The New Hardware Development Trend and the Challenges in Data Management and Analysis

Wei Pan¹ · Zhanhuai Li¹ · Yansong Zhang² · Chuliang Weng³

Received: 28 May 2018 / Accepted: 29 August 2018 / Published online: 24 September 2018
© The Author(s) 2018

Abstract

Hardware techniques and environments underwent significant transformations in the field of information technology, represented by high-performance processors and hardware accelerators characterized by abundant heterogeneous parallelism, nonvolatile memory with hybrid storage hierarchies, and RDMA-enabled high-speed network. Recent hardware trends in these areas deeply affect data management and analysis applications. In this paper, we first introduce the development trend of the new hardware in computation, storage, and network dimensions. Then, the related research techniques which affect the upper data management system design are reviewed. Finally, challenges and opportunities are addressed for the key technologies of data management and analysis in new hardware environments.

Keywords Data management · Data analysis · Heterogeneous hardware

1 Introduction

The development of data management and analysis systems largely benefits from the advancement of the hardware and software technologies. Hardware and software are the two major elements that constitute a computing system. The software performance could directly benefit from the advancement of hardware, but it is also limited by the characteristics of the hardware. Therefore, there has to be trade-offs in the design of software frameworks and systems. In the meantime, the demands for software performance also promote the advancements and innovations of hardware technologies. For data management and analysis systems, the hardware determines the performance limit of data access and query processing. To fully utilize the hardware, the software should optimize the design of algorithms and data structure according to the hardware. The traditional compute-intensive applications and recent data-intensive applications both have placed high demands on hardware performance such as access latency, capacity, bandwidth, energy consumption

and cost performance. Under the various application workloads, the traditional data management and analysis techniques are facing unprecedented challenges. Essentially, the challenges posed by big data are caused by the contradiction where the existing data processing infrastructures are incapable of meeting the diverse demands in data processing.

Data processing infrastructure generally includes the underlying hardware environment and the upper software system. The design principles, architecture selection, core functions, strategy mode, and optimization techniques of the upper software system largely depend on the computer hardware. Today's hardware technologies and environments are undergoing dramatic changes. Specifically, the advent of high-performance processors and hardware accelerators, new nonvolatile memory (NVM), and a high-speed network are rapidly changing the base support for traditional data management and analysis systems. These new hardware is expected to break through the architecture of the entire computer system and convert the assumptions of the upper software. They also require that the architecture of data management and analysis software and related technologies have hardware awareness.

✉ Zhanhuai Li
lizhh@nwpu.edu.cn

¹ Northwest Polytechnical University, Xi'an, China

² Renmin University of China, Beijing, China

³ East China Normal University, Shanghai, China

2 The Trend of Hardware

In recent years, storage, processor, and network technologies have made a great breakthrough. As shown in Fig. 1, a growing set of new hardware, architecture, and features are becoming the foundation of the future computing platforms. The current trends indicate that these techniques are significantly changing the underlying environment of traditional data management and analysis systems, including high-performance processors and hardware accelerators, NVM, RDMA-capable (remote direct memory access) networks. Significantly, the ongoing underlying environments, marked by heterogeneous multi-core architecture and hybrid storage hierarchy, make the already complicated software design space become more sophisticated [1–4].

2.1 The Trend of Processor Technologies

The development of processor technology has gone through for more than 40 years. Its development roadmap has shifted from scale-up to scale-out, and the aim dramatically shifts away from pursuing higher clock speed and instead focuses on creating more cores per processor. According to Moore's law, pushing the computing frequency of the processor continuously is one of the most important ways to improve the performance of the computer in the era of serial computing. At the same time, lots of optimization techniques, such as the instruction-level parallelism (ILP), pipeline, prefetching, branch prediction, out-of-order instruction execution, multi-level cache, and hyper-threading, can be automatically identified and utilized by the processor and the compiler. Therefore, software can consistently and transparently enjoy free and regular performance gains. However, limited by the heat, power consumption, instruction-level parallelism, manufacturing processes, and other factors, the scale-up approach reaches the ceiling.

Fig. 1 New hardware and environment

After 2005, high-performance processor technology has entered the multi-core era and multi-core parallel processing technology has become the mainstream. But although data processing capability has been significantly enhanced in multi-core architectures, software cannot automatically gain the benefits. Instead, programmers have to transform the traditional serial programs into parallel programs, and optimize the algorithm performance for the LLC (Last Level Cache) of multi-core processors. Nowadays, the performance of multi-core processors has been significantly improved with the semiconductor technology. For example, the 14-nm Xeon processor currently integrates up to 24 cores, supporting up to 3.07 TB memory and 85 GB/s memory bandwidth. However, x86 processor still has the disadvantages of low integration, high power consumption, and high price. Also the general-purpose multi-core processors can hardly to meet the demands of the highly concurrent applications. The development of the processor is going to be specifically optimized for an application, i.e., specialized hardware accelerators.

GPU, Xeon Phi, field programmable gate array (FPGA), and the like are representative of dedicated hardware accelerators. By exploiting GPUs, Xeon Phi coprocessors, and FPGAs, parts of compute-intensive and data-intensive workload can be offloaded from the CPU efficiently. Some fundamental hardware characteristics of these accelerators are given in Table 1. There is no doubt that the processing environment within the computer system becomes more and more complicated, and correspondingly, the data management and analysis systems might try to seek diversified ways to actively adapt to new situations.

2.2 The Trend of Storage Technologies

As high-performance processors and hardware accelerator technologies develop rapidly, the performance gap between CPU and storage keeps widening year by year [5]. The “memory wall” makes the data access become a non-negligible performance bottleneck. Faced with the slow I/O capabilities of traditional secondary storage devices, data management and analysis systems have had to adopt some design strategies such as cache pools, concurrency control, and disk-oriented algorithms and data structure to mitigate or hide I/O performance gap. However, I/O bottlenecks still severely constrain the processing power of data-intensive computing.

It is especially interesting that the new storage medium represented by NVM [6] provides a potential avenue to break the I/O bottleneck. The NVM is actually a general term for a type of storage technology which does not represent a specific storage technology or medium. It is also referred to as storage class memory (SCM) in some research literature [7]. Typically, NVMs include phase change memory (PCM),

Table 1 Processor characters

Type	Xeon E7-8890 V4	Xeon Phi 7290F	Xeon Phi 724P	NVIDIA Tesla V100
#core/#thread	24/48	72/288	68/272	5120 CUDA cores/640 tensor cores
Core frequency	2.20 GHz	1.50 GHz	1.3 GHz	1.455 GHz
Memory capacity	3.07 TB	384 GB	16 GB	16 GB HBM2 VRAM
Cache capacity	60 MB L3	36 MB L2/16 GB HBM	34 MB L2	6 MB L2
Memory type	DDR4-1866/4 channels	DDR4-2400/6 channels	MCDRAM/16 channels	HBM2
Memory bandwidth (GB/s)	85	115.2	500	900
Price	\$7174.00	\$3368.00	\$3324.00	\$149,000.00

magnetoresistive random access memory (MRAM), resistive random access memory (RRAM), and ferroelectric RAM (FeRAM). Although the characteristics and manufacturing processes of these memories are obviously different, they generally have some common features, including durability, high storage density, low-latency random read/write, and fine-grained byte addressing. The specifications are given in Table 2. From a performance point of view, NVM is close to the DDR memory, but also has a nonvolatile feature. Therefore, it may gradually become the main storage device, while DDR memory is used as a temporary data cache. At present, flash memory technology is today a mature technology. Take a single PCIe flash memory for example. Its capacity can reach up to 12.8 TB, and read/write performance is also high. Based on this, it can be a cache between RAM and hard disk and also can be an alternative of the hard drive as a persistent storage device. In terms of energy consumption, DRAM consumes less energy under high load. On the contrary, it consumes more energy under low load than other storage devices because refreshing the entire DRAM is required. The common feature of the NVM is that they have dual capabilities of both DRAM-like high-speed access and disk-like persistence, effectively breaking the “performance wall” of traditional storage medium that cannot overcome.

