新型高速互联协议 CXL

丁金戈[1)]

[1)](华中科技大学 计算机科学与技术学院, 武汉 430074)

摘 要 大数据、AI 大模型等应用在当前的计算架构下面临内存扩展的巨大挑战。支持异构计算/存储设备互连的新型高速互连协议如 CXL 等在统一大内存扩展和池化方面，具有广阔的应用前景，是实现下一代数据中心最佳资源利用的重大变革性技术。本文首先分析了当前计算和存储领域面临的四个挑战，为了解决这些挑战提出了新型高速互联协议 CXL，文章简要介绍了 CXL 协议的内容，指出当前 CXL 在分级内存、分离式内存、池化内存、数据中心以及 CXL-Flash 等领域的研究现状，最后分析给出 CXL 在未来潜在的研究方向。

关键词 互联；CXL；分布式内存；存储通道技术

CXL: A Novel High-Speed Interconnect Protocol

Jinge Ding[1)]

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

Abstract The current computing architecture faces significant challenges in memory expansion for applications such as big data and AI large models. New high-speed interconnect protocols that support heterogeneous computing/storage device interconnection, such as CXL, have broad application prospects in unified large memory expansion and pooling. They represent a major transformative technology for achieving optimal resource utilization in next-generation data centers. This article first analyzes four challenges in the computing and storage domains. To address these challenges, it proposes the new high-speed interconnect protocol, CXL. The article provides a brief introduction to the CXL protocol and highlights its current research status in areas such as hierarchical memory, disaggregated memory, pooled memory, data centers, and CXL-Flash. Finally, it analyzes potential research directions for CXL in the future.

Key words Interconnect；CXL；Distributed memory；Storage channel technology

1 引言

随着计算机系统的发展和各种应用需求的增加，如大数据分析、通用大模型等。产生了“内存墙”和“IO 墙”问题，内存墙问题是指内存性能严重制约 CPU 性能的发挥，内存带宽成为瓶颈，在计算和存储之间存在性能鸿沟 其中一个典型的例子如图 1 所示，近几年 NLP（Natural Language Processing）模型的大小以 14.1 x/year 的速度增长，而 GPU 所能提供的内存增长速度仅为 1.3 x/year。IO 墙是指当数据量庞大时，内存不能满足存储需求时会访问外部存储设备，在通过 IO 方式访问数据时，会带来额外的时延开销，拖累整体性能。

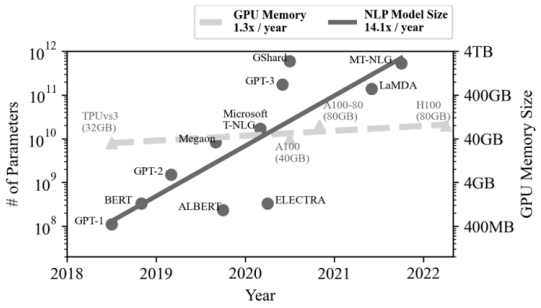

图 1 近年内存容量与模型大小增长变化趋势

内存系统的性能和容量已成为限制系统整体性能的瓶颈，支持异构计算/存储设备互联的新型高速互联技术如 Computer Express Link（CXL）、NVIDIA NVLink（NVLink）、Remote Direct Memory Access（RDMA）等在统一大内存拓展和池化方面，

最终修改稿日期：2023年11月25日；丁金戈，男，2001年生，硕士，计算机学会(CCF)会员（I2597G）,主要研究领域为计算机系统结构、内存型存储系统. E-mail: jingeding@hust.edu.cn

具有广阔的应用前景，是下一代数据中心最佳资源利用的重大变革性技术。

2 问题与挑战

传统的内存架构面临着许多挑战，如内存带宽瓶颈、内存容量限制和访问延迟等。传统上，设备之间的互联采用 Peripheral Component Interconnect Express®（PCI Express®或 PCIe®）串行接口，在 CPU 和内存之间采用 Double Data Rate（DDR）并行接口连接。虽然 PCIe 和 DDR 是各种设备的绝佳接口，但它们也有一些固有的局限性。这些限制导致了以下挑战，这些挑战共同推动了 CXL 的产生与发展[1]。

（1）对系统和设备内存之间一致性访问：系统存储器通常通过 DDR 连接，并可通过 CPU 高速缓存层次结构进行高速缓存。相反，从 PCIe 设备到系统内存的访问是通过非一致读/写操作进行的，PCIe 设备无法缓存系统内存以利用时间或空间局部性或执行原子操作序列。从设备到系统内存的每次读/写都通过主机的 root complex（RC），这使 PCIe 与 CPU 缓存语义保持一致。类似地，从主机非一致地访问连接到 PCIe 设备的存储器，每次访问都由 PCIe 设备处理。因此，设备内存无法映射到可缓存的系统地址空间。非一致性访问对于流式 I/O 操作（如网络访问或存储访问）非常有效。对于加速器，在将整个数据结构移回主存储器之前，将其从系统存储器移到用于特定功能的加速器，并部署软件机制以防止 CPU 和 加速器之间同时访问。这对人工智能、机器学习和智能网络接口卡等新的使用模型构成了障碍，在这些模型中，设备寻求使用设备的本地缓存与 CPU 同时访问相同数据结构的部分，而无需来回移动整个数据结构。这一挑战同样阻碍了 PIM 的广泛采用。

（2）内存扩展：对内存容量和带宽的需求与计算的指数增长成正比（图 2）。不幸的是，DDR 内存一直无法满足这一需求。这限制了每个 CPU 的内存带宽。这种缩放不匹配的一个关键原因是并行 DDR 接口的引脚效率低下。通过添加 DDR 通道进行扩展大大增加了平台成本，并带来了信号完整性挑战。原则上，PCIe 引脚将是一个很好的选择，因为它们每个引脚都具有卓越的内存带宽。例如，32 GT/s 的 x16 Gen5 PCIe 端口提供 256 GB/s 和 64 个信号引脚。DDR5-6400 提供 50 GB/s，约 200

