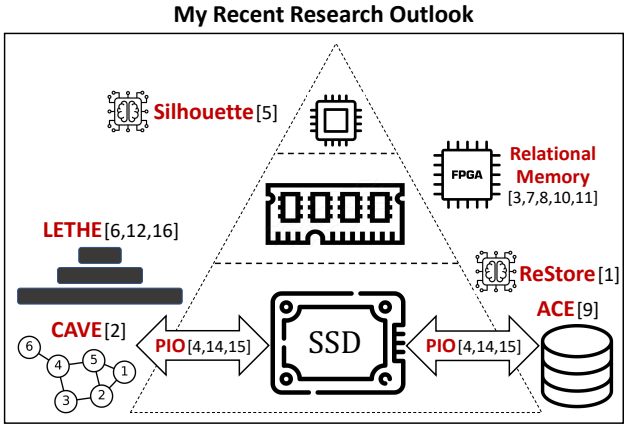
Optimizing Data Systems for Modern Memory Technologies

Tarikul Islam Papon (Boston University, USA)

1 Introduction

My research focuses on developing hardware and storage-aware systems that can enable efficient big data processing. Data-intensive applications place significant strain on the memory hierarchy due to both *excessive data movement* and the integration of *emerging storage technologies*. In my research, I address these challenges through two main avenues: (i) using modern storage devices' untapped potential through *accurate device modeling* and (ii) minimizing unnecessary data movement through *hardware specialization*.

Research Area 1: How to make the most of new SSD and NVM devices? Solid-state drives (SSDs) have become the dominant secondary storage devices replacing hard-disk drives while emerging Non-Volatile Memory (NVM) technologies are showing potential to become the next generation of storage devices. SSDs and most NVMs exhibit two distinct characteristics: (i) *read/write asymmetry*, where writes are slower than reads, and (ii) access *concurrency*, allowing multiple I/O operations to run simultaneously. Without utilizing these two properties, applications cannot fully utilize the device potential. Despite these, most storage-intensive applications do not consider asymmetry and concurrency, hence offering sub-optimal performance. In my research, I propose a new approach for storage modeling that captures these two device-specific properties and allows the designing of storage-intensive algorithms tailored to the device at hand, thus achieving maximum benefit.



Research Area 2: How to minimize data movement? Moving up in the memory hierarchy, a key bottleneck for data-intensive applications is unnecessary data movement. This is further exacerbated due to static data layout decisions since the physical layout and access pattern do not always match. To address this, I propose to capitalize on the reinvigorated trend of *hardware specialization* which has recently gained momentum as Moore's law slows down. The core idea is to perform on-the-fly near-data transformation via specialized hardware through software/hardware co-design. Since the custom hardware sits between processor and memory, this transformation happens quickly without unnecessary cache pollution.

My research on both these avenues has been published in top database conferences such as **ACM SIGMOD**, **VLDB**, **IEEE ICDE**, **EDBT**, and **CIDR**, as well as in top journals like **ACM TODS** and **IEEE TKDE**.

Summary Contributions. Starting from the lower part of the memory hierarchy, I propose the **Parametric I/O Model** [14, 15] to capture the two key properties (read/write asymmetry and concurrency) of modern SSDs and NVMs. Using this novel storage model, (i) I develop an **asymmetry & concurrency-aware DBMS bufferpool management** [9], (ii) I present an SSD **concurrency-aware graph manager** [2] for out-of-core systems, and (iii) we propose a **reinforcement learning-based data migration policy for tiered storage system** [1] that incorporates SSD properties and workload features. Moving higher in the memory hierarchy, we propose a new **software/hardware co-design** approach that performs on-the-fly in-memory data transformation without duplication [7, 11]. This creates **opportunities for innovation in the data system software stack**, including physical design, query optimization, and concurrency control [3, 10].

2 Building Storage-Aware Data Systems

2.1 The Parametric I/O Model [CIDR 21, DaMoN 21 @ ACM SIGMOD]

I propose the Parametric I/O Model (PIO) to capture the fundamental differences between traditional and modern storage devices: *asymmetry* and *concurrency* as parameters [14, 15]. I developed a custom tool to quantify them and empirically measure their values for several types of SSDs. Then I perform an abstract modeling of different applications and experiment with different state-of-the-art devices to show the benefits of carefully exploiting their asymmetry and concurrency: **more informed storage modeling leads to better overall application performance** [4].

