2025 Design, Automation & Test in Europe Conference (DATE 2025)

DuSGAI: A Dual-side Sparse GEMM Accelerator with Flexible Interconnects

Wujie Zhong and Yangdi Lyu[†]
Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China
[†]Corresponding author: yangdilyu@hkust-gz.edu.cn

***Abstract*—Sparse general matrix multiplication (SpGEMM) is a crucial operation of deep neural networks (DNNs), leading to the development of numerous specialized SpGEMM accelerators. These accelerators leverage flexible interconnects, thereby outperforming their rigid counterparts. However, the suboptimal utilization of sparsity patterns limits overall performance efficiency. In this work, we propose DuSGAI, a sparse GEMM accelerator that employs a parallel index intersection structure to utilize dual-side sparsity. Our evaluation of DuSGAI with five popular DNN models demonstrates a 3.03× performance improvement compared to the state-of-the-art SpGEMM accelerator.**

***Index Terms*—Sparse Matrix Multiplication, Hardware Accelerator, Inner-product Dataflow, Flexible Interconnects**

I. Introduction

Sparse general matrix multiplication (SpGEMM) accelerators have been developed to improve the performance and energy efficiency of deep neural network (DNN) workloads. The existing SpGEMM accelerators can be classified into three categories [1] based on their dataflows: inner-product [2], outer-product [3], and row-based dataflows [4], each with its own benefits and limitations. Compared to the other two categories, the inner-product dataflow can make a good reuse of output partial sums, but it introduces additional overhead for the index intersection of input and weight matrices. This paper focuses on optimizing the performance of the inner-product dataflow.

Existing inner-product accelerators like SIMGA [5] adopt a flexible tree-based architecture to conduct parallel reductions across various partial sum clusters. These accelerators map the non-zero (NNZ) values of the stationary matrix to the reduction tree and stream the corresponding value from the streaming matrix. This approach enables the bypassing of zero computations within the stationary matrix, thereby enhancing performance and energy efficiency compared to conventional rigid DNN accelerators such as TPU [6]. Nonetheless, these accelerators are unable to skip zero computations in the streaming matrix, which constrains the overall throughput.

To leverage the sparsity from both matrices, we propose DuSGAI, a novel inner-product SpGEMM accelerator to address challenges posed by the irregular shape and sparsity patterns of sparse matrices. The primary contributions of our work can be summarized as follows:

- We develop an efficient non-zero matcher for the index intersection of input and weight matrices, allowing for the parallel processing of non-zero values from both matrices.
- We conduct evaluations of DuSGAI using five popular DNN models with diverse characteristics. The results indicate that DuSGAI achieves a 3.03× improvement in performance over SIGMA.

II. DuSGAI

A. Overview

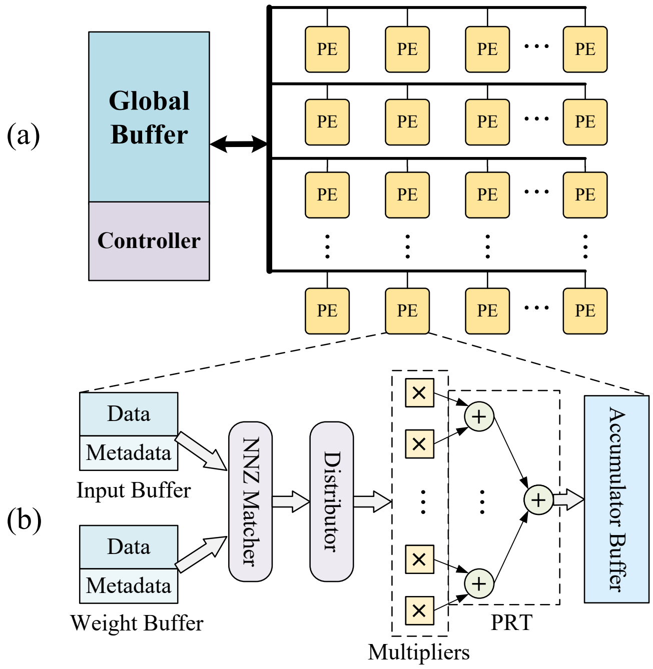

Fig. 1. (a) The high-level overview of DuSGAI, and (b) the PE Microarchitecture.

DuSGAI is composed of a global buffer, a controller, and a PE array, as illustrated in Fig. 1(a). Each PE is composed of several key components, including buffers, multipliers, an NNZ matcher, a distributor, and a pipelined reduction tree (PRT), as illustrated in Fig. 1(b). There are three types of buffers in each PE, i.e., input, weight, and accumulator buffers, each serving a distinct purpose. The input and weight buffers are responsible for storing compressed data comprising non-zero values and corresponding bitmask metadata. After identifying non-zero indices of the input and weight matrices by the NNZ matcher, the distributor transfers data from the input and weight buffers to the multipliers. The partial sums generated by the multipliers

