4× and 3.1× speed-up and the energy efficiency increases by 62.3×, 86×, and 304× (on average) over OuterSPACE, MKL and cuSPARSE.

4 Conclusion

In this paper, an asynchronous matrix multiplication accelerator is proposed, which can select the appropriate calculation method according to different data densities. The sparse matrices are condensed to reduce the amount of partial matrices' data. This accelerator takes a "one-to-one" merging strategy, which reduces the uncertainty associated with the merging process.

Finally, the accelerator is implemented using asynchronous circuits, which consumes only 0.00468 W at UMC 110nm technology. We evaluate it on matrix datasets which have different structures and densities. The performance is

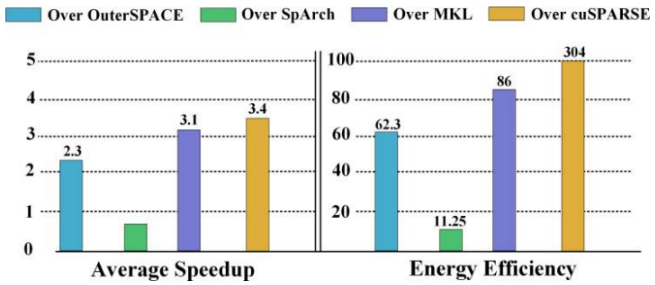

Fig. 2. The average performance comparison

improved by 3.4×, 3.1×(on average) and the energy efficiency is increased by 86×, 304×(on average) compared to MKL and cuSPARSE. Our Accelerator also achieves 62.3× and 11.25× energy improvement comparing to OuterSPACE and SpArch. The results show that the asynchronous architecture is promising to provide a low-power solution for future matrix applications.

5 Acknowledgments

This work is supported by the Gansu Provincial IC Manufacturing Materials Innovation Consortium Project 2022 (Grant No.22ZD6GE016).

References

1. Qin, E., Samajdar, A., Kwon, H., Nadella, V., Srinivasan, S., Das, D., Kaul, B., Krishna, T.: Sigma: A sparse and irregular gemm accelerator with flexible interconnects for dnn training. In: 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 58–70. IEEE (2020)
2. Srivastava, N., Jin, H., Smith, S., Rong, H., Albonesi, D., Zhang, Z.: Tensaurus: A versatile accelerator for mixed sparse-dense tensor computations. In: 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 689–702 (2020). https://doi.org/10.1109/HPCA47549.2020.00062
3. Asgari, B., Hadidi, R., Krishna, T., Kim, H., Yalamanchili, S.: Alrescha: A lightweight reconfigurable sparse-computation accelerator. In: 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 249–260 (2020). https://doi.org/10.1109/HPCA47549.2020.00029
4. Zhang, Z., Wang, H., Han, S., Dally, W.J.: Sparch: Efficient architecture for sparse matrix multiplication. In: 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 261–274 (2020). https://doi.org/10.1109/HPCA47549.2020.00030
5. Pal, S., Beaumont, J., Park, D.H., Amarnath, A., Feng, S., Chakrabarti, C., Kim, H.S., Blaauw, D., Mudge, T., Dreslinski, R.: Outerspace: An outer product based sparse matrix multiplication accelerator. In: 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 724–736. IEEE (2018)