高性能存储系统中的分层和缓存研究综述

Morteza Hoseinzadeh
加州大学圣地亚哥分校

摘要

虽然每个发明存储技术的人都向完美迈进了一大步，但没有一个人是一尘不染的。不同的数据存储要素(如性能、可用性和恢复要求)尚未在一个经济实惠的介质中同时得到满足。最重要的因素之一是价格。因此，在拥有一组理想的存储选择和成本之间一直存在权衡。为了解决这个问题，各种类型的存储介质的网络被用来提供诸如固态驱动器和非易失性存储器的昂贵设备的高性能，以及诸如硬盘驱动器的廉价设备的高容量。在软件中，缓存和分层是一个由来已久的概念，用于在这样的存储网络中自动处理文件操作和移动数据，以及管理低成本介质中的数据备份。基于需求在不同设备之间智能移动数据是解决这一问题的关键。在本次调查中，我们将讨论一些最新的研究成果，这些成果旨在通过缓存和分层技术来改进高性能存储系统。

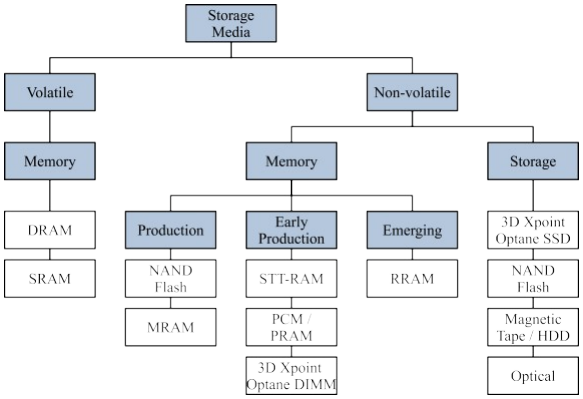


图 1:内存技术

1 介绍[1]

随着计算和网络技术的进步，特别是围绕互联网的计算和网络技术的进步，出现了大量新的数据源，如物联网端点、可穿戴设备、移动平台、智能车辆等。，企业数据密集型分析输入现已扩展至千兆字节，预计到 2020 年将超过 44 兆字节[58].关于这种快速的数据扩展，硬件一直在努力提供更大的容量和更高的密度来支持高性能存储系统。数字 1 代表目前可用的和新兴的存储技术。在存储技术方面，硬盘驱动器(HDD)现在由快速、可靠的固态驱动器(SSD)支持。添加-

此外，随着英特尔推出 Optane DIMMs，一次性新兴持久存储设备将在市场上上市[56].就价格而言，当新技术出现时，过时的技术会变得更便宜。如今，固态硬盘非常普遍，以至于它们被用作数据中心的全闪存阵列(AFA)[66].然而，存储 IO 仍然是大规模数据中心的最大瓶颈。如所示[2]，等待 I/O 所消耗的时间是闲置和浪费 CPU 资源的主要原因，因为许多流行的云应用程序都是 I/O 密集型的，例如视频流、文件同步、备份、机器学习的数据迭代等。

为了解决由输入/输出瓶颈引起的问题，在独立磁盘冗余阵列中并行输入/输出到多个硬盘成为一种常见的方法。然而，RAID 带来的性能提升仍然有限。因此，许多大数据应用程序都努力将中间数据尽可能多地存储到内存中，如 Apache Spark。遗憾的是，内存太贵，容量又极小(例如每台服务器 64 个 128GB)，单靠内存是无法支持超大规模云计算用例的。一些研究建议使用基于非易失性存储器的固态硬盘，如 3D XPoint Optane DIMM[40,22]和 PCM-

[1] 本节的部分内容摘自我发表的论文 Parts[66, 17, 69].

表 1:不同存储技术的比较[3, 59, 14, 11]

	STT-RAM	动态随机存取存储器	NVDIMM	Optane 固态硬盘	与非固态硬盘	硬盘驱动器
容量*	100 MBs	高达 128GB	100 英镑	高达 1TB	高达 4TB	高达 14TB
稍后阅读。	6ns	10-20 ns	50ns	9 秒	35 秒	10ms
稍后写。	13n	10-20 ns	150ns	30 岁	68 岁	10ms
价格	1 美元-3K/GB	7.6 美元/GB	每 GB 3-13 美元	1.30 美元/GB	0.38 美元/GB	每 GB 0.03 美元
可寻址能力	字节	字节	字节/块	街区	街区	街区
波动性	非挥发性的	不稳定的	非挥发性的	非挥发性的	非挥发性的	非挥发性的

英特尔 Optane 固态硬盘 905P 系列(960 GB)(AIC PCIe x4 3Dxp 点)三星 960 Pro 1TB M.2 固态硬盘，带 48 层 3D NAND(来源:Wikibon)*每个模块

基于内存[17, 19, 22] 代替动态随机存取存储器来提供高密度和非易失性。但是，这些存储设备还不够成熟，不能立即用作主存储器，并且仍然非常昂贵。

长期以来，缓存和分层一直被用来隐藏存储层次结构中慢速设备的长延迟。过去，高端硬盘(如 15k RPM)用作性能层，低端硬盘(如 7200 RPM)用作容量层[41].如今，NAND-Flash 固态硬盘取代了快速硬盘，虽然低端硬盘已经过时，但高端硬盘用于容量需求。很快，NVM 等现代存储技术将与过去决裂，改变存储语义。在设备层面，今天的高速固态硬盘配备了写缓冲区，就像苹果的融合驱动器一样[51]. 在系统级，几乎所有的文件系统都有一个页面缓存，用于缓冲动态随机存取存储器中的数据页面，并允许应用程序访问文件内容。使用永久内存作为存储介质，一些文件系统会跳过页面缓存[63].在应用层面，很多大数据应用都努力将中间数据尽可能多地存储到内存中，比如 Apache Spark。然而，作为大型企业存储系统，NVM 在经济上并不经济，固态硬盘的写入耐久性也很有限。

在本次调查中，我们讨论了针对高性能存储系统的缓存和分层解决方案的几项研究。在部分 2，我们简单介绍一下存储设备及其技术。部分 3 将调查几项关于高速缓存解决方案的研究 4 其中讨论了几篇关于存储分层解决方案的论文。在这一节的最后 4，我们简单介绍一下我们组开发的 Ziggurat。最后，第 5 论文结束。

2 背景

本节简要介绍了计算机内存层次结构中各个技术部分的背景信息。我们还讨论了将它们联网所需的硬件和软件的对应部分。

2.1 分级存储器体系

根据响应时间，设计内存层次结构，将计算机存储分成有组织的多级结构，以提高整体性能和存储管理。不同类型的存储介质根据其性能、容量和控制技术被指定为不同的级别。一般来说，层级越低，带宽越小，存储容量越大。层次结构中有四个主要级别，如下所示[57].

2.1.1 内部的

处理器寄存器和高速缓存等片内存储单元属于这一级别。为了提供最高性能，架构使用响应类型最低的存储技术，如静态随机存取存储器、触发器或锁存缓冲器。嵌入式动态随机存取存储器是另一种用于某些专用集成电路的技术[16].近年来，一些新兴技术，如自旋扭矩转移随机存取存储器(STT-RAM)受到了对末级高速缓存的关注[54, 53].它们不仅提供低响应时间，而且还提供高密度和持久性。

请注意，内存层次结构的这个级别中有多个子级别。具有最低可能延迟的处理器寄存器文件位于离处理器最近的子级，后面是多级高速缓存(即 L1、L2 等)。尽管在对称多处理器(SMP)架构中，缓存可能是私有的，也可能在内核之间共享，但它们仍然被视为层次结构中的同一级别。

2.1.2 主要的

计算机系统的主存储器或主存储器暂时保存包括操作系统在内的运行应用程序的所有代码和数据(部分)。在这个层级中，与内部层级相比，能力更为重要。运行应用程序的全部代码和数据都停留在这个层次。尽管此级别的存储容量比内部级别大得多，但性能也应足够高，以支持

主层和内部层之间的快速数据传输。利用空间和时间局部性，存储器控制器设法通过地址和数据总线在最后一级高速缓存和主存储器之间来回移动大量数据。与以字节为单位访问数据的内部级别相反，数据的单位访问是高速缓存或内存行(通常为64字节)。

长期以来，动态随机存取存储器技术一直被用作这一级别的最佳选择。相变存储器等其他技术[26,27,18]已经被引入作为具有持久数据能力的可扩展 DRAM 替代方案。3D 点[40]已经成功原型化并宣布。有关存储技术的详细信息，请参见第节 2.2.1.

