

# Exploiting Near-Data Processing to Accelerate Time Series Analysis

Ivan Fernandez<sup>§</sup>

Ricardo Quislan<sup>§</sup>

Christina Giannoula<sup>†</sup>

Mohammed Alser<sup>‡</sup>

Juan Gómez-Luna<sup>‡</sup>

Eladio Gutiérrez<sup>§</sup>

Oscar Plata<sup>§</sup>

Onur Mutlu<sup>‡</sup>

<sup>§</sup>University of Malaga

<sup>†</sup>National Technical University of Athens

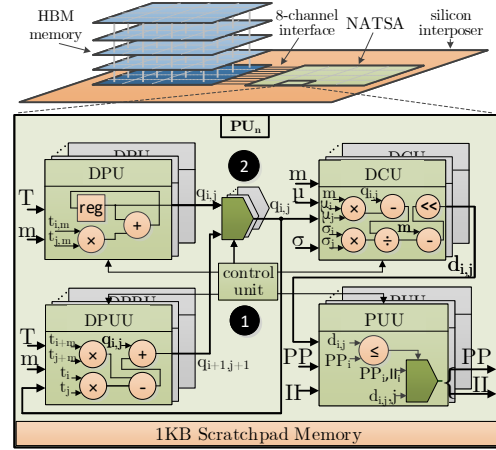
<sup>‡</sup>ETH Zürich

A time series is a chronologically ordered set of samples of a real-valued variable that can contain millions of observations. Time series analysis is used to analyze information in a wide variety of domains [128]: epidemiology, genomics, neuroscience, medicine, environmental sciences, economics, and more. Time series analysis includes finding similarities (*motifs*) and anomalies (*discords*) between every two subsequences (i.e., slices of consecutive data points) of the time series. There are two major approaches for motif and discord discovery: approximate and exact algorithms. Approximate algorithms are faster than exact algorithms, but they can provide inaccurate results or limited discord detection, which cannot be tolerated by many applications (e.g., vehicle safety systems). Unlike approximate algorithms, exact algorithms do not yield false positives or discordant dismissals, but can be very time-consuming on large time series data. Thus, *anytime* versions (aka interruptible algorithms) of exact algorithms are proposed to provide approximate solutions quickly and can return a valid result even if the user stops their execution early. The state-of-the-art exact *anytime* method for motif and discord discovery is *matrix profile* [142], which is based on Euclidean distances and floating-point arithmetic. We evaluate a recent CPU implementation of the *matrix profile* algorithm [149] on a real multi-core machine (Intel Xeon Phi KNL [76]) and observe that its performance is heavily bottlenecked by data movement. In other words, the amount of computation per data access is not enough to hide the memory latency and thus time series analysis is memory-bound. This overhead caused by data movement limits the potential benefits of acceleration efforts that do not alleviate the data movement bottleneck in current time series applications.

Several CPU and GPU implementations of *matrix profile* have been proposed in the literature. However, these acceleration efforts still require transferring the time series data from the main memory to the CPU/GPU cores, leading to the data movement bottleneck. Near-Data Processing (NDP) [1–20, 22–29, 31–57, 57, 58, 58–62, 62–74, 77–84, 86–91, 93, 94, 96–127, 129–135, 135, 136, 136–141, 143–148, 150, 151] is a promising approach to alleviate data movement by placing processing units close to memory. As a result, NDP solutions have the potential to greatly improve system performance and energy efficiency when they are carefully designed with low-cost and low-overhead near data processing cores for memory-bound applications [21].

Our **goal** in this work is to enable high-performance and energy-efficient time series analysis for a wide range of applications, by minimizing the overheads of data movement.

This can enable efficient time series analysis on large-scale systems as well as embedded and mobile devices, where power consumption is a critical constraint (e.g., heart beat analysis on a mobile medical device to predict a heart attack [95] or early earthquake detection [30]). **To this end**, we propose NATSA, the *first* Near-Data Processing Accelerator for Time Series Analysis. The key idea of NATSA (Fig. 1) is to exploit modern 3D-stacked High Bandwidth Memory (HBM) along with specialized custom processing units in the logic layer of HBM, to enable energy-efficient and fast *matrix profile* computation near memory, where time series data resides. NATSA supports a wide range of time series applications thanks to *matrix profile*'s generality and flexibility.