2.2 ACE Bufferpool Manager [IEEE ICDE 23]

To assess the PIO model's effectiveness in a practical system, I focus on bufferpool management – a critical Database Management System (DBMS) component. Since the bufferpool is closely connected to the storage device, hence, better storage modeling has the potential to improve its overall performance. However, current approaches treat reads and writes equally without leveraging device concurrency. I refactor the bufferpool design space by decoupling the write-back policy from the eviction policy and propose an asymmetry/concurrency-aware bufferpool manager named **ACE that utilizes the underlying device concurrency to amortize the high asymmetric write cost** [9]. The write-back policy always writes multiple pages concurrently (utilizing the device's *write concurrency*) to amortize the write cost. The eviction policy evicts one or multiple pages at the same time to enable parallel prefetching, exploiting the device's *read concurrency*. A key advantage of ACE is that it can be integrated with any existing page replacement policy with low engineering effort, while, any prefetching technique can also be integrated, essentially allowing **any existing bufferpool manager to be augmented by this approach**. I implement several popular page replacement policies and their ACE counterparts in PostgreSQL and evaluate ACE's benefits using TPC-C and several microbenchmarks. The ACE counterparts of all four policies lead to significant performance improvements: up to 32.1% lower runtime for mixed workloads with a negligible increase in disk writes and buffer misses. This shows that **incorporating asymmetry and concurrency in algorithm design leads to more faithful storage modeling and, ultimately, to better performance**.

2.3 CAVE Graph Manager [SIGMOD 24]

Graph traversal operations, known for their random-access heavy pattern, can benefit from SSD concurrency by parallelizing node and edge accesses, effectively distributing the workload across the SSD's parallel architecture. Hence, to assess the impact of better storage modeling on random access-intensive applications, I next focus on developing an SSD-aware graph manager. Our goal is to parallelize graph traversal algorithms without changing their core properties while utilizing the underlying SSD concurrency. To achieve this, I identify two key ways (intra/inter-subgraph parallelization) to parallelize graph traversal algorithms based on the graph structure and algorithm. I propose an SSD-aware graph processing system, named **CAVE that can harness the concurrency of the underlying storage devices** [2]. In essence, CAVE takes advantage of the availability of multiple paths that can be explored in parallel. CAVE uses a block-based file format based on adjacency lists, ensuring that graph metadata, vertex and edge information are stored in aligned blocks while enabling support for graph traversal and analytical operations. Overall, **CAVE identifies independent storage accesses (thus parallelizable) and performs them concurrently based on the device's *optimal concurrency***. We develop in CAVE the parallelized versions of five popular graph algorithms: Breadth-First Search, Depth-First Search, Weakly Connected Components, PageRank, and Random Walk. We observe that CAVE can be up to three orders of magnitude faster than baseline GraphChi and up to one order of magnitude faster than baseline GridGraph and Mosaic.

2.4 ReStore: Reinforcement Learning-based Data Migration in Tiered Storage

In a tiered storage architecture, fast yet small storage devices serve as upper tiers, while slower and larger ones form the lower tiers. Since SSDs are now the dominant storage devices, modern tiered storage architecture often have multiple (if not all) tiers with SSDs. Existing data migration policies, such as LRU and LFU fail to consider modern SSD properties and they lack adaptability for varying workload. We propose ReStore [1], a Reinforcement Learning (RL) based data migration policy that adapts to workload patterns and device characteristics to ensure optimal performance. **ReStore incorporates workload information such as frequency and recency, along with device-specific characteristics like SSD read/write asymmetry and SSD concurrency, into the RL state definition**. The RL agents' value functions drive the data migration policy, and the agents' parameters are continuously updated through temporal difference learning to ensure adaptability. We observe that compared to state-of-the-art tiered storage migration policies, ReStore achieves up to 50% lower runtime and up to $20\times$ lower number of migrations.

3 On-the-fly Data Transformation via SW/HW Co-design

In data systems, a key decision is between row-stores, which support transactional workloads, and column-stores, which are better for analytical queries. The evolution of these designs has given rise to hybrid transactional/analytical processing (HTAP) architectures, aiming to combine the advantages of both paradigms. However, existing approaches often involve data duplication, additional bookkeeping, and the cost of converting data between different layouts, leading to compromises in analytics efficiency and data freshness.

Relational Memory [TKDE 24, EDBT 23, VLDB 23 (Best Demo), IEEE ICDE 23 (invited for TKDE special issue)]. We enable access to any data layout, shipping only the relevant data through the memory hierarchy by transparently converting rows to arbitrary column groups. To achieve this, we introduce **Relational Memory (RM), a novel FPGA-based hardware design that functions as a near-data vertical partitioner enabling on-the-fly data transformation** from stored rows to any column group [7, 8, 11]. RM, situated between the CPU and main memory, seamlessly transforms base data to any group of columns with minimal overhead at runtime, introducing the first *software/hardware co-design* paradigm for general-purpose query engines and ensuring **optimal data layout access for every query**.