2.1.3 辅助存储器

二级存储或在线海量存储层由永久块设备组成，用于永久存储海量数据。与上述两个级别相比，处理器不能直接访问存储。相反，存储介质通过输入输出端口连接到处理器。固态硬盘、硬盘和旋转光学设备都是辅助存储介质的例子。当进程正在执行时，处理器通过输入输出总线(如 PCIe、集成开发环境或 SATA)向块设备发送输入输出请求，以便使用直接内存访问(DMA)功能将数据块(通常为 4KB 的块)加载到主内存中的特定位置。

2.1.4 三级存储

第三级存储或离线大容量存储包括任何种类的可移动存储设备。如果对数据的访问在处理单元的控制之下，则称之为三级存储或近线存储。例如，机器人机构按需安装和拆卸可移动设备。否则，当用户物理连接和分离存储介质时，称为离线存储。在一些存储分类中，第三级存储和离线存储是有区别的。然而，我们认为它们在本文中是相同的。本节的其余部分将讨论最相关的技术及其特点。

2.2 技术

使存储介质彼此不同的主要因素是它们的技术。纵观计算机历史，存储技术已经有了巨大的发展。图 1 一目了然地展示了当前可用的和新兴的技术。通常，计算机存储器系统可以分为易失性和非易失性存储器。另外，通常属于二级和三级存储组的非易失性存储器被用来持久地存储数据。相比之下，通常使用易失性存储器因为高速缓存暂时保持接近处理器是它们高性能的原因。然而，它们的用法可能会经常改变。例如，高端固态硬盘可以用作慢速存储设备的缓存。同样，最近出现的存储类存储器可以用作非易失性介质来永久存储数据，尽管它位于主存储位置。桌子 1 比较不同的计算机存储技术。

2.2.1 存储技术

长期以来，静态随机存取存储器和动态随机存取存储器一直被认为是分别充当处理器内部高速缓存和系统主存储器的主要技术。由于静态随机存取存储器单元的性质，它可以在短时间内保留信息。静态随机存取存储器单元由两个背靠背反相器组成。在待机状态下，只要有电源，这两个逆变器就会相互加强。其中一个表示位数据，另一个对应于位数据的反转值。读取时，读出放大器读取反相器的输出端口，找出哪个电压更高，并确定存储值。虽然 SRAM 几乎和逻辑门电路一样快，但它的密度太低，因为它的电子结构至少由四个晶体管组成。此外，它与互补金属氧化物半导体兼容，因此，在处理器芯片中集成静态随机存取存储器单元是可能的。另一方面，动态随机存取存储器单元仅包括一个晶体管和一个电容器。与静态保存数据的静态随机存取存储器相比，动态随机存取存储器由于电容器的电荷泄漏特性，需要对数据进行刷新。动态随机存取存储器的密度比静态随机存取存储器高得多，但与互补金属氧化物半导体不兼容。因此，在处理器芯片中集成动态随机存取存储器并不容易。此外，读写操作需要更大的外围电路。因为从动态随机存取存储器单元读取是破坏性的，所以每次读取之后都应该进行写入以恢复数据。总的来说，到目前为止，较高的容量和较低的动态随机存取存储器成本使其成为主存储器的最佳选择。然而，动态随机存取存储器面临着一堵缩放墙，因为它使用电容器中的电荷来维护数据。因此，随着技术的发展，不仅电容器的可靠性会显著下降，而且电池间也会产生干扰。更不用说刷新开销的有效功耗是另一个具有挑战性的问题。

许多新兴技术已被调查，以解决缩放问题等。研究人员一直在寻找一种可靠的解决方案来替代动态随机存取存储器，实现字节寻址和高能效。自旋转移扭矩随机存取存储器(STT-RAM)是高性能解决方案之一[20].具有铁磁材料的固定层和自由层，它根据自由层的自旋取向以固定层的高和低电阻特性的形式存储比特。虽然它提供了比动态随机存取存储器更高的性能，以及避免刷新的非易失性，但其昂贵的成本使其成为不可避免的选择

DRAM 替代。其超高密度和低功耗使其成为 CPU 上高速缓存技术的潜在候选。

然而，相变存储器是另一项比其他技术更有前途的新兴技术。它以相变材料电阻级别的形式存储数字信息，其范围从晶态的低电阻到非晶态的高电阻[26].如表所示 1,与动态随机存取存储器相比，相变存储器的性能较低，尤其是在写操作中。它还可以承受较少的写入次数，并且需要刷新以防止电阻漂移。有大量研究致力于解决这些问题[18, 67, 42].

尽管如此，PCM 仍是用作存储级内存技术和固态硬盘的最佳选择之一。桌子 1 显示了 PCM 和 3D 点设备的受益者，而不是 NVMe 驱动程序。连接到内存总线，它们提供接近动态随机存取存储器的性能，同时具有存储设备的大容量。这种类型的内存技术被认为是存储类内存(SCM)，可分为具有快速访问延迟和低容量的内存类型(M-SCM、持久内存或 NVM)(如 3D XPoint DIMM)，或具有高容量和低访问延迟的存储类型(S-SCM)(如 Optane SSD，请参阅部分 2.2.2) [64].

2.2.2 存储技术

除了内部存储器和主存储器之外，永久数据应该驻留在某个存储设备中，以便在需要时访问。长期以来，硬盘一直扮演着这一角色。硬盘驱动器由围绕主轴的刚性快速旋转磁盘和使用致动器臂重新定位的磁头组成。数字数据是以每个磁盘上磁性材料薄膜磁化强度变化的形式存储的。硬盘驱动器的机电方面和存储数据的串行化使得硬盘驱动器的数量级比上述非易失性存储器技术慢。然而，其低廉的价格和极高的密度使其成为二级和三级存储的理想选择。根据表格 1,硬盘的容量可以比动态随机存取存储器大 1000 倍，而操作延迟大约慢 106 倍。

固态硬盘通过使用闪存技术，以更高的价格提供更高的性能、抗冲击性和紧凑的存储。闪存单元由一个金属氧化物半导体场效应晶体管组成，具有一个字线控制栅和另一个浮栅。它将数据以电子开关的形式保存在浮动栅中，浮动栅可以被编程为开或关。无论 MOSFETs 的网络类似于与非门还是或非门逻辑，它都被称为与非门闪存或或非门闪存固态硬盘。读取操作就像在对字线充电的同时读取位线一样简单。然而，写入闪存单元会重新

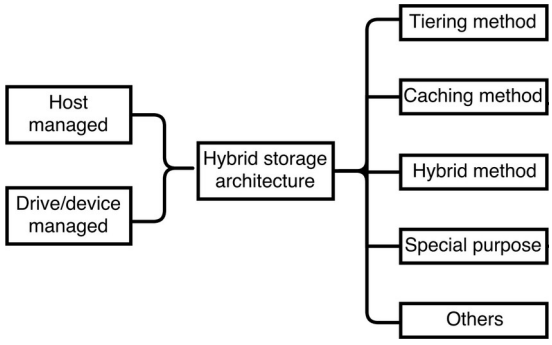


图 2:混合存储体系结构[43]

要求使用隧道释放擦除大部分(MBs)存储区域，然后使用隧道注入放入数据。固态硬盘可能使用传统协议和文件系统，如 SATA、SAS、NTFS、FAT32 等。还有一些接口，如 mSATA、m.2、u.2 和 PCIe，以及一些协议，如 NVMe，是专门为固态硬盘设计的。基于 NAND 闪存的固态硬盘的容量从 128GB 到 100TB 不等，性能最高可达 10GB/s。尽管 NAND 闪存固态硬盘提供了所有优势，但其寿命仅限于每个单元 104 次写入。英特尔和 Micron 最近发布了采用 3D XPoint 新技术的 Optane 固态硬盘[40] 这提供了更长的寿命和更高的性能。三维点单元格根据体电阻的变化保存数据[9]. 由于可堆叠的交叉网格排列，3D 点的密度远远高于传统的非易失性存储器技术。英特尔还推出了一款名为 3D XPoint DIMM 的外形，可以为非易失性存储提供内存带。

2.2.3 海量存储困境

上面提到的技术涉及内存层次结构的不同层次。一方面，存储系统在层次结构中的组织可以根据数据紧张程度而变化。另一方面，数据中心和云存储服务提供商的数据增长速度要求服务器管理员寻求一种高性能的海量存储系统，这需要在网络存储设备和服务器机器上运行数据管理软件。因此，选择一种技术来设计大规模存储系统并不是最好的解决方案。因此，数据中心专家选择开发混合存储系统[43].数字 2 描述了混合存储体系结构的总体类别。在本研究中，我们重点关注主机管理的分层和缓存方法。

呈指数级增长的数字信息要求快速、可靠、海量的数据中心不仅要归档数据，还要快速处理数据。因此，高性能和大容量都是必须的。然而，这部分

不同价值的数字信息可能不均衡。国际数据中心的报告[58] 预测以每两年翻一番的速度，到2020年，数字宇宙的大小可能会超过44 zet bytes(270兆字节)。这些极其广泛的信息并没有被同等地触及。虽然到2020年，云将只触及数字宇宙的24%，但13%将存储在云中，63%可能根本不会被触及[58].需要保护的数据速度超过40%，甚至比数字世界本身还要快。因此，大部分数据通常存储在更便宜、更可靠、更大的设备中，而尚未处理的小部分数据则保存在快速存储介质中。因此，无疑需要一个具有缓存/分层机制的混合存储系统。

3 存储缓存解决方案

为了缓解慢速设备的长延迟，可以在混合存储系统中使用缓存机制。缓存子系统有两个主要原则:
1) 在将原始数据保持在分级结构的中等级别的同时，正在处理的数据的副本驻留在高速缓存中；以及2)高速缓存层中的数据寿命很短，这意味着是暂时的。具有缓存的存储系统的性能主要受四个因素的影响[43]:

1. 数据分配策略本质上控制数据流，并相应地确定缓存的有用性。数据在多个设备中的分布由缓存策略反映，如只读、回写等。
2. 根据其机制，翻译也可能影响性能。在混合存储系统中，相同的数据可能保存在多个设备的不同位置，并且数据的每个副本都应该是可寻址的。地址转换机制对于快速的数据检索和紧凑的元数据空间使用非常重要。
3. 为了更好地利用缓存，需要一种准确的数据热度识别方法。它有助于防止不必要的数据污染缓存，从而通过即时提供热数据来提高整体性能。
4. 缓存使用效率是另一个重要因素，它受到管理队列、同步和执行顺序的调度算法的影响。

缓存机制可以通过设备的硬件管理，也可以通过主机操作系统的软件管理(见图2). 设备管理的缓存系统超出了本研究的范围，因此我们重点关注主机管理的

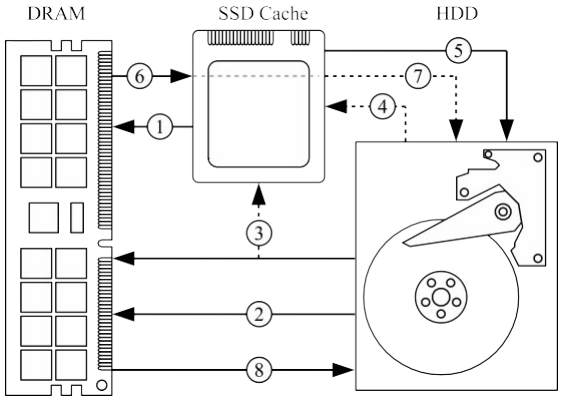


图3:固态硬盘缓存中的数据流

方法。借助主机管理缓存机制，主机可以使用单独的设备来提高性能。最常见的情况之一是使用固态硬盘作为缓存，因为它与慢速硬盘相比具有高性能，与动态随机存取存储器相比具有高容量。除固态硬盘之外，新兴的非易失性存储器(NVM)设备有望参与存储缓存机制。在本文的这一部分，我们将讨论一些存储缓存技术，包括使用固态硬盘或单片机作为存储缓存。

