ISSN 2186-7437

NII Shonan Meeting Report

No. 187

Theoretical Foundations of Nonvolatile Memory

Martin Farach-Colton
Sam H. Noh
Gala Yadgar

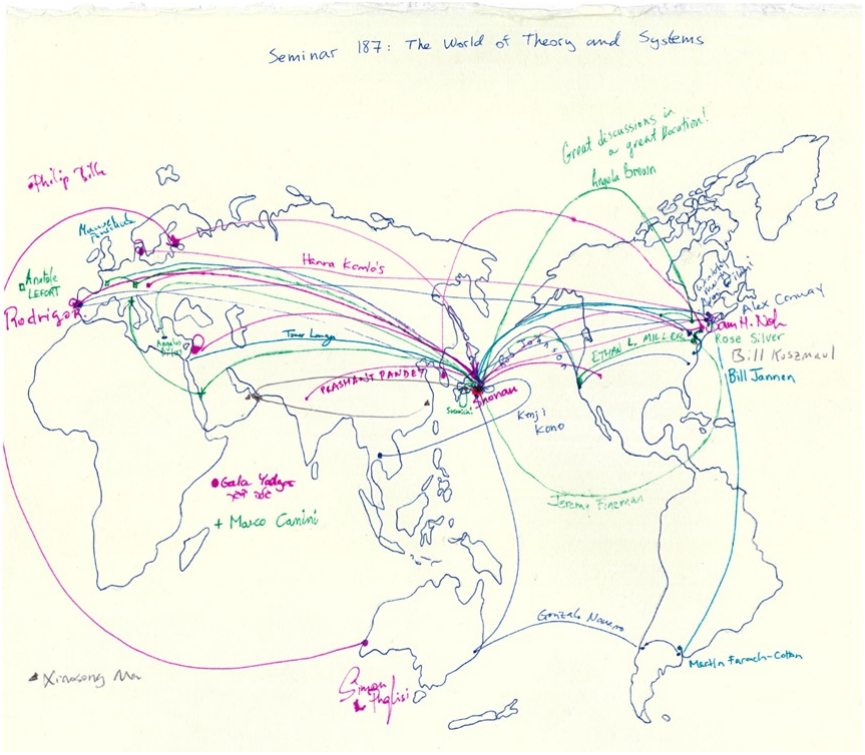
July 1–4, 2024

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Theoretical Foundations of Nonvolatile Memory

Organizers:
Martín Farach-Colton (New York University)
Sam H. Noh (Virginia Tech)
Gala Yadgar (Technion)

July 1–4, 2024



Background and introduction

In the days of hard disks, a single I/O took several milliseconds, whereas a single CPU instruction took roughly only one nanosecond. Furthermore, effectively utilizing disk bandwidth required performing large I/Os, whereas RAM could be accessed at byte granularity.

Because RAM and disks were so different in their performance characteristics, research in algorithms and data structures bifurcated into the RAM and Disk-Access Machine (DAM) models. In the RAM model, the goal is to minimize the number of instructions executed. In the DAM model, all data is stored on disk and must be brought into RAM when needed. The goal in the DAM model is to minimize the number of I/Os performed.

In the stark world of RAM and hard disks, these models were good at capturing the most important performance features of algorithms and data structures. However, we are currently in the middle of multiple upheavals in storage hardware.

On the storage side, devices are getting faster, they now have internal parallelism, and they are becoming more diverse. For example, a cutting edge NVMe drive may have a latency of only 10µsecs, which is only about 100 times slower than RAM. Moreover, NVMe storage bandwidth is now approaching 10GB/sec on very recent devices, which is approximately 100 times faster than a typical hard drive.

Furthermore, locality is being replaced by parallelism. On a hard drive, the way to get good performance is locality, i.e. performing large I/Os. On solid state storage devices, the way to get good performance is parallelism, i.e. to present many I/Os to the device at the same time. On some devices, locality offers essentially no additional performance benefit, i.e. a highly concurrent random I/O workload will be just as fast as a workload of large, sequential I/Os.