At the same time, the development of new storage technologies has also had a significant impact on processor technology. The 3D stacking technology that enhances higher

bandwidth can be applied to the on-board storage of many-core processors, delivering high-performance data cache support for the powerful parallel processing. With the NVM technology, the multi-level hybrid storage environment will certainly break the balance among the CPU, main memory, system bus, and external memory in the traditional computer architecture. It will also change the existing storage hierarchy and optimize data access critical paths to bridge the performance gap between storage tiers, providing new opportunities for data management and analytics [8].

2.3 The Trend of Network Technologies

In addition to the local storage I/O bottleneck, the network I/O bottleneck is also the main performance issue in the datacenter. Under traditional Ethernet network, the limited data transmission capability and the non-trivial CPU overhead of the TCP/IP stack have severely impacted the performance of distributed data processing. Therefore, the overall throughput of distributed database system is sharply reduced under the influence of the high proportion of distributed transactions, which lead to potentially heavy network IO. Based on this, the existing data management systems have to resort to some specific strategies such as coordinated partitioning, relaxed consistency assurance, and deterministic execution scheme to control or reduce the ratio of distributed transactions. However, most of these measures suffer

Table 2 Performance metrics of different storage devices

Attribute	DRAM	PCM (25 nm)	MRAM	RRAM	FeRAM	Flash SSD	HDD
Nonvolatile	No	Yes	Yes	Yes	Yes	Yes	Yes
Density ($\mu\text{m}^2/\text{bit}$)	0.00380	0.00250	0.00740	0.00580	0.00355	0.00210	0.00006
Read latency (ns)	55	48	20	116	55	25,000	3,000,000
Write latency (ns)	55	150	20	145	55	200,000	3,000,000
Read power consumption (pJ/bit)	12.5	2	0.02	4.81	2.13	250	2500
Write power consumption (pJ/bit)	12.5	19.2	0.03	13.8	10.12	250	2500
Lifespan	$> 10^{15}$	10^8	10^{15}	10^8	10^{14}	10^4	$> 10^{15}$

from unreasonable assumptions and applicable conditions, or opaqueness to application developers. In particular, the scalability of the system is still greatly limited, especially when the workload does not have the distinguishable characteristics to be split independently.

It is important to note that the increased contention likelihood is the most cited reasons when discussing the scalability issue of distributed transactions, but in [9], the author showed that the most important factor is the CPU overhead of the TCP/IP stack incurred by traditional Ethernet network. In other words, software-oriented optimization will not fundamentally address the scalability issue within distributed environments. In recent years, the high-performance RDMA-enabled network is dramatically improving the network latency and ensures that users can bypass the CPU when transferring data on the network. InfiniBand, iWARP, and RoCE are all RDMA-enabled network protocols, with appropriate hardware that can accelerate operations to increase the value of application. With price reduction in RDMA-related hardware, more and more emerging industry clusters working on the RDMA-related network environment, requiring a fundamental rethinking of the design of data management and analysis systems, include but not limited to distributed query, transaction processing, and other core functions [10].

Although the development of new hardware exhibits complicated variety and the composition of the hardware environment is also uncertain, it is foreseeable that they eventually will become the standard hardware components in the future. Data management and analysis on modern hardware will become a new research hotspots field.

3 Research Status

The new hardware will change the traditional computing, storage, and network systems and put a directly impact on the architecture and design scheme of the data management and analysis systems. It will also pose challenges on the core functionalities and related key technologies including indexing, analysis, and transaction processing. In the next part, we will introduce the present state of domestic and international relevant research.

3.1 System Architecture and Design Schemes in Platforms with New Hardware

The advent of high-performance processors and new accelerators has led to the shift from single-CPU architectures systems to heterogeneous, hybrid processing architectures. Data processing strategies and optimization techniques have evolved from standardization to customization and from software optimization to hardware optimization.

From the classical iterative pipeline processing model [11] to the column processing model [12] and finally to the vector processing model optimization [13] that is the combination of the former two, the traditional software-based data processing model has reached a mature stage. Though, the advent of JIT real-time compilation techniques [14] and optimization techniques combined with vector processing models [15] provides new optimization space at the register level. However, with the deepening and development of research, software-based optimization techniques are gradually touching their “ceiling.” Academics and industry are beginning to explore some new ways to accelerate the performance of data processing through software/hardware co-design. Instruction-level optimization [16], coprocessor query optimization [17, 18], hardware customization [19], workload hardware migration [20], increasing hardware-level parallelism [21], hardware-level operators [22], and so on are used to provide hardware-level performance optimization. However, the differences between the new processor and x86 processor fundamentally change the assumptions of traditional database software design on hardware. Databases are facing with more complicated architectural issues on heterogeneous computing platforms. In the future, the researcher needs to break the software-centric design idea which is effective for the traditional database systems.

The design of traditional databases has to trade-off among a number of important factors such as latency, storage capacity, cost effectiveness, and the choice between volatile and nonvolatile storage devices. The unique properties of NVM bring the opportunity to the development of data management systems, but it also introduces some new constraints. The literature [23] conducted a forward-looking research and exploration in this area. The results show that neither the disk-oriented systems nor the memory-oriented systems are ideally suited for NVM-based storage hierarchy, especially when the skew in the workload is high. The authors also found that storage I/O is no longer the main performance bottleneck in the NVM storage environment [24]; instead, a significant amount of cost on how to organize, update, and replace data will become new performance bottlenecks. For NVM-based storage hierarchy, WAL (Write-Ahead Logging) and logical logs that are common in the traditional database also have a large variety of unnecessary operations [25]. These issues indicate that diversified NVM-based storage hierarchy will lead to the new requirements for cache replacement [26], data distribution [27], data migration [28], metadata management [29], query execution plan [30], fault recovery [25], and other aspects to explore corresponding design strategies to better adapt to new environments. Therefore, NVM-specific or NVM-oriented architecture designed to utilize the nonvolatile property of NVM is necessary. The research about this area is just beginning; CMU’s N-Store

[31] presents exploratory research on how to build prototype database system on NVM-based storage hierarchy.

RDMA-enabled network is changing the assumption in traditional distributed data management systems in which network I/O is the primary performance bottleneck. Some systems [32, 33] have introduced RDMA inside, but they just carry out some add-on optimizations for RDMA later; the original architecture is obsolete. As a result, they cannot take full advantage of the opportunities presented by RDMA. It has been demonstrated [34] that migrating a legacy system to an RDMA-enabled network simply cannot fully exploit the benefits of the high-performance network; neither the shared-nothing architecture nor the distributed shared-memory architecture can bring out the full potential of RDMA. For a shared-nothing architecture, the optimization goal is to maximize data localization, but the static partitioning technique [35] or dynamic partitioning strategy does not fundamentally resolve the problem of frequent network communication [36]. Even with IPoIB (IP over Infiniband) support, for shared-nothing architecture, it is difficult to gain the most improvement on data-flow model and control-flow model simultaneously [9]. Similarly, for distributed shared-memory architectures, there is no built-in support for cache coherency; accessing cache via RDMA could have significant performance side effects if mismanaged from client [37]. At the same time, garbage collection and cache management might also be affected by it. In the RDMA network, the uniform abstraction for remote and local memory has proven to be inefficient [38]. Research [9] shows that the emerging RDMA-enabled high-performance network technologies necessitate a fundamental redesign of the way we build distributed data management system in the future.

3.2 Storage and Indexing Techniques in Platforms with New Hardware

Since NVM can serve as both internal and external memory, the boundaries between the original internal and external memory are obscured, making the composition of the underlying NVM storage diverse. Because different NVMs have their own features on access delay, durability, etc., it is theoretically possible to replace traditional storage medium without changing the storage hierarchy [39–41] or mix with them [42, 43]. In addition, NVMs also can enrich the storage hierarchies as a LLC [44] or as a new cache layer between RAM and HDD [45], which further reduce read/write latency across storage tiers. The variety of the storage environment also places considerable complexity in implementing data management and analysis techniques.