3.1 固态硬盘作为缓存

固态硬盘弥补了磁盘驱动器和主内存之间的性能差距，已被广泛用于缓存慢速驱动器。数字3 显示了使用固态硬盘设备的缓存系统中的常见数据流。1 当读取请求在固态硬盘缓存内完成而不涉及硬盘时发生。如果请求的数据块不在固态硬盘中，则可以通过2访问硬盘以检索动态随机存取存储器中的数据，如果它被识别为热数据，则通过3将其缓存在固态硬盘中。执行热数据识别的后台进程可以通过以下方式将数据从HDD迁移到SSD ④ 不管是否被请求。刷新命令或回写可以在5分钟内将脏块复制回硬盘。当块已经在固态硬盘中时，写操作可以直接在固态硬盘中完成，如6所示，无论缓存使用直写还是回写策略，脏块都可通过7复制到硬盘中。固态硬盘高速缓存中的直写策略已经过时，因为它是为易失性高速缓存设计的，在易失性高速缓存中，脏块应该在某个时间点保持不变。在使用只读或写回策略的情况下，所有新的写操作都由硬盘驱动器中的8直接执行。根据缓存策略，数据可能会流经这些路径。

4.1.1 固态硬盘作为只读缓存

当新的写请求到达只读高速缓存架构时[34, 55, 10, 68, 65, 39] 其中访问块

不在固态硬盘中，请求通过成功完成-通过 8 将其完全记录到硬盘上。当它已经缓存在固态硬盘中用于优先读取操作时，只有在成功更新数据的硬盘副本并丢弃固态硬盘副本后，该请求才被视为已完成。这种缓存架构有助于提高固态硬盘设备的耐用性，因为写入固态硬盘的流量仅限于从硬盘读取数据。同时，缓存空间可以更好地用于读取操作，并且它可以提高总体读取性能，这与写入操作不同，是在关键路径上。但是，固态硬盘的寿命仍然容易受到缓存更新策略的影响。如果数据选择不够准确，缓存可能会被不必要的数据污染，应该运行垃圾收集(GC)进程或替换机制来为要求高的数据腾出空间。此过程可能会导致固态硬盘的写入开销，并缩短寿命。

替换算法对于缓解固态硬盘缓存的写入压力至关重要。部分 3.3 将讨论更多关于常见算法的内容。此外，区块热度识别也极大地影响固态硬盘的使用寿命。MOLAR[34]根据降级计数的控制指标确定数据热度，并明智地将逐出的数据块从第 1 层缓存(动态随机存取存储器)放入第 2 层缓存(固态硬盘)。使用高性能计算系统上应用程序的输入/输出模式，[68]提出了一种启发式文件放置算法来提高缓存性能。由于高性能计算中的应用程序与具有不可预测的输入/输出模式的最终用户应用程序相比更加机械化，假设预先已知的模式离现实不远了。为了理解输入/输出模式，分布式缓存中间件在用户级检测和操作频繁访问的块。

4.1.2 固态硬盘作为读写缓存

由于其非易失性特性，固态硬盘缓存不使用直写策略来保持原始数据的最新状态，而动态随机存取存储器缓存是易失性的，需要同步或持久。因此，固态硬盘读写缓存可能只采用回写或刷新机制。使用固态硬盘作为读写缓存来提高读写操作的性能是非常常见的[21, 28, 35].在这种体系结构中，新的写入在固态硬盘缓存中执行，如图所示 3: 6，稍后它们将被写回磁盘。由于同一数据有两个版本，因此通常会运行定期刷新操作来防止数据同步问题。虽然使用读写固态硬盘缓存通常会提高存储性能，但当缓存接近满时，它会触发垃圾回收进程来清除无效数据，这可能会干扰主进程并降低性能。同时，如果工作负载是写密集型的，并且数据重用率很低，则硬盘驱动器可能会承受很大的写负载，从而使磁盘无法长时间空闲

例假。这一事实避免了固态硬盘刷新过程，并给系统带来额外的性能开销。但是，固态硬盘可以永久保存数据，因此无需刷新所有写入数据。因此，回写缓存策略可以提高存储性能。尽管如此，在固态硬盘出现故障的情况下，仍需要以较小的性能降级为代价进行偶尔的刷新操作。此外，固态硬盘限制的写入耐久性是另一个问题，与只读缓存相比，读写缓存的问题更大。请注意，固态硬盘设备中的随机写入速度大约比顺序写入慢 10 倍，并且会导致过多的内部碎片。许多算法[7, 21, 10, 33] 和建筑[65, 35, 44] 旨在缓解写流量并控制固态硬盘缓存中的垃圾收集过程。

随机接入优先[35] 缓存管理通过将固态硬盘拆分为读写缓存，延长了固态硬盘的使用寿命。前者维护从文件缓存中逐出的随机访问数据，目的是减少闪存损耗和写入命中。后一种是循环直写日志，以更快地响应写请求并执行垃圾收集。内核中的监控模块拦截页面级操作，并将它们发送给调度程序，调度程序是执行随机访问数据检测的用户级守护程序，并在缓存之间分发操作。在…里[44], 在缓存的两个不同部分平衡读写流量对性能和固态硬盘寿命都有好处。这些部件可以使用不同的技术，如动态随机存取存储器、非易失性存储器或固态硬盘。在部分 3.1.1 我们将 SSD 描述为 RO 缓存，其中写流量可能会进入 DRAM 缓存。在其他设计中，固态硬盘可用作硬盘的写缓存。

4.1.3 虚拟化环境中的固态硬盘缓存

在多个虚拟机以不同的输入输出模式运行的虚拟化环境中，写操作的随机性是固态硬盘闪存的一个痛点。为了减少随机写入的次数，[28] 提出了一种缓存方案，在该方案中，他们将日志结构文件系统的思想应用到虚拟磁盘层，并将随机写入转换为顺序写入。在具有多个虚拟机(其中同步随机写入占主导地位)的家庭云服务器的虚拟环境中利用顺序虚拟磁盘(SVD)，它完全使用固态硬盘驱动器，从而在提高性能的同时延长其寿命。vCacheShare[39] 是虚拟集群上的固态硬盘缓存体系结构，它只是跳过固态硬盘缓存进行写入操作。通过跟踪每个虚拟磁盘的输入输出流量并定期进行分析，vCacheShare 可以为每个虚拟磁盘的固态硬盘缓存进行最佳分区。

4.1.4 固态硬盘缓存中的重复数据消除

为了延长固态硬盘的使用寿命，重复数据消除是最有效的方法之一。一些研究[10, 7] 预防

如果内容已经缓存，则将数据写入固态硬盘。例如，[7] 通过避免虚拟化环境中的重复数据来减少对固态硬盘的写入次数，在虚拟化环境中，虚拟机的高度集成会导致大量数据重复。使用散列函数(SHA-1)，将在缓存未命中后的数据提取时计算数据签名，并且如果签名已经在缓存中，则地址将被映射到内容，并且它保存一个写操作。

缓存备份[32] 是一种用于客户端计算机和服务器中闪存缓存的串联重复数据消除机制。这种设计是对硝基的补充[31]对支持重复数据消除的缓存管理的体系结构和算法进行了一系列修改。重复数据消除的好处不仅是更好地利用缓存空间，还有助于提高命中率。此外，由于闪存设备的写入持久性有限，它还可以避免重复数据导致的过度写入，从而延迟设备的磨损。

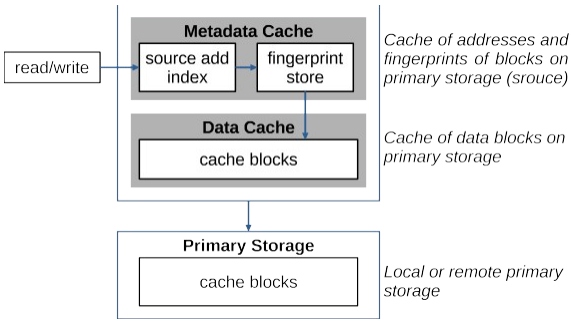


图 4:缓存备份的体系结构[32]

如图所示 4，缓存备份由两种数据结构组成:元数据缓存和数据缓存。元数据高速缓存维护用于跟踪主存储器中源地址的脚印的信息。它有一个源地址索引表和一个足迹存储。当出现读/写操作时，从源到索引的映射中获得相应的足迹索引，然后足迹到块地址的映射给出数据缓存中相应内容的块地址。由于消除了重复数据，源到缓存的地址空间具有多对一的关系，因此映射的大小不受缓存大小的限制。此外，为了防止从主存储中重新获取数据，缓存备份会保留那些已经准备好被逐出的数据块的历史指纹。缓存备份可以部署在客户端和服务器上。当它在客户机上运行时，它可以更好地隐藏重复数据的网络输入/输出，从而为应用程序获得更好的性能。在服务器端，多个客户端可能请求相同的数据，缓存备份可以帮助减少数据。请注意，在服务器端应该有缓存一致性协议

保持数据一致性的网络。虽然提出的设计都是用软件描述的，但作者声称它也可以嵌入硬件设备的闪存转换层中。所描述的系统与引用源块地址的块输入/输出级一起工作，但是它也可以在具有(文件处理程序，偏移)元组的文件系统级中使用。设计的主要部分之一是重新放置算法。有两种算法:D-LRU 和 D-ARC。详情见第节 3.3.D-ARC 算法比 D-LRU 算法复杂。D-ARC 具有抗扫描特性，可防止单次访问的数据污染缓存容量。虽然这两种算法都可以用于缓存备份，但是 D-ARC 的性能更好，而 D-LRU 的算法简单。这两种算法都具有无缓存浪费的特性，即不允许孤立地址和孤立数据块同时存在。这项研究显示了在缓存命中率、输入/输出延迟和发送到缓存设备的写入数量方面的改进。

4.1.5 固态硬盘作为 SMR 硬盘的缓存

尽管随机写入固态硬盘比顺序写入慢，但它比硬盘等叠瓦式磁记录(SMR)设备中的随机写入快一个数量级。因此，为了受益于高容量和低$/GB 的 SMR 以及固态硬盘的高性能，混合存储系统可能会将所有随机写入重定向到固态硬盘缓存，而将顺序写入留到 SMR，如中所示[62, 60, 36].

