

MIT Open Access Articles

*Accelerating Sparse Data Orchestration via
Dynamic Reflexive Tiling (Extended Abstract)*

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Odemuyiwa, Toluwanimi, Asghari-Moghaddam, Hadi, Pellauer, Michael, Hegde, Kartik, Tsai, Po-An et al. 2023. "Accelerating Sparse Data Orchestration via Dynamic Reflexive Tiling (Extended Abstract)."

As Published: <https://doi.org/10.1145/3597635.3598031>

Publisher: ACM|Proceedings of the 2023 ACM Workshop on Highlights of Parallel Computing

Persistent URL: <https://hdl.handle.net/1721.1/152175>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of Use: Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.



Accelerating Sparse Data Orchestration via Dynamic Reflexive Tiling (Extended Abstract)

Toluwanimi O. Odemuyiwa
University of California, Davis
California, USA

Hadi Asghari-Moghaddam
University of Illinois Urbana-Champaign
Illinois, USA

Michael Pellauer
NVIDIA
Massachusetts, USA

Kartik Hegde
University of Illinois Urbana-Champaign
Illinois, USA

Po-An Tsai
NVIDIA
Massachusetts, USA

Neal C. Crago
NVIDIA
Massachusetts, USA

Aamer Jaleel
NVIDIA
Massachusetts, USA

John D. Owens
University of California, Davis
California, USA

Edgar Solomonik
University of Illinois Urbana-Champaign
Illinois, USA

Joel S. Emer
MIT/NVIDIA
Massachusetts, USA

Christopher W. Fletcher
University of Illinois Urbana-Champaign
Illinois, USA

CCS CONCEPTS

- **Computer systems organization** → **Special purpose systems**;
- **Hardware** → **Hardware accelerators**.

KEYWORDS

Tensor Algebra, Sparse Computation, Hardware Acceleration

ACM Reference Format:

Toluwanimi O. Odemuyiwa, Hadi Asghari-Moghaddam, Michael Pellauer, Kartik Hegde, Po-An Tsai, Neal C. Crago, Aamer Jaleel, John D. Owens, Edgar Solomonik, Joel S. Emer, Christopher W. Fletcher. 2023. Accelerating Sparse Data Orchestration via Dynamic Reflexive Tiling (Extended Abstract). In *Proceedings of the 2023 ACM Workshop on Highlights of Parallel Computing (HOPC '23)*, June 16, 2023, Orlando, FL, USA. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3597635.3598031>

1 MOTIVATION

Tensor algebra is a ubiquitous primitive in numerical computations. In particular, *sparse* tensor algebra involves sparse tensors, where most elements are set to zero. The simplest multi-sparse kernel is the multiplication of two sparse matrices, or sparse-sparse matrix multiplication (SpMSPM). SpMSPM arises in solvers for linear systems of equations, eigenvalue computations, and graph algorithms such as computation of shortest paths and centrality measures. The higher-order analogues to SpMSPM involve tensor contraction—the higher-order equivalent of a dot product. Contractions of sparse tensors are important primitives in many of the core areas of tensor computations, including computational chemistry methods and sparse tensor decomposition. Despite their importance, however,

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
HOPC '23, June 16, 2023, Orlando, FL, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0218-1/23/06.
<https://doi.org/10.1145/3597635.3598031>

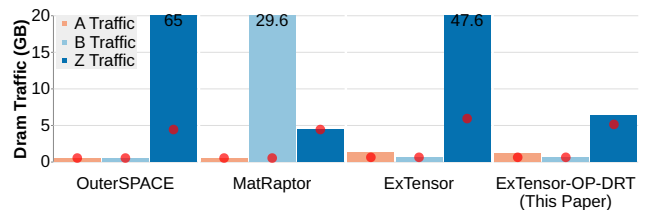


Figure 1: DRAM traffic for each input operand (A , B) and output (Z) in SpMSPM ($A \cdot B = Z$), aggregated over the matrices used in our evaluation, setting $B = A$. Each bar indicates actual traffic and each red square indicates the lower bound on traffic (read each of A and B once, write Z once for each matrix). OuterSPACE [3], MatRaptor [4] and ExTensor [1] are representative accelerators that apply the outer-product, Gustavson’s and inner-product dataflows, respectively. ExTensor-OP-DRT (this paper) using dynamic reflexive tiling achieves significantly closer to the lower bound for all operands/outputs. Different accelerators use similar but not identical compressed representations, and hence have slightly different traffic lower bounds.

the performance of these sparse tensor kernels on modern architectures is memory-bound.