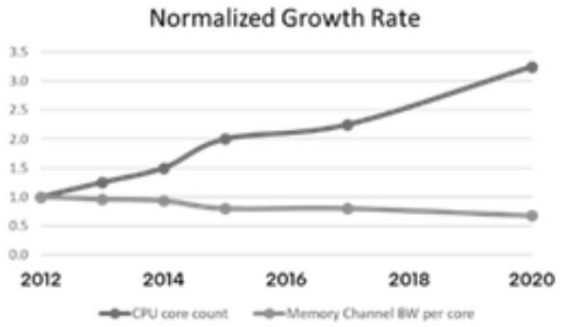

图 2 内存带宽不匹配

个信号引脚。PCIe 还支持更长的重定时器覆盖范围。不幸的是，PCIe 不支持一致性访问，并且设备连接的内存无法映射到一致的内存空间。因此 PCIe 无法取代 DDR。另一个扩展挑战是每比特的 DRAM 内存成本一直较高。虽然有多种新的介质类型，如 Managed DRAM、ReRam、3DXP/Optane 等，但是 DDR 标准依赖于 DRAM 特定的命令进行访问和维护，这阻碍了新介质类型的采用。

（3）因滞留导致内存和计算效率低下：如今的数据中心由于资源闲置而效率低下。当空闲容量仍然存在，而另一个资源（如计算）已完全使用时，资源（如内存）将被搁置。根本原因是资源紧密耦合，其中计算、内存和 I/O 设备只属于一个服务器。因此，每台服务器都需要为其提供过多的内存和加速器，以处理具有峰值容量需求的工作负载。例如，应用程序的服务器需要比可用内存更多的内存（或加速器），无法从同一机架中另一个未被充分利用的服务器借用内存（或加速卡），并且必须承担页面未命中的性能后果。另一方面，工作负载使用所有核心的服务器通常有未使用的内存。搁浅具有不利的影响，涉及到成本和可持续性问题等，造成阿里巴巴、亚马逊、谷歌、Meta 和微软等公司的数据中心的内存资源利用率较低。

（4）在分布式系统中进行细粒度数据共享：分布式系统经常需要进行细粒度同步。底层的更新通常很小且对延迟敏感，因为工作在更新上阻塞。例如，在 Web 规模的应用程序中，如 Web 搜索、社交网络内容组合和广告选择等，采用了分区/聚合的设计模式。在这些系统中，查询更新通常不到 2KB（比如搜索结果）。其他例子包括依赖 KB 级页面的分布式数据库以及具有更小更新的分布式一致性。以这种细粒度共享数据意味着在典型的数据中心网络中，通信延迟主导了更新的等待时间，降低了这些重要用例的速度。例如，以 50GB/s（400Gbit/s）

传输 4KB 只需不到 2 微秒，但在当前网络上通信延迟超过 10 微秒。一个一致的共享内存实现可以帮助减少通信延迟到亚微秒级别。

CXL 正是为了应对这些挑战而产生的。自 2019 年发布以来，CXL 已经发展到了第三代。每一代都拓展了协议的互联特性，同时保持完全向后兼容。CXL 1.0 规范在 PCIe 的基础上增加了一致性和内存语义，这解决了挑战 1（一致性）和挑战 2（内存扩展）的问题，使得 CXL 设备能够缓存系统内存。这还标准化了一个一致的接口，以促进 PIM 系统和编程模型的广泛采用。CPU 也可以缓存设备内存，从而解决异构计算的细粒度数据共享问题。此外，连接到 CXL 设备的内存可以映射到系统可缓存的内存空间。这有助于异构处理，并有助于图 3 所示的内存带宽和容量扩展的挑战。CXL1.0 还继续支持 PCIe 的非一致性生产者-消费者语义。CXL 2.0 通过在多个主机之间启用资源池化来进一步解决挑战 3（资源滞留）。这里的主机指的是在单一操作系统或虚拟机监控下的单插槽或多插槽系统。通过随时间重新分配资源，池化克服了资源滞留和碎片化问题，而无需重启这些主机。CXL 协议通过引入构建主机和内存设备的小型网络的 CXL 开关来实现池化。CXL 3.0 通过在多级 CXL 交换上解决挑战 3，使得在数据中心级别构建动态组合的系统成为可能。此外，CXL 3.0 通过在主机边界上启用细粒度内存共享来解决挑战 4（分布式数据共享）。

3 高速互联协议 CXL

CXL 是一种行业支持的高速缓存一致性互连技术，如图 3 所示，可用于处理器、GPU、拓展内存和加速器等设备之间的互联，将内存与计算解耦。CXL 技术保持了 CPU 内存空间和附加设备上的内存之间的内存一致性，从而允许资源共享以获得更高的性能，降低软件栈的复杂性，并降低系统的成本。这允许用户只关注目标工作负载，而不是加速器中的冗余内存管理硬件。CXL 最初于 2019 年由 Intel 公司提出，目前业界已经有超过 160 家公司加入工业标准化组织。下面的小节将对 CXL 的发展和技术细节进行详细的介绍。

3.1 CXL版本

自 CXL 技术首次发布以来，它经历了多个版本的演进和改进，表 1 中包含了 CXL 每个版本具有的主要特性。

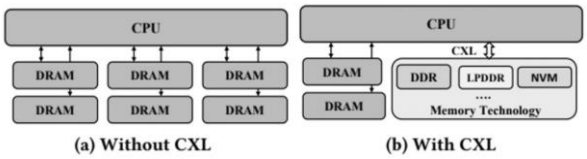

图 3 CXL 将内存和计算解耦

表 1　CXL 版本概述

版本	年份	范围	速度	使用案例
CXL1.0&1.1	2019	单机	32GT/s	加速器（挑战 1） 内存拓展（挑战 2）
CXL2.0	2020	单级交换机	32GT/s	小型资源池(挑战 3)
CXL3.0	2022	多级交换机	64GT/s	大规模资源池和数据共享(挑战 3 和 4)