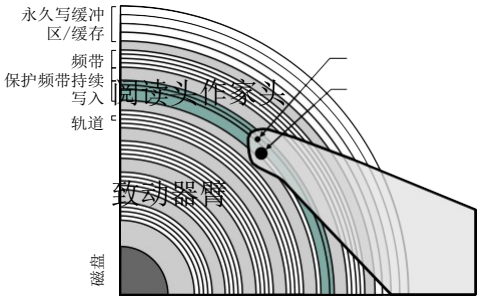


图 5:SMR 设备中的一个磁盘，正在整个磁带上写入新数据。

书写机制如图所示 5. 在 SMR 驱动器中，对磁道的写入与先前写入的磁道部分重叠，并使其变窄以提供更高的密度。这是因为写入头的物理限制，它不像读取头那样精密，在磁盘上写入时会留下更宽的轨迹。正如想象的那样，随机写入会破坏相邻的磁道，它们都应该由读-修改-写(RMW)操作重写。在 SMR 体系结构中，一个乐队是由一组连续的曲目组成的，并通过一个

狭窄的保护带。随机写作需要一整支乐队的 RMW 歌剧。因此，在将写操作写入相应的条带之前，使用一个永久缓存(闪存缓冲区或磁盘上的一些非重叠磁道)来缓冲写操作。高速公路系统[62] 是一种基于 SMR 磁盘的带感知的混合存储系统，通过将固态硬盘作为 SWD 的写缓存来提高具有持续随机写入的叠瓦写磁盘(SWD 或 SMR 磁盘)的性能。要在磁盘阵列系统中使用磁盘阵列设备，[36] 提出了三种将固态硬盘用作固态硬盘缓存的架构模型。使用此选项，RAID 系统可以以相同的性能甚至稍好的成本提供更多的存储容量。用于驱逐的部分开放区域[60] 缓存策略是固态硬盘作为 SMR 设备缓存的另一种用途。由于逻辑块地址(LBA)范围广，以及替换决策的流行，它考虑了 SMR 写入放大。简单来说，固态硬盘处理随机写入，并按顺序刷新到 SMR 设备。

3.2 非易失性存储器高速缓存

如第节所述，非易失性存储器技术的出现 2.2.1,允许在接近动态随机存取存储器的延迟下进行持久数据操作，这比固态硬盘快一个数量级。研究[29] 关于将非易失性存储器用作固态硬盘或硬盘的输入/输出缓存，揭示了当前的输入/输出缓存解决方案不能完全受益于非易失性存储器的低延迟和高吞吐量。最近的研究试图克服将非易失性存储器用作直接存取(DAX)存储设备以及将其用作固态硬盘/硬盘的高速缓存的复杂性[4,12,24,61].近年来，英特尔提供了持久内存开发套件(PMDK)[1] 它提供了几个 API 来直接从用户级别访问持久内存。银行拍摄[4] 是一个用户级库，通过向应用程序实现缓存功能并通过传递内核来降低命中延迟，从而公开了 NVM。然而，PMDK 在许多方面都优于班克肖特，因为它是最近的。与 NAND 固态硬盘相比，大多数 NVM 技术可以承受更多数量级的写入，但仍然有限。它们还提供了就地字节大小更新，这比固态硬盘中的 RMW 操作要快得多。有了这些功能，大多数动态随机存取存储器缓存策略都可以用作基于非易失性存储器的缓存，并进行重大修改，以小心管理写入流量。

分级电弧[12] cache 是一种基于 NVM 的缓存，它优化了 ARC 算法，将最近、频繁、脏和干净四种状态考虑在内，首先将缓存拆分为脏/干净页面缓存，然后将每个部分拆分为最近/频繁页面缓存。基于与 ARC 类似的机制(参见第节 3.3)，它在每一个层次上调整每个部分的大小。因此，H-ARC 将较高频率的脏页保留在缓存中的时间更长。输入输出缓存[13]还使用 NVM 作为缓冲区用于硬盘的缓存，将多个脏块合并为单个顺序写入。这种技术也用于许多其他基于非易失性存储器的设计[25,69].事务性 NVM 磁盘缓存(Tinca)[61] 旨在通过事务支持实现崩溃一致性，同时通过利用基于 NVM 的磁盘缓存避免双重写入。Tinca 利用 NVM 的字节寻址功能，维护细粒度的缓存元数据，以便在写入数据块时启用写时复制(COW)。Tinca 还使用角色切换方法，其中每个块都有一个角色，可以是正在进行的提交事务中的日志块，也可以是已完成事务中的缓冲区块。有了 COW 和角色切换两种机制，Tinca 支持提交协议，在单个事务中协作和写入多个块。

3.3 缓存替换算法

为了将最受欢迎的块保留在缓存中，设计了几种通用和特定于域的算法。一般来说，这些算法中的大多数是基于两个经验假设，即时间局部性和偏斜的流行性[21].前者假设最近使用的块很可能很快会再次被请求。后者假设与其他块相比，一些块被更频繁地访问。相应地，众所周知的最近最少使用(LRU)和最少使用(LFU)机制已经被创建，并且由于它们的简单性和 O(1)开销而被普遍用于高速缓存中的数据替换。与中央处理器不同，存储应用程序可能对时间局部性不感兴趣，因为在动态随机存取存储器中有一个页面高速缓存，可以有效地管理局部性。此外，对整个存储空间的简单搜索操作可能会刷新缓存中所有受欢迎的块，并用很少访问的块替换它们。已经提出了许多更先进的算法来解决这个问题，这些算法大多是通用的。

3.3.1 通用算法

基于频率的替换(FBR)[48]算法受益于 LRU 和 LFU 算法。它保持 LRU 排序，并主要决定一个部分中块的频率计数。其复杂度根据截面大小从 O(1)到 O(log2n)不等。使用最近信息的集合进行块引用行为识别，早期驱逐 LRU[52] 旨在为所有参考模式提供一种在线自适应替换方法。它将执行 LRU，除非许多最近获取的区块刚刚被驱逐。在这种情况下，回退算法要么驱逐 LRU 块，要么驱逐 eth MRU 块，其中 e 是预先确定的最近位置。低参考间最近集(LIRS)[23] 算法将重用度作为动态排列访问块的度量。它将缓存分为低引用间最近度(LIR)

对于大多数高排序的块和对于其他块的高内部参考相关性(HIR)。当一个 HIR 街区被关闭时，它去了 LIR，当 LIR 被填满时，来自 LIR 的排名最低的街区变成了排名最高的 HIR 街区。为了快速移除冷块，2Q[50]使用一个先进先出队列 A1in 和两个 LRU 列表 A1out 和 Am。第一个被访问的块进入 A1in，在驱逐时，它进入 A1out。重用该块会将其提升为 Am。类似地，多队列[70] 算法使用多
例如 Q0 的 LRU 队列，...，Qm 1，其中块寿命
Qj 比 Qi 长(i < j)，因为 Qi 中的一个区块至少被击中
2i 次。自适应替换缓存[38] 将缓存空间划分为 T1 和 T2，其中 T1 存储一次性访问的数据块，而 T2 保留其余的数据块。使用 B1 和 B2，T1 和 T2 的大小由划分点 P 动态调整，以在新近性和根据命中率调整的频率之间进行平衡。

3.3.2 特定领域算法

基于固态硬盘的写入性能和寿命，固态硬盘缓存通常考虑两个因素:1)在缓存中保留脏页更长时间，以避免多次获取页面；2)避免低流行块对缓存空间的污染。清洁第一 LRU[45] 将缓存空间分为干净页缓存和脏页缓存，并且仅从干净页缓存中逐出，除非没有干净页。这个基本算法试图在缓存中保留脏页更长时间，但是它忽略了跳过一次性访问页。懒人 ARC (LARC)[21] 专为固态硬盘缓存设计，可防止写入开销并延长固态硬盘的使用寿命。它过滤很少访问的块，并跳过缓存它们。与 2Q 和 ARC 类似，它考虑了这样一个事实，即最近至少被击中两次的街区更有可能受到欢迎。它有一个重影缓存来保存第一次访问的块的标识符。如果重影缓存中的数据块被重新访问，它将被视为常用数据块并放入缓存中。由于它可以防止对固态硬盘进行不必要的写入，因此也可以归类为数据热识别方法。二级电弧[15, 30]还针对固态硬盘缓存进行了优化，因为它减少了对设备的写入次数。它已在 Solaris ZFS 文件系统中使用。它使用固态硬盘作为动态随机存取存储器 ARC 缓存的二级缓存，定期用动态随机存取存储器缓存中最流行的数据内容填充它。头顶上有一个很大的空间，西维斯塔[46] 保留存储系统中每个块的未命中数信息，只允许那些未命中数大的块缓存在 SSD 中。模拟算法用于一些企业产品，如英特尔睿频存储器[37].