The big picture is that storage hardware is getting faster, more complex, more concurrent, and more diverse. Furthermore, the gap between each level of the memory hierarchy is narrowing. These small but non-trivial performance gaps put both the RAM and DAM models on shaky ground.

These are not just theoretical problems. For example, most (if not all) high-performance key-value stores are CPU bound on NVMe devices, even when they have numerous computing cores at their disposal. We recently built a new key-value store specifically designed to be able to fully utilize the performance of high-end NVMe storage. We had to revisit every level of the software stack, from caches and locks all the way up to our sorting algorithms, in order to be able to drive the device at its full potential hardware.

Overview of the meeting

This meeting was organized to bring together members of the architecture, operating systems and algorithms communities. Our main goal was to catalyze the formation of cross-disciplinary groups who can both create the new models needed for understanding modern hardware and use those models for many concrete systems-level advances. Indeed, many of our attendees stood out at previous theory/systems workshops as bridges between communities and as communication facilitators.

Our workshop followed a standard format of talks and open problem sessions, but with some changes. We encouraged attendees to prepare talks suitable for a general CS audience. We also held "Say What?" sessions, which is a technique used in theory/systems workshops in the past. During the workshop, any time a presenter would use a term that some members of the audience did not understand, the audience members were encouraged to say "Say What?" Then, the term was added to the "Say What?" Board. We then had sessions in which systems researchers explained the systems terms form the "Say What?" Board to the theoreticians, and vice versa.

The storage system world is in a ferment as new hardware becomes available. Our workshop took place in the midst of this process, in the best time to establish deep partnerships across disciplines in computer science to solve some of the most pressing big-data infrastructure problems.

Overview of Talks

Algorithms for asymmetric read/write costs

Jeremy Fineman, Georgetown University

Some non-volatile memory technologies have the feature that writes are significantly more expensive (at least in terms of energy) than reads. This work considers the problem of designing algorithms optimized for writes. For example, for the problem of sorting n objects, an algorithm that performs n writes and $O(n \log n)$ reads is significantly cheaper than one that performs $O(n \log n)$ writes. Most textbook sorting algorithms fall into the latter category, but there are existing sorting algorithms that use only n writes. This work considers write-efficient version of more difficult problems such as dynamic-programming problems.

Black-box crash-consistency and concurrency bug detection for persistent memory

Rodrigo Rodrigues, The Instituto Superior Tecnico, INESC-ID

Efficiently leveraging persistent memory (PM) is challenging—*stores* that traverse the cache must be explicitly persisted, creating a temporal window between visibility and persistency. This opens the door to new classes of bugs, namely, inconsistent application state upon machine or application crashes, and a more specific class called *persistency races*, which happen when a thread reads unpersisted data and a crash happens. Detecting these races is daunting as they are only exposed by a few interleavings within a narrow window.

This talk overviewed recent efforts in developing black-box tools to detect both crash-consistency bugs and persistency races. The approach for detecting persistency races, in particular, is based on the novel concept of *Persistent Interference-Free Regions (PIFRs)*, which soundly extend the race detection window such that an interleaving within that larger window is sufficient to detect a persistency race. PIFRs are delimited by a hybrid algorithm combining static and dynamic analysis at the binary level. The experimental results show that the tools that were presented are very effective in detecting PM bugs in a variety of systems. The ensuing discussion stressed the need to develop new abstractions to prevent these problems by construction.

Locality of reference in compact data structures

Gonzalo Navarro, Universidad de Chile

Compact data structures allow us to represent data and its accompanying data structures within compressed space. While using them usually requires more operations than classic data structures, they may be faster when they manage to fit in a faster level of the memory hierarchy—particularly RAM vs. disk. When this is not the case, however, these structures tend to feature poor locality of reference compared to classic ones. This talk surveyed some examples where using compression improves time even in the same level of the memory hierarchy, and examples where it provably worsens time.

