计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS

固态硬盘存储的技术研究

邵旭[1)]

[1)](华中科技大学计算机科学与技术学院，武汉 430074)

摘 要 随着计算机技术的快速发展，人们的电子设备时时刻刻在产生海量的数据，全球数据总量呈指数增长趋势。因此在数据中心中，存储系统面临着许多新的挑战。典型的存储系统主要有存储介质层、文件系统层和存储系统软件应用层三部分构成。在存储介质层，随着固态硬盘技术的发展，SSD 已经被广泛的应用于各大数据中心的存储系统中，各大厂商也不断研发基于不同介质更稳定和快速的固态硬盘。固态硬盘相比传统 HDD 磁盘，在随机读写性能有巨大的优势。但是传统的容错模型与技术方法无法直接应用在新的固态硬盘存储系统，随着固态硬盘存储系统日益大型化，面临的挑战也越来越多。在文件系统层，由于固态硬盘所提供的低延时访问，传统的文件 I/O 接口不适应固态硬盘所提供的访问特性，使得软件开销大于物理开销。在频繁的文件访问环境中，降低这部分的开销对于提高整体系统的性能有重要意义。研究者对文件系统层接口进行相应的优化，降低了文件访问的开销，向上对文件系统应用提供了更好的接口支持。在上层的存储系统软件应用中，键值存储系统因为对非结构化数据的友好支持已经被广泛应用。但是过去键值存储系统无法充分发挥现有的 NVMe SSD 的全部性能，导致了性能上的浪费。所以，针对基于 SSD 的存储系统，依然存在许多在安全性、高性能等方向上的挑战。我们将从这三个角度说明基于 SSD 的存储系统面临的主要挑战和已有的研究解决方法，并展望未来的研究方向。

关键词 固态存储硬盘；存储系统；高性能

Technical Research on Solid State Disk Storage

Xu Shao[1)]

[1)](School of Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China)

Abstract With the rapid development of computer technology, people's electronic devices are generating massive amounts of data all the time, and the total amount of global data is increasing exponentially. Therefore, in the data center, the storage system faces many new challenges. A typical storage system mainly consists of three parts: storage medium layer, file system layer, and storage system software application layer. At the storage medium layer, with the development of solid state drive technology, SSDs have been widely used in storage systems in major data centers, and major manufacturers have also continued to develop more stable and fast solid state drives based on different media. Compared with traditional HDD disks, solid state drives have huge advantages in random read and write performance. However, traditional fault-tolerant models and technical methods cannot be directly applied to new solid-state hard disk storage systems. As solid-state hard disk storage systems become larger and larger, more and more challenges are faced. At the file system layer, due to the low-latency access provided by the solid state drive, the traditional file I/O interface does not adapt to the access characteristics provided by the solid state drive, making the software overhead greater than the physical overhead. In a frequent file access environment, reducing this part of the overhead is of great significance for improving the performance of the overall system. Researchers optimized the file system layer interface

accordingly, which reduced the overhead of file access and provided better interface support for file system applications. In the upper-level storage system software application, the key-value storage system has been widely used because of its friendly support for unstructured data. However, in the past, the key-value storage system could not fully utilize the full performance of the existing NVMe SSD, resulting in a waste of performance. Therefore, for SSD-based storage systems, there are still many challenges in terms of security and high performance. We will explain the main challenges faced by SSD-based storage systems and existing research solutions from these three perspectives, and look forward to future research directions.
Key words Solid state storage hard disk; storage system; high performance

1 绪论

1.1 海量数据存储

随着互联网应用的高速发展和通信技术的发展，全球的数据总量呈现指数级的增长趋势。以机械硬盘为主的传统存储设备因为高访问延迟、高能耗开销、防震效果差等原因，整体性能无法应对高速增加的数据，成为了限制计算机系统性能的瓶颈。因此，为了应对快速增加的数据，更高速、更稳定、更高性能的存储设备不断出现。但是新的设备在其物理性质上与传统设备存在差异，导致需要改变使用新型存储设备的方法。同样，在指数级增长的数据中，非结构化数据的比例越来越大[4]。根据国际数据公司 IDC 预测，预计到 2025 年，全球产生的数据量将会增长到 175ZB，且其中超过 80%的数据会是处理难度较大的非结构化数据。因此，传统的文件存储系统不再适合新结构组成的数据存储。各行各业随着互联网技术的发展，每天都有大量的数据产生。如何将这些海量的数据存储起来并加以有效的利用，成为现在数据中心、各大互联网公司需要面对的主要挑战之一。