类似于 ARC，复制感知 ARC (D-ARC)[7, 32]包括四个 LRU 贮藏处。D-ARC 使用缓存块控制帐篷或指纹代替地址。根据脏比率的高低和块的引用计数，它将数据划分为四个组，并总是驱逐引用最少和最脏的缓存块。因此，移除的数据块更有可能是最不受欢迎的数据块，在不久的将来不会被重用，固态硬盘也不会将其弹出到不再需要的程度。这将减少固态硬盘设备的写入带宽，并节省因误驱逐而产生的额外写入。同样，到期时间驱动(ETD)[10]高速缓存算法将高速缓存块逐出延迟到其到期时间，而不是在未命中时更新高速缓存，而是在块到期时逐出块，然后从候选块列表中选择替换块。LRU[32] 是一个意识到重复的 LRU，由两个独立的 LRU 政策组成。首先，它使用 LRU 在元数据缓存中插入地址 x。其次，使用另一个 LRU 将地址 x 的相应指纹插入数据缓存。

气孔[60]是另一种特定于域的策略，在固态硬盘-SMR 混合存储系统中非常有用。它将 SMR LBA 山脉分为开放区和禁区。重新放置策略可能只驱逐开放区域中的脏块。写回的块存储在 SMR 写缓冲器或永久高速缓存中，用于随后写入相应的带。开放区域会定期更改，以覆盖 SMR LBA 范围内的所有脏区块。这种算法有助于避免在随机带上写入数据，因为这会严重破坏性能。

4 存储分层解决方案

过去，高端和低端硬盘分别用作性能和容量层。如今，在多层存储系统中使用了具有不同特性和容量的多种类型的存储介质。缓存和分层的主要区别在于，在缓存系统中，数据的拷贝保存在缓存中，而在分层系统中，原始数据通过升级和降级两个操作在多个层之间迁移。数据根据应用程序需求和可用层的特征进行分类，通常分为热层和冷层。热数据重新进入性能层，而冷数据留在容量层。考虑到多种因素，如运行状况、传输速度等。，可能有两层以上。

数字 6 展示了由四个阶段组成的通用存储分层机制。在数据收集阶段，系统收集决策所需的信息。IO 模式的应用配置文件可以在线或离线获取。在线分析模块可能会在应用程序运行时收集 IO 信息，代价是潜在的性能开销。当涉及到用户时，例如个人计算机或虚拟环境云系统，这种机制非常有用。离线分析

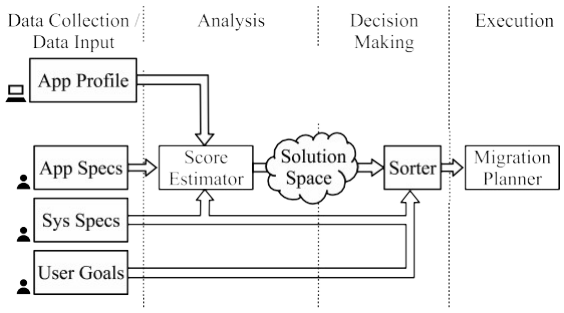


图 6:常规存储分层阶段

另一方面，模块在运行之前获得应用程序 IO pro- file。这种分析机制适用于集群分析应用程序，在这些应用程序中，除了可预测的正在运行的应用程序之外，没有任何 ran- dom 参数会干扰 IO 路径。其他一些信息，如应用程序/系统规格，也可以由用户或机器同时输入分层算法。在分析阶段，系统评估几个可能的计划或模型，并以解决方案空间的形式生成建议列表。一些分层算法可能会跳过这一阶段，通过一些分析直接找到答案。解决方案空间由成本函数或性能模型评估的不同情况下的几个估计组成(例如，在各层之间特定的数据分布下运行特定的应用程序或整个系统)。每个解决方案都有一个成本估算，稍后将用于下一阶段的决策。在这个阶段，排序算法可能足以决定哪个迁移计划值得采用。根据目标、每个计划的分数以及它们的成本，分层算法决定了是否向哪个方向迁移一大块数据。

4.1 固态硬盘作为性能层

中提供了对可用存储类型的全面研究[24].它将 Micron 全 PCM 固态硬盘原型与 eMLC 闪存固态硬盘在性能方面进行了比较，并将其评估为分层存储系统的一个有前途的选项。它采用模拟方法，估计/获得每个设备的性能特征，测试 PCM 固态硬盘、eMLC 固态硬盘和硬盘的每种可能组合。虽然现在市场上有来自英特尔和 Micron 的 Optane 固态硬盘，并且我们知道它比过时的全 PCM 固态硬盘 pro- totype 提供更好的性能，但本文假设 PCM 固态硬盘的写入操作比 eMLS 固态硬盘慢 3.5 倍。有了这一假设，再加上非常简单的基于 IOPS 的动态分层算法，他们展示了在多层存储系统中使用 PCM SSD 作为企业解决方案在各种实际工作负载中的优势。

在线操作系统级数据分层[49] 基于访问频率、数据类型(元数据或用户数据)、访问模式(随机或顺序)和读/写操作频率，在多个层之间有效地分配数据。使用加权优先级函数，OODT 根据数据块的随机程度、读取比率和请求类型对每一层的数据块进行排序。OODT 可以解释固定大小的请求(4KB)。如果请求比这个大，它将被分解成几个小的子请求，并在一个叫做 dispatcher 的模块中独立处理它们。通过使用映射表，可以将所有数据块追踪到层号和物理块索引。为了实现在线迁移，它获取数据块的统计信息，并将其保存在访问表中，该表由数据修补程序更新。面向对象技术和其他分层方案最重要的部分是优先级计算(在其他技术中可以称为评分、排序或匹配)，它为每个数据确定匹配的层。OODT 使用一个简单的加权线性公式，四个输入分别为 Paccess、Prandom、Pread 和 Pmetadata，计算潜在迁移的优先级。

云分析存储分层[8]，听起来，是一个用于云中数据分析应用程序的存储分层解决方案。借助离线工作负载分析，CAST 为不同云存储服务上的每个租户建立作业性能预测模型。然后，它将获得的预测模型与工作负载规格及其目标相结合，以执行经济高效的数据放置和存储预测计划。他们将数据放置和存储资源调配问题建模为一个非线性优化问题，在这个问题中，他们根据完成时间和成本最大化租户利用率。CAST 的增强版本也在中提出[8] 它被称为 CAST++并为 CAST 增加了数据重用模式和工作流意识。

基于虚拟化环境中的实测 IOPS，自动分层[66]在全闪存存储系统中，将虚拟磁盘文件(VMDK)从一层动态迁移到另一层。它使用采样机制来估计在其他层上运行虚拟机的 IOPS。基于这一测量和成本，它根据分数对所有可能的变动进行分类。对于每个 VMDK，虚拟机管理程序上的守护程序会收集与 IO 相关的统计信息，包括延迟注入测试的 IOPS 结果，以类似于每个采样周期结束时较慢的层。为了模拟更快的层，自动化利用了线性回归模型的优势。如果 IOPS 没有通过减慢 IO 进程来改变，并且队列中有一个虚拟机在等待性能层，则降级会将 VMDK 提升到容量层，并让另一个 VMDK 通过提升它来接管性能层。

4.2 作为性能层的非易失性存储器

NVMFS[47] 是一种混合文件系统，通过利用

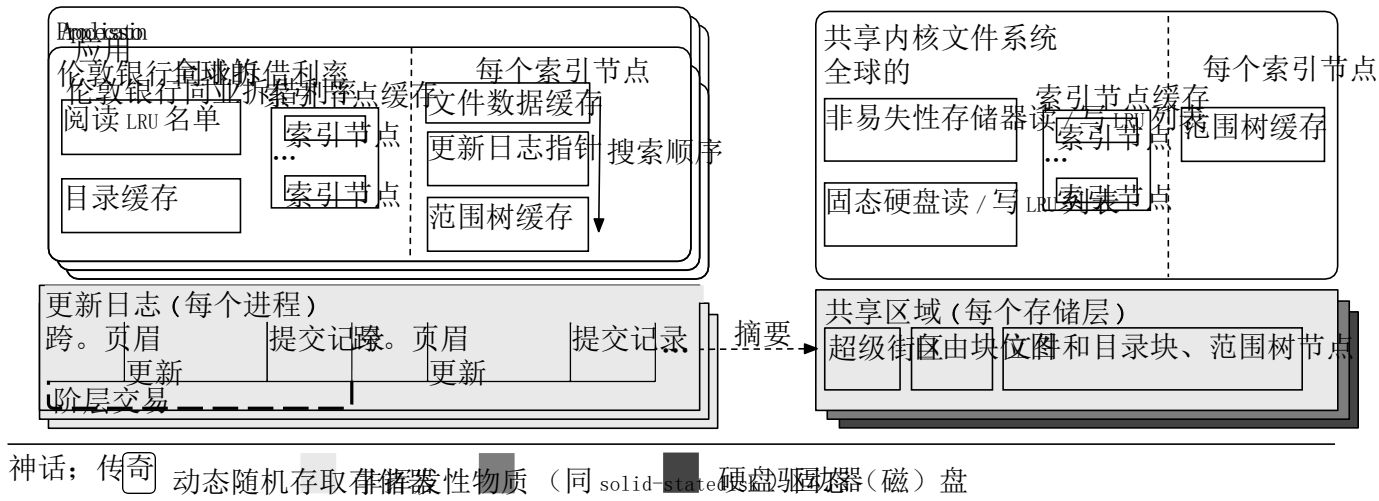


图 7:地层设计[25]. LibFS 将写入定向到更新日志，并为共享区域的读取提供服务。文件数据缓存是只读缓存，包含来自固态硬盘或硬盘的数据。

辅助非易失性存储器设备的字节寻址能力。这个文件系统的关键特征是它重定向 NVM 上的小随机 IOs，其中包括元数据和热文件数据块。该方案有助于减少固态硬盘的写入流量，从而提高固态硬盘的耐用性。该技术将文件系统级别的随机写入转换为固态硬盘级别的顺序写入。它将具有相同更新可能性的数据分组，并提交单个大型固态硬盘写入请求。