**Figure 1: NATSA design and integration next to HBM memory. NATSA is connected directly to the HBM interface.**

Our evaluation shows that NATSA provides up to  $14.2\times$  ( $9.9\times$  on average) higher performance and up to  $27.2\times$  ( $19.4\times$  on average) lower energy consumption compared to a state-of-the-art multi-core system. NATSA consumes  $11.0\times$  and  $4.1\times$  less energy over optimized implementations of *matrix profile* on an Intel Xeon Phi KNL [76] and NVIDIA GTX 1050 GPU [85], respectively. NATSA has  $9.6\times$  and  $1.8\times$  smaller area than these two accelerators, at equivalent performance points. NATSA outperforms a general-purpose NDP platform by  $6.3\times$  while consuming  $10.2\times$  less energy.

This work makes the following *contributions*:

- We propose NATSA, the first near-data processing accelerator for accelerating time series analysis using modern 3D-stacked High Bandwidth Memory (HBM) [75, 92].
- We propose a new workload partitioning scheme that preserves the *anytime* property of the algorithm, while provid-

ing load balancing among near-data processing units.

- We perform a detailed analysis of NATSA in terms of both performance and energy consumption. We compare different versions of NATSA (DDR4 and HBM) with four different architectures (8-core CPU, 64-core CPU, GPUs and NDP-CPU) and find that NATSA provides the highest performance and lowest energy consumption.

This invited extended abstract is a summary version of our prior work [41] published at ICCD 2020. NATSA's full-paper, video and codes are available at <https://arxiv.org/abs/2010.02079>, [https://www.youtube.com/watch?v=PwhtSAVa\\_W4](https://www.youtube.com/watch?v=PwhtSAVa_W4) and <https://github.com/CMU-SAFARI/NATSA>, respectively.

## Acknowledgments

This work has been supported by TIN2016-80920-R and UMA18-FEDERJA-197 Spanish projects, and Eurolab4HPC and HiPEAC collaboration grants. We also acknowledge support from the SAFARI Group's industrial partners, especially ASML, Facebook, Google, Huawei, Intel, Microsoft, and VMware, as well as support from the Semiconductor Research Corporation.