现如今，基于各种介质的固态硬盘（SSD）正取代传统机械硬盘称为数据存储中心的主流存储设备。它能提供更为出色的访问性能，更低的能耗开销和高抗震特性，在性能和存储容量方面全面超越了传统机械硬盘。并且随着固态硬盘价格的下降和技术的成熟，固态硬盘在嵌入式系统、笔记本和 PC 机中得到了广泛的应用，在大规模的数据中心中也得到了广泛应用。虽然相比传统机械硬盘，固态硬盘具有更好的性能，但是因为与适配机械硬盘的文件系统 I/O 和存储系统以及仍存在的可靠性和安全性问题，在实际应用中，不断地修改整体结构和硬件软件接口，充分发挥其性能。

1.2 存储系统的发展

为了应对海量非结构化数据地快速存取，存储系统的各个方面经历了许多发展。典型的存储系统通常由三个部分组成：底层的存储介质、中间层的文件系统 I/O 层和最上层的存储系统应用层。这三个层次和之间的接口设计影响了整个存储系统的性能。在数据规模不断增长、用户需求不断增长和数据结构不断变化的情况下，存储系统的这三个层次设计也在不断地修改完善中发展。

图 1-1 存储系统层次图

在存储介质层，存储系统对于存储设备的要求越来越高，主要可以分为两个方面：性能和安全性。常见的存储设备从最早的磁带、光盘等，发展到现在依然使用的机械硬盘，再到如今常用的基于闪存和其他介质的 SSD，存储介质变得更加高效。存储介质的性能和安全性提升使得存储系统在介质层的访问延时有了明显的降低，高端的 NVMe SSD 甚至可以将读写延迟降低到 10 微秒以下，比机械硬盘快了几个数量级。现在基于闪存的 SSD 已经被广泛应用到各种存储系统中，而高端的 SSD 由于材质和高昂的价格，只是选择部署在一些关键位置。

文件系统 I/O 层是沟通存储介质和上层应用的桥梁，它的设计对整个系统影响巨大。上层的软件应用使用文件系统提供的接口将数据以文件的形式存储到存储介质中，根据需要对介质中的文件进行访问和修改。文件系统利用操作系统提高的 I/O 接口与底层的设备进行交互，实现数据的读写。但是一直以来的文件系统都是针对传统的 HDD 或者

慢速的 SATA 固态进行设计的。这些传统的存储设备的速度较慢，远大于文件系统或操作系统 I/O 栈的开销。所以，当高速的 SSD 出现并应用后，SSD 硬盘的高速访问和接口之间的速度差矛盾就加剧了。为了降低因为接口导致的性能损失，研究者提出了不同的开发工具、新型的硬件访问接口，改变了访问硬件的协议等不同方法。例如 Intel 开发的 SPDK 工具包、零拷贝等技术，进一步地充分发挥底层存储介质的性能。

在软件应用层，关系型数据库一直是主流的存储软件。但是随着非结构化数据的增多，键值存储系统出现并应用广泛。键值存储系统的数据模型更加简单，比关系型数据库有更好的读写性能和可扩展性。面对非结构化的数据，键值存储系统表现出更好的性能。如今工业界主流的键值存储系统包括 RockDB、LevelDB 等。这些系统的优化为整体存储系统提升了性能。

1.3 存储系统面临的挑战

经过 1.2 节的分析，存储系统的各个层次的设计都会影响整体系统展现的性能，也即每个层次的瓶颈会限制整个存储系统的性能。

在存储介质层，存储介质的读写速度直接影响了数据存取性能。基于闪存的 SSD 由于闪存的一些物理限制，硬盘内部通常需要一个 FTL 来实现 SSD 的内部数据管理，所以 SSD 的性能很大程度上还受限于 FTL 层算法的影响。缓存管理和映射表管理是 FTL 的两个主要核心功能。在缓存管理方面，现有的缓存管理算法以提高混村命中率为目标，但是由于上层工作负载的不均衡性，会导致不同队列上的闪存芯片负载不均衡，同时没有考虑到缓存替换时的代价开销，从而没有充分发挥 SSD 的性能。在映射表管理方面，随着 SSD 的容量不断增大，SSD 的页级映射表的大小也不断增加，以至于不能完整地放在 SSD 的内置 DRAM 或 SRAM 上。为了快速定位映射记录，冷热映射记录都混合在闪存页中。所以为了充分发挥 SSD 的优势，应当开发适合 SSD 的访问接口和协议。

在文件系统 I/O 层，随着 SSD 技术的发展，新的高端 SSD 将读写延迟降低，这就导致原本相比传统机械硬盘可以忽略不计的 I/O 栈开销变得不可忽视。为了进一步发挥 SSD 的高速优势，我们应当对 I/O 栈协议进行完善。各大厂商推出了一些 I/O 工具。但如何利用这些 I/O 工具，提供更高效的性能是一个重大挑战。

在存储系统应用层，随着下面两层的优化和性能提升，广泛使用的键值存储系统在运行中，有大量的同步操作，使得线程间的同步开销远大于 I/O 开销，无法充分地利用底层存储介质的高并发性。另一方面，高端存储介质尽管可以大幅提升存储系统的性能，但是通常成本也更高，所以，在硬件成本和性能之间进行权衡也是存储系统面临的主要挑战之一。