RM offers contiguous access to a specific column (or column-group) in memory by creating references to data that does not exist in main memory, which *the CPU can use as if the data does exist* in main memory. In other words, RM enables accessing the same content in main memory under different strides, but it can be accessed as if it was stored contiguously from the CPU's perspective. Reorganizing data to improve locality minimizes the waste of constrained CPU cache real-estate. In turn, this translates to better efficiency for the query at hand and lower cache pollution. Furthermore, operating closer to the data allows to exploit the inherent parallelism of memory cells. By providing an intuitive API, *Relational Memory shifts vertical partitioning to the hardware*, offering advantages such as eliminating data duplication, simplifying memory management, and reducing unnecessary data movement through the memory hierarchy. Our current design performs projection of column groups with fewer cache misses creating the opportunity for a radical data system redesign including simpler query optimization, physical design, and concurrency control [3, 10]. We implement and deploy RM in real hardware, demonstrating up to a $1.63\times$ faster access to desired columns compared to a row-wise layout. Moreover, RM matches the performance of pure columnar access for low projectivity and outperforms it by up to $2.23\times$ as projectivity (and tuple reconstruction cost) increases. Native data access to both rows and column-groups leads to better cache utilization and paves the way toward **a unified HTAP architecture**.

4 Other Work

Data Management: Efficient Deletes in LSM Engines [ACM SIGMOD 20, IEEE DEBull, ACM TODS]. This work enables *privacy-through-deletion* in modern Log-structured merge (LSM)-based data systems by empowering them with the ability to delete data timely and persistently, without hurting performance [6, 12, 16]. We highlight that all out-of-place key-value stores treat deletes as *second-class citizens*, and are not designed to efficiently realize deletes. We introduce **Lethe [16], a delete-aware out-of-place key-value store** that introduces two new key design components for LSM-based engines: a family of compaction strategies termed FADE and a new storage layout named Key Weaving Storage Layout (KiWi). The key intuitions behind our solution are: (i) use the file-level metadata to track the age of a tombstone (special marker denoting deleted entry), (ii) trigger inter-level data consolidation based on tombstone age, and (iii) choose the appropriate files for consolidation to timely purge the tombstones. The proposed solution not only guarantees timely persistence of logical deletes, but by doing so, also achieves $2-9\times$ lower space amplification and $1.2-1.4\times$ higher read performance. The solution is simple and easily integrable to production systems, and **a variant of this solution is now supported by RocksDB**, a widely used, open-source key-value store.

CPU Performance Modeling via Learned Embeddings [MLforSys@Neurips 23]. During my internship time at Intel, I worked on developing learned embeddings for general-purpose CPUs. Learned embeddings are widely used to obtain concise data representation and enable transfer learning between different data sets and tasks. We present ***Silhouette*** [5], which leverages publicly available CPU performance data sets to learn CPU performance embeddings and enable transfer learning across different types of CPUs.

Social Network Analysis via Distributed k-core Decomposition [SNAM]. I worked on developing a new distributed approach for determining the most influential spreader in a social network by combining both user-specific and topological information. We implemented a distributed modification of the popular *k-core decomposition* algorithm by incorporating user attributes [13]. Our evaluation shows that the proposed approach can process a very large network while being on average 12.5% more accurate and $175\times$ faster.

Mobile Health Applications [NSysS 15, B.Sc. & M.Sc. Thesis]. As part of my B.Sc. thesis, I designed a cost-effective non-invasive infrared sensor-based device to capture the vital signs (heart rate and blood pressure) using Photoplethysmographic (PPG) signal [19, 20, 21]. During my M.Sc. thesis, I worked on detecting diabetic retinopathy from the images of interior surface of eye (known as fundus image) [17]. I trained a deep convolutional neural network using TensorFlow with a large public data set of fundus images and applied several image processing and machine learning techniques to develop the model.

5 Future Research

My long-term research goal is to design versatile data systems meeting a wide range of application needs and performance goals in light of ever-evolving storage and hardware technologies. Toward this, I have identified three key research directions: (i) designing scalable data systems for emerging SSDs like computational SSDs, open-channel SSDs, and zoned namespace SSDs, as well as reprogrammable technologies, (ii) developing a CXL-optimized disaggregated database system, and (iii) incorporating machine learning techniques into data systems components to navigate performance, heterogeneous workload and application requirements tradeoff. With my expertise in storage modeling [4, 14, 15], data systems [1, 2, 6, 9, 12, 16], sw/hw co-design paradigms [3, 7, 8, 10, 11], machine learning for systems [1, 5, 17] and other inter-disciplinary works [13, 18, 19, 20, 21], I am well-equipped to address these research challenges.

