Distributed and Parallel Databases (2023) 41:163–189
https://doi.org/10.1007/s10619-022-07421-x

Challenges and future directions for energy, latency, and lifetime improvements in NVMs

Saeed Kargar[1] · **Faisal Nawab**[2]

Accepted: 7 September 2022 / Published online: 21 September 2022

Abstract

Recently, non-volatile memory (NVM) technology has revolutionized the landscape of memory systems. With many advantages, such as non volatility and near zero standby power consumption, these byte-addressable memory technologies are taking the place of DRAMs. Nonetheless, they also present some limitations, such as limited write endurance, which hinders their widespread use in today's systems. Furthermore, adjusting current data management systems to embrace these new memory technologies and all their potential is proving to be a nontrivial task. Because of this, a substantial amount of research has been done, from both the database community and the storage systems community, that tries to improve various aspects of NVMs to integrate these technologies into the memory hierarchy. In this work, which is the extended version of Kargar and Nawab (Proc. VLDB Endowment 14(12):3194–3197, 2021), we explore state-of-the-art work on deploying NVMs in database and storage systems communities and the ways their limitations are being handled within these communities. In particular, we focus on (1) the challenges that are related to high energy consumption, low write endurance and asymmetric read/write costs and (2) how these challenges can be solved using hardware and software solutions, especially by reducing the number of bit flips in write operations. We believe that this area has not gained enough attention in the data management community and this tutorial will provide information on how to integrate recent advances from the NVM storage community into existing and future data management systems.

Keywords Non-volatile memory · Write endurance · Energy consumption · Reducing bit flips

✉ Saeed Kargar
skargar@ucsc.edu

Faisal Nawab
nawabf@uci.edu

[1] CSE Department, UC Santa Cruz, Santa Cruz, CA, USA

[2] CSE Department, UC Irvine, Irvine, CA, USA

1 Introduction

Nowadays, we are witnessing the emergence of new non-volatile memory (NVM) technologies that are remarkably changing the landscape of memory systems. They are persistent, have high density, byte addressable, and require near-zero standby power [1], which makes them of great interest in the database and storage systems communities. Furthermore, NVMs provide up to 10x higher bandwidth and 100x lower access latency compared to SSDs [2, 3]. However, because they also present some limitations, such as limited write endurance, which is significantly lower than DRAM write endurance, and high write energy consumption (Fig. 1), adopting the current systems to use NVM while maximizing their potential is proving to be challenging. Additionally, unlike flash storage, cells are written individually in many NVM technologies such as PCM. This means that some cells may receive a much higher number of writes than others during a given period, and as a result wear out sooner. So, any system design needs to take these limitations into consideration before deciding to utilize NVMs.

The scientific community has conducted an extensive amount of research on integrating these new technologies in current systems. Furthermore, these emerging technologies are carving out their own place in the memory hierarchy. As persistent memories are usually larger than DRAM in capacity, researchers in database community usually take advantage of its consistency to boost performance of the system by making the in-memory database persistent, which results in capacity expansion

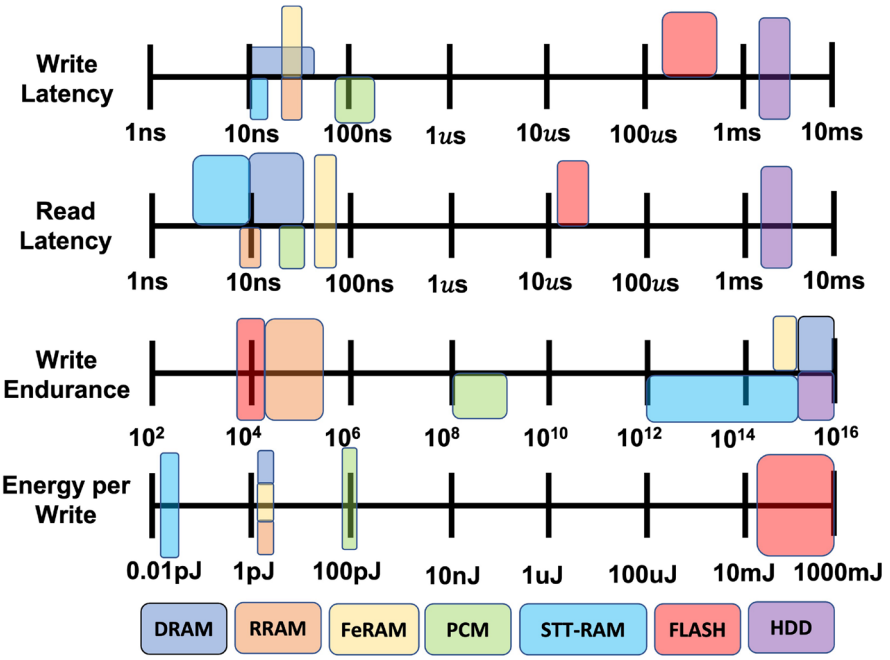

Fig. 1 Comparison of device properties of memory and storage technologies [4]

and recovery cost reduction. However, when designing a database management system, it is critical to fully consider the characteristics of NVMs to take advantage of their hardware potentials. For example, most NVM technologies have limited write endurance, which needs special considerations when using them as a memory device. Another characteristic is their asymmetric write/read costs in most NVM technologies. So, the databases systems or data structures that are designed for them usually avoid write operations as much as possible.

Therefore, in this tutorial we survey recent work in both storage and database communities, where a substantial amount of work has been done in the adoption of NVMs. In particular, we focus on the common limitations that are shared among most NVMs—i.e. low write endurance, high energy consumption, and intrinsic asymmetric properties of reads and writes. Table 1 compares the main characteristics of these memory technologies. This tutorial provides information on how to integrate recent advances from the NVM storage community into existing and future data management systems.

Contributions In this paper, which is the extended version of [4], we focus on the techniques for improving NVM's performance, lifetime and energy consumption although these challenges might vary from technology to technology. For example, while limited lifetime is a key challenge in most NVM technologies, it might not be an issue in Optane and FeRAM. In this work, we, especially, explore how the mentioned challenges are related to the number of bit flips in write operations in one of the most mature and widespread NVM technologies, i.e. phase change memory (PCM). Figure 2 shows an overview of this paper. In Sect. 2, we present a background on key terms, concepts, and properties of different NVM technologies. In Sect. 2.3, we underscore the motivation for and challenges in improving lifetime, energy consumption and latency of the NVM device. Further, we classify the importance of works based on three main views to give an overall view of the whole field. We discuss three main methods for overcoming some of the limitations that most

Table 1 Device properties of emerging nonvolatile memories

	Series storage class memory device property							
	Cost/bit	Reliability	Flexibility	Write endurance	Write Energy	Scalability	Write latency	Density
PCM	✓	✓	✓	✗	✗	✓	✗	✓
FeRAM	✗	✗	✓	✓	✓	✗	✓	✗
STT-RAM	✗	✗	✗	✓	✗	✗	✓	✗
ReRAM	✓	✗	✓	✗	✗	✓	✗	✓
SRAM	✗	✗	✗	✓	✗	✗	✓	✗
NAND Flash	✓	✗	✓	✗	✗	✗	✓	✓
NOR Flash	✗	✓	✓	✗	✗	✗	✗	✗
3D XPoint	✓	✓	✓	✗	✗	✓	✗	✓

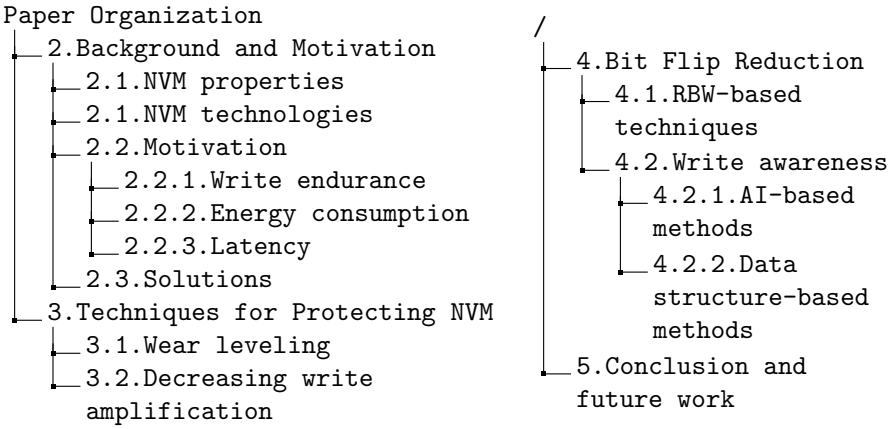

Fig. 2 Organization of the paper

NVM technologies have in common in Sects. 3 and 4. In Sect. 4 we elaborate bit flip reduction technique, which is the main focus of this survey. We conclude this paper in Sect. 5 with a brief mention of future research challenges.

Scope For sake of a concise presentation, we limit the scope of this paper as follows. Although We discuss the storage and hardware-level techniques to improve performance, energy consumption and lifetime of various NVM technologies, in this work, we specifically explore the potential of decreasing bit flips in various aspects of PCMs. We believe that this paper will be useful not only for memory designers and database and storage researchers who are new to this field but also for researchers from other fields, such as IoT, distributed systems, and HPC, who are interested to integrate NVMs in their work.

2 Background and motivation

We now introduce some concepts which will be useful throughout this paper. We refer the reader to previous works for a detailed background on NVM architecture [5–7].

2.1 NVM properties

Non-volatile memory has the potential of transforming the memory architecture in data management systems due to their characteristics such as persistency, high density, byte addressability, and requiring near-zero standby power [1]. However, they also have their own drawbacks that may vary from technology to technology: some have problems with write endurance, some have higher write latency than the read latency, some have low density and do not scale well, and so on. So, researchers have been trying to overcome these limitations to efficiently utilize them.