1.4 本文主要内容

本文将简要介绍 SSD 的发展过程并描述应用 SSD 的存储系统各层次之间主要面临的挑战，并介绍几种解决这些问题的最新成果。由于存储系统可以大概分成存储介质、文件系统 I/O 和存储系统软件这三层，本文将从这三个方面一起阐述他们的优化和对整个系统的性能提升。

2 SSD 及其使用优化

2.1 SSD存储介质及其存储结构

最早期出现的存储介质是以磁介质的载体为基础，如磁盘、磁带等等。但是磁介质的存储设备无法满足越来越高的性能目标和可靠性需求。固态硬盘得到了广泛的关注。固态硬盘可以达到十倍于传统机械磁盘的读写吞吐，百分之一的延迟和降低了一半的功耗。根据介质的类型可以将磁盘划分为基于闪存 (Nand Flash) 的固态硬盘 (即 SSD) 和基于相变（Phase Changing Memory）存储的固态硬盘。

闪存是一种非易失性随机访问存储介质，组成上类似 NMOS 双层浮栅 (Floating Gate) MOS 晶体管。通过高压将电子压入绝缘层的方式用以锁存电荷，数据因此得以以电荷的形式储存在浮栅中。不同于易失性内存，闪存即使在无电源供应的情况下仍然可以保持数据。而由于所有操作都电子化了，不涉及机械部件的操作，因此闪存的读 / 写时延大大降低。当前，固态硬盘闪存单元基本时延大体上只有主存时延的 4 至 5 倍，较传统机械式硬盘有数十倍的提高。在固态硬盘的生产中，有四种结构的：单层式存储（SLC）、多层式存储（MLC，通常是双层式存储）、三层式存储（TLC，亦称为 3-bit MLC）以及 3D Nand 结构。SLC、MLC、TLC 的读写速度依序从快至慢（约 4:2:1），使用寿命依序从长至短（约为 6:3:2），成本依序从高至低，需

要纠错比特数（ECC）则是相反地从低至高。

基于相变的随机访问存储器（PRAM）作为新出现的非易失性存储器设备，使用含一种或多种硫族化物的玻璃制成。当前的主流为 GeSbTe 系合金。硫属玻璃的特性是经由加热可以改变它的状态，成为晶体或非晶体。PRAM 的最大特点就是可以实现按字节寻址，即每次可只修改其中的一个字节。因此除了作为硬盘替代品，基于相变存储的固态硬盘也被认为是主存的可持久化替代品。目前，PRAM 的主流产品包括 Intel 的 Optane 存储器等等。

上述两种新型的高性能存储介质出现极大地促进了接口与配套协议的研发，例如 PCI-E 硬件存储接口与 NVME 协议等等。在该协议下，固态硬盘大大突破了传统协议和接口固有的性能限制。另外，传统的单节点存储系统通过装载多个硬盘构成磁盘阵列满足高性能或者高容错的需求，如 RAID 阵列。随着新型的存储介质出现，研究者也提出了适应 SSD 的新型存储结构组合方式：混合存储硬盘、混合存储阵列、全闪存阵列。

混合存储硬盘将机械硬盘和固态硬盘相结合，通过少量的固态集成在硬盘的核心架构上，不仅发挥了固态硬盘的性能优势，又实现了普通硬盘的存储容量，同时成本并没有提高多少。全闪存阵列指的是存储阵列中全部使用基于闪存的固态硬盘。但是它并不是简单的将所有的机械硬盘换成固态硬盘，因为传统的磁盘阵列软件是基于机械磁盘设计的。但是因为固态硬盘本身的性质，它不需要考虑一些问题，又有一些机械硬盘没有考虑过的问题。所以真正的全闪存阵列是面向固态硬盘特别设计的系统，更好的支持固态硬盘。混合存储阵列在全闪存阵列和全硬盘存储之间进行了价格和性能的平衡。它一定程度上既提供了全闪存的高速度和低延迟，又不需要负担全固态硬盘存储的高昂成本。大多数混闪存储使用两层存储介质或更多类型的存储来为不同的工作负载提供最佳的性能。一般而言，混合存储系列使用少量固态硬盘用以缓存写入的内容，并定期将内容写入后方的传统硬盘。

2.2 NAND闪存介绍

NAND 闪存（以下简称闪存）属于半导体电子设备，不像 HDD 那样包含机械装置。闪存通过控制电路，利用充放电的形式存储信息。在物理特性方面，闪存有着良好的抗震性、质量轻、体积小同时能耗低。在存取性能方面，单个闪存芯片就可以提供几十 MB/s 的读写带宽，一个 SSD 内部通常含有几十个或者更多的闪存芯片，可以对外提供几百甚至几千 MB/s 的读写带宽。