Emerging Storage/Hardware-Aware Data Systems. I aim to enable on-the-fly data transformation from stored rows to any column groups at the storage level via near-storage computation in emerging SSDs. While near-storage computation is traditionally challenging due to limited logic capabilities, modern storage devices like computational SSDs and OpenSSDs offer processing power that we can leverage. I plan to experiment with a flipped design, storing base data in columnar format on storage for efficient compression like RLE. The processing capabilities of modern SSDs, coupled with custom logic in embedded FPGAs, will handle decompression and tuple reconstruction, reducing the load on the database system's software stack.

I also intend to redefine the design of LSM (Log-Structured Merge) trees to align with zoned namespace (ZNS) SSDs. Typically, traditional SSDs invalidate older versions of pages in an out-of-place manner, similar to the LSM *compaction* process. However, this process is duplicated for LSMs on SSDs, leading to excessive writes. ZNS SSDs divide space into equal-sized zones and depart from the block API concept, enabling fast sequential writes and flexibility in data placement and garbage collection. I plan to leverage these features by (i) treating erase blocks within SSDs as fragments of a sorted immutable run in LSM trees, (ii) incorporating host-side garbage collection, and (iii) optimizing data placement by grouping related data together.

CXL-Optimized Database System. Resource disaggregation has recently emerged as a promising architecture in modern data centers. The separation of memory resources (both storage and memory) from compute nodes (CPU, GPU) offers advantages such as flexibility, scalability, and efficient resource sharing. In disaggregated data centers (DDCs), local accesses translates to network communication, and despite progress in data center networking, communication remains a bottleneck. Compute Express Link (CXL) has emerged as a promising interconnect technology, surpassing classical RDMA interconnects in speed and approaching the performance of local memory. I aim to develop a CXL-optimized disaggregated database system that provides a seamless programming environment, unifying local and CXL memory without requiring application modifications. My initial plan is to start by developing novel storage and memory management strategies to harness the full potential of CXL's high-speed, coherent connections, ensuring seamless integration with databases while ensuring compatibility across different generations of CXL.

Machine Learning for Data Systems. In light of increasing data complexity, diverse workloads, and the advent of new hardware, machine learning (ML) can play a pivotal role in improving the performance of data system components while maintaining application requirements. I intend to delve into several research challenges on this front: (i) applying ML techniques to optimize energy consumption in data centers by factoring in considerations such as access patterns and device characteristics, thereby steering toward green computing for data centers or warehouses, (ii) exploring how reinforcement learning can enhance the efficiency of LSM compaction file-picking policies, (iii) investigating how ML algorithms can contribute to query optimization by exploring the learned index paradigm, and (iv) examining how ML models can be trained and applied to sensitive data without compromising individual privacy via federated learning and homomorphic Encryption.

References

[1] (Under Review) Tianru Zhang, **Tarikul Islam Papon**, Salman Toor, Manos Athanassoulis. *`ReStore`: Reinforcement Learning-Based Data Migration Policy in Tiered Storage Architecture.*

[2] **Tarikul Islam Papon**, Taishan Chen, Shuo Zhang, Manos Athanassoulis. *CAVE: Concurrency-Aware Graph Processing on SSDs.* In Proceedings of the ACM on Management of Data (**PACMMOD/SIGMOD**), 2024.

[3] **Tarikul Islam Papon**, Ju Hyoung Mun, Shahin Roozkhosh, Denis Hoornaert, Ahmed Sanaullah, Ulrich Drepper, Renato Mancuso, Manos Athanassoulis. *Effortless Locality on Data Systems using Relational Fabric.* IEEE Transactions on Knowledge and Data Engineering (**TKDE**), 2024.

[4] **Tarikul Islam Papon**. *Enhancing Data Systems Performance by Exploiting SSD Concurrency & Asymmetry*. IEEE International Conference on Data Engineering (**ICDE**) PhD Symposium, 2024.

[5] **Tarikul Islam Papon**, Abdul Wasay. *Silhouette: Toward Performance-Conscious and Transferable CPU Embeddings*. Workshop on **ML for Systems** at **NeurIPS**, 2023.

[6] Subhadeep Sarkar, **Tarikul Islam Papon**, Dimitris Staratzis, Zichen Zhu, Manos Athanassoulis. *Enabling Timely and Persistent Deletion in LSM-Engines*. ACM Transactions on Database Systems (**TODS**), 2023.