978-3-9826741-0-0/DATE25/© 2025 EDAA

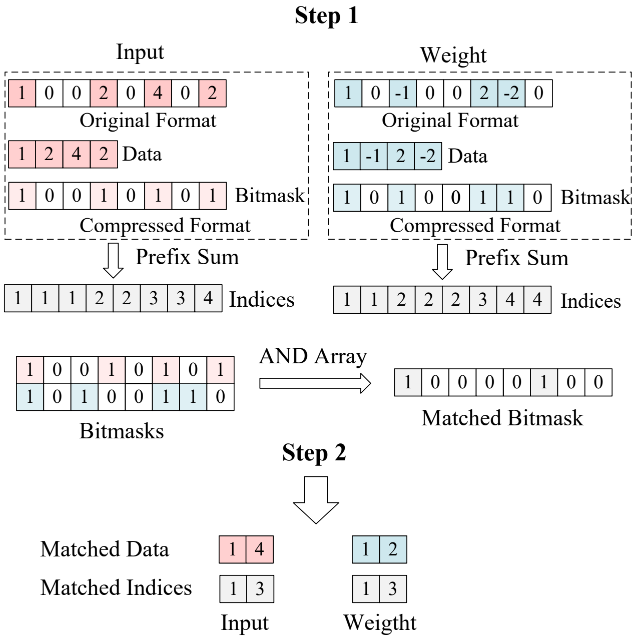

Fig. 2. An example for the NNZ matcher.

are then accumulated by PRT and subsequently stored in the accumulator buffer. The accumulator buffer maintains the uncompressed format of the generated output matrix.

B. Non-zero (NNZ) Matcher

The NNZ matcher operates in a two-step process, as illustrated in Fig. 2. Initially, the prefix sum module calculates the prefix sum for the bitmasks of the input and weight arrays to obtain the indices of the non-zero elements. Concurrently, the AND array performs a bitwise AND operation on the two bitmask arrays to identify the positions where both the input and weight elements are non-zero. The resulting matched bitmask contains a "1" at each position where both corresponding elements are non-zero. In the subsequent step, the indices whose corresponding positions in the matched bitmask are "1" are stored in register files. In the provided example, the first and sixth bits of the matched bitmask vector are "1". Therefore, the matched non-zero indices of the input and weight data are both [1, 3], according to the outputs of the prefix sum modules. Here the indices start from one, so the matched data of the input and weight arrays are [1, 4] and [1, 2], respectively. In terms of time complexity, this intersection process can be completed in a single cycle due to its parallel structure.

III. Experiment

In this study, we conduct a comparative analysis between DuSGAI and SIGMA [5], using an identical configuration of 1024 multipliers for both accelerators to ensure a fair comparison. Additionally, the bandwidth, on-chip buffer capacity, and data width are also kept consistent between the two accelerators. A cycle-accurate simulator developed in C++ is used to evaluate the performance of DuSGAI. The benchmark consists of five popular DNN models: ResNet50, VGG16, MobileNetV2, BERT, and T5. These models are chosen due to their widespread use and representation of various DNN architectures in current research and applications.

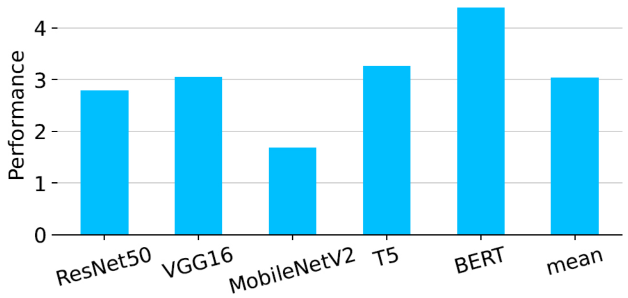

Fig. 3. The normalized performance of DuSGAI over SIGMA for the model workloads.

The normalized performance of DuSGAI over SIGMA for the model workloads is illustrated in Fig. 3. On average, DuSGAI achieves a 3.03× improvement in performance compared to SIGMA. This improvement can be attributed to two factors. First, DuSGAI effectively utilizes dual-side sparsity, which enhances its performance. Second, DuSGAI employs PE local buffers to mitigate the bandwidth limitation of the global buffer. The performance speedup of MobileNetV2 is a little less pronounced compared to other models, attributable to its limited sparsity. The adoption of advanced pruning techniques may lead to additional performance gains.

IV. Conclusion

We propose DuSGAI, a SpGEMM accelerator with the ability to utilize the benefits of dual-side sparsity and flexible interconnects. We evaluate DuSGAI with five popular DNN models. The experiment results show that DuSGAI can achieve a 3.03× improvement in performance compared to the state-of-the-art SpGEMM accelerator baseline.

References

[1] Z. Li *et al.*, "Spada: accelerating sparse matrix multiplication with adaptive dataflow," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 2023, pp. 747–761.

[2] H. Kwon *et al.*, "Maeri: Enabling flexible dataflow mapping over dnn accelerators via reconfigurable interconnects," *ACM SIGPLAN Notices*, vol. 53, no. 2, pp. 461–475, 2018.

[3] S. Pal *et al.*, "Outerspace: An outer product based sparse matrix multiplication accelerator," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2018, pp. 724–736.

[4] G. Zhang *et al.*, "Gamma: Leveraging gustavson's algorithm to accelerate sparse matrix multiplication," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 687–701.

[5] E. Qin *et al.*, "Sigma: A sparse and irregular gemm accelerator with flexible interconnects for dnn training," in *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2020, pp. 58–70.

[6] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.