As far as write endurance is concerned, most NVM technologies, such as PCM, ReRAM, 3D XPoint and Flash memory, have lower write endurance (10^5–10^9) compared to the traditional memory technologies, such as DRAM, which have virtually unlimited lifetime (10^{16}). Even in some NVM technologies such as STT-RAM with higher lifetime (at least 10^{12}), certain usage conditions, such as frequent accesses with a high applied bias voltage or a long pulse, may aggravate their lifetime and reliability [79, 80]. Furthermore, in many use cases, some of NVM technologies, such as PCM, STT-RAM and ReRAM, are used in the lower-level cache hierarchies instead of memory level, which expose them to aggressively high number of writes. So, in these cases, write endurance can prevent the adoption of NVM in lower-level caches.

Improving NVM lifetime is not the only concern that researchers need to deal with. NVM write operations demand a significant amount of current and power. For flipping an individual bit in PCM, for instance, it requires around 50 pJ/b compared to writing a DRAM page, which needs only 1 pJ/b [23]. Table 2 shows some selected studies that are done in recent years to improve lifetime, performance or energy efficiency of different NVM technologies. To solve these main limitations, researchers usually use one or a combination of the following techniques: (1) designing new data structures and database management systems based on the limitations of NVMs, (2) redesign the existing data structures, which are proved to be efficient, to be more NVM-friendly, and (3) designing some special hardware-level or software-level solutions to meet the limitations of NVMs or alleviate their drawbacks.

2.2 NVM technologies

Despite having so many benefits over DRAMs, NVM technologies have lots of limitations in common, such as high energy consumption, low write endurance and asymmetric read/write costs, which needs to be considered before using them. So, researchers come up with different solutions to deal with these limitations, from redesigning the conventional data structures to proposing hardware-level methods, to deploy these new technologies in their systems. Among all the challenges that

Table 2 Selected studies categorized by their contribution

Contribution	Series selected studies
Lifetime improvement	[4, 8–17]
	[18–27]
	[28–34]
Performance	[4, 15–17, 26, 28, 35–38]
	[39–45]
	[46–55]
	[56–64]
Energy efficiency	[4, 9, 13, 18, 23–27, 35]
	[41, 65–71]
	[72–78]

NVMs face, in this paper, we focus on (1) low endurance, high energy consumption, and asymmetric read/write related problems and (2) how researchers in different communities, from databases to storage systems to embedded systems and distributed systems, overcome these limitations. Table 3 classifies the research studies from Table 2 based on their memory technologies. The NVM technologies that we cover are:

Spin-Torque Transfer RAM (STT-RAM), which is a variation of MRAM, switches the memory states using spin-transfer torque. Using spin-polarized current for setting bits makes the cell structure simple and small. The most noticeable advantage of this memory is to have a high efficiency and write endurance even compared to DRAM. These characteristics make it as one of the top alternatives to the current technologies such as DRAM and NOR Flash. Despite having the mentioned advantages, this technology comes at a price: having low density which makes it hard to scale [81].

Resistive RAM(ReRAM) is one of the most promising NVM technologies that can take the place of DRAM. It's simple structure, easy fabrication, high scalability, and compatibility with the existing CMOS technology make this technology a good candidate to be used in many applications in various fields from Neuromorphic Computing to logic applications. In ReRAM, by applying current to a cell, the state of the cell can be switched. Despite all its advantages, it has some drawbacks, such as having low write endurance and inconsistent switching mechanism, which makes researchers look for methods to improve its performance and mitigate its disadvantages to use it in the existing systems [82].

Phase-Change Random Access Memory (PCM), which is probably the most mature of the NVM technologies, consists of phase-change materials that switches between two different phases with distinct properties: an amorphous phase, with high electrical resistivity, and a crystalline phase, with low electrical resistivity. So, in PCMs, there are two main operations: SET operation and RESET operation. These operations are controlled by electrical current as follows: while in the RESET operation High-power are used to place the memory cell into the high-resistance RESET state, for the SET

Table 3 Selected studies categorized by their SCM type

SCM Type	Series selected studies
PCM	[4, 18–20, 22–26, 30, 34]
	[35, 36, 40, 43, 53, 67, 71, 77, 78]
ReRAM	[8–11, 17, 18, 29, 66, 72, 75, 76]
STT-RAM	[8, 11–14, 31–33]
	[41, 42, 63, 65, 70, 73]
NAND Flash	[15, 16, 28, 37, 45, 46, 49]
	[51, 54, 56, 61, 64]
3D XPoint	[25–27, 38, 40, 43, 44, 47, 48, 50]
	[51, 52, 55, 57–60, 62, 69]

operation, moderate power but longer duration pulses are used to return the cell to the low-resistance SET state. Although PCM scales well and has write endurance comparable to that of NAND Flash (10^8–10^9), which makes it a viable alternate for future high-speed storage devices. However, its asymmetric read/write costs in terms of energy consumption and latency, and lower write endurance compared to DRAM (10^{16}), limit the PCM adoption in system architecture.

3D XPoint is a recent NVM technology that is developed by the Intel and Micron jointly since 2015. This technology is usually considered as a type of PCM although Intel has never disclosed its technical details [82]. This technology presents many advantages, such as having a high density and low access latency, which makes it probably one of the best candidates to replace DRAM. However, this technology pose some challenges, which need to be considered. First, its write energy consumption is relatively high. Second, although its write latency is relatively low, it is still orders of magnitude high compared to DRAM. Furthermore, its memory cell write endurance (10^8–10^9) is orders of magnitude lower than that for DRAM although it is claimed that it is enough to survive continuous operation as the main memory for a few years [83].

Flash memory is an electronic NVM storage medium that can be electrically erased and reprogrammed. It is probably the most wide-spread NVM technology since it has the performance better than traditional storage. There are two types of nonvolatile semiconductor flash memory: NOR and NAND. While NOR Flash has low latency and can be programmed at byte granularity, which is suitable for IoT system devices like GPS and e-readers that do not require as much memory, NAND Flash is more like HDD with page-based programing granularity, which is suitable for sequential data such as video, audio, and so on [84]. Despite all the mentioned advantages, this technology shares some disadvantages that most NVM technologies have in common: having limited write endurance, low density, energy constraints, asymmetric read/write latency and persistence.

FeRAM, which has similar structure as DRAM bit cell except, achieve nonvolatility, it utilizes a ferroelectric layer instead of a dielectric layer. This technology has been gaining popularity, especially in industrial IoT applications and autonomous vehicles field, for its unique characteristics such as having very high life endurance, low write time speed, and low write energy consumption [83]. However, like any other technology in NVM category, it faces some limitations that hinders this technology from its widespread use, such as much lower storage densities than Flash, storage capacity limitations and higher cost.

Figure 3 illustrates the number of studies that are carried out in the three main categories, i.e., improving lifetime (LIF), performance (Perf), and energy consumption (EC), on different types of NVM technologies from 2009. These results can reveal how much each of the mentioned fields could draw researchers' attention in different NVM technologies over time. In this figure, we try to give some direction for future work to those researchers who are new to this field.

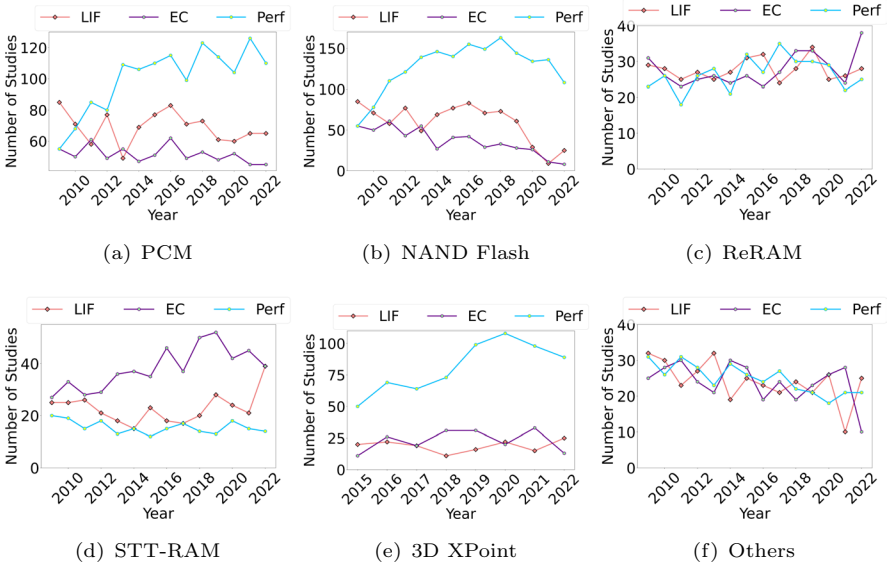

Fig. 3 The number of studies that are done to improve the lifetime (LIF), energy consumption (EC), and performance (Perf) of different NVM technologies

2.3 Motivation for bit flip reduction

2.3.1 Write endurance

Most NVM technologies, such as PCM, ReRAM, 3D XPoint and Flash memory, have low write endurance (the number of writes that can be applied to a segment of storage media before it becomes unreliable.) The write endurance in these technologies is on the order of 10^5–10^9 writes, which is significantly lower than DRAM write endurance (in the order of 10^{16} write) [4, 5]. Even in some NVM technologies such as STT-RAM, which has an endurance of at least 10^{12} writes, certain usage conditions, such as frequent accesses with a high applied bias voltage or a long pulse, may aggravate their lifetime and reliability [79, 80]. Furthermore, in many use cases, some of these technologies, such as PCM, STT-RAM and ReRAM, are used in the lower-level cache hierarchies instead of memory level, which expose them to aggressively high number of writes. So, in these cases, write endurance can prevent the adoption of NVM in lower-level caches.

Table 2 illustrates some of the selected studies that are proposed recently in an attempt to solve the limited write endurance of NVMs. Although most of the solutions that are presented to improve the lifetime of NVM, such as reducing write amplification, local write optimization and memory awareness, can be applied to various NVM technologies with write endurance problem, in this paper, we focus on PCM and how its lifetime can be improved through bit flip reduction.