3.1.1 CXL1.0/1.1

第一代 CXL 为直接连接到主机的设备引入了一致性和内存语义。这支持对 CPU 和加速器的共享数据结构进行细粒度异构处理以及经济高效地扩展内存带宽和容量。CXL 是一种类似于 PCIe 的非对称协议，主处理器包含 RC，每个 CXL 连接一个，每个连接到一个作为“端点”的设备。主机处理器协调缓存一致性，软件通过在主机处理器中执行指令来配置系统，主机处理器生成访问每个设备的配置事务。CXL 原生支持 x16、x8、x4 链路宽度，降级支持 x2、x1 链路宽度。降级模式是指 PCIe 链路自动进入更窄的宽度或更低的频率，以克服给定通道上高于预期的错误率。CXL 原生支持 32.0 GT/s 和 64.0 GT/s 的数据速率，降级模式下支持 16.0 GT/s 和 8.0 GT/s 的数据速率。

3.1.2 CXL2.0

CXL2.0 是 CXL 的第二代协议标准，如图 4 所示，其支持内存资源池化特性，允许在一段时间内将相同的资源分配给不同的主机。在运行时重新分配资源的能力解决了资源搁浅问题（挑战 3），因为它克服了资源与各个主机的紧密耦合。如果一台主机运行计算密集型工作负载，并且不使用从池中分配的设备内存，操作人员可以将该设备内存重新分配给另一台主机，这可能会运行内存密集型工作负载。由于运营商在设计时通常不知道哪些工作负载在哪些主机上运行，因此资源池可以节省大量内存，运营商可以根据平均情况提供内存，而不是根据任何类型的内存密集型工作负载使用的最坏情况对两台主机进行调整。同样的池化结构也适用于加速器等其他资源。此外，CXL2.0 协议中还引入了交换机结构，使得其可以连接 2-16 台设备。

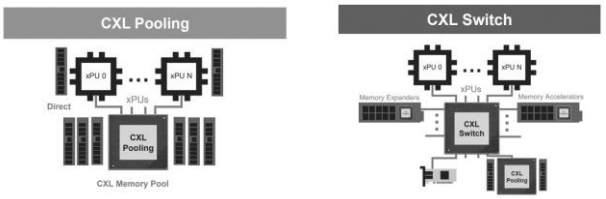

图 4 CXL2.0 支持的池化和交换机结构

3.1.3 CXL3.0

CXL 3.0 侧重于物理和逻辑层面的升级。在物理层面，基于 PCIe 6.0 技术，编码采用 PAM4 方式，图 5 为传统的 NRZ 编码与 PAM4 编码的信号传输特性，可以看到，同一时段采用 PAM4 编码传输的数据量提高了一倍，传输速率达到 64GT/s。在逻辑层面，CXL 3.0 扩招了标准逻辑能力，允许更复杂的连接拓扑，以及一组 CXL 设备内可以灵活实现内存共享和内存访问。为了应对大型分布式系统中的数据共享挑战，CXL 3.0 将 CXL 2.0 中引入的资源池扩展到更大的规模，支持多达 4096 个终端设备的多级交换结构，同时利用更大的 Flit 大小，使其具有低延迟和足够的带宽。终端设备可以是主机 CPU(代表独立的服务器或节点)、内存、加速器或任何其他 I/O 设备(如网卡)。总体目标是根据工作负载动态组合系统，以提供较低 TCO 的节能性能，为了实现这些目标，CXL 3.0 引入了以下内容:

•将每引脚带宽加倍，同时保持扩展到更大拓扑所需的平坦延迟。

•对 Fabric 拓扑的支持:这是任何负载存储互连标准的第一个，它在事务之间有排序约束。见图 6 所示，通过在任意源目标对之间使用多条路径，CXL 摆脱了树拓扑的限制，这对于扩展到数千个设备是必不可少的。这样可以实现更低的延迟、更高的对分带宽和故障转移功能。

•如果没有冲突，从 PCIe/CXL 设备直接点对点访问到由 Type-2/Type-3 设备托管的相干 HDM 存储器，而不涉及主机处理器。这将导致低延迟、较少拥塞和高带宽效率，这对大型系统至关重要。例如，使用直接 P2P 从网卡到内存的距离是 8 跳，而通过 CPU 往返的距离是 16 跳。

•可以通过硬件或软件来实现在主机之间共享一致内存和消息传递。共享一致内存允许多个系统共享数据结构、执行同步或使用低延迟负载存储语义传递消息。消息传递也可以通过使用 load-store CXL.io 来完成。

•近内存处理，允许在近内存执行计算，以获得更好的性能和能源效率。

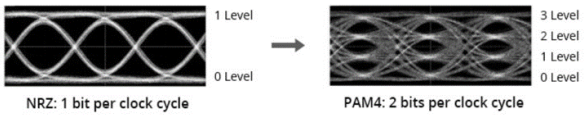

图 5 NRZ 编码与 PAM4 编码信号对比

CXL 1.0/1.1 是最早发布的 CXL 版本，它定义了基本的互连协议和电气规范。CXL 1.0 支持高带宽、低延迟的内存访问，并提供了 Cache Coherency（一致性缓存）和 Memory Semantics（内存语义）功能。这使得 CXL 能够实现 CPU 与加速器设备之间的高效通信和协同计算。CXL1.0/1.1 可以归纳为“直联”，也就是让主机 CPU 可以直接访问 PCIe 设备的内存，具体分为三个子协议（将在 2.2 节进行介绍）。CXL 2.0 是对 CXL 技术的进一步改进和扩展。这个版本引入了一些重要的功能和特性。其中包括：多路复用（Multiplexing）支持，使得多个设备可以共享同一条 CXL 连接；更高的带宽和更低的延迟，提升了数据传输的效率和响应时间；支持更大的地址空间，满足了更大规模的内存访问需求；增强的安全性和可靠性特性，提供了更好的数据保护和容错能力。CXL2.0 可归纳为“池化”，就是让多个主机 CPU 和多个设备可通过一个 Switch 硬件连接在一起，可以互相访问，实现内存池化。CXL 3.0 发布于 2022 年 8 月，它进一步加强了 CXL 技术在计算和存储领域的应用。CXL 3.0 引入了一些重要的新特性。其中包括：存储级别协议（Storage Class Protocol，SCP）的支持，使得 CXL 可以与存储设备进行高速、低延迟的通信；对持久内存（Persistent Memory）的原生支持，提供了更高效的数据持久化和访问机制；增强的安全性功能，包括硬件加密和认证功能，保护数据的安全性和完整性。CXL3.0 可归纳为 Fabric，可以让多个 Switch 实现级联结构，支持更复杂的结构。