From data management perspective, how to integrate NVM into the I/O stack is a very important research topic. There are two typical ways to abstract NVM, as persistence heap [46] or as file system [47]. Because NVM can exchange

data directly with the processor using a memory bus or a dedicated bus, memory objects that do not need to be serialized into disk can be directly created with heap. Typical research works include NV-heap [46], Mnemosyne [48], and HEAPO [49]. In addition, the NVM memory environment also brings a series of new problems to be solved, such as ordering [47], atomic operations [50], and consistency guarantee [51]. In contrast, file-based NVM abstraction can take advantage of the semantics of existing file systems in namespaces, access control, read–write protection, and so on. But, in the design, in addition to take full advantage of the fine-grained addressing and in-place update capability of NVM, the impact of frequent writes on NVM lifetime needs to be considered [52]. Besides, file abstraction has a long data access path which implies some unnecessary software overhead [53]. PRAMFS [54], BPFS [47], and SIMFS [55] are all typical NVM-oriented file systems. Whether abstracting the NVM in the way of persisting memory heap or file system can become the basic building block for the upper data processing technology [56]. However, they can only provide low-level assurance on atomicity. Some of the high-level features (such as transaction semantics [57], non-volatile data structures [58], etc.) also require corresponding change and improvement in upper data management system.

Different performance characteristics of NVMs will affect the data access and processing strategies of the heterogeneous processor architecture. On the contrary, the computing characteristics of the processors also affect the storage policy. For example, under the coupled CPU/GPU processor architecture, the data distribution and the exchange should be designed according to the characteristics of low-latency CPU data access and the large-granularity GPU data processing. In addition, if NVM can be used to add a large-capacity, low-cost, high-performance storage tier under traditional DRAM, hardware accelerators such as GPU can access NVM storage directly across the memory through dedicated data channel, reducing the data transfer cost in traditional storage hierarchies [59]. We need to realize that hybrid storage and heterogeneous computing architecture will exist for a long time. The ideal technical route in the future is to divide the data processing into different processing stages according to the type of workload and to concentrate the computing on the smaller data set to achieve the goal of accelerating critical workloads through hardware accelerators. It is the ideal state to concentrate 80% of the computational load on 20% of the data [60], which simplifies data distribution and computing distribution strategies across hybrid storage and heterogeneous computing platforms.

In addition to improving data management and analytics support at the storage level, indexes are also key technologies for efficiently organizing data to accelerate the performance of upper-level data analytics. Traditional local indexing and optimization techniques based on B+ tree, R-tree,

or KD-tree [61–63] are designed for the block-level storage. Due to the significant difference between NVM and disk, the effectiveness of the existing indexes can be severely affected in NVM storage environments. For NVM, the large number of writes caused by index updates not only reduces their lifespan but also degrades their performance. To reduce frequent updates and writes of small amounts of data, merge updates [64] or late updates [65], which are frequently used on flash indexes, are typical approaches. Future indexing technologies for NVM should be more effective in controlling the read and write paths and impact areas that result from index updates, as well as enabling layered indexing techniques for the NVM storage hierarchies. At the same time, concurrency control on indexes such as B+ tree also shows obvious scalability bottlenecks in highly concurrent heterogeneous computing architectures. The coarse-grained range locks of traditional index structures and the critical regions corresponding to latches are the main reasons for limiting the degree of concurrency. Some levels of optimization techniques such as multi-granularity locking [66] and latch avoidance [67] increase the degree of concurrency of indexed access updates, but they also unavoidably introduce the issue of consistency verification, increased transaction failure rates, and higher overhead. In the future, the indexing technology for highly concurrent heterogeneous computing infrastructure needs a more lightweight and flexible lock protocol to balance the consistency, maximum concurrency, and lock protocol simplicity.

3.3 Query Processing and Optimization in Platforms with New Hardware

The basic assumption of the traditional query algorithms and data structures on the underlying storage environment does not stand in NVM storage environment. Therefore, the traditional algorithms and data structures are difficult to obtain the ideal effect in the NVM storage environment.

Reducing NVM-oriented writes is a major strategy in previous studies. A number of technologies, which include unnecessary write avoiding [39], write cancelation and write pausing strategies [68], dead write prediction [69], cache coherence enabled refresh policy [70], and PCM-aware swap algorithm [71], are used to optimize the NVM writes. With these underlying optimizations for drivers, FTL, and memory controller, the algorithms can directly benefit, but algorithms can also be optimized from a higher level. In this level, there are two ways to control or reduce NVM writes: One is to take advantage of extra caches [72, 73] to mitigate NVM write requests with the help of DRAM; the other is to utilize the low-overhead NVM reads and in-time calculations to waive the costly NVM writes [58, 74]. For further reducing NVM writes, even parts of constraints on data structures or algorithms [74–76] can be appropriately

relaxed. In the future, how to design, organize, and operate write-limited algorithms and data structures is an urgent question. However, it is important to note that with NVM asymmetric read/write costs, the major performance bottlenecks have shifted from the ratio of sequential and random disk I/O to the ratio of NVM read and write. As a result, previous cost models [77] inevitably fail to characterize access patterns of NVM accurately. Furthermore, heterogeneous computing architectures and hybrid storage hierarchy will further complicate the cost estimation in new hardware environments [30, 73]. Therefore, how to ensure the validity and correctness of the cost model under the new hardware environment is also a challenging issue. In the NVM storage environment, the basic design principle for NVM-oriented algorithms and data structures is to reduce the NVM write operations as much as possible. Put another way, write-limited (NVM-aware, NVM-friendly) algorithms and data structures are the possible strategy.

From the point of view of processor development, query optimization technologies have gone through several different stages with the evolution of hardware. During different development stages, there are significant differences in query optimization goals, from mitigating disk-oriented I/O to designing cache-conscious data structures and access methods [78–81] and developing efficient parallel algorithm [40, 82, 83]. Nowadays, the processor technology moves from multi-core to many-core which greatly differs from the multi-core processor in terms of core integration, number of threads, cache structure, and memory access. The object that should be optimized has been turned into SIMD [84, 85], GPUs, APUs, Xeon Phi coprocessors, and FPGAs [18, 86–88]. The query optimization is becoming more and more dependent on the underlying hardware. But current query optimization techniques for new hardware are in an awkward position: lacking the holistic consideration for evolving hardware, the algorithm requires constant changes to accommodate different hardware features, from predicate processing [89] to join [90] to index [91]. From a perspective of the overall architecture, the difficulty of query optimization is further increased under new hardware architecture.

In optimization techniques for join algorithm, a hot research topic in recent years is to explore whether hardware-conscious or hardware-oblivious algorithm designs are the best choices for new hardware environments. The goal of hardware-conscious algorithms is the pursuit of the highest performance, whose guiding ideology is to fully consider the hardware-specific characteristics to optimize join algorithms; instead, the goal of hardware-oblivious algorithms is the pursuit of generalizing and simplify, whose guiding principle is to design the join algorithm based on the common characteristics of the hardware. The competition between the two technology routes has intensified in recent years, from the basic CPU platform [92] to the NUMA platform

[93], and it will certainly be extended to the new processor platforms in the future. The underlying reason behind this phenomenon is that it is difficult to quantify the optimization techniques, as in the field of in-memory database technology, although there are numerous hash structures and join algorithms currently [94–96], the simple question of which is the best in-memory hash join algorithm is still unknown [97]. In the future, when the new hardware environment is more and more complicated, performance should not be the only indicator to evaluate the advantages and disadvantages of algorithm. More attention should be paid to improving the adaptability and scalability of algorithms on heterogeneous platforms.

3.4 Transaction Processing in Platforms with New Hardware

Recovery and concurrency control are the core functions of transaction processing in DBMS. They are closely related to the underlying storage and computing environment. Distributed transaction processing is also closely related to the network environment.