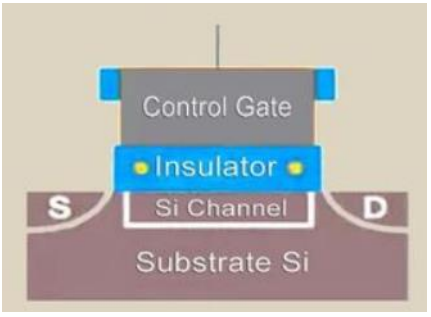

图 2-1 闪存结构图

SSD 与主机通过总线连接，常用的总线标准有 SATA 或 PCIe。由于 PCIe 的传输速度远远快于 SATA，所以现在基于 PCIe 的 NVMe SSD 的速度要远远快于 SATA SSD。SSD 内部的主要架构包括三个部分：主机逻辑接口(HIL)，主控单元(Controller)和闪存存储芯片(chip)。主机逻辑接口主要负责接收来自主机端的各类请求，并对请求进行预处理（如大请求的划分）后发送给内部的主控芯片，同时还负责将请求的数据返回给主机端。主控单元是 SSD 的核心部件，其内置小型的处理器，并管理一个小容量 SRAM 或 DRAM，是 SSD 的核心功能部件，对 SSD 的性能和寿命等关键指标有非常大的影响。

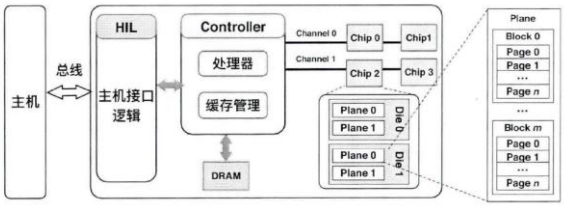

图 2-2 SSD 存储系统结构

2.3 SSD的并行结构研究

为了提升固态硬盘性能，在固态硬盘内部存在四级并行结构：分组间并行、晶圆间并行、芯片间并行和通道间并行。目前在固态硬盘并行研究上并不是很多。Chen 等人[5]测试并验证了固态硬盘并行结构能够有效提高其设备性能。Jung 等人和 Hu 等人[6]验证了两种分配策略对于固态硬盘性能的影响，并阐述了固态硬盘四级并行性之间的区别以及对性能的影响。

闪存固态硬盘的并行性挖掘已经被广泛用于闪存固态硬盘的性能提升。受限于固态硬盘中分组级别并行性所需的两个限制条件，闪存固态硬盘的并行性效率仍然较低，从而导致固态硬盘性能较

差。固态硬盘(SSD)的多级并行性对 SSD 的性能优势有重要的作用。随着 3D 层叠 flash 技术的快速发展，3D SSD 因为其容量大幅增加，逐渐称为 SSD 市场的主流。但是 3D SSD 的优势因为数据放置、请求调度、垃圾收集等策略还是针对 2D SSD 设计的，没有完成展示出来 3D SSD 的性能优势。因此 XU 等人设计了基于闪存的 3D SSD 分层请求调度策略。

2.3.1 3D 闪存固态硬盘

基于 CT(charge trap)的闪存是广泛应用于 3D SSD 的特殊类型的 3D 闪存内存，它利用一种有效的方法构建了垂直的 flash 结构。三维 CT 闪片中存在多层栅叠加层和垂直圆柱通道，如图 2-3 所示。各层电场强度不同，开孔越大，电场强度越大，接入速度越慢。因此，底层的访问速度要比上层快。这种现象称为层速变化，如图 2-4 所示。这种物理现象导致了上层的访问速度可能比底层块好几倍。因此，将合适的数据放入合适的层中，将给并行结构下的 3D SSD 带来性能提升。