Daily development-friendly bug detection in Linux

Kenji Kono, Keio University

Bug detection tools are not widely used in daily development, probably due to long analysis time. There is a well-known trade-off between analysis time and bug detection capabilities. In this talk, we explored an approach that balances this trade-off by a careful combination of computationally lightweight analyses. The proposed approach successfully identified 47 new bugs in the Linux kernel version 5.15, outperforming existing tools in both speed and bug detection.

Algorithms for the ultra-wide word RAM

Philip Bille, Technical University of Denmark

We discussed an interesting extension of the classic word RAM model of computation that adds support for slightly longer words and scattered memory reads and writes. Surprisingly, this simple extension allows us to circumvent well-known lower bounds and significantly improve several classic data structure bounds.

Algorithms for SSD management

Tomer Lange, Technion

SSDs have gained a central role in the infrastructure of large-scale datacenters, as well as in commodity servers and personal devices. The main limitation of flash media is its inability to support update-in-place: after data has been written to a physical location, it has to be erased before new data can be written to it. Moreover, SSDs support read and write operations in granularity of pages, while erasures are performed on entire blocks, which often contain hundreds of pages. When erasing a block, any valid data it stores must be rewritten to a clean location. As an SSD eventually wears out with a progressing number of erasures, the efficiency of the management algorithm has a significant impact on its endurance.

This talk described the first formal definition of the SSD management problem. We explored this problem from an algorithmic perspective, considering it in both offline and online settings. In the offline setting, we present a near-optimal algorithm that, given any input, performs a negligible number of rewrites (relative to the input length). In the online setting, we first considered algorithms that have no prior knowledge about the input. We proved that no deterministic algorithm outperforms the greedy algorithm in this setting, and discuss the possible benefit of randomization. We then augmented our model, assuming that each request for a page arrives with a prediction of the next time the page is updated. We designed an online algorithm that uses such predictions, and show that its performance improves as the prediction error decreases. We also showed that the performance of our algorithm is never worse than that guaranteed by the greedy algorithm, even when the prediction error is large.

Applicability of program differentiation to reducing timing parameters of DRAM

Soramichi Akiyama, Ritsumeikan University

Our previous work established a method to statistically model the distribution of errors of program outputs when the variables inside that program are converted from floating-point to fixed-point [Akiyama et al., ISQED'24]. In this Shonan meeting, we discussed if and how we can apply the same methodology to a different type of approximation, reducing the timing parameters of DRAM. Reducing the timing parameters is known to lower both the energy consumption and random access latency of DRAM, both of which are growing challenges on large-scale computers today and in the future.

Tail latency in LSM-based key-value stores

Angelos Bilas, Foundation for Research and Technology (FORTH), University of Crete

LSM-based key-value (KV) stores suffer from high tail latency, in the order of several seconds, making them less attractive for user-facing applications. In this talk, we introduced the notion of compaction chains and we analyse how they affect tail latency. Then, we show that modern designs reduce tail latency, by trading I/O amplification or requiring large amounts of memory.

Based on our analysis, we presented vLSM, a new KV store design that improves tail latency significantly without compromising on memory or I/O amplification. vLSM reduces (a) compaction chain width by using small SSTs and eliminating the tiering compaction required in L_0 by modern systems and (b) compaction chain length by using a larger than typical growth factor between L_1 and L_2 and introducing overlap-aware SSTs in L_1.

We implemented vLSM in RocksDB and evaluated it using db_bench and YCSB. Our preliminary evaluation highlights the underlying trade-off among memory requirements, I/O amplification, and tail latency, as well as the advantage of vLSM over current approaches. vLSM improves P99 tail latency by up to 4.8× for writes and by up to 12.5× for reads, reduces cumulative write stalls by up to 60% while also slightly improving I/O amplification at the same memory budget.

Searching and indexing deduplicated data

Gala Yadgar, Technion

Deduplication is widely used to effectively increase the logical capacity of large-scale storage systems, by replacing redundant chunks of data with references to their unique copies. As a result, the logical size of a storage system may be many multiples of the physical data size. The many-to-one relationship between logical references and physical chunks complicates many functionalities supported by traditional storage systems, but, at the same time, presents an opportunity to rethink and optimize others. This talk focused on the functionalities of search and indexing.