WAL-based recovery methods [98] can be significantly affected in NVM environments. First, because the data written on the NVM are persistent, transactions are not forced to be stored to disk when submitted, and the design rules for flush-before-commit in the WAL are broken. Moreover, because of NVM high-speed random read/write capabilities, the advantages of the cache turn into disadvantages. In extreme cases, transactional update data can also have significant redundancies in different locations (log buffers, swap areas, disks) [25]. The NVM environment not only has an impact on the assumptions and strategies of WAL, but also brings some new issues. The way to ensure the atomic NVM write operation is the most fundamental problem. Through some hardware-level primitives [25] and optimization on processor cache [50], this problem can be guaranteed partially. In addition, due to the effect of out-of-order optimization in modern processor environments, there is a new demand in serializing the data written into NVM to ensure the order of log records. Memory barriers [47] became the main solution, but the read/write latencies caused by memory barriers in turn degrade the transactional throughput based on WAL. Thus, the introduction of NVM changes the assumptions in the log design, which will inevitably introduce new technical issues.

There is a tight coupling between the log technology and NVM environment. In the case of directly replacing the external storage by NVM, the goal of log technology optimization is to reduce the software side effects incurred by ARIES-style logs [99]. In the NVM memory environment, it can be further subdivided into different levels, including hybrid DRAM/NVM and NVM only. Currently there are

many different optimization technologies on NVM-oriented logging, including two-layer logging architecture [56], log redundancies elimination [100], cost-effective approaches [101], decentralized logging [102] and others. In some sense, the existing logging technologies for NVM are actually stop-gap solutions [102]. For future NVM-based systems, the ultimate solution is that to develop logging technology on pure-NVM environment, where the entire storage system, including the multi-level cache on processor, will consist of NVM. Current research generally lacks attention on the durability of NVM; thus, finding a more appropriate trade-off among high-speed I/O, nonvolatility, and poor write tolerance is the focus of future NVM logging research.

Concurrent control effectively protects the isolation property of transactions. From the execution logic of the upper layers of different concurrency control protocols, it seems that the underlying storage is isolated and transparent. But in essence, the specific implementation of concurrency control and its overhead ratio in system are closely related to the underlying storage environment. Under the traditional two-tier storage hierarchy, the lock manager for concurrency control in memory is almost negligible, because the disk I/O bottlenecks are predominant. But in NVM storage environment, with the decrease in the cost of disk I/O, the memory overhead incurred by lock manager becomes a new bottleneck [103]. In addition, with the multi-core processors, the contradiction between the high parallelism brought by rich hardware context and the consistency of data maintenance will further aggravate the complexity of lock concurrency control [104–106]. The traditional blocking strategies, such as blocking synchronization and busy-waiting, are difficult to apply [107]. To reduce the overhead of concurrency control, it is necessary to control the lock manager's resource competition. The research mainly focuses on reducing the number of locks with three main approaches, including latch-free data structures [108], lightweight concurrency primitives [109], and distributed lock manager [110]. In addition, for MVCC, there is a tight location coupling between the index and the multi-version records in the physical storage. This will result in serious performance degradation when updating the index. In a hybrid NVM environment, an intermediate layer constructed by low-latency NVM can be used to decouple the relationship between the physical representation and the logical representation of the multi-version log [111]. This study that builds a new storage layer to ease the bottleneck of reading and writing is worth learning.

The approach to improving the extensibility of distributed transactions has always been the central question in building distributed transactional systems. On the basis of a large number of previous studies [112–114], the researchers have already formed a basic consensus that it is difficult to guarantee the scalability of systems with a large number of distributed transactions. Therefore, research focuses

on how to avoid distributed transactions [115, 116] and to control and optimize the proportion of distributed transactions [117–119]. Most of these technologies are not transparent to application developers and need to be carefully controlled or preprocessed at the application layer [120]. On the other hand, a large number of studies are also exploring how to deregulate strict transactions semantic. Under this background, the paradigm of data management is also shifting from SQL, NoSQL [121] to NewSQL [122, 123]. This development once again shows that, for a large number of critical applications, it is impossible to forego the transaction mechanism even with requirements of scalability [124]. However, these requirements are hard to meet in the traditional network. In traditional network environments, limited bandwidth, high latency, and overhead make distributed transactions not scalable. With RDMA-enabled high-performance network, the previously unmanageable hardware limitations and software costs are expected to be fully mitigated [38]. The RDMA-enabled network addresses the two most important difficulties encountered with traditional distributed transaction scaling: limited bandwidth and high CPU overhead in data transfers. Some RDMA-aware data management systems [32] have also emerged, but such systems are primarily concerned with direct RDMA support at the storage tier, and transaction processing is not the focus. Other RDMA-aware systems [125] focus on transactional processing, but they still adopt centralized managers that affect the scalability of distributed transactions. In addition, although some data management systems that fully adopt RDMA support distributed transactions, they have only have limited consistent isolation levels, such as serialization [38] and snapshot isolation [9, 34]. Relevant research [9] shows that the underlying architecture of data management should to be redesigned, such as the separation of storage and computing. Only in this way, it is possible to fully exploit all the advantages of RDMA-enabled networks to achieve fully scalable distributed transactions.

The new hardware and environment has many encouraging and superior features, but it is impossible to automatically enjoy the “free lunch” by simply migrating existing technologies onto the new platform. Traditional data management and analysis techniques are based on the x86 architecture, two-tier storage hierarchy, and TCP/IP-Ethernet. The huge differences from heterogeneous computing architectures, nonvolatile hybrid storage environments, and high-performance networking systems determine that the traditional design principles and rules of thumb are difficult to apply. In addition, in the new hardware environment, inefficient or useless components, and technologies in traditional data management and analysis systems also largely limit the efficiency of the hardware. Meanwhile, under the trend of diversified development of the hardware environment, there lacks the corresponding architectures and technologies. In

the future, it is necessary to research with the overall system and core functions. In view of the coexistence of traditional hardware and new hardware as well as the common problems of extracting and abstracting differentiated hardware, the future research will be based on perception, customization, integration, adaptation, and even reconstruction in studying the appropriate architecture, components, strategies, and technologies, in order to release the computing and storage capacity brought by the new hardware.

4 Research Challenges and Future Research Directions

4.1 Challenges

Software’s chances of being any more successful depend on whether they can accurately insight into holistic impacts on the system design; define performance bounds of the hardware; put forward new assumptions on new environment; and seek for the good trade-off. These are all the challenges that data management and analytics systems must cope with.

1. Firstly, at the system level, new hardware and environment have a systemic impact on existing technologies. This may introduce new performance bottlenecks after eliminating existing ones. Therefore, it is necessary to examine their impact in a higher-level context. In the heterogeneous computing environment composed by new processors and accelerators, although the insufficiency of large-scale parallel capabilities can be offset, the problems of the memory wall, von-Neumann bottleneck, and energy wall may become even worse in the new heterogeneous computing environment. The communication delay between heterogeneous processing units, limited cache capacity, and the non-uniform storage access cost may become a new performance problem. In the environment of new nonvolatile storage, the restrictions in disk I/O stack can be eliminated, but the new NVM I/O stack will significantly magnify the software overhead that is typically ignored in traditional I/O stack. Therefore, redesigning the software stack to reduce its overhead ratio has become a more important design principle. In high-performance network architectures, while network I/O latency is no longer a major constriction in system design, the efficiency of processor caches and local memory becomes more important.
2. Secondly, the design philosophy of the algorithm and data structure in the new hardware environment needs to be changed. Directly migrating or partially tuning algorithms and data structures cannot fully exploit the characteristics of the new hardware. At the processor level,

data structures and cache-centric algorithms designed for x86-based processors are not designed to match the hardware features of the compute-centric many-core processor. Many mature query processing techniques may fail in platforms with many-core processors. Database has long been designed based on the ideas of serial and small-scale parallel processing-based programming. This makes the traditional query algorithms difficult to convert to a large-scale parallel processing mode. At the storage level, although the NVM has both the advantages of internal and external memory, NVM also has some negative features such as asymmetric read/write costs and low write endurance. These features are significantly different with traditional storage environment, so previous studies on memory, disk, and flash cannot achieve ideal performance within a new storage hierarchy containing NVM. At the network level, the RDMA-enabled cluster is a new hybrid architecture, which is distinct from message-passing architecture or shared-memory architecture. Therefore, the technologies in the non-uniform memory access architecture cannot be directly applied to the RDMA cluster environment.