## References

- [1] S. Aga *et al.*, "Compute Caches," in *HPCA*, 2017.
- [2] H. Ahmed *et al.*, "A Compiler for Automatic Selection of Suitable Processing-in-Memory Instructions," in *DATE*, 2019.
- [3] J. Ahn *et al.*, "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing," in *ISCA*, 2015.
- [4] J. Ahn *et al.*, "PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture," in *ISCA*, 2015.
- [5] B. Akin *et al.*, "Data Reorganization in Memory Using 3D-Stacked DRAM," in *ISCA*, 2015.
- [6] M. F. Ali *et al.*, "In-Memory Low-Cost Bit-Serial Addition Using Commodity DRAM Technology," in *TCAS-I*, 2019.
- [7] S. Angizi *et al.*, "PIMA-Logic: A Novel Processing-in-Memory Architecture for Highly Flexible and Energy-efficient Logic Computation," in *DAC*, 2018.
- [8] S. Angizi *et al.*, "CMP-PIM: An Energy-efficient Comparator-based Processing-in-Memory Neural Network Accelerator," in *DAC*, 2018.
- [9] S. Angizi *et al.*, "AlignS: A Processing-in-Memory Accelerator for DNA Short Read Alignment Leveraging SOT-MRAM," in *DAC*, 2019.
- [10] S. Angizi *et al.*, "Graphide: A Graph Processing Accelerator Leveraging In-DRAM-computing," in *GLSVLSI*, 2019.
- [11] B. Asgari *et al.*, "FAFNIR: Accelerating Sparse Gathering by Using Efficient Near-Memory Intelligent Reduction," in *HPCA*, 2021.
- [12] H. Asghari-Moghaddam *et al.*, "Chameleon: Versatile and Practical Near-DRAM Acceleration Architecture for Large Memory Systems," in *MICRO*, 2016.
- [13] O. O. Babarinsa *et al.*, "JAFAR: Near-Data Processing for Databases," in *SIGMOD*, 2015.
- [14] R. Balasubramanian *et al.*, "Near-Data Processing: Insights from a MICRO-46 Workshop," *IEEE Micro*, 2014.
- [15] M. Besta *et al.*, "SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems," in *MICRO*, 2021.
- [16] D. Bhattacharjee *et al.*, "ReVAMP: ReRAM based VLIW Architecture for In-memory Computing," in *DATE*, 2017.
- [17] A. Boroumand *et al.*, "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks," in *ASPLOS*, 2018.
- [18] A. Boroumand, "Practical Mechanisms for Reducing Processor-Memory Data Movement in Modern Workloads," Ph.D. dissertation, Carnegie Mellon University, 2020.
- [19] A. Boroumand *et al.*, "Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks," *arXiv preprint arXiv:2109.14320*, 2021.
- [20] A. Boroumand *et al.*, "Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks," in *PACT*, 2021.
- [21] A. Boroumand *et al.*, "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks," *ASPLOS*, 2018.
- [22] A. Boroumand *et al.*, "Polynesia: Enabling Effective Hybrid Transactional/Analytical Databases with Specialized Hardware/Software Co-Design," *arXiv:2103.00798 [cs.AR]*, 2021.
- [23] A. Boroumand *et al.*, "Polynesia: Enabling Effective Hybrid Transactional Analytical Databases with Specialized Hardware Software Co-Design," in *ICDE*, 2022.
- [24] A. Boroumand *et al.*, "CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators," in *ISCA*, 2019.
- [25] A. Boroumand *et al.*, "LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory," *CAL*, 2016.
- [26] D. S. Cali *et al.*, "GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis," in *MICRO*, 2020.