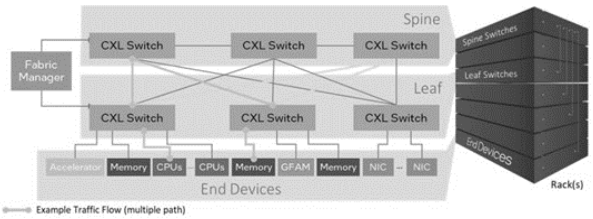

图 6 CXL 多级交换机结构

CXL 技术的不断演进和改进为计算和存储领域带来了许多新的机遇和挑战。通过提供高带宽、低延迟的互连能力，CXL 使得计算和存储设备能够更紧密地协同工作，提升系统性能和效率。尽管

CXL 技术在不断发展，但仍面临一些挑战，如成本、兼容性和生态系统的建设等，未来对 CXL 技术的研究可以聚焦在这些领域进行改进和突破，以满足不断增长的计算和存储需求。

3.2 CXL子协议

CXL 技术通过其三个主要的子协议：CXL.io、CXL.cache 和 CXL.mem，提供了卓越的高性能互连能力，以下内容是对三个子协议的简要介绍。

3.2.1 CXL.io

CXL.io 是 CXL 规范中定义的物理层接口，可以提供比传统 PCIe 更低的延迟、更高的带宽和更好的可扩展性。它通过使用 SerDes 技术（一种将串行数据转换为并行数据以及反向转换的技术），在单个物理通道上同时传输多个不同的数据流。这些数据流可以包括带宽密集型的数据流、低延迟的命令和控制信息以及配置寄存器和状态信息。其本质就是具有一些增强功能的 PCIe5.0 协议，用于初始化、链接、设备发现和枚举以及寄存器访问。

3.2.2 CXL.cache

CXL.cache 定义了主机和设备之间的交互，允许连接的 CXL 设备使用请求和响应方法以极低的延迟高效缓存主机内存，还可以通过将内存缓存到外部设备中来提高性能，提高整体系统性能，即允许设备访问主存和 cache。

3.2.3 CXL.mem

CXL.mem 协议使得主机处理器能够使用 load/store 命令访问设备附加内存，其允许 CPU 将外部设备看作是扩展内存，从而可以存储更多的数据。这种方式可以提高系统的可靠性，因为即使发生了内存故障，CPU 仍然可以通过外部设备继续运行。归纳为 CPU 可以访问设备的内存。其中主机 CPU 充当主设备，CXL 设备充当从属设备，并且可以支持易失性和持久性内存体系结构。

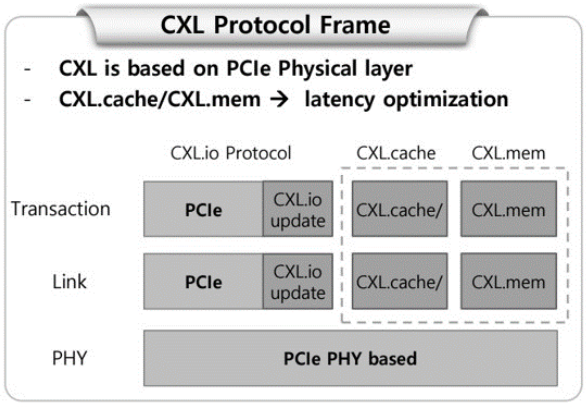

图 7 CXL 子协议的层次结构

3.3 CXL设备类型

CXL 定义了三种设备类型，见图 8 所示，它们通过 CXL 接口与主机处理器直接通信，提供了更快的数据传输速度和低延迟的性能，三种类型即 Type1、Type2、Type3。

其中 Type 1 设备是通过 PCIe 插槽安装的加速卡或附加卡。这些设备可以与现有系统集成，并通过 CXL 接口与 CPU 直接通信。其适用于各种高速缓存设备，如网卡和各种计算设备，例如 CPU 和 AI 加速器。通过 CXL 接口，Type 1 设备能够提供更快的数据传输速度，从而提高系统的性能。

Type 2 设备具备 Type 1 设备的所有功能，并且通常用于高密度计算的场景。典型的 Type 2 设备是 GPU 加速器。这些设备不仅具有计算功能，还集成了内存。通过 CXL 接口，Type 2 设备能够在高密度计算环境中提供强大的计算能力和高速的数据传输。

Type 3 设备是一种专用的存储设备，与主机处理器直接通信，并且使用 CXL 协议实现低延迟和高吞吐量的数据传输。Type 3 设备通常用作内存缓冲器，用于扩展系统的内存带宽和容量。通过 CXL 接口，Type 3 设备能够提供快速的存储访问，从而提高系统的性能和响应速度。

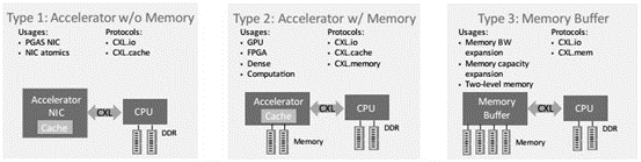

图 8 CXL 的三种设备类型

4 研究现状分析

新型高速互联技术 CXL 凭借其低延迟、高带宽、高灵活性和高拓展性等受到工业界与学术界的广泛关注，有着广泛的应用场景，CXL 技术不仅适用于数据中心等高性能计算领域，还可以应用于人工智能、区块链、物联网等多个领域。目前，对 CXL 的研究主要集中在分级内存、分离式内存、构建共享内存池、优化数据中心的性能以及 CXL-Flash，下面我们将依次介绍各个领域的研究现状。

4.1 基于CXL的分级内存

大规模应用产生的内存需求导致内存成为了数据中心的主要花费，对于许多数据中心的应用，虽然它们分配了大量的内存，但是很快需要经常使用的内存量远低于已经分配的内存，如果页面放置机制能够将这些不活跃的页面移动到较低的内存层次，同时将活跃的热页面及时迁移到更高的内存层次，分层存储系统可以很好地适应这种冷热内存切换。CXL 技术支持主内存拓展可以提供有效的解