3. Thirdly, the impact of new hardware and environments on data management and analytics technologies is comprehensive, deep, and crosscutting. Due to the new features of the new hardware environment, the functionalities of data management systems cannot be tailored to adapt to the new hardware environment. In heterogeneous computing environments with new processors and accelerators, the parallel processing capacity is greatly improved. However, the richer hardware contexts also pose stringent challenges on achieving high throughput and maintaining data consistency. The impact of NVM on the logging will fundamentally change the length of the critical path of transactions. The reduction in the transaction submission time will result in lock competition, affecting the overall system concurrency capacity and the throughput. With low-latency and high-bandwidth, high-performance networks will change the system's basic assumptions that distributed transactions are difficult to extend. This is also true with the optimization objective of minimizing network latency when designing distributed algorithms. Cache utilization in multi-core architectures will become the new optimization direction. In addition, some of the existing data management components have composite functions. For example, the existing buffer not only relieves the I/O bottleneck of the entire system, but also reduces the overhead of the recovery mechanism. A more complicated scenario is the crosscutting effect between the new hardware and the environment. For example, the out-of-order instruction execution techniques in processors can cause the cached data to be not accessed and executed in the application's

logical order. If a single NVM is used to simplify traditional storage tiers, the data serializing on the NVM must be addressed.

4. Finally, under the new hardware environment, hardware/software co-design is the inevitable way for the data management and analysis system. The new hardware technology has its inherent advantages and disadvantages, which cannot completely replace the original hardware. For a long time, it is inevitable that the traditional hardware and new hardware coexist. While providing diversified hardware choices, this also leads to more complicated design strategies, more difficult optimization techniques and more difficult performance tuning space. In a heterogeneous computing environment, using coprocessor or co-placement to achieve customized data processing acceleration has created significant differences in system architecture. Moreover, the threshold of parallel programming has become increasingly high, and the gap between software and hardware is becoming larger than ever before. In some cases, the development of software lags behind hardware. In many applications, the actual utilization rate of hardware is well below the upper limit of the performance [126]. However, the new memory devices have significant differentiation and diversification. There is great flexibility and uncertainty in making use of the new nonvolatile memory to construct a NVM environment. Whether the components constitute a simple or a mixed form, and whether the status is equivalent or undetermined, the upper data management and analysis technology has also brought great challenges to research. In high-performance network system, although InfiniBand has considered RDMA from the beginning, the traditional Ethernet has also proposed the solution to support RDMA. There is no exact answer at present on which kind of plan can form the intact industry ecology at the end. Therefore, it is even more necessary to conduct cutting-edge research as soon as possible to explore a new data management architecture suitable for high-performance network environments.

4.2 Future Research

New hardware environments such as heterogeneous computing architectures, hybrid storage environments, and high-performance network will surely change the base support of traditional data management and analysis systems. It will bring significant opportunities for the development of key technologies. The future of research can start from the following aspects.

1. *Lightly coupled system architecture and collaborative design scheme* The computing, storage, and networking environments have heterogeneity, diversity, and hybridism. The different environmental components have a significant impact on the design of the upper data management system architecture. To effectively leverage the power of new hardware, the seamless integration of new hardware into the data management stack is an important fundamental research topic. To be compatible with diverse hardware environments and to reduce the risk of failure in highly coupled optimization techniques with specific hardware, the different heterogeneous, hybrid hardware environments must be effectively abstracted and virtualized. Abstraction technology can extract common features for hardware, reduce the low-level over coupling while ensuring hardware awareness, and provide flexible customization and service support for upper-layer technologies. In the meantime, the execution cost and the proportions of different operations will change under the new hardware environment, and the bottleneck of the system is also shifting. As a result, the negligible overhead in traditional software stacks would be significantly magnified. Therefore, based on this, new performance bottlenecks need to be found, a reasonable software stack needs to be redesigned, and the software overhead in the new hardware environment needs to be reduced. In addition, the new hardware environment has advantages including low latency, high capacity, high bandwidth, and high speed read and write. This has brought new opportunities for development of the integration of OLTP and OLAP system and achieving convergent OLTP system.
2. *Storage and index management with mixed heterogeneous hardware environments* Because of both internal and external memory capabilities, the new nonvolatile memory obscures the clear boundaries between existing storages. This also offers a considerable degree of freedom for the construction of the new nonvolatile storage environment and storage methods. The method can provide a powerful guarantee for accelerating the processing of data in the upper layer. Although the high-speed I/O capabilities of NVM offer opportunities for enhancing data access performance, NVM only ensures nonvolatility only at the device level. At the system level, however, the caching mechanisms may introduce the inconsistency issue. Therefore, in the future, collaborative technology needs to be studied at different levels such as architecture, strategy, and implementation. In addition, as a dedicated acceleration hardware, FPGA has its own unique advantages in accelerating data processing. In particular, the combination of NVM features can further enhance the data processing efficiency. Therefore, with the data storage engine optimization and reconstruction techniques as well as the data access accelerating and data filtering technology in the FPGA, the preprocessing on the original data can be effectively completed. This will reduce the amount of data to be transferred, thereby alleviating bottlenecks of data access in large-scale data processing. Moreover, the NVM environment has a richer storage hierarchy, while new processor technology also provides additional data processing capabilities for indexing. Therefore, multi-level and processor-conscious indexing technology is also a future research direction.
3. *Hardware-aware query processing and performance optimization* Query processing is the core operation in data analysis, involving a series of complex activities in the data extraction process. The high degree of parallelism and customizable capabilities provided by heterogeneous computing architectures, as well as the new I/O features of NVM environments, make previous query processing and optimization mechanisms difficult to apply. Future research may focus on two aspects. One is query optimization technology in NVM environment: high-speed NVM read and write, byte addressable, asymmetric read and write, and other features. It will exert a significant impact on traditional query operations such as join, sort, and aggregation. At the same time, NVM has changed the composition of traditional storage hierarchy and also affected the traditional measurement hypothesis of estimating query cost, which is based on the cost of disk I/O. Therefore, it is necessary to study the cost model in NVM environment and the design and optimization of write-limited algorithms and data structures, so that the negative impact of NVM write operations is alleviated. On the other hand, query optimization technology under heterogeneous processor platforms: the introduction of a new processor increases the dimension of heterogeneous computing platforms, resulting in the increased complexity of the query optimization techniques. This poses a huge challenge to the design of query optimizers. Collaborative query processing technology, query optimization technology, and hybrid query execution plan generation technology are all ways to improve the query efficiency in the heterogeneous computing platform.
4. *New hardware-enabled transaction processing technologies* Concurrency control and recovery are core functions in data management which ensure transaction isolation and persistence. Their design and implementation are tightly related to the underlying computing and storage environment. At the same time, high-performance network environments also provide new opportunities for distributed transaction processing which were difficult to scale out. First of all, the storage hierarchy and the new access characteristics have the most significant

impact on transaction recovery technology. Database recovery technology needs to be optimized according to the features of NVM. Recovery technology for NVM, partitioning technology, and concurrent control protocol based on NVM are all urgent research topics. Second, transactions typically involve multiple types of operations as well as synchronization among them. However, general-purpose processors and specialized accelerators have different data processing modes. Separating the transactional processing performed by the general-purpose CPU, and offloading part of the workload to specialized processor, is capable of enhancing the performance of transaction processing. Therefore, studying the load balancing and I/O optimization technologies that accelerate the transaction processing is an effective way to solve the performance bottlenecks in transaction processing. Furthermore, in RDMA-enabled network, the assumption about distributed transactions cannot scale will cease to exist. RDMA-enabled distributed commit protocols and pessimistic and optimistic concurrency control approaches for RDMA may all be potential research directions.

5 Conclusion

The new hardware and its constructed environment will deeply influence the architecture of the entire computing system and change the previous assumptions of the upper-layer software. While providing higher physical performance, software architectures and related technologies for data management and analysis are also required to sense and adapt to the new hardware features. The new hardware environment has made the trade-off between data management and analytics system design space more complex, posing multidimensional research challenges. In the future, there is an urgent need to break the oldness of the traditional data management and analysis software. It is also necessary to manage and analyze the core system functions based on the characteristics of hardware environments and explore and research new data processing modes, architectures, and technologies from the bottom up.