- [27] K. K. Chang *et al.*, "Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM," in *HPCA*, 2016.
- [28] P. Chi *et al.*, "PRIME: A Novel Processing-In-Memory Architecture for Neural Network Computation In ReRAM-Based Main Memory," in *ISCA*, 2016.
- [29] S. Cho *et al.*, "McDRAM v2: In-Dynamic Random Access Memory Systolic Array Accelerator to Address the Large Model Problem in Deep Neural Networks on the Edge," *IEEE Access*, 2020.
- [30] A. Christophersen *et al.*, "Bayesian network modeling and expert elicitation for probabilistic eruption forecasting: Pilot study for Whakaari/White Island, New Zealand," *Frontiers in Earth Science*, vol. 6, p. 211, 2018.
- [31] G. Dai *et al.*, "GraphH: A Processing-in-Memory Architecture for Large-scale Graph Processing," *IEEE TCAD*, 2018.
- [32] Q. Deng *et al.*, "DrAcc: A DRAM Based Accelerator for Accurate CNN Inference," in *DAC*, 2018.
- [33] A. Denzler *et al.*, "Casper: Accelerating stencil computation using near-cache processing," *arXiv preprint arXiv:2112.14216*, 2021.
- [34] S. Diab *et al.*, "High-throughput Pairwise Alignment with the Wavefront Algorithm using Processing-in-Memory," *arXiv preprint arXiv:2204.02085*, 2022.
- [35] S. Diab *et al.*, "High-throughput Pairwise Alignment with the Wavefront Algorithm using Processing-in-Memory," in *HICOMB*, 2022.
- [36] J. Draper *et al.*, "The Architecture of the DIVA Processing-in-Memory Chip," in *SC*, 2002.
- [37] M. Drummond *et al.*, "The Mondrian Data Engine," in *ISCA*, 2017.
- [38] C. Eckert *et al.*, "Neural Cache: Bit-serial In-cache Acceleration of Deep Neural Networks," in *ISCA*, 2018.
- [39] D. G. Elliott *et al.*, "Computational RAM: Implementing Processors in Memory," *IEEE Design & Test of Computers*, 1999.
- [40] A. Farmahini-Farahani *et al.*, "NDA: Near-DRAM acceleration architecture leveraging commodity DRAM devices and standard memory modules," in *HPCA*, 2015.
- [41] I. Fernandez *et al.*, "NATSA: A Near-data Processing Accelerator for Time Series Analysis," in *ICCD*, 2020.
- [42] J. D. Ferreira *et al.*, "pLUTo: In-DRAM Lookup Tables to Enable Massively Parallel General-Purpose Computation," *arXiv:2104.07699 [cs.AR]*, 2021.
- [43] D. Fujiki *et al.*, "In-Memory Data Parallel Processor," in *ASPLOS*, 2018.
- [44] D. Fujiki *et al.*, "Duality Cache for Data Parallel Acceleration," in *ISCA*, 2019.
- [45] P.-E. Gaillardon *et al.*, "The Programmable Logic-in-Memory (PLiM) Computer," in *DATE*, 2016.
- [46] F. Gao *et al.*, "ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs," in *MICRO*, 2019.
- [47] M. Gao *et al.*, "Practical Near-Data Processing for In-Memory Analytics Frameworks," in *PACT*, 2015.
- [48] M. Gao *et al.*, "HRL: Efficient and Flexible Reconfigurable Logic for Near-Data Processing," in *HPCA*, 2016.
- [49] M. Gao *et al.*, "Tetris: Scalable and Efficient Neural Network Acceleration with 3D Memory," in *ASPLOS*, 2017.
- [50] N. M. Ghiasi *et al.*, "GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis," in *ASPLOS*, 2022.
- [51] S. Ghose *et al.*, "Processing-in-Memory: A Workload-Driven Perspective," *IBM JRD*, 2019.