决方案，在分级内存领域，CXL-Memory 的延迟介于 DRAM 与 NVM 之间，见图 9 所示。其可以充当新的存储层次优化系统的整体性能。一些研究者以 Meta 服务器集群为例演示 TPP[2]。

为了分析应用程序中内存页面的类别，作者开发了 Chameleon 工具，Chameleon 的主要使用例子是理解应用程序的内存访问机制，及时去识别内存中的冷热页面，该工具由两部分组成，包括 Collector 和 Worker，其中 Collector 利用现代 CPU 的 PEBS 机制来收集与内存访问相关的硬件级性能事件。Worker 使用采样信息来生成结果。

随着 CXL 技术的出现，超大规模计算机系统正在采用 CXL 支持的异构分层内存系统，不同的内存层具有不同的性能特征。为了在这样的系统中进行性能优化，需要使用透明页面放置机制（TPP）来处理具有不同热度特征的页面，并将其放置在适当的温度层上。一个有效的页面放置机制应该能够高效地将内存中的冷页面迁移到较慢的 CXL 内存，同时适当地识别在 CXL 节点中的热页面，并将它们提升到快速内存层。由于 CXL 内存没有 CPU 且独立于与 CPU 连接的内存，因此它应该足够灵活，以支持具有不同特征的异构内存技术。分配页面到 NUMA 节点不应频繁中断，因为较慢的回收机制需要释放空间。此外，有效的策略应该能够了解应用程序对不同页面类型的敏感性。考虑数据中心工作负载特性和我们的设计目标，我们提出 TPP：一种智能的操作系统管理的分层内存系统机制。如图 10 所示，TPP 将“更热”的页面放置在本地内存中，同时如图 11 所示，将“更冷”的页面移动到 CXL 内存中。TPP 的设计空间可以划分为四个主要领域，(1)轻量级的降级冷页面到 CXL 内存，(2)分离的分配和回收路径，(3)将热页面提升到本地节点，以及(4)页面类型感知的内存分配。

根据图 12 的测试结果以及其他分析可知，TPP 可以将默认 Linux 上的应用程序性能提高 18%。TPP 还在 NUMA 平衡和 AutoTiering 这两种最先进的分层内存管理机制上表现出 5-17%的优势。作者提出了以下三种洞察观点。

（1）多租户云的分层内存。在典型的云平台环境下，当多个租户共存于一台主机上时，TPP 可以有效地使它们竞争地共享不同的内存层级。当本地内存大小占据系统总内存容量的主导地位时，这可能不会造成太大问题。然而，如果具有不同优先级的应用程序具有不同的 QoS 要求，TPP 可能会提供次优的性能。通过在 TPP 上集成一个经过良好设计的 QoS 感知内存管理机制可以解决这个问题。

（2）对于内存带宽受限的应用程序来说，CPU 与 DRAM 内存带宽往往成为瓶颈。CXL 的额外内存带宽可以通过将内存分布在顶层和远程节点上来提供帮助。与仅将冷页面放入 CXL-Memory（其

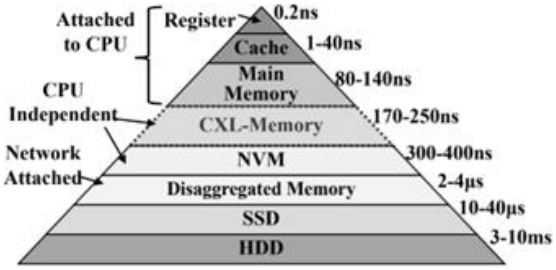

图 9 不同存储介质的延迟特性

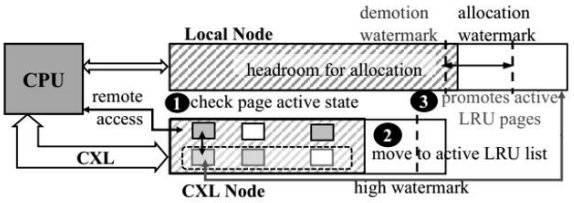

图 10 TPP 将热页面提升到本地节点

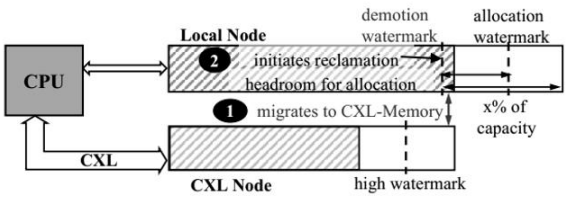

图 11 TPP 将冷页面降级到 CXL-Memory

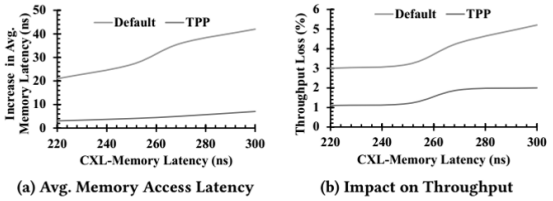

图 12 TPP 在延迟和流量方面的测试结果

带宽消耗非常低）不同，最优解决方案应该将带宽需求较高、对延迟不敏感的页面合理地放置在 CXL-Memory 中。确定这类工作集的理想比例的方法可能需要硬件支持。我们希望在未来的工作中探索透明的内存管理，以应对内存带宽扩展的使用场景。

（3）硬件特性可以进一步提升 TPP 的性能。在支持 CXL 的 ASIC 上的内存侧缓存及其相关预取器可以帮助减少 CXL-Memory 的有效延迟。硬件支持内存层间数据移动可以帮助减少页面迁移开销。

4.2 利用CXL实现分离式内存

由于其高内存利用率、透明的弹性和资源管理效率，内存分离引起了广泛关注。许多研究已经探索了各种软件和硬件方法来实现内存分离，并在大规模系统中实现内存分离方面投入了大量的努力。目前内存分离的相关工作主要有三类，一是借助 RDMA；二是 Swap，基于页面的交换内存池；三是 KVS，基于对象实现的内存池。它通过不同的方法实现高效的内存利用和资源管理[3]。基于页面的