As mentioned earlier, PCM is currently the most mature and widespread of the NVM memory technologies under research. It is used in different memory

hierarchies from L2 cache to storage medium, which makes it of great importance in terms of replacing conventional memory technologies. In Sect. 4, we explore the effects of reducing the number of bit flips on various aspects of PCM from performance to energy consumption and lifetime. Based on the experiments that we have done in that section, we will show that not only does reducing the number of written cells increase the average lifetime of the device, but also it decreases the system's power consumption and improves the system's latency significantly. We also believe that this area has not gained enough attention in the data management community and this tutorial will provide information on how to integrate recent advances from the NVM storage community into existing and future data management systems.

2.3.2 Energy consumption

Bit flip reduction represents the design principle of minimizing how many bits are flipped from 0 to 1 or 1 to 0 when a write is applied to a memory segment. In a PCM device, as the number of bit flips decreases, the write endurance improves. Reducing the number of written cells also means that the energy consumption of the system drops significantly. For instance, based on an experiment conducted by [66], when the number of different bits in the write data and the overwritten content varies from 0 to 100%, the energy consumption can vary from nearly 0 to over 10,000 pJ. So, targeting bit flip reduction is a worthy investment in the NVM context, and there are a large body of research, such as [18, 23, 25–27, 66, 67], targeting bit flip reduction to extend the NVM lifetime and decrease their energy consumption.

To see how this difference affects the system's energy consumption, we have conducted a simple experiment on a real Optane memory device (Sect. 4). For the experiment, we have used the Persistent Memory Development Kit (PMDK) [85], formerly known as NVML. As we will see in the results, reducing the number of bit flips can have positive effects on the energy consumption of the NVM device. This experiment alongside the ones that we have done to analyze the impacts of bit flip reduction on latency and lifetime of PCMs show that reducing the number of bit flips in the NVM devices, such as PCM, whose controllers are optimized by only flipping bits when the old value of a cell differs from the value being written to it can lead to improvement in lifetime, latency, and energy consumption of the device.

2.3.3 Latency

Not only does reducing the number of bit flips save the energy consumption of the system, but also it can improve the latency of write operations. Figure 6 (the top one) shows that write latency also improves when bit flips are reduced. The main reason behind this is when the content of the old data and the data that is being written is similar, the total number of writes can also be reduced, which results in improving write latency. This process can reduce the number of writes in two ways: (1) the first way is by writing new items in-place to replace a similar old value in terms of hamming distance. This leads to decreasing NVM word writes (i.e., the number of modified words in a cache line.) (2) In the second way, writing similar contents decreases the number of NVM line writes, respectively cache lines needed

to be written per item. For example, suppose that the page size in a system is 4 KB as shown in Fig. 1 [25]. In this scenario, if the items are similar to each other in terms of the hamming distance, fewer number of cache lines are needed to fulfill the request (suppose each part in Fig. 4 is a cache line). This enables PNW to decrease NVM word writes in addition to NVM line writes.

2.4 Solutions

There have been plenty of methods proposed for improving NVM write endurance and energy consumption from hardware-focused methods to software-focused ones. According to their general strategies, these methods can be categorized into three main groups:

(1) The storage community developed write optimization techniques that are mostly based on a *Read-Before-Write* (RBW) pattern [86]. In RBW, a write operation w to a memory location x is always preceded with a read of x. The value to be written by w is compared with the old content of x, and only the bits that are different are written. This reduces the number of flipped bits, which increases write endurance [86]. Other approaches built on RBW to increase write endurance by masking or changing the value to be written if it leads to reducing bit flips [87–91]. Flip-N-Write (FNW), for example, checks whether flipping the bits of the write operation would lead to decreasing the number of bit flips [87].
(2) The data management community tackled the problem of write endurance by minimize write operations via techniques such as caching [92–94] and delayed merging [40, 43], or by designing specialized data structures that require fewer writes [24]. Therefore, data structures that are designed to be deployed on NVM should be designed in a way to exploit the advantages and avoid the disadvantages of the technologies. For example, data structures for disks are block-oriented and work the best for sequential access. However, those designed for flash reduce write amplification, which is the main concern in flash technologies [23]. As we will explain later, techniques that focus on reducing write amplification can result in increasing write endurance although this is not always the case [23].
(3) The proposed methods in the third group also increase the write endurance of PCMs through a new approach called *memory-awareness*. The main idea behind this approach is to take advantage of having knowledge of the memory content in advance because read operations are less expensive than write operations in PCMs. Prior methods pick the memory location for a write operation arbitrarily (new data items select an arbitrary location in memory, and updates to data items

	Content
Old item	0xABCDA23B BCDABCDA BCDAB01A ... 1CDA23BA CCDAAB00 ACCDAB01
New item	0xABCDABC1 BCDABCDA BCDAB01A ... 1CDA23BA CCDAAB00 ABCDABC8

Fig. 4 An example of replacing a memory content with a similar content used in PNW [25]

overwrite the previously-chosen location.) The methods in this group judiciously pick a memory location that is similar to the value to be written. When the new value and the value to be overwritten are similar, this means that the number of bit flips is going to be lower. There are lots of methods that use this approach as the backbones of their methods, which will be explored in Sect. 4.2.

3 Techniques for protecting NVM

In this section, we present the main concerns, challenges, and limitations of state-of-the-art methods that have utilized NVMs in their designs by dividing them into three main groups based on the trends and solutions they propose to solve the problem of write endurance and energy consumption. We also identify the short- and long-term research opportunities in this space.

3.1 Wear leveling

Wear leveling has been studied extensively for Flash-based storage devices [95–99]. In these methods, the wear-leveling algorithms usually keep track of the storage blocks and remap those blocks that are written heavily in a given time quanta to the lowest wear-out blocks. In storage class memory, wear leveling has almost the same objective, which is extending the lifetime of NVM devices by distributing the writes evenly across the memory blocks of NVM so that no *hot* area reaches its maximum lifespan by extremely high concentration of write operations [18, 66, 87, 88, 100–102]. These methods usually are implemented in the memory controller level to protect NVMs. Wear-leveling is usually transparent for upper-level applications and they can simply access to the same content using the same logical address (LA) (they are unaware of the physical address where the data are stored.) According to the mapping strategies, existing wear-leveling schemes can be divided into two main groups:

(1) Table-based wear-leveling (TBWL) techniques that store the LAs, their corresponding physical addresses (PA), and the frequency of accesses. In this way, the storage table of the wear-leveling can be eliminated. When the number of writes to a specific physical address (PA) goes beyond a threshold, its content is swapped by the wear-leveling method [18, 103].

For example, In [104], the authors propose Fine-Grain Wear Leveling (FGWL), which shifts cache lines within a page to achieve uniform wear out of all lines in the page. Also, when a PCM page is read, it is realigned. The pages are written from the Write Queue to the PCM in a line-shifted format. For a system with 4 lines per page, for instance, the rotate amount is between 0 and 3 lines. The rotate value of 0 means the page is stored in a traditional manner. If it is 1, then the Line 0 of the address is being shifted one line and stored in Line 1 of the physical PCM page, line 1 of the address in stored in line 2, line 2 in 3, and finally, line 3 of address space is stored in Line 0. When a PCM page is

read, it is realigned. The pages are written from the Write Queue to the PCM in a line-shifted format.

Self-adaptive wear-leveling (SAWL) algorithm [18] is another wear-leveling scheme that dynamically adjusts the wear-leveling granularity to accommodate more useful addresses in the cache, thus improving cache hit rate. This method distributes the writes across the cells of entire memory, thus achieving suitable tradeoff between the lifetime and cache hit rate.

(2) Algebraic-based wear-leveling methods [101, 105, 106] are another group that try to distribute the incoming writes to avoid concentrating writes on specific physical locations and creating the hot areas. To this end, they usually replace the address-mapping table in the table-based wear-leveling algorithms with hardware structure, which are more space efficient [101]. In this way, they can increase the lifetime of NVMs orders of magnitude compared to the based line where there is no wear-leveling.

Start-Gap [101] is one of the first methods that proposed a wear-leveling method based on the algebraic mapping between the logical and physical addresses. In this technique, there are two registers (Start and Gap) that do the wear-leveling. When a new write comes to the memory, Start-Gap moves one line from its location to a neighboring location. While the Gap register keeps track of the number of lines moved, Start register counts the number of times that all the available lines have moved. Finally, the mapping between logical and physical address is done by a simple arithmetic operation of Gap and Start registers, which eliminates the need for storing the address-mapping table in the memory.

In [107], the authors propose a hardware-based wear-leveling scheme named Security Refresh, which performs dynamic randomization for placing PCM data. In this method, an embedded controller inside each PCM is responsible for preventing adversaries from tampering the bus interface or aggregating meaningful information via side channels [107]. They also applied designed some attacks to analyze the wear-out distribution using Security Refresh.

Although wear-leveling strategies have been successful in preventing creation of hot locations and extending the lifetime of NVMs, the controller cannot guarantee that some cells will not wear out much faster than the average. The reason is that distributing writes evenly across the memory space does not necessarily mean that the individual cells within the words also be flipped/written evenly. That is why, in some extreme cases, even with the protection of state-of-the-art wear-leveling schemes, wear-out attacks such as Remapping Timing Attack[108] and Row Buffer Hit [109] can wear out NVM as fast as 137 seconds [109]. Therefore, hardware techniques such as FNW [87], CDE [88], FPC [11], and Flip-Mirror-Rotate [90], which will be explained in the next section, have been proposed to focus on reducing the number of bit flips within a given word instead of just distributing writes uniformly across the device.