NVMFS 包含两个 LRU 列表:脏和干净。脏 LRU 列表吸收非易失性随机存取存储器中的更新。当页面写回固态硬盘设备时，它会从脏列表移动到干净列表。NVMFS 动态调整脏和干净的 LRU 列表。一旦 NVRAM 使用率达到 80%，后台线程就会开始刷新脏列表中的数据，并将它们移到干净的 LRU 列表中，直到使用率降至 50%。NVRAM 在 SSD 上有不覆盖策略:定期清理内部碎片，将多个部分扩展区集成到一个中，并回收可用空间。

作者通过 5 个步骤来解释文件系统的一致性。1:检查 NVRAM 使用率是否超过 80%；2:如果是，将脏 LRU 列表中随机的小 IOs 分组到大(512K)区中；3:然后，依次将扩展区写入 SSD(在 FTL 更好的块擦除)；4:将刷新的页面插入干净的 LRU 列表；最后 5:通过在 page_info 结构中记录新的数据位置来更新元数据。因此，当崩溃发生在任何时候，它都可以恢复。

为了防止段清理不一致，NVMFS 在碎片整理期间对事务进行扩展，类似于日志结构文件系统中的事务。选择候选扩展区后，它会将该扩展区的有效数据块迁移到 NVRAM，然后更新相应的信息节点。当信息节点更新时，它会释放固态硬盘中的空间。

即使在过程中发生崩溃，数据也将始终保持一致。

地层[25]是一个多层文件系统，利用非易失性存储器作为性能层，固态硬盘/硬盘作为容量层。它由两部分组成:内核函数和 LibFS。为了启动 Strategy，应用程序需要用 LibFS 重新编译，LiBfs 重新实现了标准的 POSIX 接口。在内核端，内核文件系统应该运行，以授予应用程序对共享存储区域的访问权限，共享存储区域是 NVM、固态硬盘和硬盘的组合。它使用 NVM 的字节寻址能力来合并日志，并将它们迁移到较低层，以最大限度地减少写入放大。文件数据只能在 Strata 中的 NVM 中分配，并且只能从较快的层迁移到较慢的层。Strata 的剖析粒度是一个页面，增加了记账开销，浪费了文件访问的局部性信息。

Strata 通过将登录和消化任务分别分离和分解到用户空间和内核空间来实现快速写操作。内核允许 LibFS 直接访问 NVM 以获取自己的私有更新日志，并允许共享区域进行只读操作，如图所示 7. 内核通过多个线程并行执行摘要操作。此操作的一个好处是，尽管对最新日志的初始写入具有随机性和较小的大小，但它们可以合并并按顺序写入共享区域，这有助于最大限度地减少碎片和元数据开销。这也有助于高效的闪存擦除和叠瓦写入操作。

为了保证崩溃的一致性，LibFS 使用了一个持久的 Strata 事务单元，它为应用程序更新日志提供了 ACID 语义。为了实现这一点，Strata 在一个或多个 Strata 事务中包装每个 POSIX 系统调用。数字 7 代表地层设计和伦敦银行同业拆借利率

和内核组件。

4.3 作为元数据层的非易失性存储器

在像 Ext4 这样的日志文件系统中，元数据会更新通常非常小(例如，索引节点大小为 256 字节)。尽管修改信息节点需要少量的写操作，但由于存储设备的块大小操作，整个信息节点块(例如，4K)将被替换。近年来，非易失性存储器由于其通过存储器总线连接的特性而引起了广泛关注。这个特性意味着中央处理器可以发布字节级(高速缓存行大小)持久更新。

文件系统元数据加速器[6]由于非易失性存储器的访问规模小，受欢迎程度高，因此它取消了数据和元数据的输入/输出路径，并使用非易失性存储器来存储文件系统元数据。元数据永久存储在 NVM 中，默认情况下，从不定期刷新回磁盘。对元数据的所有更新都是就地更新。不仅在元数据更新操作过程中断电后，元数据会损坏，而且作者认为，由于 NVM 和块设备之间的性能差距，数据更新落后于元数据更新，元数据更新一旦更新就会在 NVM 中持续。由于字节大小版本的实现非常复杂和繁琐，并且块大小版本化会造成写入放大和 NVM 空间浪费，因此 FSMAC 使用细粒度版本化(索引节点大小，即 128 字节)，可以在合理的实现和空间成本下保持一致性。

为了解决数据和元数据的写入顺序问题，同时又不破坏由于 NVM 的字节寻址能力而获得的性能，FSMAC 使用了细粒度版本控制和事务的轻量级组合。元数据的原始版本是在更新之前创建的，以安全地覆盖崩溃。只有在更新事务成功完成后，才会将其删除。之后，整个文件系统将保持一致。

利用这个机会，陈等人提出了在 NVM 上进行细粒度的元数据日志记录[5]. 虽然它与分层和缓存解决方案没有直接关系，但使用 NVM 来保留一部分存储数据是一种分类问题，这是分层方法的基础。

与传统的日志文件系统相反，在基于 NVM 的细粒度日志文件系统中，在 DRAM 中的页面缓冲区中修改的索引节点块以事务的形式保存到磁盘[5],只有修改过的索引节点被链接在一起并保存在 NVM 中(图 8).使用缓存刷新指令和内存栅栏，它提供了有序写入的一致性。不使用总计 8K 的大描述符和提交(或撤销)块，而是引入了新的数据结构 TxnInfo，它包含列表中修改过的索引节点的数量(计数)，以及用于在恢复时间内识别 TxnInfo 的幻数。

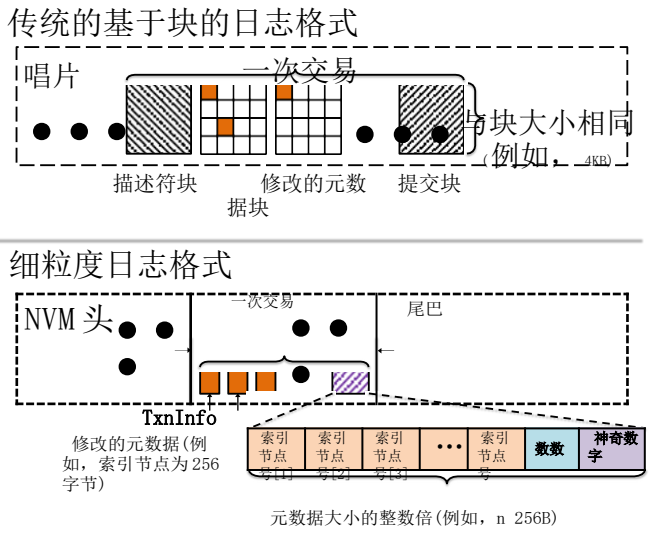
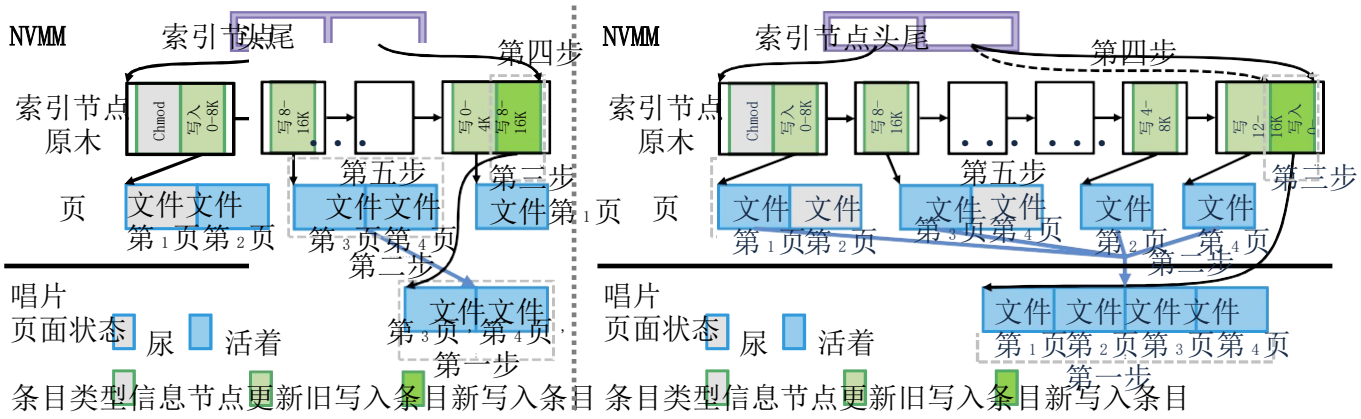


图 NVM 上的细粒度元数据日志格式[5]

NVM 中的日志区是一个带有头指针和尾指针的环形缓冲区。写入它由三个步骤组成:1) memcpy 修改从 DRAM 到 NVM 的索引节点；2)刷新相应的高速缓存线并发布存储器屏障；和
3) 使用原子 8 字节写操作自动更新 NVM 中日志区的尾指针，刷新其缓存行，并发出内存屏障。

在传统的日志文件系统中，提交运行事务(已修改信息节点块的链接列表)是由预定义的计时器或预定义数量的已修改信息节点块触发的。在细粒度日志中，当预定义的计时器启动时，类似于传统的文件系统，提交过程开始。这个过程的未设置也由修改过的索引节点的数量控制，因为 TxnInfo 可以保存有限数量的修改过的索引节点的信息。提交过程从将所有修改过的索引节点从正在运行的事务重新链接到提交事务开始，以便正在运行的事务可以接受新的修改过的索引节点。然后，所有修改过的索引节点从尾部开始被保存到 NVM 中，然后计算出 TxnInfo。此后，刷新相应的高速缓存行，并发出存储器围栏。最后，尾部指针将自动更新，确认事务已提交。事实上，在这个过程中数据是一致的，即使中间发生了崩溃，因为尾部控制着数据的可见性。与传统日志记录相比，这种方法最多可将事务写入减少 99%。

为了防止过长的日志降低性能，文件系统通常会定期使用检查点。细粒度日志记录每 10 分钟触发一次检查点，或者在 NVM 利用率达到 50%时触发一次检查点。像传统的日志文件系统一样，它接管修改后的索引节点块列表，并一个接一个地写入块。然后，