[7] Ju Hyoung Mun, Konstantinos Karatsenidis, **Tarikul Islam Papon**, Shahin Roozkhosh, Denis Hoornaert, Ulrich Drepper, Ahmed Sanaullah, Renato Mancuso, Manos Athanassoulis. *On-the-fly Data Transformation in Action*. In Proceedings of the **VLDB** Endowment, 2023 **[Best Demo Award]**.

[8] Renato Mancuso, Shahin Roozkhosh, Denis Hoornaert, Ju Hyoung Mun, **Tarikul Islam Papon**, Manos Athanassoulis. *Software-Shaped Platforms*. In Proceedings of the Cyber-Physical Systems and Internet of Things Week (**CPS-IoT Week Workshops**), 2023.

[9] **Tarikul Islam Papon**, Manos Athanassoulis. *ACEing the Bufferpool Management Paradigm for Modern Storage Devices*. In Proceedings of the IEEE International Conference on Data Engineering (**ICDE**), 2023.

[10] **Tarikul Islam Papon**, Ju Hyoung Mun, Shahin Roozkhosh, Denis Hoornaert, Ahmed Sanaullah, Ulrich Drepper, Renato Mancuso, Manos Athanassoulis. *Relational Fabric: Transparent Data Transformation [Vision]*. In Proceedings of the IEEE International Conference on Data Engineering (**ICDE**), 2023 **[Invited for TKDE]**.

[11] Shahin Roozkhosh, Denis Hoornaert, Ju Hyoung Mun, **Tarikul Islam Papon**, Ahmed Sanaullah, Ulrich Drepper, Renato Mancuso, Manos Athanassoulis. *Relational Memory: Native In-Memory Accesses on Rows and Columns*. In Proceedings of the International Conference on Extending Database Technology (**EDBT**), 2023.

[12] Manos Athanassoulis, Subhadeep Sarkar, **Tarikul Islam Papon**, Zichen Zhu, Dimitris Staratzis. *Building Deletion-Compliant Data Systems*. IEEE Data Engineering Bulletin (**IEEE DEBull**), 2022.

[13] T. M. Tariq Adnan, Md. Saiful Islam, **Tarikul Islam Papon**, Shourav Nath, Muhammad Abdullah Adnan. *UACD: A Local Approach for Identifying the Most Influential Spreaders in Twitter in a Distributed Environment*. Social Network Analysis and Mining (**SNAM**), 2022.

[14] **Tarikul Islam Papon**, Manos Athanassoulis. *A Parametric I/O Model for Modern Storage Devices*. In Proceedings of the International Workshop on Data Management on New Hardware (**DaMoN**), 2021.

[15] **Tarikul Islam Papon**, Manos Athanassoulis. *The Need for a New I/O Model*. In Proceedings of the bienial Conference on Innovative Data Systems Research **(CIDR)**, 2021 **[Honorable Mention for Best Gong Show Talk]**.

[16] Subhadeep Sarkar, **Tarikul Islam Papon**, Dimitris Staratzis, Manos Athanassoulis. *Lethe: A Tunable Delete-Aware LSM Engine*. In Proceedings of the **ACM SIGMOD** International Conference on Management of Data, 2020.

[17] **Tarikul Islam Papon**. *Design and Development of A Deep Learning Based Applica-tion for Detecting Diabetic Retinopathy*. **M.Sc. Thesis**, Department of CSE, BUET 2019.

[18] Sadia Tamanna Khan, Ashef Ainan Baksh, **Tarikul Islam Papon**, Muhammad Ashraf Ali. *Rainwater Harvesting System: An Approach for Optimum Tank Size Design and Assessment of Efficiency*. International Journal of Environmental Science and Development (**IJESD**), 2017.

[19] **Tarikul Islam Papon**, Ishtiyaque Ahmad, Nazmus Saquib, Ashikur Rahman. *Non-invasive Heart Rate Measuring Smartphone Applications using On-board Cameras: A Short Survey*. 1st International Conference on Networking Systems and Security (**NSysS**), 2015.

[20] Nazmus Saquib, **Tarikul Islam Papon**, Ishtiyaque Ahmad, Ashikur Rahman. *Measurement of Heart Rate Using Photoplethysmography*. 1st International Conference on Networking Systems and Security (**NSysS**), 2015.

[21] **Tarikul Islam Papon**, Nazmus Saquib, Ishtiyaque Ahmad,. *Photoplethysmographic Analysis of Optical Signals : A Single Device to Measure All the Vital Signs*. **B.Sc. Thesis**, Department of CSE, BUET 2015.