- [52] C. Giannoula *et al.*, "SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems," *arXiv preprint arXiv:2201.05072*, 2022.
- [53] C. Giannoula *et al.*, "Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-in-Memory Architectures," in *SIGMETRICS*, 2022.
- [54] C. Giannoula *et al.*, "SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures," in *HPCA*, 2021.
- [55] M. Gokhale *et al.*, "Processing in Memory: The Terasys Massively Parallel PIM Array," *IEEE Computer*, 1995.
- [56] M. Gokhale *et al.*, "Near Memory Data Structure Rearrangement," in *MEMSYS*, 2015.
- [57] J. Gómez-Luna *et al.*, "Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-In-Memory Hardware," in *JGSC*, 2021.
- [58] J. Gómez-Luna *et al.*, "Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture," *arXiv:2105.03814 [cs.AR]*, 2021.
- [59] B. Gu *et al.*, "Biscuit: A Framework for Near-Data Processing of Big Data Workloads," in *ISCA*, 2016.
- [60] P. Gu *et al.*, "iPIM: Programmable In-Memory Image Processing Accelerator using Near-Bank Architecture," in *ISCA*, 2020.
- [61] Q. Guo *et al.*, "3D-Stacked Memory-Side Acceleration: Accelerator and System Design," in *WoNDP*, 2014.
- [62] J. Gómez-Luna *et al.*, "Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System," *IEEE Access*, 2022.
- [63] N. Hajinazar *et al.*, "SIMDRAM: A Framework for Bit-Serial SIMD Processing Using DRAM," in *ASPLOS*, 2021.
- [64] S. Hamdioui *et al.*, "Memristor for Computing: Myth or Reality?" in *DATE*, 2017.
- [65] S. Hamdioui *et al.*, "Memristor Based Computation-in-Memory Architecture for Data-intensive Applications," in *DATE*, 2015.
- [66] M. Hashemi *et al.*, "Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads," in *MICRO*, 2016.
- [67] M. Hashemi *et al.*, "Accelerating Dependent Cache Misses with an Enhanced Memory Controller," in *ISCA*, 2016.
- [68] J. M. Herruzo *et al.*, "Enabling Fast and Energy-Efficient FM-Index Exact Matching Using Processing-Near-Memory," *The Journal of Supercomputing*, 2021.
- [69] K. Hsieh *et al.*, "Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation," in *ICCD*, 2016.
- [70] K. Hsieh *et al.*, "Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems," in *ISCA*, 2016.
- [71] Y. Huang *et al.*, "A Heterogeneous PIM Hardware-Software Co-Design for Energy-Efficient Graph Processing," in *IPDPS*, 2020.
- [72] W.-M. Hwu *et al.*, "Rebooting the Data Access Hierarchy of Computing Systems," in *ICRC*, 2017.
- [73] A. C. Jacob *et al.*, "Compiling for the Active Memory Cube," Tech. rep. RC25644 (WAT1612-008). IBM Research Division, Tech. Rep., 2016.
- [74] S. Jain *et al.*, "Computing-in-memory with spintronics," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 1640–1645.
- [75] JEDEC, "High Bandwidth Memory (HBM) DRAM," Standard No. JESD235, 2013.
- [76] J. Jeffers *et al.*, *Intel Xeon Phi Processor High Performance Programming: Knights Landing Edition*. Morgan Kaufmann, 2016.
- [77] M. Kang *et al.*, "An Energy-Efficient VLSI Architecture for Pattern Recognition via Deep Embedding of Computation in SRAM," in *ICASSP*, 2014.
- [78] Y. Kang *et al.*, "FlexRAM: Toward an Advanced Intelligent Memory System," in *ICCD*, 1999.
- [79] W. H. Kautz, "Cellular Logic-in-Memory Arrays," *IEEE TC*, 1969.
- [80] D. Kim *et al.*, "Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory," in *ISCA*, 2016.
- [81] G. Kim *et al.*, "Toward Standardized Near-Data Processing with Unrestricted Data Placement for GPUs," in *SC*, 2017.
- [82] J. Kim *et al.*, "The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency-Reliability Tradeoff in Modern DRAM Devices," in *HPCA*, 2018.
- [83] J. Kim *et al.*, "D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput," in *HPCA*, 2019.
- [84] J. S. Kim *et al.*, "GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies," *BMC Genomics*, 2018.