The first was the offline task of searching for one or more byte strings (keywords) in a large data repository. The traditional, naïve, search mechanism traverses the directory tree and reads the data chunks in the order in which they are referenced, fetching them from the underlying storage devices repeatedly if they are referenced multiple times. In contrast, the DedupSearch algorithm operates in two phases: a physical phase that first scans the storage sequentially and processes each data chunk only once, recording keyword matches in a temporary result database, and a logical phase that then traverses the system's metadata in its logical order, attributing matches within chunks to the files that contain them.

Second, We showed that indexing deduplicated data with deduplication-oblivious mechanisms might result in extreme inefficiencies: the index size would increase in proportion to the logical data size, regardless of its duplication ratio, consuming excessive storage and memory and slowing down lookups. In addition, the logically sequential accesses during index creation would be transformed into random and redundant accesses to the physical chunks. Indeed, to the best of our knowledge, term indexing is not supported by any deduplicating storage system. We described the design of a deduplication-aware term-index that addresses these challenges, and discussed intriguing remaining challenges in extending its applicability to general types of queries and indexed file formats.

Scalable sampling techniques for random benchmark data

Manuel Penschuck, Goethe University

Random graph models are frequently used as a controllable and versatile data source for experimental campaigns in various research fields. Generating such data-sets at scale is a non-trivial task as it requires design decisions typically spanning multiple areas of expertise. Challenges begin with the identification of relevant domain-specific network features, continue with the question of how to compile such features into a tractable model, and culminate in algorithmic details arising while implementing the pertaining model. In this talk, we focused on one important aspect and discuss sampling techniques that were applied successfully in scalable random graph generators. We also connected these techniques to more general benchmark problems beyond random graph models.

Exploring Tiered Heterogeneous Cache

Avani Wildani, Emory University and Cloudflare

Storage systems persist large volumes of data and provide fast data access to applications that are ubiquitous in our society, such as banking, social networks, machine learning, video streaming, and ride hailing. The cheap, high-capacity, low bandwidth backing store provides persistence, whereas the expensive, low-capacity, high-bandwidth cache delivers performance. Multiple storage nodes with backing store and cache provide the required capacity and performance. Large storage clusters with expensive hardware can be costly, both financially and ecologically. In order to reduce the size and consequently the cost of storage systems, we need to squeeze more performance from a single server. An

approach is to add a flash cache to support the DRAM cache, which increases the potential throughput of a storage server. This can reduce the number of storage servers that are required to meet the performance requirement. However, it is challenging to determine when using a flash cache can be beneficial.

We compared the performance of storage servers with and without multi-tier caches using diverse servers and workloads, and demonstrate techniques to determine when to use a multi-tier cache. First, we evaluated the potential performance and cost benefit of using multi-tier caches using simulation and analysis. We developed an algorithm, Cydonia, to determine cost-effective tier sizes given a workload and storage devices on the server. Next, we used trace replay to validate the performance improvement from multi-tier caches and demonstrate the importance of request rate along with miss ratio in determining performance. We trained decision tree models that accurately predict whether using a multi-tier cache will improve performance using the large corpus of data collected using trace replay for a given server.

A short tale on programmable networks for distributed deep learning

Marco Canini, King Abdullah University of Science and Technology (KAUST)

Training large deep learning (DL) models is challenging due to high communication overheads that distributed training entails. Embracing the recent technological development of programmable network devices, this talk described our efforts to rein in distributed deep learning's communication bottlenecks and offered an agenda for future work in this area.

We demonstrated that an in-network aggregation primitive can accelerate distributed DL workloads, and can be implemented using modern programmable network devices. We discussed various designs for streaming aggregation and in-network data processing that lower memory requirements and exploit sparsity to maximize effective bandwidth use. We also touched on gradient compression methods, which contribute to lower communication volume and adapt to dynamic network conditions. Lastly, we considered how to continue our research in light of the enormous costs of training large models at scale, which make it quite hard for researchers to tackle this problem area. We described our ongoing work to create a new approach to emulate DL workloads at a fraction of the necessary resources.