3.2 Reducing write amplification

Many data storage and indexing solutions target the reduction of write amplification to optimize the utilization of I/O bandwidth. This is done via various techniques, including delaying the consolidation of writes [40, 43], caching [92–94], and others [24]. With the introduction of NVM to the memory hierarchy, it turns out that reducing write amplification can have the positive side-effect of increasing NVM write endurance since less data is written. However, this is not an easy task to do due to the fact that all the existing data structures and database systems have been designed for DRAMs and HDDs, where the challenges of the lifespan of memory segments and the energy consumption of writes are not as significant in DRAM/HDD as they are in NVM. However, as discussed before, when it comes to NVMs, write operation needs to be performed wisely. So, the proposed methods in this group reduce the write amplification in an attempt to decrease the average number of updated cells and as a result increases the lifetime of NVMs.

To achieve this, many methods re-design existing data structures and database systems to mitigate the write amplification issue caused by them instead of designing and building new ones from scratch. The reason behind this is that existing data structures and database systems have undergone decades of research that makes them extremely efficient and makes building alternatives from scratch an arduous task.

Log-Structured Merge-tree (LSM-tree) is one of those data structures that has been widely adopted for use in the storage layer of modern NoSQL systems, and as a result, has attracted a large body of research, from both the database community and the storage systems community, that try to improve various aspects of LSM-trees by using NVMs [37, 38, 40]. NoveLSM [40] is one of these methods. This method is a persistent LSM-based key-value storage system designed to take advantage of having a NVM in its design. To tackle NVM's limited write endurance, NoveLSM comes up with a new design, where only the parts of the key/value store that do not need to be changed frequently, such as immutable memtables, are handled by NVM. On the other hand, other parts, such as mutable memtables, which need constant updates and data movements, are placed on DRAM, which do not have any restrictions on write operation.

WiscKey [37] is another work, which proposes a persistent LSM-tree-based key-value store, which has been derived from the popular LSM-tree implementation, LevelDB. Although, like the other methods in this category, WiscKey focuses on decreasing write amplification, it achieve this through a different and simple way, which is separating keys from values. This method observes that since the indexing is done by keys, and not values, they do not need to be bundled together when they are stored in the LSM-tree. So, in this method, only keys are kept sorted in the LSM-tree, while values are stored separately in a log. Through this insight, they have reduced write amplification by avoiding the unnecessary movement of values while sorting. Although this technique is originally proposed for SSDs, it can be generalized to storage class memories, which suffer from the same limitation.

Another data structure that has been redesigned to utilize NVMs is B+-Trees, which is used widely in K/V data stores [47, 93]. Fingerprinting Persistent Tree

(FPTree) [94] is a hybrid SCM-DRAM persistent and concurrent B+-Tree that is designed specifically for NVMs. This method aims to decrease write amplification on NVMs. To do so, in this method, leaf nodes are persisted in SCM while inner nodes are placed in DRAM and rebuilt upon recovery.

LB+-Tree [44] is another method that changes the structure of the conventional data structures to take advantage of NVMs. In this work, to improve the insertion performance of LB+-Tree, three techniques are proposed: (1) entry moving, which creates empty slots in the first line of a leaf node to reduce the number of NVM line writes, (2) logless node split, which targets the logging overhead, and (3) distributed headers. LB+-Tree improves the performance of the insertion operation compared to the other methods.

DPTree (Differential Persistent Tree) [110] batches multiple writes in DRAM persistently and later merge them into a PM component to decrease the persistence overhead, which is imposed to the system because of the persist primitive that the existing PM index structures use to guarantee consistency in case of failure.

In [111], the authors target the tail latency of tree-based index structures, which the result of the internal structural refinement operations (SROs) and the inter-thread interferences. To reach to this aim, uTree introduces a shadow linked-list layer to the leaf nodes of a B+-tree to minimize the SRO overhead. This method has succeeded in improving throughput and latency.

Hash-based indexing structures have also been good candidates to utilize NVMs due to their nature of typically causing high write amplification and that they are vastly used in various applications and systems [24, 27, 112, 113]. A lot of effort has been made to improve hash-based indexing structures for byte-addressable persistent memory, and almost all of them focus on decreasing the write amplification to reach their goal. Path hashing [24] is an example of these hash-based indexes, which is designed specifically for NVMs. The basic idea of path hashing is to leverage a position sharing method to resolve the hash-collision problem, which usually results in a high number of extra writes or write amplifications.

Dash [114] proposes a persistent hashing scheme that aims to solve the performance (scalability) and functionality challenges that persistent hash tables face by reducing both unnecessary reads and writes. To achieve this, Dash takes advantage of some previous techniques, such as fingerprinting, optimistic locking, and combine them with some novel methods such as bucket load balancing technique. They have tested their method on real Optane DCPMM and the results show that Dash can improve throughput significantly. It is worth noting that the methods that the methods that we mentioned in this section are just some examples of the whole group of methods (Table 2) that utilize NVMs in their work in different ways.

All these methods offer different advantages and disadvantages. These methods invite exploring how they can be integrated with existing data management systems to enable them to improve the lifetime of NVMs. Some of these methods are independent from the application (and often implemented as a hardware method) which means that augmenting them within existing data management systems is a straight-forward task. Other approaches—especially ones based on masking—require domain knowledge on the application using them. There is an opportunity for data management researchers to find ways to adapt these methods to work with

existing data management systems. This would entail learning the write patterns of data management systems and translating this knowledge into appropriate masking techniques that are based on the methods presented above.

4 Bit flip reduction

Although reducing write amplification is a promising way to extend the PCMs' lifespan, it does not necessarily lead to the best opportunities to reduce bit flipping and increasing write endurance [23, 25]. This is because—unlike flash—PCM cells are written individually, which means that the number of flipped bits is more important to optimize than the number of written words [23]. Therefore, focusing on reducing bit flips is a viable solution that can both save energy and extend the life of PCMs.

To see how this difference affects NVM's performance in terms of lifetime, energy consumption, and latency, we have conducted some experiments on a PCM device and a real Optane memory. Figure 5 shows the lifetime of a PCM device is

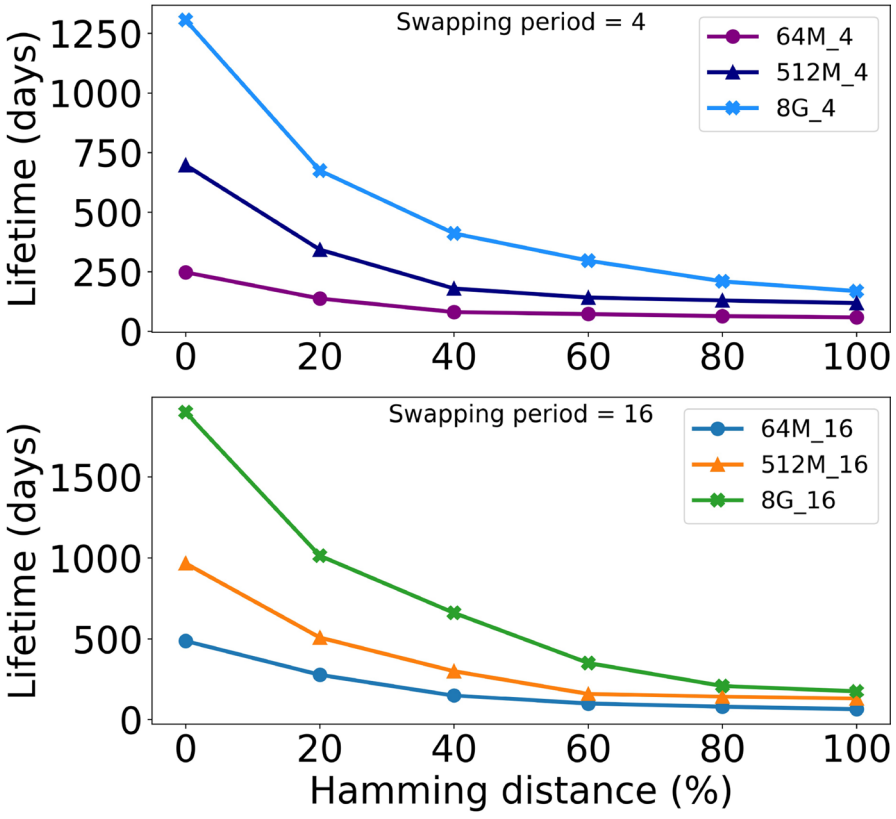

Fig. 5 The impact of capacity and swapping period on PCM's lifetime when the percentage of hamming distance between the write data and the overwritten content changes

affected by the number of bit flips that occur during write operations. As we can see in this figure, when the average percentage of the hamming distance between the old value of the cell and the value that is going to be written increases, the lifetime of the device decreases. This figure also shows swapping period plays an important role in the lifetime of the device. Based on the results, for a certain hamming distance, when the swapping period is high, the device lasts longer because having lower swapping period means more write amplification, which increases the overall flipping bits. However, setting lower swapping period increases the risk of certain cells being worn out sooner than others, especially when there is a malicious attack. Figure 5 also shows that when the capacity of the device is low, e.g., in mobile and embedded systems, memory cells can wear out faster.

To see how bit flip reduction affects the latency and energy consumption of a NVM device, we have conducted a simple experiment on a real Optane memory device (Sect. 4) using the Persistent Memory Development Kit (PMDK) [85], formerly known as NVML. In this test, first, we allocate a contiguous region of N Optane blocks of 256B. During each "round" of the experiment, we first initialize all the blocks with random data, and then update the blocks with new data with content that is x% different than the data that is already in the block (hamming distance). We use PMDK's transactions to persist writes. We measure the latency and energy consumption of the socket for each round. Figure 6 shows that by overwriting similar content, which needs less bit flipping, we can save a huge amount of energy and improve latency.

Generally, the existing bit flipping reduction methods can be divided into two main categories: RBW-based techniques and Memory-awareness.