Acknowledgements The authors would like to thank Mike Dixon for polishing the paper. He is an older gentleman from Salt Lake City. This work is supported by the National Natural Science Foundation of China under Grant Nos. 61472321, 61672432, 61732014, and 61332006 and the National High Technology Research and Development Program of China under Grant No. 2015AA015307.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no competing interests.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Bordawekar RR, Sadoghi M (2016) Accelerating database workloads by software–hardware–system co-design. In: 2016 IEEE 32nd international conference on data engineering (ICDE). IEEE, pp 1428–1431
2. Viglas SD (2015) Data management in non-volatile memory. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, pp 1707–1711
3. Ailamaki A (2015) Databases and hardware: the beginning and sequel of a beautiful friendship. *Proc VLDB Endow* 8(12):2058–2061
4. Swanson S, Caulfield AM (2013) Refactor, reduce, recycle: restructuring the i/o stack for the future of storage. *Computer* 46(8):52–59
5. Lim K, Chang J, Mudge T et al (2009) Disaggregated memory for expansion and sharing in blade servers. *ACM SIGARCH Comput Archit News* 37(3):267–278
6. Xue CJ, Sun G, Zhang Y et al (2011) Emerging non-volatile memories: opportunities and challenges. In: 2011 Proceedings of the 9th international conference on hardware/software codesign and system synthesis (CODES + ISSS). IEEE, pp 325–334
7. Freitas R (2010) Storage class memory: technology, systems and applications. In: 2010 IEEE hot chips 22 symposium (HCS). IEEE, pp 1–37
8. Arulraj J, Pavlo A (2017) How to build a non-volatile memory database management system. In: Proceedings of the 2017 ACM international conference on management of data. ACM, pp 1753–1758
9. Zamanian E, Binnig C, Harris T et al (2017) The end of a myth: distributed transactions can scale. *Proc VLDB Endow* 10(6):685–696
10. Barthels C, Alonso G, Hoeffler T (2017) Designing databases for future high-performance networks. *IEEE Data Eng Bull* 40(1):15–26
11. Graefe G (1994) Volcano/spl minus/an extensible and parallel query evaluation system. *IEEE Trans Knowl Data Eng* 6(1):120–135
12. Idreos S, Groffen F, Nes N et al (2012) MonetDB: two decades of research in column-oriented database architectures. *Q Bull IEEE Comput Soc Tech Comm Database Eng* 35(1):40–45
13. Zukowski M, van de Wiel M, Boncz P (2012) Vectorwise: A vectorized analytical DBMS. In: 2012 IEEE 28th international conference on data engineering (ICDE). IEEE, pp 1349–1350
14. Neumann T (2011) Efficiently compiling efficient query plans for modern hardware. *Proc VLDB Endow* 4(9):539–550
15. Lang H, Mühlbauer T, Funke F et al (2016) Data blocks: hybrid OLTP and OLAP on compressed storage using both vectorization and compilation. In: Proceedings of the 2016 international conference on management of data. ACM, pp 311–326
16. Polychroniou O, Raghavan A, Ross KA (2015) Rethinking SIMD vectorization for in-memory databases. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, pp 1493–1508
17. Sukhwani B, Thoennes M, Min H et al (2015) A hardware/software approach for database query acceleration with fpgas. *Int J Parallel Program* 43(6):1129–1159

18. Meister A, Breß S, Saake G (2015) Toward GPU-accelerated database optimization. *Datenbank Spektrum* 15(2):131–140
19. Jha S, He B, Lu M et al (2015) Improving main memory hash joins on Intel Xeon Phi processors: an experimental approach. *Proc VLDB Endow* 8(6):642–653
20. Woods L, István Z, Alonso G (2014) Ibex: an intelligent storage engine with support for advanced SQL offloading. *Proc VLDB Endow* 7(11):963–974
21. Woods L, Alonso G, Teubner J (2015) Parallelizing data processing on FPGAs with shifter lists. *ACM Trans Reconfig Technol Syst (TRETTS)* 8(2):7
22. He B, Yang K, Fang R et al (2008) Relational joins on graphics processors. In: *Proceedings of the 2008 ACM SIGMOD international conference on management of data*. ACM, pp 511–524
23. DeBrabant J, Arulraj J, Pavlo A et al (2014) A prolegomenon on OLTP database systems for non-volatile memory. In: *ADMS@VLDB*
24. Pavlo A (2015) Emerging hardware trends in large-scale transaction processing. *IEEE Internet Comput* 19(3):68–71
25. Coburn J, Bunker T, Schwarz M et al (2013) From ARIES to MARS: transaction support for next-generation, solid-state drives. In: *Proceedings of the twenty-fourth ACM symposium on operating systems principles*. ACM, pp 197–212
26. DeBrabant J, Pavlo A, Tu S et al (2013) Anti-caching: a new approach to database management system architecture. *Proc VLDB Endow* 6(14):1942–1953
27. Kim D, Lee S, Chung J et al (2012) Hybrid DRAM/PRAM-based main memory for single-chip CPU/GPU. In: *Proceedings of the 49th annual design automation conference*. ACM, pp 888–896
28. Yoon HB, Meza J, Ausavarungnirun R et al (2012) Row buffer locality aware caching policies for hybrid memories. In: *2012 IEEE 30th international conference on computer design (ICCD)*. IEEE, pp 337–344
29. Meza J, Chang J, Yoon HB et al (2012) Enabling efficient and scalable hybrid memories using fine-granularity DRAM cache management. *IEEE Comput Archit Lett* 11(2):61–64
30. Manegold S, Boncz P, Kersten ML (2002) Generic database cost models for hierarchical memory systems. In: *Proceedings of the 28th international conference on very large data bases*. VLDB endowment, pp 191–202
31. Carnegie Mellon Database Group [EB/OL] (2016) N-Store. <http://db.cs.cmu.edu/projects/nvm/>. Accessed 22 Nov 2017
32. Tinnefeld C, Kossmann D, Grund M et al (2013) Elastic online analytical processing on ramcloud. In: *Proceedings of the 16th international conference on extending database technology*. ACM, pp 454–464
33. Li F, Das S, Syamala M et al (2016) Accelerating relational databases by leveraging remote memory and RDMA. In: *Proceedings of the 2016 international conference on management of data*. ACM, pp 355–370
34. Binnig C, Crotty A, Galakatos A et al (2016) The end of slow networks: it's time for a redesign. *Proc VLDB Endow* 9(7):528–539
35. Elmore AJ, Arora V, Taft R et al (2015) Squall: fine-grained live reconfiguration for partitioned main memory databases. In: *Proceedings of the 2015 ACM SIGMOD International conference on management of data*. ACM, pp 299–313
36. Polychroniou O, Sen R, Ross KA (2014) Track join: distributed joins with minimal network traffic. In: *Proceedings of the 2014 ACM SIGMOD international conference on management of data*. ACM, pp 1483–1494
37. Kalia A, Kaminsky M, Andersen DG (2014) Using RDMA efficiently for key-value services. *ACM SIGCOMM Comput Commun Rev* 44(4):295–306
38. Dragojević A, Narayanan D, Hodson O et al (2014) FaRM: fast remote memory. In: *Proceedings of the 11th USENIX conference on networked systems design and implementation*, pp 401–414
39. Bishnoi R, Oboril F, Ebrahimi M et al (2014) Avoiding unnecessary write operations in STT-MRAM for low power implementation. In: *2014 15th international symposium on quality electronic design (ISQED)*. IEEE, pp 548–553
40. Yue J, Zhu Y (2013) Accelerating write by exploiting PCM asymmetries. In: *2013 IEEE 19th international symposium on high performance computer architecture (HPCA2013)*. IEEE, pp 282–293
41. Yang J, Minturn DB, Hady F (2012) When poll is better than interrupt. *FAST* 12:3
42. Qureshi MK, Srinivasan V, Rivers JA (2009) Scalable high performance main memory system using phase-change memory technology. *ACM SIGARCH Comput Archit News* 37(3):24–33
43. Dhiman G, Ayoub R, Rosing T (2009) PDRAM: a hybrid PRAM and DRAM main memory system. In: *46th ACM/IEEE design automation conference, 2009, DAC'09*. IEEE, pp 664–669
44. Wang Z, Shan S, Cao T et al (2013) WADE: writeback-aware dynamic cache management for NVM-based main memory system. *ACM Trans Archit Code Optim (TACO)* 10(4):51
45. Seok H, Park Y, Park KH (2011) Migration based page caching algorithm for a hybrid main memory of DRAM and PRAM. In: *Proceedings of the 2011 ACM symposium on applied computing*. ACM, pp 595–599
46. Coburn J, Caulfield AM, Akel A et al (2011) NV-Heaps: making persistent objects fast and safe with next-generation, non-volatile memories. *ACM Sigplan Not* 46(3):105–118
47. Condit J, Nightingale EB, Frost C et al (2009) Better I/O through byte-addressable, persistent memory. In: *Proceedings of the ACM SIGOPS 22nd symposium on operating systems principles*. ACM, pp 133–146
48. Volos H, Tack AJ, Swift MM (2011) Mnemosyne: lightweight persistent memory. *ACM SIGARCH Comput Archit News* 39(1):91–104
49. Hwang T, Jung J, Won Y (2015) Heapo: heap-based persistent object store. *ACM Trans Storage (TOS)* 11(1):3
50. Giles E, Doshi K, Varman P (2013) Bridging the programming gap between persistent and volatile memory using WrAP. In: *Proceedings of the ACM international conference on computing frontiers*. ACM, p 30
51. Zhao J, Li S, Yoon DH et al (2013) Kiln: closing the performance gap between systems with and without persistence support. In: *2013 46th annual IEEE/ACM international symposium on microarchitecture (MICRO)*. IEEE, pp 421–432
52. Lee S, Kim J, Lee M et al (2014) Last block logging mechanism for improving performance and lifetime on SCM-based file system. In: *Proceedings of the 8th international conference on ubiquitous information management and communication*. ACM, p 38
53. Caulfield AM, De A, Coburn J et al (2010) Moneta: a high-performance storage array architecture for next-generation, non-volatile memories. In: *Proceedings of the 2010 43rd annual IEEE/ACM international symposium on microarchitecture*. IEEE Computer Society, pp 385–395
54. PRAMFS Team (2016) Protected and persistent RAM file system [EB/OL]. <http://pRamfs.SourceForge.net>. Accessed 16 Mar 2018
55. Sha EHM, Chen X, Zhuge Q et al (2016) A new design of in-memory file system based on file virtual address framework. *IEEE Trans Comput* 65(10):2959–2972
56. Fang R, Hsiao HI, He B et al (2011) High performance database logging using storage class memory. In: *2011 IEEE 27th international conference on data engineering (ICDE)*. IEEE, pp 1221–1231