Successor queries in optimal external-memory dictionaries

Rob Johnson, Broadcom Inc.

Dictionaries, which allow users to store and retrieve key-value pairs, are fundamental and widely used data structures. They are so important that they are built into many programming languages. External-memory dictionaries are used to store, on disk, data sets that are too large to fit in RAM. In 2011, Iacono and Patrascu gave a lower bound on the trade-off between insertion and query costs in external-memory dictionaries and presented a data structure meeting their lower bound. Later, Conway, Farach-Colton and Shilane gave simpler

constructions. However, none of these optimal dictionaries supported successor queries.

This talk described the mapped B^ϵ-tree, a data structure that meets the Iacono-Patrascu lower bound while supporting efficient successor queries. Furthermore, whereas previous designs were fundamentally based on hashing, the mapped B^ϵ-tree is essentially an atomic-key comparison-based dictionary with a little bit of hashing stuck on the side, suggesting an alternative approach to designing external-memory dictionaries that meet the Iacono-Patrascu bound. The talk also described SplinterDB, a high-performance and highly concurrent mapped B^ϵ-tree implementation.

Adaptive filters: how to learn from your mistakes

Prashant Pandey, The University of Utah

Adaptive filters, such as telescoping and adaptive cuckoo filters, update their representation upon detecting a false positive to avoid repeating the same error in the future. Adaptive filters require an auxiliary structure, typically much larger than the main filter and often residing on slow storage, to facilitate adaptation.

However, existing adaptive filters are not practical and have seen no adoption in real-world systems due to two main reasons. Firstly, they offer weak adaptivity guarantees, meaning that fixing a new false positive can cause a previously fixed false positive to come back. Secondly, the sub-optimal design of the auxiliary structure results in adaptivity overheads so substantial that they can actually diminish the overall system performance compared to a traditional filter.

In this talk, I described AdaptiveQF, the first practical adaptive filter with minimal adaptivity overhead and strong adaptivity guarantees, which means that the performance and false-positive guarantees continue to hold even for adversarial workloads. The AdaptiveQF is based on the state-of-the-art quotient filter design and preserves all the critical features of the quotient filter such as cache efficiency and mergeability. Furthermore, we employ a new auxiliary structure design which results in considerably low adaptivity overhead and makes the AdaptiveQF practical in real systems.

List of Participants

- Soramichi Akiyama, Ritsumeikan University, Japan
- Angelos Bilas, Foundation for Research and Technology (FORTH), University of Crete, Greece
- Philip Bille, Technical University of Denmark, Denmark
- Marco Canini, King Abdullah University of Science and Technology (KAUST), Saudi Arabia
- Alex Conway, Cornell Tech, USA
- Angela Demke Brown, University of Toronto, Canada
- Martin Farach-Colton, New York University, USA
- Jeremy Fineman, Georgetown University, USA
- Inge Goertz, Technical University of Denmark, Denmark
- William Jannen, Williams College, USA
- Rob Johnson, Broadcom Inc., USA
- Hanna Komlos, New York University, USA
- Kenji Kono, Keio University, Japan
- William Kuszmaul, Harvard and CMU, USA
- Tomer Lange, Technion, Israel
- Anatole Lefort, Technical University of Munich, Germany
- Xiaosong Ma, Qatar Computing Research Institute, Qatar
- Ethan Miller, UC Santa Cruz, Pure Storage, USA
- Gonzalo Navarro, Universidad de Chile, Chile
- Sam H. Noh, Virginia Tech, USA
- Prashant Pandey, The University of Utah, USA
- Manuel Penschuck, Goethe University, Germany
- Simon Puglisi, University of Helsinki, Finland
- Rodrigo Rodrigues, The Instituto Superior Tecnico, INESC-ID, Portugal
- Rose Silver, Northeastern University, USA
- Avani Wildani, Emory University, Cloudflare, USA
- Gala Yadgar, Technion, Israel