方法提供了一种无需代码更改即可使用分离内存的方式，而基于对象的方法通过使用自己的数据库来处理分离内存，解决了地址转换带来的挑战。这些研究为实现大规模系统中的实用内存分离做出了重要贡献。

有研究者提出了直接可访问的内存分离方案 DirectCXL[4]，具体的系统设计结构见图 13、14、15。通过 CXL 的内存协议（CXL.mem）直接连接主机处理器复杂和远程内存资源。为此该团队探索了基于 CXL 的内存分离的实际设计并实现了它。由于目前还没有支持 CXL 的操作系统，论文还提供了 CXL 软件运行时，允许用户通过纯粹的加载/存储指令利用底层的分离内存资源。由于 DirectCXL 不需要在主机内存和远程内存之间进行任何数据拷贝，它可以向用户展示远程分离内存资源的真实性能。通过 DirectCXL，用户可以直接访问远程的分离内存资源，而不需要进行数据拷贝，从而实现了更高的性能。这种直接的连接方式使得用户能够充分利用远程分离内存资源。

该论文的评估结果表明，当工作负载能够利用主机处理器的缓存时，DirectCXL 的分离内存结构可以展现出类似 DRAM 的性能。DirectCXL 的延迟平均比 RDMA 的最佳延迟短 6.2 倍。图 16 展示的是实际应用程序的测试结果，DirectCXL 的性能平均比基于 RDMA 的内存分离提升了 3 倍。

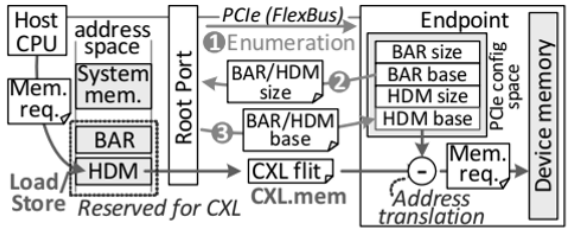

图 13 DirectCXL 连接方法

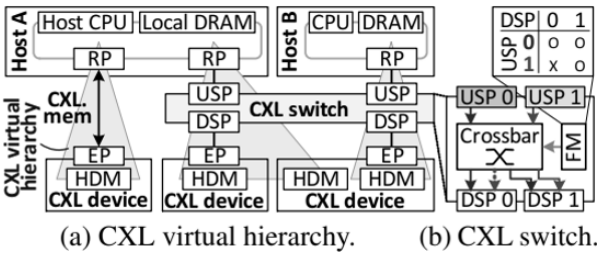

图 14 DirectCXL 网络层次和交换机

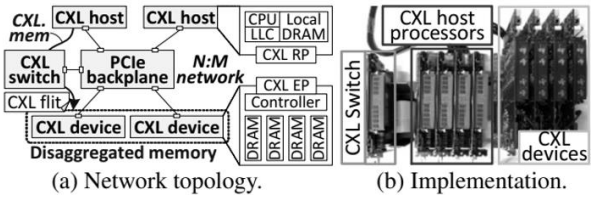

图 15 CXL 网络拓扑和实施

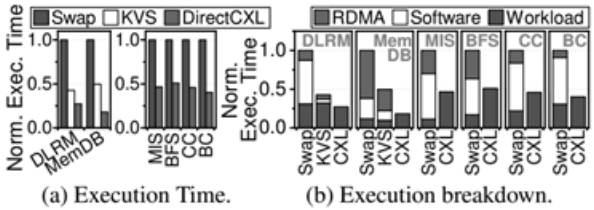

图 16 运行时间测试结果

4.3 基于CXL构建共享内存池

内存池内存系统的核心思想是在应用程序启动时预先分配一定数量的内存块，并将其存储在内存池中。应用程序可以从内存池中获取内存块来存储数据，并在不需要时将其归还给内存池。这种预分配和重复利用内存块的方式可以避免频繁的内存分配和释放操作，从而提高内存的利用率和性能。其可以提供更快的内存访问速度。由于内存块已经预先分配并存储在内存池中，应用程序可以直接从内存池中获取内存块，而无需进行额外的内存分配操作。这些操作减少了内存访问的开销，并且可以更高效地利用硬件缓存，从而提高系统的整体性能。

CXL 互联技术可以借助其低延迟，高带宽等特性去构建内存池，CXL 池化内存作为一种可行的内存分离解决方案，提供了内存扩展和减轻内存超配的功能，受到了行业的关注。对于有效利用池化内存，一个关键特性是根据主机的需求动态分配或释放内存。称这个特性称为动态容量服务（DCS）[5]。目前有研究者提出了 CXL 池化内存行业首个 DCS 实现。其通过实现基于 FPGA 的 CXL 池化内存原型和完整的软件堆栈，演示了完全功能的 DCS。实验结果表明，DCS 可以通过按需动态分配和释放内存资源，显着提高系统的内存利用率。

借助 CXL 实现的内存池架构如图 17 所示，传统上，在系统引导时静态分配每个主机的内存容量。随着每个主机的工作集大小（WSS）随时间变化，整个系统可能会因为交换操作而出现内存低利用率或性能下降的问题。该研究者据此提出了动态容量服务（DCS），通过根据每个主机的不同 WSS 动态分配和释放 CXL 内存资源，解决了上述低效问题。他们期望 DCS 通过避免不必要的内存资源分配，提高 CXL 池化内存的多主机系统的内存利用率和性能。设计的 DCS 是一个基于 CXL 3.0 中描述的动态容量设备（DCD）的硬件/软件集成 CXL 池化内存解决方案，并在 2022 年的 Flash Memory Summit 上进行了演示。通过基于多家 FPGA 平台的测试，展示了 DCS 在 CXL 池化内存系统中的潜在优势。