57. Caulfield AM, Molloy TI, Eisner LA et al (2012) Providing safe, user space access to fast, solid state disks. *ACM SIGARCH Comput Archit News* 40(1):387–400
58. Venkataraman S, Tolia N, Ranganathan P et al (2011) Consistent and durable data structures for non-volatile byte-addressable memory. *FAST* 11:61–75
59. Zhang J, Donofrio D, Shalf J et al (2015) Nvmmu: a non-volatile memory management unit for heterogeneous GPU-SSD architectures. In: 2015 international conference on parallel architecture and compilation (PACT). IEEE, pp 13–24
60. Zhang Y, Zhou X, Zhang Y et al (2016) Virtual denormalization via array index reference for main memory OLAP. *IEEE Trans Knowl Data Eng* 28(4):1061–1074
61. Wu S, Jiang D, Ooi BC et al (2010) Efficient B-tree based indexing for cloud data processing. *Proc VLDB Endow* 3(1–2):1207–1218
62. Wang J, Wu S, Gao H et al (2010) Indexing multi-dimensional data in a cloud system. In: Proceedings of the 2010 ACM SIGMOD international conference on management of data. ACM, pp 591–602
63. Ding L, Qiao B, Wang G et al (2011) An efficient quad-tree based index structure for cloud data management. In: WAIM, pp 238–250
64. Jin R, Cho HJ, Chung TS (2014) A group round robin based B-tree index storage scheme for flash memory devices. In: Proceedings of the 8th international conference on ubiquitous information management and communication. ACM, p 29
65. On ST, Hu H, Li Y et al (2009) Lazy-update B+-tree for flash devices. In: Tenth international conference on mobile data management: systems, services and middleware, 2009, MDM'09. IEEE, pp 323–328
66. Lomet D (2004) Simple, robust and highly concurrent B-trees with node deletion. In: Proceedings. 20th international conference on data engineering. IEEE, pp 18–27
67. Levandoski JJ, Lomet DB, Sengupta S (2013) The Bw-tree: a B-tree for new hardware platforms. In: 2013 IEEE 29th international conference on data engineering (ICDE). IEEE, pp 302–313
68. Qureshi MK, Franceschini MM, Lastras-Montano LA (2010) Improving read performance of phase change memories via write cancellation and write pausing. In: 2010 IEEE 16th international symposium on high performance computer architecture (HPCA). IEEE, pp 1–11
69. Ahn J, Yoo S, Choi K (2014) DASCA: dead write prediction assisted STT-RAM cache architecture. In: 2014 IEEE 20th international symposium on high performance computer architecture (HPCA). IEEE, pp 25–36
70. Li J, Shi L, Li Q et al (2013) Cache coherence enabled adaptive refresh for volatile STT-RAM. In: Proceedings of the conference on design, automation and test in Europe. EDA consortium, pp 1247–1250
71. Ferreira AP, Zhou M, Bock S et al (2010) Increasing PCM main memory lifetime. In: Proceedings of the conference on design, automation and test in Europe. European Design and Automation Association, pp 914–919
72. Vamsikrishna MV, Su Z, Tan KL (2012) A write efficient pcm-aware sort. In: International conference on database and expert systems applications. Springer, Berlin, pp 86–100
73. Garg V, Singh A, Haritsa JR (2015) Towards making database systems PCM-compliant. In: International conference on database and expert systems applications. Springer, Cham, pp 269–284
74. Chen S, Gibbons PB, Nath S et al (2011) Rethinking database algorithms for phase change memory. In: Conference on innovative data systems research, pp 21–31
75. Viglas SD (2012) Adapting the B+-tree for asymmetric I/O. In: East European conference on advances in databases and information systems. Springer, Berlin, pp 399–412
76. Chi P, Lee WC, Xie Y (2014) Making B+-tree efficient in PCM-based main memory. In: Proceedings of the 2014 international symposium on low power electronics and design. ACM, pp 69–74
77. Bausch D, Petrov I, Buchmann A (2012) Making cost-based query optimization asymmetry-aware. In: Proceedings of the eighth international workshop on data management on new hardware. ACM, pp 24–32
78. Ailamaki A, DeWitt DJ, Hill MD (2002) Data page layouts for relational databases on deep memory hierarchies. *VLDB J Int J Very Large Data Bases* 11(3):198–215
79. Lee IH, Lee SG, Shim J (2011) Making T-trees cache conscious on commodity microprocessors. *J Inf Sci Eng* 27(1):143–161
80. Manegold S, Boncz P, Kersten M (2002) Optimizing main-memory join on modern hardware. *IEEE Trans Knowl Data Eng* 14(4):709–730
81. Hankins RA, Patel JM (2003) Data morphing: an adaptive, cache-conscious storage technique. In: Proceedings of the 29th international conference on very large data bases, vol 29. VLDB Endowment, pp 417–428
82. Răducanu B, Boncz P, Zukowski M (2013) Micro adaptivity in vectorwise. In: Proceedings of the 2013 ACM SIGMOD international conference on management of data. ACM, pp 1231–1242
83. Johnson R, Raman V, Sidle R et al (2008) Row-wise parallel predicate evaluation. *Proc VLDB Endow* 1(1):622–634
84. Balkesen C, Teubner J, Alonso G et al (2013) Main-memory hash joins on multi-core CPUs: tuning to the underlying hardware. In: 2013 IEEE 29th international conference on data engineering (ICDE). IEEE, pp 362–373
85. Balkesen C, Alonso G, Teubner J et al (2013) Multi-core, main-memory joins: sort vs. hash revisited. *Proc VLDB Endow* 7(1):85–96
86. He J, Zhang S, He B (2014) In-cache query co-processing on coupled CPU-GPU architectures. *Proc VLDB Endow* 8(4):329–340
87. Cheng X, He B, Lu M, et al (2016) Efficient query processing on many-core architectures: a case study with Intel Xeon Phi processor. In: Proceedings of the 2016 international conference on management of data. ACM, pp 2081–2084
88. Werner S, Heinrich D, Stelzner M et al (2016) Accelerated join evaluation in Semantic Web databases by using FPGAs. *Concurr Comput Pract Exp* 28(7):2031–2051
89. Sitaridi EA, Ross KA (2016) GPU-accelerated string matching for database applications. *VLDB J* 25(5):719–740
90. He J, Lu M, He B (2013) Revisiting co-processing for hash joins on the coupled cpu-gpu architecture. *Proc VLDB Endow* 6(10):889–900
91. Zhang Y, Zhang Y, Su M et al (2013) HG-Bitmap join index: a hybrid GPU/CPU bitmap join index mechanism for OLAP. In: International conference on web information systems engineering. Springer, Berlin, pp 23–36
92. Blanas S, Li Y, Patel JM (2011) Design and evaluation of main memory hash join algorithms for multi-core CPUs. In: Proceedings of the 2011 ACM SIGMOD international conference on management of data. ACM, pp 37–48
93. Albutiu MC, Kemper A, Neumann T (2012) Massively parallel sort-merge joins in main memory multi-core database systems. *Proc VLDB Endow* 5(10):1064–1075
94. Pagh R, Wei Z, Yi K et al (2010) Cache-oblivious hashing. In: Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems. ACM, pp 297–304
95. Mitzenmacher M (2016) A new approach to analyzing robin hood hashing. In: 2016 proceedings of the thirteenth workshop