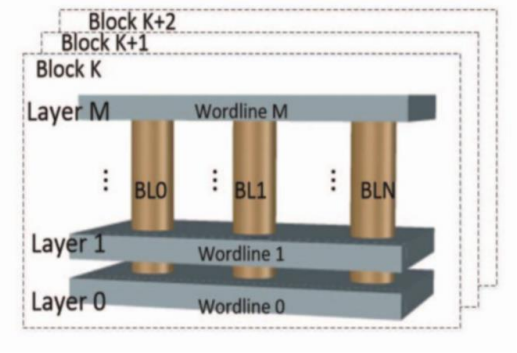

图 2-3 3D 闪存物理结构

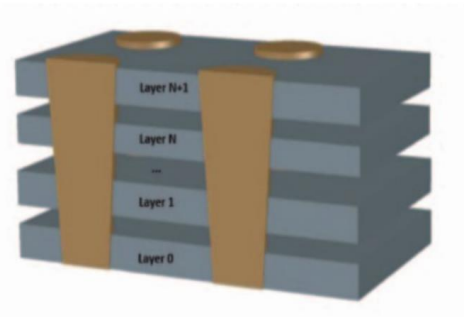

图 2-4 不同层的速度差距

2.3.2 并行优化研究

现有工作从多个角度对并行设计进行了优化，提升了 SSD 的性能。Elyasiet 等人[7]利用子请求之间的空闲时间，提出了一种新的调度方法，对子请求进行重新排序，以提高总体请求访问时间。通过估计冗余时间，该算法可以在冗余时间中偶尔插入子请求。Guo 等[8]提出了一种请求调度方法，利用信道资源，避免垃圾回收的影响。为了避免 GC 冲突，Mao 等[9]提出了一种感知垃圾收集的请求调度方法。Chenet 等[10]建立了一个延迟模型，根据得到的延迟值分别对读写请求进行调度和调度。Cui 等.[11]建议根据保留时间来调度读请求，根据热度来调度写请求，以了解设备过程的变化。Liu 等[12]指出，当添加层信息时，3D 闪存芯片级并行度利用率降低，导致并行性能不佳。从这些工作中可以发现，延迟是衡量 SSD 并行性能的一个重要指标，其中许多工作都使用了一种方法来估计延迟值。现有研究发现，3D flash 存在显著的层间访问速度变化。因此，由于这种显著的变化，上述方法可能不适用于 3D flash。

在 3D 闪存 SSD 中，多个闪存芯片被分为多个通道、多个连接方式，请求数据可以同时加载或存储到闪存上。因此如何调度队列请求和将请求从命令队列分派到芯片将大幅度影响 SSD 的性能表现。所以 Xu 等人[2]分析了 3D 闪存存储器的分层速度变化特性，首先建立了一种分层速度感知的访问时间估计方法，然后利用该模型提前提供更准确的芯片队列时间判断，即计算每个芯片中请求的估计访问时间之和。为了验证该估计方法的有效性，xu 等人提出了一种贪婪请求调度算法，以最短的估计队列时间将读/写请求分配到芯片上。

他们在时间驱动的 SSD 模拟器 SSDSim 上开发实验，他们设计的 LaCR 方法(layer aware chip re-direction method)与静态方法、原始芯片重定向(Original chip redirection, OCR)方法进行实验对比。通过实验得到以下结果：与静态的方法相比，OCR 和 LaCR 都大大提高了并行度，IO 请求响应时间平均分别降低了 3.8%和 6.5%。写请求的改善更为明显，平均分别为 6.2%和 8.3%。因为它们都考虑了芯片的排队时间。其次，与其他算法相比，LaCR 算法工作效率高，并行性好。这是因为在 3D 闪存中，层对层访问速度的显著变化。LaCR 利用了这一特性，可以提供更准确的芯片队列时间判断。写队列时间和整体队列时间都降低了。最后，LaCR 在写密集型工作负载下的性能优于读密集型工作负载。

现有对闪存特性的研究主要可以分为三个方面。第一是考虑根据当前队列的长度，如何将数据分配到并行芯片中。第二个是减少读、写和垃圾回收操作之间的干扰。第三是完善地址翻译相关的操

作，通过与数据访问解耦来减少地址转换开销。

3 固态硬盘接口优化

块接口将存储设备表示为固定大小的逻辑数据块的一维数组，这些数据块可以按照任何顺序进行读、写和重写。块接口最初是为了隐藏硬盘介质特性和简化主机软件而引入的，它适用于许多代存储设备，并且在存储设备接口的两边都有很大的创新。然而，当前基于闪存的 SSD 保持了几十年前的块接口，这在容量过剩、页面映射表的 DRAM、垃圾收集开销以及试图减轻垃圾收集的主机软件复杂性方面带来了巨大的损失。

这些成本是由于允许的操作和基础闪存介质的性质之间的不匹配。虽然单个的逻辑块可以写入闪存，但介质必须在称为擦除块的更大单元的粒度上被擦除。SSD 的 Flash 转换层(FTL)通过使用大量的 DRAM 进行动态逻辑到物理页面映射结构来隐藏这种不匹配，并通过保留大量的驱动器介质容量(过度供应)来降低擦除块数据的垃圾收集开销。尽管有这些途径进行改善，但垃圾收集经常导致吞吐量限制、写放大、性能不可预测性和高尾延迟等问题。

3.1 接口与SSD的不匹配问题

因为原本为HDD设计的块接口随着时间迁移，成为了上层应用程序不成文的契约，并被广泛使用。所以当 SSD 推出了，为了不改变上层应用程序，只能通过对 SSD 添加复杂的固件(FTL)使得 SSD 能为应用程序提供相同的块接口。这也就导致现有的固态硬盘本身的物理特性和性能优势与块接口不匹配，进而导致了写放大、垃圾回收复杂、过度供给三方面问题，没有充分发挥出 SSD 的性能优势。