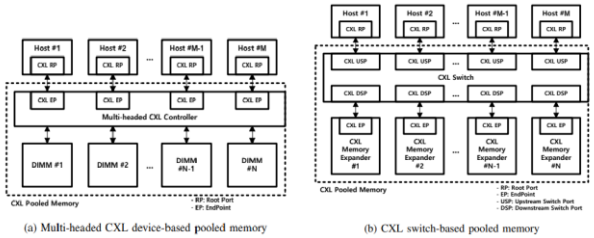

图 17 CXL 内存池架构

使用 CXL 内存池可以带来许多优势。首先，它提供了更大的内存容量，可以满足大型工作负载的需求。其次，通过动态分配和释放内存资源，可以提高内存利用率，减少内存超配的问题。此外，CXL 内存池还可以提供更快的数据访问速度和更好的系统性能。然而，构建和实现 CXL 内存池也面临一些挑战。这包括设计合适的内存池大小、范围和拓扑结构，以及解决与性能、虚拟化和管理等方面的复杂设计约束。因此，需要深入研究和合理的系统设计来实现最佳的性能和资源利用率。

总之，CXL 内存池是一种创新的内存管理解决方案，通过动态分配和释放内存资源，提供了更大的内存容量和更高的利用率，为多主机系统带来了许多优势。随着技术的不断发展，CXL 内存池有望在数据中心和云计算领域发挥重要作用。

4.4 优化数据中心的性能

在基于云的数据中心中，内存利用率是一个非常关键的指标。目前，数据中心在应对各种应用程序对内存需求的能力方面存在不足。另一方面是数据中心服务器集群中有着大量闲置的内存资源没有被充分利用，这些资源被称为 Memory stranding。引入 CXL 实现内存池化提供了提高利用率的途径。然而，基于 CXL 的内存系统的设计空间问题中，关键问题在于内存池的大小、范围和拓扑结构[6]。同时，使用内存池需要在性能、虚拟化和管理等方面解决复杂的设计约束。当下一些研究者考虑要充分利用这些滞留的内存，提出了一种软硬件协同优化设计方案，称之为 Pond[7]，这是第一个既满足云性能目标又显著降低 DRAM 成本的内存池化系统。

Pond 基于 Compute Express Link（CXL）标准，用于对池内存进行加载/存储访问，并基于两个关键点进行构建。首先，论文作者在云环境下的跟踪数据的分析显示，跨 8-16 个插槽进行池化足以实现大部分的性能。这使得可以采用小型池设计，使其具有低访问延迟。其次，通过创建机器学习模型，准确预测分配给虚拟机（VM）的本地和池内存的数量，以模拟相同 NUMA 节点内存性能。分析结果表明 Pond 成功地解决了云中的内存池化挑战，实现了降低成本和满足性能要求的双重目标。

论文作者根据在微软 Azure 上的测量结果，定义了以下设计目标。G1：与 NUMA 本地 DRAM 相当的性能。G2：与虚拟化加速器兼容。G3：与不透明的 VM 和不变的客户操作系统/应用程序兼容。G4：低主机资源开销。在整个内存系统的优化设计方案上，为了量化分析性能（G1），定义了性能降级边界（PDM），作为给定工作负载相对于完全在 NUMA 本地 DRAM 上运行时的允许放慢速度。Pond 旨在实现可配置的 PDM，例如对于可配置的尾部比例的虚拟机（例如 98%）。为了实现这种高性能，Pond 使用一个小而快速的 CXL 池。由于 Pond 的内存节省来自于池化而不是超额分配，Pond 必须在其系统软件层中最小化池碎片化和浪费。为了实现（G2），Pond 在 VM 启动时预分配本地和池化内存。Pond 在其分配、性能监控和缓解流水线中决定使用具体分配的策略。该流水线使用新颖的预测模型来实现 PDM，模型的预测流程见图 18 所示。最后，Pond 通过使用轻量级的硬件计数器来克服 VM 不透明性（G3）和主机开销（G4）。

论文通过对 158 个工作负载进行评估，见图 19，结果显示 Pond 可以在性能与同一非一致内存访问(NUAM)节点虚拟机分配相差 1-5%的情况下，降低 DRAM 成本 7%。这意味着云服务器成本总体上降低了 3.5%。Pond 通过利用 CXL 并采用小型内存池设计思路和基于机器学习的内存分配，该系统展示了提高 DRAM 利用率和降低成本、同时满足严格性能目标的潜力。

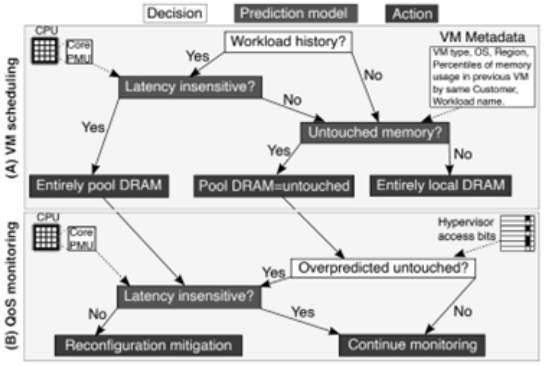

图 18 基于机器学习的内存分配预测模型

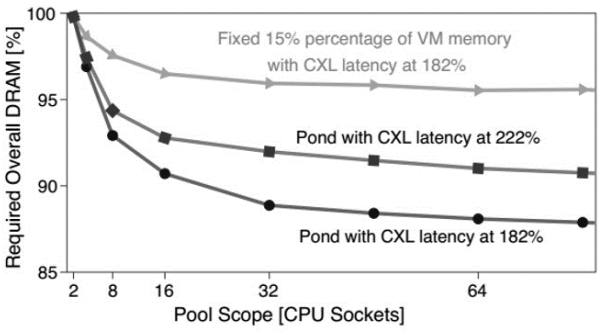

图 19 Pond 测试结果

4.5 CXL-Flash

计算能力和内存容量需求之间的日益不平衡发展成为一个称为“内存墙”的挑战，一些常见的解决方法是近数据处理、存内计算、分离内存等，本文作者探讨了使用闪存（Flash）内存来突破“内存墙”瓶颈[8]，Flash 是一种通常用于固态硬盘的存储技术，其具有高密度和容量扩展性。虽然 DRAM 只能扩展到几 GB 的容量，但基于闪存的固态硬盘（SSD）可达到 TB 级的容量，足够大以应对“内存墙”的挑战。