Meeting Schedule

Check-in Day: June 30 (Sun)

- Welcome Banquet

Day1: July 1 (Mon)

- Welcome and introduction
- Algorithms for asymmetric read/write costs (Jeremy Fineman)
- Black-box crash-consistency and concurrency bug detection for persistent memory (Rodrigo Rodrigues)
- Locality of reference in compact data structures (Gonzalo Navarro)
- Daily development-friendly bug detection in Linux (Kenji Kono)
- Algorithms for the ultra-wide word RAM (Philip Bille)
- "Say what?" session

Day2: July 2 (Tue)

- Algorithms for SSD management (Tomer Lange)
- Automated synthesis of verified CXL bridges for heterogeneous architectures (Anatole Lefort)
- Optane is dead, what's next? (Ethan Miller)
- Applicability of program differentiation to reducing timing parameters of DRAM (Soramichi Akiyama)
- Group photo shooting
- Tail latency in LSM-based key-value stores (Angelos Bilas)
- Searching and indexing deduplicated data (Gala Yadgar)

Day3: July 3 (Wed)

- Some scalable sampling techniques for random benchmark data (Manuel Penschuck)
- Thoughts from recent experience working on storage for LLM training (Xiaosong Ma)
- Exploring tiered heterogeneous cache (Avani Wildani)
- A short tale on programmable networks for distributed deep learning (Marco Canini)
- Excursion and main banquet

Day4: July 4 (Thu)

- Successor queries in optimal external-memory dictionaries (Rob Johnson)
- Adaptive filters: how to learn from your mistakes (Prashant Pandey)
- Summary discussion and farewell

Summary of discussions

By design, participants of the workshop were mainly from two different camps, the theory camp and systems camp, that typically do not interact with one another. Thus, much of the discussions were concentrated on understanding each others' terminology, methodology, and challenges. The "Say What?" session in the schedule was naturally extended to the free discussions that followed a similar structure: they were especially helpful by allowing the participants to ask basic questions without any hesitation.

The two camps came together under the topic of nonvolatile memory, the theme of the workshop. However, the diversity of the meaning of nonvolatile memory as well as the diversity of the participants allowed the discussion to digress to a vast range of subjects from flash memory to machine learning to compact data structures to KV stores to adaptive filters. While the formal talks were structured to introduce topics and recent results in related areas, the discussions often took the form of exploring connections between the areas and between camps. Participants investigated similarity and differences between the challenges they address, to what extent open questions of one camp can be addressed by methods and approaches of the other camp, and which challenges are worth addressing together as part of collaborative interdisciplinary projects.

Summary of new findings

The workshop did not result in direct and explicit new findings in any scholarly aspect. Nevertheless, both the theory and systems camps were able to come closer to understanding the thinking process of the "other" camp and their approach to addressing challenges in their respective fields. This is an important step forward from each participants' perspective, as it helps build a solid foundation for future interaction and collaboration between the two camps.

The participants found this to be a highly valuable experience. They all aspired that opportunities for these kinds of interaction between siloed disciplines would become more frequent. Moreover, even in the short duration of the workshop, initial steps in defining shared challenges were already taken. We anticipate that some of these challenges will be further studied in collaborative projects, and we are looking forward to seeing their outcome and results in the future.

Identified issues and future directions

The academic part of the workshop lasted for three and a half days, which is, realistically, very short. This is especially so for our type of workshop where participants are from two different camps. It takes time for participants to acclimate themselves to the world of "other" camp: they need to familiarize themselves with new terminology, conventions, objectives, and constraints. Similarly, they must articulate their thoughts and concerns in simpler and broader terms—a challenging yet highly rewarding task for researchers.

Nevertheless, all participants found this kind of interaction to be helpful in better understanding one another as well as the workings of nonvolatile memory as it becomes more integrated into modern computer systems. Thus, we look forward to future workshops that encourage interaction of different disciplines.