在文件系统中，文件系统以 4KB 为一个块进行划分，作为最小单位进行操作。但 SSD 的最小写单位远比 4KB 要大。SSD 采用 append 的方式进行更新页，即需要将原来的数据重定向到新的位置，对原数据的整个块进行擦除，这个块才能再次被利用。因此为了满足上述的垃圾回收机制，垃圾收集需要分配设备上的物理资源。为了在物理地址之间转移数据，这就需要 SSD 预留一部分空间(7%-28%).这就导致了实际可使用的空间的损失，也即过度供给问题(over-provided, OP)。

3.2 降低损耗的策略

现在有两种主要方式可以降低这方面的损失：支持流的 SSD(Stream SSDs) 和开放通道 SSD(Open-Channel SSDs ,OCSSDs)。

流 SSD 允许主机使用流提示标记其写命令。流提示由流 SSD 解释，允许它将传入数据区分到不同的擦除块，从而提高整体 SSD 性能和介质寿命。流 SSD 要求主机仔细标记具有相似生存期的数据，以减少垃圾收集。如果主机将不同生存期的数据混合到同一个流中，则流 SSD 的行为类似于块接口 SSD。流 SSD 必须携带用于管理此类事件的资源，因此流 SSD 不会通过资源调配和 DRAM 降低块接口 SSD 用于额外介质的成本。

开放通道 SSD 允许主机和 SSD 通过一组连续的 LBA 块进行协作。OCSSD 可以公开这些块，使它们与介质的物理擦除块边界对齐。这消除了设备内的垃圾收集开销，并降低了媒体资源调配和 DRAM 的成本。对于 OCSSD，主机负责数据放置。这包括基础介质可靠性管理，如磨损均衡和特定介质故障特征（取决于 OCSSD 类型）。这有可能改善 SSD 性能和流 SSD 上的介质寿命，但主机必须管理 SSD 实现之间的差异，以确保耐用性，从而使接口难以采用，并需要持续的软件维护。

3.3 ZNS接口的SSD

ZNS(Zoned Namespace)接口是在 OCSSD 和 SMR-HDD 的基础上构建的。它利用并兼容 ZAC/ZBC 规范中定义的分区存储模型。ZNS 旨在消除 SSD 介质和设备接口之间的不匹配。它还通过避免直接管理 OCSSD 等特定于介质的特性，提供了一个与介质无关的下一代存储接口。

3.3.1 ZNS 分区划分

ZNS 将连续的硬盘 block 划分为一个个 zone，在 zone 分区之间允许任意顺序的读写，但是在 zone 之内的 block 只能顺序写。每个区域状态机使用以下状态确定给定区域是否可写：空、打开、关闭或已满。区域从空状态开始，在写入时转换为打开状态，最后在完全写入时转换为完全。例如，由于设备资源或媒体限制，设备可进一步对可同时处于打开状态的区域的数量施加打开区域限制。如果达到限制并且主机尝试写入新区域，则另一个区域必须从打开状态转换为关闭状态，从而释放设备资源（如写入缓冲区）。闭合区域仍然是可写的，但必须在服务其他写入之前再次转换为打开状态。

ZNS 放弃传统上由 FTL 执行的与支持随机写入相关的职责。ZNS 接口使 SSD 能够将连续区域写入转换为不同的擦除块，从而消除接口和物理媒体之间的不匹配。由于接口不允许随机写入，并且主机必须显式重置区域，因此设备管理的数据放置发生在区域的粗粒度级别。所以为了给用户更好的支持，ZNS 在 FTL 的设计中引入以下三个方面的设计权衡。

分区大小：分区的写入容量与 SSD 实现的擦除块大小之间存在直接关联。SSD 通常有一个由 16-128 个芯片的闪存块组成的条带，该条带可转换为一个可写容量从数百兆字节到低一位数千兆字节的区域。大型分区会降低主机放置数据的自由度，因此我们主张尽可能减小分区大小，在这种情况下，仍然提供芯片级保护，并且在较低的 I/O 队列深度下可以实现足够的每个分区读/写性能。

映射表：在块接口 SSD 中，FTL 维护 LBA 与其物理位置之间的完全关联映射表。这种细粒度映射提高了垃圾收集性能，但表大小通常要求每 1TB 媒体容量有 1GB 映射。由于 ZNS-SSD 区域写入要求是顺序的，因此我们可以从复杂的、完全关联的映射表过渡到完全在擦除块级别维护的粗粒度映射或以某种混合方式维护的粗粒度映射。由于这些映射占 SSD 上 DRAM 的最大使用量，因此可以显著减少甚至完全消除对 DRAM 的需求。

设备资源：由于前面的设计和不需要额外为垃圾回收提供空间，同时降低了奇偶校验要求和写回策略的改变，进一步的增加了活动区域的大小。

ZNS 将传统 SSD 中的 FTL 功能层，转交给 HOST，由 host 来完成垃圾回收、空间分配等任务。并且通过 host 端对 zone 进行管理，处理数据的放置操作，因此在操作粒度上与硬件达成了一致。同时垃圾回收等由 host 来进行，那么就不需要在硬盘上预留空间，同时降低了对 SSD 的读写次数。这意味着通过 ZNS 机制，将避免写放大、过度供给等问题。