4.1 RBW-based techniques

One of the most important characteristics of Read Before Write (RBW) technique and its variants is their simplicity and efficiency in dealing with the lifetime issue of NVM. So, there has been a large body of research that uses various types of this method in their works. In this category, there are various techniques, such as caching [92–94] and the RBW technique [86], to decrease the number of bit flips.

RBW is one of the most popular techniques, which has been widely utilized by various approaches [87–91], to reduce the number of bit flips is the RBW technique [86], in which the content of an old memory block is read before it is overwritten with the new data (Fig. 7). This technique replaces each PCM write operation with a more efficient read-modify-write operation. Reading before writing allows comparing the bits of the old and new data, updating only the bits that differ.

Flip-n-Write (FNW) [87] is one of the most popular methods and became the building block of many other techniques in this area. This method compares the current content of the memory location (the old data) with the content to-be-written (the new data). This enables FNW to decide whether to write the new data in its original format or to flip it before writing it if that leads to reducing the number of bit flips. (A flag is used so that future operations know whether to flip the content

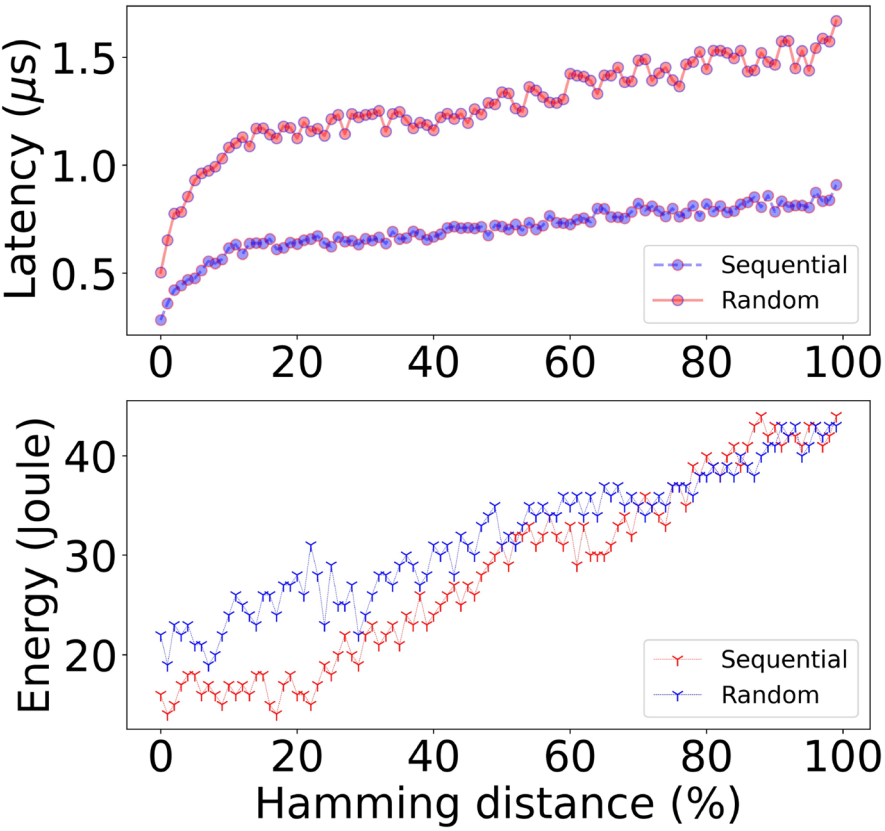

Fig. 6 The latency and memory energy consumption on a real Intel Optane memory device for read and write operations with different percentages of hamming distance

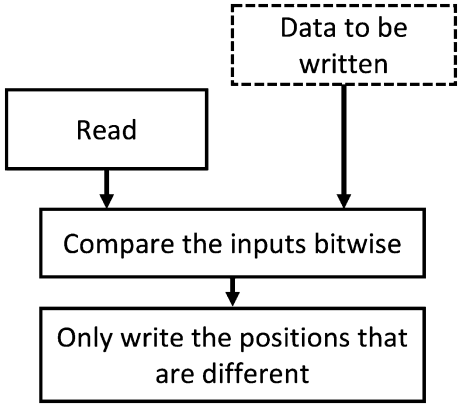

Fig. 7 Read Before Write (RBW) technique

before reading.) This method guarantees that the number of bit flips in PCM is always less than half the total number of written bits (excluding the flag bit).

DCW [88] finds common patterns and then compresses data to reduce the number of bit flips in PCM. Like Flip-N-Write, DCW replaces a write operation with a read-modify-write process. It starts comparing the new data and the old data from the first bit to the last one. The most significant difference between DCW and Frip-N-Write is that in DCW, the maximum number of bit flips is still N (the word width).

Captopril [89] is another recent proposal for reducing bit flips in PCMs. This method masks some "hot locations", where bits are flipped more, to reduce the number of bit flips. In this method, the authors compare every write with 4 predefined sequences of bits to decide which bits need to be flipped and which ones need to be written in their original form. This method suffers from relatively high overhead. More importantly, it is rigid and would only work on predefined applications.

Flip-Mirror-Rotate [90] is another method that is built upon Flip-N-Write [87] and FPC [91] to reduce the number of flipped bits. Like Captopril, this method uses only predefined patterns to mask some bits, which means it would only work on predefined applications.

MinShift [115] proposes reduces the total number of update bits to SCMs. The main idea of this method is that if the hamming distance falls between two specific bounds, the new data is rotated to change the hamming distance. Although this method is simple, it suffers from high overhead.

In [116], the authors use a combination of MinShift and Flip-N-Write to decrease the number of written bits. They compute the minimum amount of some possible states to choose a pattern to encode the data. This method has advantages and disadvantages of both methods.

4.2 Memory-awareness

As we discussed earlier, the writing operations in PCM takes much more energy than reading operation. To save energy, the PCM controller can avoid writing to a cell whose content is the desired value. It means that the lifetime of the cell and the consumed energy in the write operation depends on the number of bits that are actually being flipped by the write rather than the number of words or bits that are written.

Although focusing on bit flipping reduction technique seems a reasonable choice, the methods in this category fail to achieve its full potential because the existing methods miss a crucial opportunity. Prior methods pick the memory location for a write operation arbitrarily (new data items select an arbitrary location in memory, and updates to data items overwrite the previously-chosen location.) This misses the opportunity to judiciously pick a memory location that is similar to the value to be written. When the new value and the value to be overwritten are similar, this means that the number of bit flips is going to be lower. Reducing the number of bit flips increases write endurance and reduces power consumption. This approach is called memory awareness.

4.2.1 Content-aware methods

The methods in this group, by being aware of the memory content, try to redirect write requests to overwrite specific memory locations based on their content-aware replacement policies to decrease the energy consumption of the system and extend the lifetime of the device. For example, the authors in [102] proposes an encoded content-aware cache replacement policy to reduce the total switched bits in spin-torque magnetic random-access memory (STT-MRAM) caches. To do this, instead of replacing the LRU block, the victim block is chosen among the blocks whose contents are most similar to the missed one. To avoid the comparison of the entire 512 bits of the blocks, each block is encoded using 8 bits, which incur low space overhead. The reason behind this encoding is that when the contents of two blocks are dominated by certain bit value, there is a good chance that the content similarity of the two blocks is high, hence may lower the switch bits when one double word replaces the other [102].

Data Content-aware (DATACON) [66] is a recent mechanism that reduces the latency and energy of PCM writes by redirecting the write requests to a new physical address within memory to overwrite memory locations containing all-zeros or all-ones depending on the content of the incoming writes. DATACON is implemented inside the memory controller. To keep track of the all-zeros and all-ones memory locations and their address translations, the memory controller needs to maintain a table. Although this method takes advantage of being content aware, it needs to be implemented inside the existing memory controller, which is non-trivial task. Moreover, the average number of bit flips it can save is highly dependent on the workload since it uses two fixed patterns to save bit flips.

In [23], the authors modifies the common data structures based on the idea of pointer distance to minimize the number of bit flips on PCMs. In this method, instead of building a doubly-linked list, for instance, XOR linked lists are used, which allows each node to store only the XOR between the previous and next node instead of storing the previous and next nodes. The results show that storing the XOR of two pointers, which are likely to contain similar higher-order bits, reduces the number of bit flips, which can lead to reducing power consumption.

Another method that leverages memory awareness in its structure is called Hamming-Tree [26], which is an auxiliary data structure that can be augmented with existing indexes. Hamming-Tree is a data structure that organizes free memory locations based on their hamming distance. It can be built upon any existing tree-based data structure—whether they are designed for PCM or not—to improve their performance in terms of PCM write endurance. One of the unique qualities of this method, which makes it highly adoptable by any existing key/value stores, is its ability to be augmented with a data indexing structure from B+-tree to LSM-based persistent K/V stores to cache optimized PCM index, and write-friendly hashing schemes. In this method, the data indexing structure handles the regular indexing of keys and values, and Hamming-Tree handles the mapping of free memory locations for future writes and updates. This method also reduces bit flipping considerably.

4.2.2 AI-based methods

Machine learning and deep learning are changing the world by transforming all segments from power systems to transport to storage systems and database systems [117–119]. Using machine learning in the field of PCMs is not new, but utilizing a machine learning-based method to extend the lifetime of PCMs was introduced in [25].

Predict and Write (PNW) [25] is a memory-aware mechanism that uses machine learning to extend the lifetime of PCMs. PNW is a K/V store that is designed specifically for PCMs. This method uses a clustering-based approach to extend the lifetime of PCMs using machine learning. Writes are directed to clusters with similar content to reduce the number of bit flips. Like the previous methods in Sect. 4.1, PNW also targets bit flip reduction but through software techniques. PNW decreases the number of bit flips for PUT/UPDATE operations by determining the best memory location an updated value should be written to. This method leverages the indirection level of K/V-stores to freely choose the target memory location for any given write based on its value. In this method, PCM addresses are organized in a dynamic address pool clustered by the similarity of the data values they refer to.