- [85] D. Kirk *et al.*, "NVIDIA CUDA Software and GPU Parallel Computing Architecture," in *ISMM*, 2007.
- [86] P. M. Kogge, "EXECUBE - A New Architecture for Scaleable MPPs," in *ICPP*, 1994.
- [87] S. Kvatinsky *et al.*, "MAGIC—Memristor-Aided Logic," *IEEE TCAS II: Express Briefs*, 2014.
- [88] S. Kvatinsky *et al.*, "Memristor-Based IMPLY Logic Design Procedure," in *ICCD*, 2011.
- [89] S. Kvatinsky *et al.*, "Memristor-Based Material Implication (IMPLY) Logic: Design Principles and Methodologies," *TVLSI*, 2014.
- [90] J. Landgraf *et al.*, "Combining Emulation and Simulation to Evaluate a Near Memory Key/Value Lookup Accelerator," 2021.
- [91] D. Lavenier *et al.*, "Variant Calling Parallelization on Processor-in-Memory Architecture," in *BIBM*, 2020.
- [92] D. Lee *et al.*, "Simultaneous Multi-Layer Access: Improving 3D-Stacked Memory Bandwidth at Low Cost," *TACO*, 2016.
- [93] J. H. Lee *et al.*, "BSSync: Processing Near Memory for Machine Learning Workloads with Bounded Staleness Consistency Models," in *PACT*, 2015.
- [94] Y. Levy *et al.*, "Logic Operations in Memory Using a Memristive Akers Array," *Microelectronics Journal*, 2014.
- [95] K. H. C. Li *et al.*, "The Current State of Mobile Phone Apps for Monitoring Heart Rate, Heart Rate Variability, and Atrial Fibrillation: Narrative Review," *JMIR Mhealth Uhealth*, 2019.
- [96] S. Li *et al.*, "DRISA: A DRAM-Based Reconfigurable In-Situ Accelerator," in *MICRO*, 2017.
- [97] S. Li *et al.*, "Pinatubo: A Processing-in-Memory Architecture for Bulk Bitwise Operations in Emerging Non-Volatile Memories," in *DAC*, 2016.
- [98] Z. Liu *et al.*, "Concurrent Data Structures for Near-Memory Computing," in *SPAA*, 2017.
- [99] S. Lloyd *et al.*, "In-memory Data Rearrangement for Irregular, Data-intensive Computing," *Computer*, 2015.
- [100] S. Lloyd *et al.*, "Near Memory Key/Value Lookup Acceleration," in *MEMSYS*, 2017.
- [101] S. Lloyd *et al.*, "Design Space Exploration of Near Memory Accelerators," in *MEMSYS*, 2018.
- [102] K. Mai *et al.*, "Smart Memories: A Modular Reconfigurable Architecture," in *ISCA*, 2000.
- [103] A. Morad *et al.*, "GP-SIMD Processing-in-Memory," *ACM TACO*, 2015.
- [104] R. C. Murphy *et al.*, "The Characterization of Data Intensive Memory Workloads on Distributed PIM Systems," in *Intelligent Memory Systems*. Springer, 2001.
- [105] O. Mutlu, "Memory Scaling: A Systems Architecture Perspective," *IMW*, 2013.
- [106] O. Mutlu *et al.*, "Research Problems and Opportunities in Memory Systems," *SUPERFRI*, 2014.
- [107] L. Nai *et al.*, "GraphPIM: Enabling Instruction-Level PIM Offloading in Graph Computing Frameworks," in *HPCA*, 2017.
- [108] R. Nair, "Evolution of Memory Architecture," *Proceedings of the IEEE*, 2015.
- [109] R. Nair *et al.*, "Active Memory Cube: A Processing-in-Memory Architecture for Exascale Systems," *IBM JRD*, 2015.
- [110] A. Olgun *et al.*, "QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAMs," in *ISCA*, 2021.
- [111] G. F. Oliveira *et al.*, "DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks," *IEEE Access*, 2021.
- [112] G. F. Oliveira *et al.*, "DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks," *arXiv:2105.03725 [cs.AR]*, 2021.
- [113] M. Oskun *et al.*, "Active Pages: A Computation Model for Intelligent Memory," in *ISCA*, 1998.
- [114] D. Patterson *et al.*, "A Case for Intelligent RAM," *IEEE Micro*, 1997.
- [115] A. Pattanaik *et al.*, "Scheduling Techniques for GPU Architectures with Processing-in-Memory Capabilities," in *PACT*, 2016.
- [116] S. H. Pugsley *et al.*, "NDC: Analyzing the Impact of 3D-Stacked Memory+Logic Devices on MapReduce Workloads," in *ISPASS*, 2014.
- [117] S. H. S. Rezaei *et al.*, "NoM: Network-on-Memory for Inter-Bank Data Transfer in Highly-Banked Memories," *CAL*, 2020.
- [118] A. Rodrigues *et al.*, "Towards a Scatter-Gather Architecture: Hardware