3.3.2 ZNS 主机软件修改

为了适配 ZNS 接口，上层的应用程序或文件系统需要修改对数据的访问方式。这里可以设计一种在主机端的 FTL 功能层，将 ZNS SSD 看作块接口的 SSD。这样，上层的应用程序不需要额外进行修改，便可以直接利用 ZNS SSD。但是这样将加大 host 端的处理压力，并且也有一定的处理延时。第二种方法是使用传统的文件系统特性放置数据，放置的效率更高，但是需要为垃圾回收等任务腾出空间和增加了执行压力。第三种方式就是在应用程序中修改驱动，使得他们支持 ZNS 的存取方式，避免 FTL 的开销。但这需要用户执行查看、检查、备份恢复等操作，对于程序的开发有更高要求。

3.3.3 ZNS 实验测试

通过实验验证，ZNS 在覆写方面具有巨大的优势，对整体的性能提升和容量利用提升明显。根据硬件的实际特性，修改不适配的硬件接口，从而充分发挥硬件本身的优势。虽然该工作思路会增加上层人员的开发负担，但整体应用中提高了性能降低了损失。

4 存储系统优化

随着互联网应用的快速发展，对非结构化数据快速存取的需求与日俱增。传统的关系型数据库不再适用这种对非结构化数据的高并发和低延迟的访问需求。因此，键值存储系统应运而生，键值存储系统读写速度更块，可拓展性更好。

4.1 键值存储系统

键值存储系统是一种非关系型存储系统，它使用简单的键值方法来存储数据。键值存储系统将数据存储为键值对的集合，其中键(key)作为唯一的标识符。键和值(value)都可以是从简单对象到复杂复合对象的任何内容，如文本、图片和视频等。图 4-1 展示了键值存储系统的数据模型。

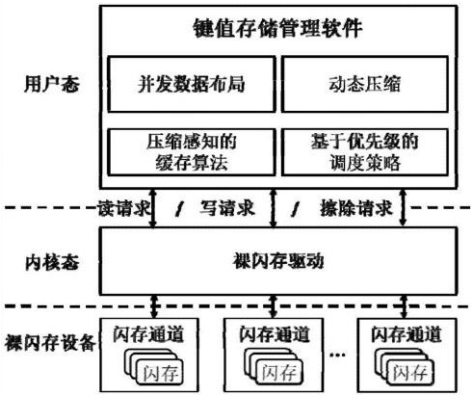

图 4-1 kv 模型架构

键值存储系统已经被广泛地用于支持各种关键的应用程序和服务。键值存储系统可以快速地执行各种内存处理操作，但是仍然经常受到 I/O 性能的限制。最近出现的高速商用 NVMe SSD 推动了新的键值存储系统的发展，这些键值存储系统利用了 NVMe SSD 的超低延迟和高带宽的优势。

4.2 kv存储系统优化方案

为了提高 KV 存储器的性能，最近的系列先进技术附件(SATA)和非易失性存储快速(NVMe)固态硬盘(SSD)被广泛采用。与现有的硬盘驱动器(HDD)相比，SSD 具有独特的特性，必须仔细考虑这些特性才能充分利用性能。例如，由于写前擦除的限制，负载的访问方式会影响 SSD 的性能和耐久性。因此，SSD 在顺序工作负载下的性能要高于随机工作负载下的性能。

4.2.1 kv 存储的优势及面临的问题

由于最近的应用程序会产生大量的数据，因此有效地存储和访问数据至关重要。为此，广泛使用关系数据库来管理数据。但是，由于关系数据库支持丰富的关系，因此很难高效和高性能地管理大量数据。为了提高效率，与关系数据库相比，KV 存储由于更简单的数据管理和更高的性能，被广泛应用于工业和学术界。为了进一步提高 KV 存储的性能，SATA、NVMe SSD 等新兴的高性能存储设备被广泛应用于存储设备。

与现有的 HDD 相比，SSD 提供了低延迟和高带宽。由于 SSD 使用 flash 块存储数据，因此必须仔细考虑 flash 块的特性，才能充分利用 SSD 的性能。例如，由于 NAND flash 的特性，当数据写入一个 flash 块时，就不能在同一个块中更新数据。要更新已保存的数据，必须先读取原始块，并将修改后的数据写入另一个块。然后，带有旧的和未修改的数据的原始块必须清除以备将来使用，这被称为垃圾收集(GC)操作。因此，重要的是要考虑数据局部性和工作负载的访问模式，以充分利用 SSD 的性能。