(a) 齐格拉特文件结构与基本迁移(二)群体迁移

图 9:齐格拉特的迁移机制[69].Ziggurat 使用其基本迁移和组迁移机制在各层之间迁移文件数据。蓝色箭头表示数据移动，而黑色箭头表示指针。

它通过使头尾指针相等来丢弃 NVM 中的日志，这保证了可恢复性，因为当崩溃发生时，我们在恢复中仍然有 NVM 中修改过的索引节点。

细粒度日志记录中的恢复过程从 NVM 中的尾部向后开始。它将相应的索引节点块检索到动态随机存取存储器。过时的信息节点数据块通过在数据块内应用修改后的信息节点来更新。在动态随机存取存储器中更新所有信息节点块后，它会将它们刷新回磁盘。最后，让头部和尾部原子一样。保证一致性类似于检查点过程。

4.3.1 我们的多层文件系统:齐格拉特[2]

Ziggurat[69]是一个基于 NVM 的多层文件系统一个跨越 NVNM 和磁盘的分层文件系统，它是在我们的研究小组中开发的。该论文发表在第 17 届 USENIX 文件和存储技术会议(FAST '19)上。它基于我们众所周知的基于 NVM 的文件系统 NOVA[63].Ziggurat 通过文件写入和数据迁移期间的智能数据放置，利用了 NVM 的优势。Ziggurat 包括两个位置预测器，用于分析文件写入序列，并预测传入的写入是否既大又稳定，以及文件更新是否可能同步。然后，它根据预测将传入的写入引导到最合适的层:对同步更新的文件的写入转到 NVM 层，以最小化同步开销。小型随机写入也会转到 NVM 层，以完全避免随机写入磁盘。异步更新的剩余大型顺序写入

2 本节的部分内容摘自 FAST'19 接受的原始论文 Parts[69]

文件转到磁盘。Ziggurat 寻求以下五个主要设计目标。

将写入发送到最合适的层。尽管 NVM 是 Ziggurat 中速度最快的层，但文件写入不应该总是流向 NVM。NVM 最适合小更新(因为对磁盘的小写入很慢)和同步写入(因为 NVM 具有更高的带宽和更低的延迟)。但是，对于较大的异步写入，以磁盘为目标更快，因为 Ziggurat 可以比写入 NVM 更快地缓冲 DRAM 中的数据，并且写入磁盘可以在后台进行。Ziggurat 使用其同步性预测器来分析对每个文件的写入顺序，并预测未来的访问是否可能是同步的(即应用程序是否会很快调用 fsync)。

仅迁移冷文件中的冷数据。在迁移过程中，Ziggurat 以冷文件的冷部分为目标。访问不均匀的文件中的热文件和热数据仍保留在较快的层中。当快速层的使用率高于阈值时，Ziggurat 会选择平均修改时间最早的文件进行迁移。在每个文件中，Ziggurat 迁移比平均时间更早的数据块。除非整个文件是冷的(即其修改时间不是最近的)，在这种情况下，我们会迁移整个文件。

NVM 空间利用率高。Ziggurat 充分利用 NVM 空间来提高性能。Ziggurat 使用 NVM 来吸收同步写入。Ziggurat 基于应用程序的读写模式，对非易失性存储器使用动态迁移阈值，从而充分利用非易失性存储器来高效地处理文件读写。当运行以读取为主的工作负载时，我们还实施反向迁移，将数据从磁盘迁移到 NVM。

分组迁移文件数据。为了最大化磁盘的写入带宽，Ziggurat 尽可能地执行磁盘迁移。放置策略确保大多数小的随机写入都进入 NVM。然而，迁移

这些直接写入磁盘的小条目会受到驱动器随机访问性能差的影响。为了提高迁移效率，Ziggurat 将相邻文件数据合并成大块进行移动，以利用顺序磁盘带宽。

高可扩展性。Ziggurat 扩展了 NOVAs 的每 CPU 存储空间分配器，以包括所有存储层。它还使用每 cpu 迁移和页面缓存回写线程来提高可扩展性。

数字 9a 显示了 Ziggurat 如何将写条目从 NVM 移动到磁盘的基本过程。第一步是在磁盘上分配连续的空间来保存迁移的数据。Ziggurat 将数据从 NVM 复制到磁盘。然后，它向索引节点日志追加一个新的写条目，其中包含迁移数据块的新位置。之后，更新 NVM 中的日志尾和 DRAM 中的基数树。最后，Ziggurat 释放了旧的 NVM 块。

数字 9b 展示了避免细粒度迁移的组迁移步骤，以提高效率并最大化磁盘的顺序带宽。它们类似于修改写条目。在步骤 1 中，它在较低层分配大块数据块。在第 2 步中，它通过一次顺序写入将多个页面复制到较低层。之后，它会追加日志条目，并更新 inode 日志尾部，从而提交组迁移。旧页面和日志随后被释放。理想情况下，组迁移大小(组迁移的粒度)应该设置为接近真实的输入/输出大小，这样应用程序就可以从磁盘顺序读取文件数据。此外，它不应超过 CPU 缓存大小，以最大限度地提高从磁盘加载写条目的性能。

简而言之，齐格拉特弥合了基于磁盘的存储和基于非易失性存储器的存储之间的差距，并为应用程序提供了高性能和大容量。

5 结论

存储技术的多样性及其不同的特性使它们各自适合一组存储需求。在软件方面，不断扩展的数字信息云需要具有高性能存储系统的大规模企业数据服务器。虽然旧的设计良好的存储技术(如硬盘)以相对较低的成本提供了大空间和高密度，但新技术(如固态硬盘和非易失性存储器)以更高的成本提供了超快且可靠的输入输出工作流程。人们普遍希望新技术具有高存储容量和低成本的高性能。尽管处理器的开发速度远高于存储技术的开发速度，但缓存和分层等软件解决方案吸引了专家的注意力，以克服上述限制。在本次调查中，我们广泛调查了针对高性能存储系统的几种缓存和分层解决方案。我们观察到，尽管有几种缓存和分层建议使用固态硬盘作为性能层，年轻的非易失性存储器技术没有得到足够的重视，不能用于这类系统。这并不出人意料，因为这项技术是最近开发的，并且这种类型的第一批产品在本出版物发布前几个月才上市。顺便说一下，我们还研究了一些最近关于使用非易失性存储器作为性能层的科学论文，我们还介绍了 Ziggu- rat，一种使用非易失性存储器作为性能层来覆盖固态硬盘和硬盘的长延迟的多分层文件系统。

参考

[1] PMDK。https://pmem.io/pmdk/。

[2] 安德森和斯旺森重新思考数据中心的闪存。IEEE 微 30，4 (2010)，52 - 54。

[3] 让我们来谈谈非易失性存储器数据库系统的存储和恢复方法。《2015 年美国计算机学会数据管理国际会议论文集》(2015 年)，美国计算机学会，第 707-722 页。

[4] 在快速非易失性存储器中缓存慢速存储。《第一届非易失性存储器/闪存与操作系统和工作负载交互研讨会论文集》(2013 年)，美国计算机学会，第 1 页。

[5] 陈，c，杨，j，魏，q，王，c，和薛，
米（meter 的缩写））nvm 上的细粒度元数据日志。大容量存储系统和技术(MSST)，2016 年第 32 届研讨会(2016 年)，IEEE，第 1-13 页。

[6] 一个具有非易失性存储器的文件系统元数据加速器。在大容量存储系统和技术(MSST)中，2013 年 IEEE 第 29 届研讨会(2013 年)，IEEE，第 1-11 页。

[7] 虚拟化环境下基于重复感知固态硬盘的主存储高速缓存架构。IEEE 系统杂志 11，4 (2017)，2578 - 2589。

[8] 郑裕彤，伊克巴尔，硕士，美国古普塔，和巴特，
A.R. Cast:为云中的数据分析进行存储分层。《第 24 届高性能并行和分布式计算国际研讨会论文集》(2015 年)，美国计算机学会，第 45-56 页。

[9] 英特尔 CLARKE，Micron 推出“批量切换”ReRAM。https://www.eetimes.com/document.asp?doc_id=1327289, 2015.

[10] 高速缓存:一种到期时间驱动的高速缓存方案，使基于固态硬盘的读高速缓存经久耐用且经济高效。《第十二届美国计算机学会国际计算前沿会议论文集》(2015 年)，美国计算机学会，第 26 页。

[11] 持久性存储器系统软件。《第九届欧洲计算机系统会议记录》(2014 年)，美国计算机学会，第 15 页。

[12] 固态硬盘基于非易失性存储器的高速缓存策略。大容量存储系统和技术(MSST)，2014 年第 30 届研讨会(2014 年)，IEEE，第 1-11 页。

[13] 范，z，哈格杜斯特，a，DU，D. H，和 VOIGT，D.输入/输出高速缓存:一种基于非易失性存储器的缓冲高速缓存策略，用于提高存储性能。在计算机和电信系统(MASCOTS)的建模、分析和仿真中，2015 年 IEEE 第 23 届(2015)国际研讨会，IEEE，第 102 - 111 页。

[14] 3D XPoint 会让它对抗 3D NAND 吗？https://wikibon.com/3d-xpoint-falters/，2017.