This recent AI-based direction has a high potential to improve the performance of the existing wear-leveling methods that use conventional techniques such as fixed patterns. Also, this new direction is in its infancy and can be improved in many ways, such as using efficient autonomous clustering methods [117, 120] or utilize advanced deep learning methods that are capable of learning the existing patterns among the existing data dynamically [121].

5 Conclusion and future work

NVMs promise to be an indispensable part of future memory systems due to their unique characteristics such as non volatility, byte addressability, high density, high scalability, and requiring near-zero standby power. They can revolutionize the performance, energy efficiency, and processing footprint of existing systems from storage systems to edge and cloud environments to distributed database systems and blockchain decentralized applications [5, 122–127]. However, their main limitations, especially limited write endurance and high write energy consumption, pose serious challenges for their full adoption, which needs to be taken into consideration to leverage their full potential.

There is an opportunity now for researchers in data management systems to adopt solutions to overcome these limitations of NVMs that would be essential for their adoption and success. Specifically, in this paper, we present the low-hanging fruits and approaches of augmenting existing techniques from the NVM storage community to be adopted in data management systems. Also, we outline future opportunities in the area of memory-awareness that promises to increase the efficacy of existing techniques to improve the lifetime and energy efficiency of PCM devices.

References

1. Hameed, F.: Efficient stt-ram last-level-cache architecture to replace dram cache. In: MEMSYS 2017, pp. 141–151 (2017)
2. Caulfield, A.M., De, A., Coburn, J., Mollow, T.I., Gupta, R.K., Swanson, S.: Moneta: A high-performance storage array architecture for next-generation, non-volatile memories. In: 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture, pp. 385–395. IEEE Computer Society, Washington DC (2010)
3. Dulloor, S.R., Kumar, S., Keshavamurthy, A., Lantz, P., Reddy, D., Sankaran, R., Jackson, J.: System software for persistent memory. In: Proceedings of the Ninth European Conference on Computer Systems, pp. 1–15 (2014)
4. Kargar, S., Nawab, F.: Extending the lifetime of nvm: challenges and opportunities. Proc. VLDB Endowment **14**(12), 3194–3197 (2021)
5. Mittal, S., Vetter, J.S.: (2015): A survey of software techniques for using non-volatile memories for storage and main memory systems. TPDS **27**(5), 1537–1550 (2015)
6. Xia, F.: A survey of phase change memory systems. J. Comput. Sci. Technol. **30**(1), 121–144 (2015)
7. Boukhobza, J., Rubini, S., Chen, R., Shao, Z.: Emerging NVM: a survey on architectural integration and research challenges. ACM Trans. Des. Autom. Electron. Syst. (TODAES) **23**(2), 1–32 (2017)
8. Agarwal, S., Kapoor, H.K.: Improving the lifetime of non-volatile cache by write restriction. IEEE Trans. Comput. **68**(9), 1297–1312 (2019)
9. Palangappa, P.M., Mohanram, K.: Compex++ compression-expansion coding for energy, latency, and lifetime improvements in mlc/tlc nvms. ACM Trans. Architect. Code Optim. (TACO) **14**(1), 1–30 (2017)
10. García, A.A., de Jong, R., Wang, W., Diestelhorst, S.: Composing lifetime enhancing techniques for non-volatile main memories. In: Proceedings of the International Symposium on Memory Systems, pp. 363–373 (2017)
11. Guo, Y., Hua, Y., Zuo, P.: Dfpc: A dynamic frequent pattern compression scheme in nvm-based main memory. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1622–1627. IEEE Computer Society, Washington DC (2018)
12. Agarwal, S., Kapoor, H.K.: Targeting inter set write variation to improve the lifetime of non-volatile cache using fellow sets. In: 2017 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC), pp. 1–6. IEEE Computer Society, Washington DC (2017)
13. Safayenikoo, P., Asad, A., Fathy, M., Mohammadi, F.: Exploiting non-uniformity of write accesses for designing a high-endurance hybrid last level cache in 3d cmps. In: 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1–5. IEEE Computer Society, Washington DC (2017)
14. Safayenikoo, P., Asad, A., Fathy, M., Mohammadi, F.: A new traffic compression method for end-to-end memory accesses in 3D chip-multiprocessors. In: 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1–4. IEEE Computer Society, Washington DC (2017)
15. Deguchi, Y., Takeuchi, K.: 3d-nand flash solid-state drive (SSD) for deep neural network weight storage of iot edge devices with 700x data-retention lifetime extention. In: 2018 IEEE International Memory Workshop (IMW), pp. 1–4. IEEE Computer Society, Washington DC (2018)

16. Shi, J., Zhang, H., Bai, Y., Han, G., Jia, G.: A novel data aggregation preprocessing algorithm in flash memory for IoT based power grid storage system. IEEE Access **6**, 57279–57290 (2018)
17. Yang, J., Lin, Y., Fu, Y., Xue, X., Chen, B.: A small area and low power true random number generator using write speed variation of oxidebased rram for iot security application. In: 2017 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–4. IEEE Computer Society, Washington DC (2017)
18. Huang, J., Hua, Y., Zuo, P., Zhou, W., Huang, F.: An efficient wear-level architecture using self-adaptive wear leveling. In: 49th International Conference on Parallel Processing-ICPP, pp. 1–11 (2020)
19. Pourshirazi, B., Beigi, M.V., Zhu, Z., Memik, G.: Writeback-aware llc management for pcm-based main memory systems. ACM Trans. Des. Autom. Electron. Syst. (TODAES) **24**(2), 1–19 (2019)
20. Akram, S., Sartor, J.B., McKinley, K.S., Eeckhout, L.: Write-rationing garbage collection for hybrid memories. ACM SIGPLAN Not. **53**(4), 62–77 (2018)
21. Wu, J., Dong, J., Fang, R., Zhang, W., Wang, W., Zuo, D.: Wdbt: Non-volatile memory wear characterization and mitigation for DBT systems. J. Syst. Softw. **187**, 111247 (2022)
22. Song, S., Das, A., Kandasamy, N.: Exploiting inter-and intra-memory asymmetries for data mapping in hybrid tiered-memories. In: Proceedings of the 2020 ACM SIGPLAN International Symposium on Memory Management, pp. 100–114 (2020)
23. Bittman, D.: Optimizing systems for byte-addressable {NVM} by reducing bit flipping. In: 17th {USENIX} Conference on File and Storage Technologies ({FAST} 19), pp. 17–30 (2019)
24. Zuo, P., Hua, Y.: A write-friendly hashing scheme for non-volatile memory systems. In: Proceedings of MSST (2017)
25. Kargar, S., Litz, H., Nawab, F.: Predict and write: Using k-means clustering to extend the lifetime of nvm storage. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 768–779. IEEE Computer Society, Washington DC (2021)
26. Kargar, S., Nawab, F.: Hamming tree: The case for memory-aware bit flipping reduction for nvm indexing. In: CIDR (2021)
27. Huang, K., Yan, Y., Huang, L.: Revisiting persistent hash table design for commercial non-volatile memory. In: 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 708–713. IEEE Computer Society, Washington DC (2020)
28. Liu, C.-Y., Lee, Y., Jung, M., Kandemir, M.T., Choi, W.: Prolonging 3D NAND SSD lifetime via read latency relaxation. In: Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 730–742 (2021)
29. Cai, Y., Lin, Y., Xia, L., Chen, X., Han, S., Wang, Y., Yang, H.: Long live time: improving lifetime and security for NVM-based training-in-memory systems. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **39**(12), 4707–4720 (2020)
30. Zhou, F., Wu, S., Jia, Y., Gao, X., Jin, H., Liao, X., Yuan, P.: Vail: A victim-aware cache policy to improve nvm lifetime for hybrid memory system. Parallel Comput. **87**, 70–76 (2019)
31. Carlos, E., Pablo, I., Teresa, M., Llaberia, J.M., Viñals, V.: L2C2: Last-level compressed-cache NVM and a procedure to forecast performance and lifetime. arXiv preprint (2022). arXiv:2204.09504
32. Rani, K., Kapoor, H.K.: Write variation aware buffer assignment for improved lifetime of non-volatile buffers in on-chip interconnects. IEEE Trans. Very Large Scale Integration (VLSI) Syst. **27**(9), 2191–2204 (2019)
33. Agarwal, S., Kapoor, H.K.: Improving the performance of hybrid caches using partitioned victim caching. ACM Trans. Embed. Comput. Syst. (TECS) **20**(1), 1–27 (2020)
34. Hakert, C., Kühn, R., Chen, K.-H., Chen, J.-J., Teubner, J.: Octo+: Optimized checkpointing of b+ trees for non-volatile main memory wear-leveling. In: 2021 IEEE 10th Non-Volatile Memory Systems and Applications Symposium (NVMSA), pp. 1–6. IEEE Computer Society, Washington DC (2021)
35. Guo, Y., Hua, Y., Zuo, P.: A latency-optimized and energy-efficient write scheme in nvm-based main memory. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **39**(1), 62–74 (2018)
36. Chu, Z., Luo, Y., Jin, P.: An efficient sorting algorithm for non-volatile memory. Int. J. Softw. Eng. Knowl. Eng. **31**(11–12), 1603–1621 (2021)
37. Lu, L., Pillai, T.S., Gopalakrishnan, H., Arpaci-Dusseau, A.C., Arpaci-Dusseau, R.H.: Wisckey: Separating keys from values in SSD-conscious storage. ACM Trans. Storage (TOS) **13**(1), 1–28 (2017)