- on analytic algorithmics and combinatorics (ANALCO). Society for Industrial and Applied Mathematics, pp 10–24
96. Richter S, Alvarez V, Dittrich J (2015) A seven-dimensional analysis of hashing methods and its implications on query processing. *Proc VLDB Endow* 9(3):96–107
 97. Schuh S, Chen X, Dittrich J (2016) An experimental comparison of thirteen relational equi-joins in main memory. In: *Proceedings of the 2016 international conference on management of data*. ACM, pp 1961–1976
 98. Mohan C, Haderle D, Lindsay B et al (1992) ARIES: a transaction recovery method supporting fine-granularity locking and partial rollbacks using write-ahead logging. *ACM Trans Database Syst (TODS)* 17(1):94–162
 99. Pelley S, Wenisch TF, Gold BT et al (2013) Storage management in the NVRAM era. *Proc VLDB Endow* 7(2):121–132
 100. Gao S, Xu J, Härder T et al (2015) PCMLogging: optimizing transaction logging and recovery performance with PCM. *IEEE Trans Knowl Data Eng* 27(12):3332–3346
 101. Huang J, Schwan K, Qureshi MK (2014) NVRAM-aware logging in transaction systems. *Proc VLDB Endow* 8(4):389–400
 102. Wang T, Johnson R (2014) Scalable logging through emerging non-volatile memory. *Proc VLDB Endow* 7(10):865–876
 103. Harizopoulos S, Abadi DJ, Madden S et al (2008) OLTP through the looking glass, and what we found there. In: *Proceedings of the 2008 ACM SIGMOD international conference on management of data*. ACM, pp 981–992
 104. Larson PÅ, Blanas S, Diaconu C et al (2011) High-performance concurrency control mechanisms for main-memory databases. *Proc VLDB Endow* 5(4):298–309
 105. Johnson R, Pandis I, Ailamaki A (2009) Improving OLTP scalability using speculative lock inheritance. *Proc VLDB Endow* 2(1):479–489
 106. Pandis I, Johnson R, Hardavellas N et al (2010) Data-oriented transaction execution. *Proc VLDB Endow* 3(1–2):928–939
 107. Johnson R, Athanassoulis M, Stoica R et al (2009) A new look at the roles of spinning and blocking. In: *Proceedings of the fifth international workshop on data management on new hardware*. ACM, pp 21–26
 108. Horikawa T (2013) Latch-free data structures for DBMS: design, implementation, and evaluation. In: *Proceedings of the 2013 ACM SIGMOD international conference on management of data*. ACM, pp 409–420
 109. Jung H, Han H, Fekete A et al (2014) A scalable lock manager for multicores. *ACM Trans Database Syst (TODS)* 39(4):29
 110. Ren K, Thomson A, Abadi DJ (2012) Lightweight locking for main memory database systems. In: *Proceedings of the VLDB endowment*, vol 6, no 2. VLDB Endowment, pp 145–156
 111. Sadoghi M, Ross KA, Canim M et al (2013) Making updates disk-I/O friendly using SSDs. *Proc VLDB Endow* 6(11):997–1008
 112. Kallman R, Kimura H, Natkins J et al (2008) H-store: a high-performance, distributed main memory transaction processing system. *Proc VLDB Endow* 1(2):1496–1499
 113. Dragojević A, Narayanan D, Nightingale EB et al (2015) No compromises: distributed transactions with consistency, availability, and performance. In: *Proceedings of the 25th symposium on operating systems principles*. ACM, pp 54–70
 114. Levandoski JJ, Lomet DB, Sengupta S et al (2015) High performance transactions in deuteronomy. In: *CIDR*
 115. Zhenkun Y, Chuanhui Y, Zhen L (2013) OceanBase—a massive structured data storage management system. *E Sci Technol Appl* 4(1):41–48
 116. Bailis P, Davidson A, Fekete A et al (2013) Highly available transactions: virtues and limitations. *Proc VLDB Endow* 7(3):181–192
 117. Stonebraker M, Madden S, Abadi DJ et al (2007) The end of an architectural era: (it's time for a complete rewrite). In: *Proceedings of the 33rd international conference on very large data bases*. VLDB Endowment, pp 1150–1160
 118. Pavlo A, Curino C, Zdonik S (2012) Skew-aware automatic database partitioning in shared-nothing, parallel OLTP systems. In: *Proceedings of the 2012 ACM SIGMOD international conference on management of data*. ACM, pp 61–72
 119. NuoDB Team [EB/OL] (2016) NuoDB. <http://www.nuodb.com/>. Accessed 22 Jan 2018
 120. Brantner M, Florescu D, Graf D et al (2008) Building a database on S3. In: *Proceedings of the 2008 ACM SIGMOD international conference on management of data*. ACM, pp 251–264
 121. Stonebraker M (2010) SQL databases v. NoSQL databases. *Commun ACM* 53(4):10–11
 122. Özcan F, Tatbul N, Abadi DJ et al (2014) Are we experiencing a big data bubble? In: *Proceedings of the 2014 ACM SIGMOD international conference on management of data*. ACM, pp 1407–1408
 123. Pavlo A, Aslett M (2016) What's really new with NewSQL? *ACM Sigmod Rec* 45(2):45–55
 124. Krueger J, Kim C, Grund M et al (2011) Fast updates on read-optimized databases using multi-core CPUs. *Proc VLDB Endow* 5(1):61–72
 125. Barshai V, Chan Y, Lu H et al (2012) Delivering continuity and extreme capacity with the IBM DB2 pureScale feature. IBM Redbooks, New York
 126. Ailamaki A, Liarou E, Tözün P et al (2015) How to stop under-utilization and love multicores. In: *2015 IEEE 31st international conference on data engineering (ICDE)*. IEEE, pp 1530–1533