[15] 格雷格，B. ZFS L2ARC。甲骨文博客 2008 年 7 月 22 日。http://www.brendangregg.com/blog/2008-07-22/zfs-l2arc.html。

[16] HAMZAOGLU，f .，ARSLAN，u .，BISNIK，n .，GHOSH，s .，LAL，M. B .，LINDERT，n .，METERELLIYOZ，m .，OSBORNE，R. B .，PARK，j .，TOMISHIMA，s .，WANG，y .，AND ZHANG，K. 13.1 A 1Gb 2GHz 嵌入式 DRAM 采用 22nm 三栅 CMOS 技术。2014 年 IEEE 国际固态电路会议技术论文摘要(ISSCC)(2014 年 2 月)，第 230-231 页。

[17] 通过行分条减少多级内存的访问延迟。《第 41 届计算机体系结构国际年会论文集》(皮斯卡塔韦，美国新泽西州，2014 年)，ISCA '14，IEEE 出版社，第 277-288 页。

[18] 通过行分条降低多层通信脉码调制的访问延迟。《第 41 届计算机体系结构国际年会论文集》(皮斯卡塔韦，美国新泽西州，2014 年)，ISCA '14，IEEE 出版社，第 277-288 页。

[19] SPCM:条纹相变存储器。架构和代码优化的 ACM 交易(TACO) 12，4 (2016)，38。

[20] HOSOMI，m .，YAMAGISHI，h .，YAMAMOTO，t .，BESSHO，k .，HIGO，y .，YAMANE，k .，YAMADA，h .，SHOJI，m .，HACHINO，h .，FUKUMOTO，c .，ET AL .一种具有自旋扭矩转移磁化开关的新型非易失性存储器:自旋随机存储器。在电子设备会议，2005 年。IEDM 技术文摘。IEEE 国际(2005)，IEEE，第 459-462 页。

[21] 黄，s，魏，q，冯，d，陈，j，和陈，c .用惰性自适应替换改进基于闪存的磁盘缓存。存储上的 ACM 交易(TOS) 12，2 (2016)，8。

[22] 英特尔。英特尔 optane 技术，2018。https://www.intel.com/content/www/us/en/architecture-and-technology/intel-optane-technology.html。

[23] LIRS:一种有效的低引用间最近集替换策略来提高缓存性能。在 2002 年美国计算机学会计算机系统测量和建模国际会议记录(美国纽约，纽约，2002)中，美国计算机学会，第 31-42 页。

[24] 金，h，塞沙德里，s，迪基，C. L，和邱，长度评估企业存储系统的相变存储器:缓存和分层方法的研究。存储上的 ACM 交易(TOS) 10，4 (2014)，15。

[25] 《跨媒体文件系统》。《第 26 届操作系统原理研讨会论文集》(2017 年)，美国计算机学会，第 460-477 页。

[26] 将相变存储器设计为可扩展的 dram 替代品。在 ACM SIGARCH 计算机体系结构新闻(2009)，第 37 卷，ACM，第 2-13 页。

[27] 相变技术与主存储器的未来。IEEE 微 30，1 (2010)。

[28] 李博士、闵博士和杨博士，即用于高性能家庭云服务器的有效固态硬盘缓存。在 IEEE 消费电子国际会议(ICCE) (2015)上，第 152-153 页。

[29] 李，g，李，h，g，李，j，金，b，s，和敏，南基于非易失性存储器的块输入输出缓存的实证研究。《第九届亚太系统研讨会论文集》(美国纽约州纽约市，2018年)，美联社系统’18，美国计算机学会，第11:1－11:8页。

[30] 闪存存储器。ACM 51的通信，7 (2008)，47－51。

[31] 一种容量优化的主存储固态硬盘高速缓存。在USENIX年度技术会议(2014)上，第501-512页。

[32] 李，w，JEAN-BAPTISE，g，RIVEROS，j，NARASIMHAN，g，ZHANG，t，和ZHAO，M. Cachededup:闪存缓存的线内重复数据消除。在FAST (2016)中，第301-314页。

[33] 梁，于，柴，于，鲍，于，陈，于，刘，Y.弹性队列:用于缓存替换算法的通用ssd寿命延长插件。在第九届美国计算机学会国际系统与存储会议(2016)的会议记录中，美国计算机学会，第5页。

[34] 刘，y，GE，x，HUANG，x，AND DU，D. H. Molar:一种经济高效、高性能的混合存储缓存。在群集计算(Cluster)中，2013年IEEE国际会议(2013年)，IEEE，第1－5页。

[35] 刘延英，黄建杰，谢建忠，曹庆飞:一种改进基于固态硬盘的磁盘高速缓存的随机访问优先高速缓存管理。2010年IEEE第五届网络、架构和存储国际会议(2010年7月)，第492-500页。

[36] 混合叠瓦记录磁盘阵列系统的设计与实现。在第14届国际普及智能和计算会议(PiCom) (2016)上，IEEE，第937-942页。

[37] 主流计算机系统存储体系中的非易失性磁盘缓存。存储上的ACM事务(TOS) 4，2 (2008)，4。

[38] 一种自调整、低开销的替换高速缓存。在USENIX年度技术会议上，《总路线》(2003年)，第3卷，第115-130页。

[39] 虚拟化环境中的自动化服务器闪存空间管理。在USENIX年度技术会议(2014)上，第133-144页。

[40] MICRON。3d-xpoint技术，2017年。https://www.micron.com/products/advanced-solutions/3d-xpoint-technology。

[41] 一种具有RAID1和RAID5的多层RAID存储系统。并行和分布式处理研讨会，2000年。IPDPS 2000。诉讼程序。第14届国际会议(2000年)，IEEE，第663-671页。

[42] 通过早期读取和涡轮读取减少相变存储器的读取延迟。在高性能计算机架构(HPCA)中，2015年第21届IEEE国际研讨会(2015年)，IEEE，第309-319页。

[43] 混合存储系统:体系结构和算法综述。IEEE ACCESS 6 (2018)，13385－13406。

[44] 减少缓存以获得更好的性能:平衡混合存储系统中闪存缓存的缓存大小和更新成本。在FAST (2012)中，第12卷。

[45] 《闪存的替换算法》。《2006年国际嵌入式系统编译器、体系结构和综合会议论文集》(2006年)，美国计算机学会，第234-241页。

[46] 一种高选择性、集成级的磁盘高速缓存，以实现高性价比。在ACM SIGARCH计算机建筑新闻(2010)，第38卷，ACM，第163-174页。

[47] 一种改进nand-flash ssd中随机写入的混合文件系统。在大容量存储系统和技术(MSST)中，2013年IEEE第29届研讨会(2013年)，IEEE，第1-5页。

[48] 使用基于频率的替换的数据高速缓存管理，第18卷。澳大利亚竞争委员会，1990年。

[49] 使用在线工作负载表征的操作系统级数据分层。《超级计算杂志》71，4 (2015)，1534－1562。

[50] 2Q:一种低开销的高性能缓冲管理替代方案。《第20届超大型数据库国际会议论文集》(1994年)。

[51] 了解苹果的融合驱动，2012年10月。https://www.anandtech.com/show/6406/understanding-apples-fusion-drive。

[52] 简单有效的自适应页面替换。在 ACM SIGMETRICS 绩效评估评论(1999)中，第 27 卷，ACM，第 122-133 页。

[53] SMULLEN，C. W．，MOHAN，v．，NIGAM，a．，GURUMURTHI，s．，和 STAN，M. R．.放松非易失性，实现快速、高能效的 stt-ram 缓存。在高性能计算机架构(HPCA)中，2011 年 IEEE 第 17 届国际研讨会(2011 年)，IEEE，第 50-61 页。

[54] 具有动态刷新方案的多保留级 STT- RAM 高速缓存设计。《第 44 届 IEEE/ACM 国际微体系结构研讨会论文集》(2011 年)，ACM，第 329-338 页。

[55] 共享混合存储集群中支持服务级别协议的数据迁移。Clus- ter Computing 18，4 (2015)，1581－1593。

[56] 英特尔推出高达 512GB 的 Optane 内存:阿帕奇通行证来了！ https://www.anandtech.com/show/12828/intel-launches-optane-dimms-up-to-512gb-apache-pastse-r.isIn-hPeerrfeo。军事计算和通信

[57] 计算机硬件-软件架构。普伦蒂斯霍尔专业技术参考，1986 年。

[58] 机遇的数字宇宙:丰富的数据和物联网日益增长的价值。IDC 分析未来 16 (2014)。

[59] 《轻量级持久内存》。在第十六届编程语言和操作系统架构支持国际会议的会议记录中，ASPLOS XVI (2011)，第 39 卷，ACM，第 91-104 页。

[60] 更大，更便宜，但更快:固态硬盘-SMR 混合存储由新的面向 SMR 的缓存框架推动。在第 33 届国际大规模存储系统和技术会议(MSST'17) (2017)会议记录中。

[61] 具有高性能和崩溃一致性的事务型 nvm 缓存。《高性能计算、网络、存储和分析国际会议论文集》(2017 年)，美国计算机学会，第 56 页。

[62] 肖，魏，董，何，马，李，刘，赵，张，问:HS-BAS:一个基于带状写磁盘的带感知的混合存储系统。在 2016 年 IEEE 第 34 届国际计算机设计会议(ICCD) (2016)上，IEEE，第 64-71 页。

[63] 一种用于混合易失性/非易失性主存储器的日志结构文件系统。在 FAST (2016)中，第 323-338 页。

[64] 高性能单片机/nand 闪存混合固态硬盘的单片机特性和非易失性高速缓存算法的最佳组合。在硅纳米电子研讨会(SNW)，2016 年电气和电子工程师协会(2016 年)，电气和电子工程师协会，第 88-89 页。

[65] 杨，j，PLASSON，n，GILLIS，g，TALAGALA，n，SUNDARARAMAN，s，和 WOOD，R. HEC:提高基于闪存的高性能高速缓存设备的耐用性。《第六届国际系统和存储会议论文集》(2013 年)，美国计算机学会，第 10 页。

[66] YANG，z，HOSEINZADEH，m，ANDREWS，a，MAYERS，c，EVANS，D. T，BOLT，R. T，BHIMANI，j，MI，n，AND SWANSON，S. Autotiering:多层全闪存数据中心的自动数据放置管理器-

会议(IPCCC)，2017 年 IEEE 第 36 届国际(2017 年)，IEEE，第 1-8 页。

[67] 多级单元相变存储器的有效数据映射和缓冲技术。架构和代码优化的 ACM 交易(TACO) 11，4 (2014)，40。

[68] 面向分布式系统的高性价比缓存中间件。国际大数据情报杂志 3，2 (2016)，92－110。

[69] 非易失性主存储器和磁盘的分层文件系统。在 FAST (2019 年)中。

[70] 二级缓存的多队列替换算法。在 USENIX 年度技术会议上，通用赛道(2001)。