38. Dai, Y., Xu, Y., Ganesan, A., Alagappan, R., Kroth, B., Arpaci-Dusseau, A., Arpaci-Dusseau, R.: From WiscKey to Bourbon: a learned index for log-structured merge trees. In: 14th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 20), pp. 155–171 (2020)
39. Li, J., Pavlo, A., Dong, S.: NVMRocks: RocksDB on non-volatile memory systems (2017)
40. Kannan, S.: Redesigning lsms for nonvolatile memory with novelsm. In: 2018 {USENIX} Annual Technical Conference ({USENIX} {ATC} 18), pp. 993–1005 (2018)
41. Cai, H., Wang, Y., de Barros Naviner, L.A., Yang, J., Zhao, W.: Exploring hybrid stt-mtj/cmos energy solution in near-/sub-threshold regime for IoT applications. IEEE Trans. Magn. **54**(2), 1–9 (2017)
42. Prasad, R.S., Chaturvedi, N., Gurunarayanan, S.: A low power high speed MTJ based non-volatile sram cell for energy harvesting based IoT applications. Integration **65**, 43–50 (2019)
43. Li, W.: Hilsm: an lsm-based key-value store for hybrid nvm-ssd storage systems. In: Proceedings of the 17th ACM International Conference on Computing Frontiers, pp. 208–216 (2020)
44. Liu, J.: Lb+ trees: optimizing persistent index performance on 3dxpoint memory. Proc. VLDB Endowment **13**(7), 1078–1090 (2020)
45. Balmau, O., Didona, D., Guerraoui, R., Zwaenepoel, W., Yuan, H., Arora, A., Gupta, K., Konka, P.: {TRIAD}: Creating synergies between memory, disk and log in log structured {Key-Value} stores. In: 2017 USENIX Annual Technical Conference (USENIX ATC 17), pp. 363–375 (2017)
46. Eisenman, A., Cidon, A., Pergament, E., Haimovich, O., Stutsman, R., Alizadeh, M., Katti, S.: Flashield: a hybrid key-value cache that controls flash write amplification. In: 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19), pp. 65–78 (2019)
47. Hu, J., et al.: Understanding and analysis of b+ trees on nvm towards consistency and efficiency. CCF Trans. High Perform. Comput. (2020). https://doi.org/10.1007/s42514-020-00022-z
48. Wang, Q., Li, J., Lee, P.P., Ouyang, T., Shi, C., Huang, L.: Separating data via block invalidation time inference for write amplification reduction in logstructured storage. In: Proceedings of USENIX FAST (2022)
49. Chakraborttii, C., Litz, H.: Reducing write amplification in flash by death-time prediction of logical block addresses. In: Proceedings of the 14th ACM International Conference on Systems and Storage, pp. 1–12 (2021)
50. Zhang, W., Zhao, X., Jiang, S., Jiang, H.: Chameleondb: a key-value store for optane persistent memory. In: Proceedings of the Sixteenth European Conference on Computer Systems, pp. 194–209 (2021)
51. Chen, H., Ruan, C., Li, C., Ma, X., Xu, Y.: {SpanDB}: A fast, {Cost-Effective}{LSM-tree} based {KV} store on hybrid storage. In: 19th USENIX Conference on File and Storage Technologies (FAST 21), pp. 17–32 (2021)
52. Xanthakis, G., Saloustros, G., Batsaras, N., Papagiannis, A., Bilas, A.: Parallax: hybrid key-value placement in LSM-based key-value stores. In: Proceedings of the ACM Symposium on Cloud Computing, pp. 305–318 (2021)
53. Zhang, B., Du, D.H.: NVLSM: a persistent memory key-value store using log-structured merge tree with accumulative compaction. ACM Trans. Storage (TOS) **17**(3), 1–26 (2021)
54. Kim, S., Son, Y.: Optimizing key-value stores for flash-based SSDS via key reshaping. IEEE Access **9**, 115135–115144 (2021)
55. Li, C., Chen, H., Ruan, C., Ma, X., Xu, Y.: Leveraging nvme ssds for building a fast, cost-effective, LSM-tree-based KV store. ACM Trans. Storage (TOS) **17**(4), 1–29 (2021)
56. Doekemeijer, K., Trivedi, A.: Key-value stores on flash storage devices: a survey. arXiv preprint (2022). arXiv:2205.07975
57. Conway, A., Gupta, A., Chidambaram, V., Farach-Colton, M., Spillane, R., Tai, A., Johnson, R.: {SplinterDB}: closing the bandwidth gap for {NVMe}{Key-Value} stores. In: 2020 USENIX Annual Technical Conference (USENIX ATC 20), pp. 49–63 (2020)
58. Cui, L., He, K., Li, Y., Li, P., Zhang, J., Wang, G., Liu, X.-G.: Swapkv: a hotness aware in-memory key-value store for hybrid memory systems. IEEE Trans. Knowl. Data Eng. (2021)***
59. Sun, P., Xue, D., You, L., Yan, Y., Huang, L.: Hyperkv: a high performance concurrent key-value store for persistent memory. In: 2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), pp. 125–134. IEEE Computer Society, Washington DC (2021)

60. Krishnan, R.M., Kim, W.-H., Fu, X., Monga, S.K., Lee, H.W., Jang, M., Mathew, A., Min, C.: {TIPS}: Making volatile index structures persistent with {DRAM-NVMM} tiering. In: 2021 USENIX Annual Technical Conference (USENIX ATC 21), pp. 773–787 (2021)
61. Lee, H., Lee, M., Eom, Y.I.: Partial tiering: A hybrid merge policy for log structured key-value stores. In: 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), pp. 20–23. IEEE Computer Society, Washington DC (2021)
62. Friedman, M., Petrank, E., Ramalhete, P.: Mirror: making lock-free data structures persistent. In: Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, pp. 1218–1232 (2021)
63. Shi, Y., Oh, S., Huang, Z., Lu, X., Kang, S.H., Kuzum, D.: Performance prospects of deeply scaled spin-transfer torque magnetic random-access memory for in-memory computing. IEEE Electron Device Lett. **41**(7), 1126–1129 (2020)
64. Balmau, O., Guerraoui, R., Trigonakis, V., Zablotchi, I.: Flodb: Unlocking memory in persistent key-value stores. In: Proceedings of the Twelfth European Conference on Computer Systems, pp. 80–94 (2017)
65. Safayenikoo, P., Asad, A., Fathy, M., Mohammadi, F.: An energy efficient non-uniform last level cache architecture in 3d chip-multiprocessors. In: 2017 18th International Symposium on Quality Electronic Design (ISQED), pp. 373–378. IEEE Computer Society, Washington DC (2017)
66. Song, S., Das, A., Mutlu, O., Kandasamy, N.: Improving phase change memory performance with data content aware access. In: Proceedings of the 2020 ACM SIGPLAN International Symposium on Memory Management, pp. 30–47 (2020)
67. Song, S., Das, A., Mutlu, O., Kandasamy, N.: Aging-aware request scheduling for non-volatile main memory. In: Proceedings of the 26th Asia and South Pacific Design Automation Conference, pp. 657–664 (2021)
68. Kamath, A.K., et al.: Storage class memory: principles, problems, and possibilities. arXiv preprint (2019). arXiv:1909.12221
69. Chen, L., Zhao, J., Wang, C., Cao, T., Zigman, J., Volos, H., Mutlu, O., Lv, F., Feng, X., Xu, G.H., et al.: Unified holistic memory management supporting multiple big data processing frameworks over hybrid memories. ACM Trans. Comput. Syst. (TOCS) **39**(1–4), 1–38 (2022)
70. Lai, J., Cai, J., Chu, J.: A congestion-aware hybrid SRAM and STT-RAM buffer design for network-on-chip router. IEICE Electron. Express (2022). https://doi.org/10.1587/elex.19.20220078
71. Liu, L., Yang, S., Peng, L., Li, X.: Hierarchical hybrid memory management in OS for tiered memory systems. IEEE Trans. Parallel .Distrib. Syst. **30**(10), 2223–2236 (2019)
72. Huang, T., Dai, G., Wang, Y., Yang, H.: Hyve: Hybrid vertex-edge memory hierarchy for energy-efficient graph processing. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 973–978. IEEE Computer Society, Washington DC (2018)
73. Zhao, H., Zhao, J.: Leveraging mlc stt-ram for energy-efficient cnn training. In: Proceedings of the International Symposium on Memory Systems, pp. 279–290 (2018)
74. Sun, B., Liu, D., Yu, L., Li, J., Liu, H., Zhang, W., Torng, T.: Mram co-designed processing-in-memory cnn accelerator for mobile and iot applications. arXiv preprint (2018). arXiv:1811.12179
75. Song, S., Balaji, A., Das, A., Kandasamy, N.: Design-technology co-optimization for NVM-based neuromorphic processing elements. ACM Trans. Embedded Comput. Syst. (TECS) (2022). https://doi.org/10.1145/3524068
76. Kosta, A., Soufleri, E., Chakraborty, I., Agrawal, A., Ankit, A., Roy, K.: Hyperx: A hybrid rram-sram partitioned system for error recovery in memristive xbars. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 88–91. IEEE Computer Society, Washington DC (2022)
77. Shin, D., Jang, H., Oh, K., Lee, J.W.: An energy-efficient dram cache architecture for mobile platforms with pcm-based main memory. ACM Trans. Embed. Comput. Syst. (TECS) **21**(1), 1–22 (2022)
78. Ho, C.-C., Wang, W.-C., Hsu, T.-H., Jiang, Z.-D., Li, Y.-C.: Approximate programming design for enhancing energy, endurance and performance of neural network training on nvm-based systems. In: 2021 IEEE 10th Non-Volatile Memory Systems and Applications Symposium (NVMSA), pp. 1–6. IEEE Computer Society, Washington DC (2021)
79. Kang, W., Zhang, L., Zhao, W., Klein, J.-O., Zhang, Y., Ravelosona, D., Chappert, C.: Yield and reliability improvement techniques for emerging nonvolatile STT-MRAM. IEEE J. Emerg. Sel. Top. Circuits Syst. **5**(1), 28–39 (2014)