4.2.2 优化方案介绍

Kim 等人提出，在客户端的随机插入请求通过密钥重塑操作转化为顺序插入请求。重塑操作完成后，新请求的键会被重塑为比现有键更高的键。该方案可以通过在 B+树中创建顺序访问模式来提高 KV 存储的整体性能。将随机请求转换为顺序请求也有利于底层存储设备，特别是基于闪存的 SSD。如前面所提到的，由于 NAND 闪存块的错位特性，ssd 可以从高数据局部性中受益。

F2FS 是一个 flash 友好的文件系统，它改进了一个日志结构的文件系统，该文件系统广泛用于将随机请求转换为顺序请求，从而利用 SSD 的性能。SHRD 是一种请求重塑方案，包括存储设备驱动程序和 SSD 的 FTL。通过在存储中记录随机请求并对这些请求进行排序，SHRD 改善了 FTL 映射表访问的空间局域性。Ziggurat 是一种分级存储文件系统，缓冲文件写入 NVMM 和 DRAM。通过缓存写，它向底层的闪存发送批量顺序写请求，这可以提高性能和垃圾收集效率。

Kim 等人将随机写转换为顺序写，以改善 SSD 中的数据局部性。他们集中在改善 KV 存储的空间局部性，而不是存储设备和系统(即文件系统和 ssd)。

5 总结和展望

随着互联网应用的迅速发展和 5G 时代的来临，全球数据总量呈指数增长趋势，且绝大多数都是处理难度大的非结构化数据。为了针对环境的变化和数据处理的需求，存储系统的存储介质层、I/O 层和应用层都有着迅速的发展。本文针对发展的一些历史和这三个方面的优化做了简要介绍，并介绍了几个最新的工作情况。

随着计算机技术的发展，固态硬盘的需求会越来越多，未来也一定会寻找更高效、更稳定和安全的存储设备。同时在物理设备上的软件架构也会随着硬件的改变或者新架构的提出迎来巨大的改变和飞跃。未来一段时间内，固态硬盘存储系统仍将是高速发展得到阶段，也会面临新的问题，也会有各种新的方向提出，比如智能 SSD。

参 考 文 献

[1] Bjørling M, Aghayev A, Holmberg H, et al. {ZNS}: Avoiding the Block Interface Tax for Flash-based {SSDs}[C]//2021 USENIX Annual Technical Conference (USENIX ATC 21). 2021: 689-703.

[2] Xu J, Du Y, Ding C. Layer-Aware Request Scheduling for 3D Flash-Based SSDs[J]. IEEE Access, 2021, 9: 72025-72032.

[3] Kim S, Son Y. Optimizing Key-Value Stores for Flash-Based SSDs via Key Reshaping[J]. IEEE Access, 2021, 9: 115135-115144.

[4] Data Age 2025[EB/OL]. https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

[5] Feng C, Rubao L, Xiaodong Z. Essential roles of exploiting internal parallelism of flash memory based solid state drives in high-speed data processing, in: Proceedings of IEEE 17th International Symposium on High Performance Computer Architecture. IEEE, 2011,266-277

[6] Yang H, Hong J, Dan F, et al. Exploring and exploiting the multilevel parallelism inside ssds for improved performance and

endurance. IEEE Transactions on Computers, 2013’ 62(6): 1141-1155

[7] N. Elyasi, M. Arjomand, A. Sivasubramaniam, M. T. Kandemir, C. R. Das, and M. Jung, “Exploiting intra-request slack to improve SSD performance,” inProc. 22nd Int. Conf. Architectural Support Program. Lang. Operating Syst., Apr. 2017, pp. 375–388.

[8] J. Guo, Y . Hu, B. Mao, and S. Wu, “Parallelism and garbage collection aware I/O scheduler with improved SSD performance,” inProc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS), May 2017, pp. 1184–1193.

[9] B. Mao and S. Wu, “Exploiting request characteristics and internal parallelism to improve SSD performance,” inProc. 33rd IEEE Int. Conf. Comput. Design (ICCD), Oct. 2015, pp. 447–450.

[10] R. Chen, Q. Guan, C. Ma, and Z. Feng, “Delay-based I/O request scheduling in SSDs,”J. Syst. Archit., vol. 98, pp. 434–442, Sep. 2019

[11] J. Cui, Y . Zhang, W. Wu, J. Yang, Y . Wang, and J. Huang, “DLV: Exploiting device level latency variations for performance improvement on flash memory storage systems,”IEEE Trans. Comput.-Aided Design Integr . Circuits Syst., vol. 37, no. 8, pp. 1546–1559, Aug. 2018.

[12] C.-Y . Liu, J. B. Kotra, M. Jung, M. T. Kandemir, and C. R. Das, “SOML read: Rethinking the read operation granularity of 3D NAND SSDs,” in Proc. 24th Int. Conf. Architectural Support Program. Lang. Operating Syst., 2019, pp. 955–969.