and Software Issues,” in *MEMSYS*, 2019.

- [119] P. C. Santos *et al.*, “Operand Size Reconfiguration for Big Data Processing in Memory,” in *DATE*, 2017.
- [120] V. Seshadri *et al.*, “Buddy-RAM: Improving the Performance and Efficiency of Bulk Bitwise Operations Using DRAM,” arXiv:1611.09988 [cs:AR], 2016.
- [121] V. Seshadri *et al.*, “Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology,” in *MICRO*, 2017.
- [122] V. Seshadri *et al.*, “Fast Bulk Bitwise AND and OR in DRAM,” *CAL*, 2015.
- [123] V. Seshadri *et al.*, “RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization,” in *MICRO*, 2013.
- [124] V. Seshadri *et al.*, “Simple Operations in Memory to Reduce Data Movement,” in *Advances in Computers, Volume 106*, 2017.
- [125] A. Shafiee *et al.*, “ISAAC: A Convolutional Neural Network Accelerator with In-situ Analog Arithmetic in Crossbars,” *ISCA*, 2016.
- [126] D. E. Shaw *et al.*, “The NON-VON Database Machine: A Brief Overview,” *IEEE Database Eng. Bull.*, 1981.
- [127] H. Shin *et al.*, “McDRAM: Low latency and energy-efficient matrix computations in DRAM,” *IEEE TCADICS*, 2018.
- [128] R. H. Shumway *et al.*, “Time Series Analysis and Its Applications: With R Examples,” 2017.
- [129] G. Singh *et al.*, “FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications,” *IEEE Micro*, 2021.
- [130] G. Singh *et al.*, “Accelerating Weather Prediction using Near-Memory Reconfigurable Fabric,” *ACM TRETTS*, 2021.
- [131] G. Singh *et al.*, “NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling,” in *FPL*, 2020.
- [132] G. Singh *et al.*, “NAPEL: Near-memory Computing Application Performance Prediction Via Ensemble Learning,” in *DAC*, 2019.
- [133] H. S. Stone, “A Logic-in-Memory Computer,” *IEEE TC*, 1970.
- [134] Z. Sura *et al.*, “Data Access Optimization in a Processing-in-Memory System,” in *CF*, 2015.
- [135] UPMEM, “Introduction to UPMEM PIM. Processing-in-memory (PIM) on DRAM Accelerator (White Paper),” 2018.
- [136] UPMEM, “UPMEM Website,” <https://www.upmem.com>, 2020.
- [137] Y. Wang *et al.*, “FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching,” in *MICRO*, 2020.
- [138] Y. Xi *et al.*, “In-Memory Learning With Analog Resistive Switching Memory: A Review and Perspective,” *Proceedings of the IEEE*, 2020.
- [139] L. Xie *et al.*, “Fast Boolean Logic Papped on Memristor Crossbar,” in *ICCD*, 2015.
- [140] X. Xin *et al.*, “ELP2IM: Efficient and Low Power Bitwise Operation Processing in DRAM,” in *HPCA*, 2020.
- [141] L. Yavits *et al.*, “GIRAF: General Purpose In-Storage Resistive Associative Framework,” *IEEE TPDS*, 2021.
- [142] C.-C. M. Yeh *et al.*, “Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets,” in *ICDM*, 2016.
- [143] J. Yu *et al.*, “Memristive Devices for Computation-in-Memory,” in *DATE*, 2018.
- [144] Y. Zha *et al.*, “Hyper-AP: Enhancing Associative Processing Through A Full-Stack Optimization,” in *ISCA*, 2020.
- [145] D. P. Zhang *et al.*, “TOP-PIM: Throughput-Oriented Programmable Processing in Memory,” in *HPDC*, 2014.
- [146] M. Zhang *et al.*, “GraphP: Reducing Communication for PIM-based Graph Processing with Efficient Data Partition,” in *HPCA*, 2018.
- [147] L. Zheng *et al.*, “RRAM-based TCAMs for pattern search,” in *ISCAS*, 2016.
- [148] Q. Zhu *et al.*, “Accelerating Sparse Matrix-Matrix Multiplication with 3D-Stacked Logic-in-Memory Hardware,” in *HPEC*, 2013.
- [149] Y. Zhu *et al.*, “Matrix Profile XI: SCRIMP++: Time Series Motif Discovery at Interactive Speeds,” in *ICDM*, 2018.
- [150] Y. Zhuo *et al.*, “GraphQ: Scalable PIM-based Graph Processing,” in *MICRO*, 2019.
- [151] V. Zois *et al.*, “Massively Parallel Skyline Computation for Processing-in-Memory Architectures,” in *PACT*, 2018.