80. Reed, E., Alameldeen, A.R., Naeimi, H., Stolt, P.: Probabilistic replacement strategies for improving the lifetimes of NVM-based caches. In: Proceedings of the International Symposium on Memory Systems, pp. 166–176 (2017)
81. Puglia, G.O.: Non-volatile memory file systems: a survey. IEEE Access **7**, 25836–25871 (2019)
82. Zahoor, F., Azni Zulkifli, T.Z., Khanday, F.A.: Resistive random access memory (RRAM): an overview of materials, switching mechanism, performance, multilevel cell (MLC) storage, modeling, and applications. Nanoscale Res. Lett. **15**(1), 1–26 (2020)
83. Akram, S.: Performance evaluation of intel optane memory for managed workloads. ACM Trans. Architect. Code Optim. (TACO) **18**(3), 1–26 (2021)
84. Alsalibi, A.I., Shambour, M.K.Y., Abu-Hashem, M.A., Shehab, M., Shambour, Q., Muqat, R.: Nonvolatile memory-based Internet of Things: a survey. In: Artificial Intelligence-based Internet of Things Systems, pp. 285–304. Springer, Cham (2022)
85. https://pmem.io. Accessed Feb 2022
86. Yang, B.-D.: A low power phase-change random access memory using a data-comparison write scheme. In: ISCAS 2007, pp. 3014–3017. IEEE Computer Society, Washington DC (2007)
87. Cho, S., Lee, H.: Flip-n-write: A simple deterministic technique to improve pram write performance, energy and endurance. In: MICRO 2009, pp. 347–357 (2009)
88. Dgien, D.B.: Compression architecture for bit-write reduction in non-volatile memory technologies. In: NANOARCH 2014, pp. 51–56. IEEE Computer Society, Washington DC (2014)
89. Jalili, M., Sarbazi-Azad, H.: Captopril: Reducing the pressure of bit flips on hot locations in non-volatile main memories. In: DATE 2016, pp. 1116–1119. IEEE Computer Society, Washington DC (2016)
90. Palangappa, P.M., Mohanram, K.: Flip-mirror-rotate: an architecture for bit-write reduction and wear leveling in non-volatile memories. In: GLSVLSI 2015, pp. 221–224 (2015)
91. Alameldeen, A., Wood, D.: Frequent pattern compression: a significance-based compression scheme for L2 caches. Technical report, University of Wisconsin-Madison Department of Computer Sciences (2004)
92. Arulraj, J.: Let's talk about storage and recovery methods for non-volatile memory database systems. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 707–722 (2015)
93. Chen, S., Jin, Q.: Persistent b+-trees in non-volatile main memory. Proc. VLDB Endowment **8**(7), 786–797 (2015)
94. Oukid, I.: Fptree: A hybrid scm-dram persistent and concurrent b-tree for storage class memory. In: Proceedings of the 2016 International Conference on Management of Data, pp. 371–386 (2016)
95. Liu, Z., Liu, T., Guo, J., Wu, N., Wen, W.: An ecc-free mlc stt-ram based approximate memory design for multimedia applications. In: 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 142–147. IEEE Computer Society, Washington DC (2018)
96. Kokolis, A., Skarlatos, D., Torrellas, J.: Pageseer: Using page walks to trigger page swaps in hybrid memory systems. In: 2019 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 596–608. IEEE Computer Society, Washington DC (2019)
97. Kültürsay, E., Kandemir, M., Sivasubramaniam, A., Mutlu, O.: Evaluating STT-RAM as an energy-efficient main memory alternative. In: 2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 256–267. IEEE Computer Society, Washington DC (2013)
98. Gleixner, B., Pellizzer, F., Bez, R.: Reliability characterization of phase change memory. In: 2009 10th Annual Non-Volatile Memory Technology Symposium (NVMTS), pp. 7–11. IEEE Computer Society, Washington DC (2009)
99. Ban, A.: Wear leveling of static areas in flash memory. Google Patents. US Patent 6,732,221 (2004)
100. Che, Y., Yang, Y., Awad, A., Wang, R.: A lightweight memory access pattern obfuscation framework for NVM. IEEE Comput. Architect. Lett. **19**(2), 163–166 (2020)
101. Qureshi, M.K., Karidis, J., Franceschini, M., Srinivasan, V., Lastras, L., Abali, B.: Enhancing lifetime and security of pcm-based main memory with start-gap wear leveling. In: 2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp. 14–23. IEEE Computer Society, Washington DC (2009)
102. Zeng, Q., Peir, J.-K.: Content-aware non-volatile cache replacement. In: 2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 92–101. IEEE Computer Society, Washington DC (2017)

103. Zhou, P., Zhao, B., Yang, J., Zhang, Y.: A durable and energy efficient main memory using phase change memory technology. ACM SIGARCH Comput. Architect. News **37**(3), 14–23 (2009)
104. Qureshi, M.K.: Scalable high performance main memory system using phase-change memory technology. In: ISCA '09, pp. 24–33 (2009)
105. Seong, N.H., Woo, D.H., Lee, H.-H.S.: Security refresh: prevent malicious wear-out and increase durability for phase-change memory with dynamically randomized address mapping. ACM SIGARCH Comput. Architect. News **38**(3), 383–394 (2010)
106. Thoziyoor, S., Ahn, J.H., Monchiero, M., Brockman, J.B., Jouppi, N.P.: A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies. ACM SIGARCH Comput. Architect. News **36**(3), 51–62 (2008)
107. Young, V., Nair, P.J., Qureshi, M.K.: Deuce: write-efficient encryption for non-volatile memories. ACM SIGARCH Comput. Architect. News **43**(1), 33–44 (2015)
108. Huang, F., Feng, D., Xia, W., Zhou, W., Zhang, Y., Fu, M., Jiang, C., Zhou, Y.: Security rbsg: Protecting phase change memory with security-level adjustable dynamic mapping. In: 2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 1081–1090. IEEE (2016)
109. Mao, H., Zhang, X., Sun, G., Shu, J.: Protect non-volatile memory from wear-out attack based on timing difference of row buffer hit/miss. In: Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, pp. 1623–1626. IEEE Computer Society, Washington DC (2017)
110. Zhou, X., Shou, L., Chen, K., Hu, W., Chen, G.: Dptree: differential indexing for persistent memory. Proc. VLDB Endowment **13**(4), 421–434 (2019)
111. Chen, Y., Lu, Y., Fang, K., Wang, Q., Shu, J.: utree: a persistent b+-tree with low tail latency. Proc. VLDB Endowment **13**(12), 2634–2648 (2020)
112. Ma, Z., Sha, E.H.-M., Zhuge, Q., Jiang, W., Zhang, R., Gu, S.: Towards the design of efficient hash-based indexing scheme for growing databases on non-volatile memory. Future Gen. Comput. Syst. **105**, 1–12 (2020)
113. Xia, F.: Hikv: A hybrid index key-value store for dram-nvm memory systems. In: USENIX ATC 17, pp. 349–362 (2017)
114. Lu, B., Hao, X., Wang, T., Lo, E.: Dash: scalable hashing on persistent memory. arXiv preprint (2020). arXiv:2003.07302
115. Luo, X.: Enhancing lifetime of nvm-based main memory with bit shifting and flipping. In: RTCSA 2014, pp. 1–7. IEEE Computer Society, Washington DC (2014)
116. Dong, W.: Minimizing update bits of nvm-based main memory using bit flipping and cyclic shifting. In: HPCC 2015, CSS 2015, and ESS 2015, pp. 290–295. IEEE Computer Society, Washington DC (2015)
117. Gu, B., Kargar, S., Nawab, F.: Efficient dynamic clustering: capturing patterns fromhistorical cluster evolution. arXiv preprint (2022). arXiv:2203.00812
118. Andalibi, M., Hajihosseini, M., Teymoori, S., Kargar, M., Gheisarnejad, M.: A time-varying deep reinforcement model predictive control for dc power converter systems. In: 2021 IEEE 12th International Symposium on Power Electronics for Distributed Generation Systems (PEDG), pp. 1–6. IEEE Computer Society, Washington DC (2021)
119. Kraska, T.: The case for learned index structures. In: SIGMOD 2018, pp. 489–504 (2018)
120. Huang, S., Kang, Z., Xu, Z., Liu, Q.: Robust deep k-means: an effective and simple method for data clustering. Pattern Recogn. **117**, 107996 (2021)
121. Sharif, A., Li, J.P., Saleem, M.A., Manogran, G., Kadry, S., Basit, A., Khan, M.A.: A dynamic clustering technique based on deep reinforcement learning for internet of vehicles. J. Intell. Manuf. **32**(3), 757–768 (2021)
122. Gazzaz, S., Chakraborty, V., Nawab, F.: Croesus: multi-stage processing and transactions for video-analytics in edge-cloud systems. arXiv preprint (2021). arXiv:2201.00063
123. Kargar, S., Mohammad-Khanli, L.: Fractal: An advanced multidimensional range query lookup protocol on nested rings for distributed systems. J. Netw. Comput. Appl. **87**, 147–168 (2017)
124. Nawab, F., Chakrabarti, D.R., Kelly, T., Morrey III, C.B.: Procrastination beats prevention: timely sufficient persistence for efficient crash resilience. In: EDBT, pp. 689–694 (2015)
125. Nawab, F.: Wedgechain: a trusted edge-cloud store with asynchronous (lazy) trust. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 408–419. IEEE Computer Society, Washington DC (2021)

126. Nawab, F., Sadoghi, M.: Blockplane: A global-scale byzantizing middleware. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 124–135. IEEE Computer Society, Washington DC (2019)
127. Gu, B., Li, Z., Liu, A., Xu, J., Zhao, L., Zhou, X.: Improving the quality of web-based data imputation with crowd intervention. IEEE Trans. Knowl. Data Eng. **33**(6), 2534–2547 (2019)