2 LIMITATIONS OF THE STATE OF THE ART

The maximum achievable performance for a memory-bound kernel is a function of its arithmetic intensity: the ratio of FLOPS (or computation) to the data traffic (bytes transferred) from DRAM. Thus, a key opportunity for accelerating sparse tensor algebra is in minimizing the DRAM traffic through on-chip data reuse. Prior work improves arithmetic intensity by exploring dataflows that improve *data reuse*. For example, Figure 1 shows the DRAM traffic of prior SpMSPM accelerators that explore using the three main SpMSPM dataflows (outer-product [3, 7], row-wise Gustavson’s [4, 6] and inner-product [1]) in an attempt to significantly reduce memory traffic. Yet, as the figure shows, dataflow alone is not sufficient to bring DRAM traffic close to the lower bound.

For dense problems, significantly improving reuse beyond dataflow decisions can be achieved through *tiling*. Unfortunately, due to irregular data sparsity, traditional tiling applied to sparse problems does not enable us to mine this data reuse opportunity. Consider the state-of-the-art accelerator in sparse tiling, ExTensor [1]. It statically tiles the input and output matrices offline into

uniformly sized, *coordinate-space* regions, where coordinates correspond to the locations in Cartesian space, such as row and column ids, for elements in each matrix. Such a tiling is oblivious to data sparsity. Therefore, the amount of non-zero data in each tile—the tile’s *occupancy*—can vary widely. This can lead to decreased arithmetic intensity because low occupancy tiles typically result in low reuse per buffer fill. Indeed, Figure 1 shows how ExTensor’s tiling suffers from high output traffic due to being oblivious to data sparsity [1].

3 KEY INSIGHTS

To address the above challenges, we propose *dynamic reflexive tiling* (DRT), a novel tiling algorithm and hardware mechanism that dramatically improves reuse in the presence of irregular sparsity. To demonstrate the effectiveness of DRT, we architect an innovative accelerator called **TACTile**, built around DRT [2], and also show how DRT can improve reuse when applied to prior accelerators [1, 3, 4] that utilize other dataflows, as described above.

The key idea in DRT is to *dynamically co-tile* input and output tensors into *non-uniform coordinate-space* regions: tiles whose volumes differ when measured in coordinates. Through dynamic non-uniform coordinate-space tiling, DRT takes into account data sparsity across all participating tensors, depending on the current active region of each tensor in the computation. Tile occupancy is maximized, subject to the buffer capacity, and variation in occupancy across spatially-distributed tiles is minimized. To maximize utilization across the entire duration of the kernel, DRT not only changes tile shape across different regions of each tensor *but also over time for the same region*, based on how the data is later reused.

A challenge that arises when tiling into non-uniform size regions is how to enable *co-iteration*. That is, when performing operations such as inner or outer products on tiles, participating tiles must have corresponding coordinate ranges. For example, an inner product-style matrix multiplication requires that the column coordinates in a tile of matrix *A* match the row coordinates in a tile of matrix *B*.

To address the co-iteration problem, DRT *co-tiles* in the *coordinate space*. A co-tiling is one where co-iterated dimensions, shared between tiles mapped to each buffer, correspond to the same coordinate range in the original untiled tensors. This facilitates operations such as coordinate intersections by ensuring that the set of coordinates from each tile in the intersection covers the same coordinate range. Depending on the dataflow, co-tiling may require coordinating tile shape across many tiles. For example, if a tile of matrix *A* is broadcast to all PEs, all tiles of *B* later mapped to the PEs must be co-tiled with respect to that tile of *A*.

Finally, we propose algorithms and a hardware architecture to perform all of the above efficiently, including hiding the latency of dynamic tile construction. This is challenging, as finding optimal tile shapes implies performing a search that must be solved online and continuously for each set of tiles distributed to each accelerator buffer. We design and implement DRT in a hardware unit called the *tile extractor*, and design an accelerator, ExTensor-OP-DRT (or **TACTile**), that implements multiple levels of tile extraction to hierarchically break down data into sparsity-aware tile shapes. As shown in Figure 1, ExTensor-OP-DRT significantly reduces data traffic, achieving close to the lower bound. Finally, we show that DRT is not tied to **TACTile** but can be integrated into dataflow-

specific accelerators to improve their data traffic. Please see the full paper for details [2].

4 KEY RESULTS AND CONTRIBUTIONS

- We propose a novel mechanism, *dynamic reflexive tiling* (DRT), that dynamically co-tiles input and output tensors to maximize buffer utilization. We also propose a hardware unit, the *tile extractor*, which implements DRT with a small area overhead.
- Running SpMSPM across a range of matrix shapes and sparsity patterns, a graph algorithm, and a higher-order tensor kernel, we compare **TACTile** and its static tiling variant to the prior state of the art in sparse tiling, ExTensor [1], as well as a baseline CPU MKL kernel [5]. It achieves a geometric speedup ranging from $2.4 \times$ – $3.6 \times$ and $3.6 \times$ – $5.5 \times$ over ExTensor and the CPU, respectively. Beyond SpMSPM, we show that DRT can reduce traffic for higher-order tensor kernels. For the evaluated higher-order kernel, it achieves an improvement of $16.6 \times$ and $3.9 \times$ in DRAM traffic over its SUC variant and CPU MKL, respectively.
- We demonstrate that DRT is portable to other sparse accelerators and integrate it into outer-product and row-wise accelerators. DRT tiling provides performance and arithmetic intensity improvements over static uniform tiling for both classes of accelerators.
- Overall, when implemented in hardware, DRT successfully improves load balance and data reuse across all operands without adding significant latency overhead.
- Finally, we evaluate the potential benefits of a software DRT implementation over untiled and statically tiled software implementations, showing a $7.29 \times$ and $2.94 \times$ memory traffic improvement, respectively.

REFERENCES

- [1] Kartik Hegde, Hadi Asghari-Moghaddam, Michael Pellauer, Neal Crago, Aamer Jaleel, Edgar Solomonik, Joel Emer, and Christopher W. Fletcher. 2019. ExTensor: An Accelerator for Sparse Tensor Algebra. In *International Symposium on Microarchitecture (MICRO)*. 319–333. <https://doi.org/10.1145/3352460.3358275>
- [2] Toluwanimi O. Odemuyiwa, Hadi Asghari-Moghaddam, Michael Pellauer, Kartik Hegde, Po-An Tsai, Neal Crago, Aamer Jaleel, John D. Owens, Edgar Solomonik, Joel Emer, and Christopher Fletcher. 2023. Accelerating Sparse Data Orchestration via Dynamic Reflexive Tiling. In *28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '23, Vol. 3)*. <https://doi.org/10.1145/3582016.3582064>
- [3] Subhankar Pal, Jonathan Beaumont, Dong-Hyeon Park, Aporva Amarnath, Siying Feng, Chaitali Chakrabarti, Hun-Seok Kim, David Blaauw, Trevor Mudge, and Ronald Dreslinski. 2018. OuterSPACE: An Outer Product Based Sparse Matrix Multiplication Accelerator. In *International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 724–736. <https://doi.org/10.1109/hpca.2018.00067>
- [4] Nitish Srivastava, Hanchen Jin, Jie Liu, David Albonese, and Zhiru Zhang. 2020. MatRaptor: A Sparse-Sparse Matrix Multiplication Accelerator Based on Row-Wise Product. In *International Symposium on Microarchitecture (MICRO)*. 766–780. <https://doi.org/10.1145/MICRO50266.2020.00068>
- [5] Endong Wang, Qing Zhang, Bo Shen, Guangyong Zhang, Xiaowei Lu, Qing Wu, and Yajuan Wang. 2014. Intel math kernel library. In *High-Performance Computing on the Intel® Xeon Phi*. Springer.
- [6] Guowei Zhang, Nithya Attaluri, Joel S. Emer, and Daniel Sanchez. 2021. 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 (ASPLOS 2021)*. 687–701. <https://doi.org/10.1145/3445814.3446702>
- [7] Zhekai Zhang, Hanrui Wang, Song Han, and William J. Dally. 2020. SpArch: Efficient Architecture for Sparse Matrix Multiplication. In *International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 261–274. <https://doi.org/10.1109/HPCA47549.2020.00030>