闪存内存作为主内存的使用得益于最近出现的诸如 CXL、Gen-Z、CCIX 和 OpenCAPI 的互连技术，直接访问 PCIe 设备，拓展能力好。虽然 CXL 目前只考虑 DRAM 和 PMEM 作为主要的内存扩展设备，但由于 CXL 的一致性内存访问特性，使用 SSD 中的闪存当作内存使用也是可能的，但是面临很多问题。

（1）粒度不匹配：闪存不是随机访问的：其数据以页粒度写入和读取，每个页的大小约为几千字节，导致大量的流量放大。此外，页面不能被覆盖写入。相反，必须首先擦除一个包含数百个页的块，然后才能写入数据到已擦除的页。这种受限的接口导致任何 64B 缓存行刷新通过读取-修改-写入操作产生大量的写放大。作为一个块设备，其访问粒度要大得多（4KiB）的 SSD 拥有更少的开销。

（2）微秒级延迟：闪存的速度比 DRAM 慢几个数量级，其读取速度仍在几十微秒范围内，而较慢的编程和擦除操作则在几百微秒到几千微秒之间。此外，闪存的延迟还取决于其单元技术。例如，随着每个单元存储的位数增加，从 SLC（单级单元）到 TLC（三级单元），延迟也会增加。超低延迟（ULL）闪存是 SLC 的一种变体，以性能为代价提高了密度。然而，即使是 ULL 技术，其速度仍比 DRAM 慢几个数量级。作为一个块设备，微秒级的延迟是可以容忍的，因为存在软件开销。然而，对于直接使用 load/store 指令访问的内存设备来说，微秒级延迟是一个挑战。

（3）有限的耐久性：编程和擦除操作期间施加在闪存上的高电压会慢慢使单元失效，使它们随着时间的推移无法使用。存储器制造商规定了耐久性极限作为一个指导，表示闪存块可被擦除的次数。这个限制也取决于闪存技术。虽然这仍然是一个软限制，闪存超过限制后仍然可以继续使用，但是磨损的块表现出不可靠的行为，并且不能保证正确存储数据。由于应用程序级和内核级的缓存和缓冲，SSD 的块接口的写入量减少，因此当前的耐久性限制在 SSD 的寿命内通常是足够的。然而，作为内存设备，频繁的内存写入会使闪存内存快速变得无法使用。

CXL-Flash 的系统架构分为四个部分，第一部分是在 Flash 前面添加缓存，第二部分是增加一组 MSHR（miss status holding registers），即缺失状态处理寄存器，用于减少对闪存的重复读问题。第三部分是增加一个预取器，本文对后续还对不同的预取策略进行了详细分析，最后是探索 Flash 级别的并行性对性能的影响。具体的架构见图 20 所示。

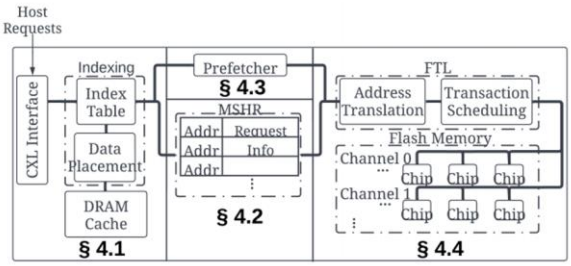

图 20 CXL-Flash 系统架构

最后的测试结果可知使用该设备作为内存，68-91%的内存访问可以在 CXL-flash 设备上实现亚微秒的延迟，即使在高负载的情况下，该设备的寿命至少可以达到 3.1 年。同时还可以得出如下三点启发。

（1）对于虚拟内存的地址转换使得 CXL-flash 的预取器难以发挥作用，建议通过传递内核级别的访问模式提示来进一步提高性能。

（2）本文所探索的 CXL-flash 的设计中并未考虑闪存的内部任务，如垃圾收集和磨损均衡，未来可对此进一步改进提升性能。

（3）此外，所考虑的主机系统可能并未完全反映 CXL 引入的新系统特性。因此可知在 CXL-flash 研究领域还需要做更多的工作，而本文的工作可以为未来的研究提供一个平台。

5 未来研究方向

目前工业界对 CXL 的探索主要有：Intel 发布第四代至强可拓展存储器，澜起科技发布全球首款 CXL 内存扩展控制芯片，Microchip 推出 CXL 智能存储控制器，SK 海力士计划 2023 年量产 CXL 内存，三星推出 512GB 内存扩展器 CXL DRAM。未来对 CXL 协议的研究方向主要是两个方面，一是从协议本身进行优化，二是借助 CXL 的互联特性构建新的使用场景。可以考虑的研究方向可以分为

以下几点：

（1）性能优化：研究进一步提升 CXL 性能的技术，如降低延迟、增加带宽和提高系统效率。这包括探索先进的信号传输和编码方案，以及对协议栈进行优化。

（2）可扩展性和互操作性：解决将 CXL 扩展到支持更大系统的挑战，并确保与现有和未来技术的互操作性。这涉及研究 CXL 对系统架构的影响，研究与不同硬件和软件生态系统的兼容性，并探索将 CXL 无缝集成到各种计算环境中的方法。

（3）安全性和可靠性：加强 CXL 的安全性和可靠性方面的研究。这包括研究安全引导、访问控制机制、数据完整性以及错误检测和纠正技术。此外，研究缓解潜在安全漏洞的方法，并确保对各种攻击向量的鲁棒性。

（4）存储扩展和异构计算：探索 CXL 在实现超越传统限制的存储扩展和有效利用异构计算资源方面的潜力。这涉及研究存储池技术、存储解聚和通过 CXL 利用不同加速器和存储类型能力的新方法。

（5）生态系统发展和标准化：推动 CXL 在行业中的采用和标准化。这包括与行业利益相关者合作，推动开发 CXL 兼容产品，并建立 CXL 实施和部署的指南和最佳实践。此外，探索将 CXL 集成到边缘计算和人工智能等新兴领域的机会。

（6）将 CXL 与目前研究方向结合：比如将 CXL 引入分级内存、分离式内存、构建内存池、分布式内存系统、优化数据中心的内存利用率、将 Flash 与 CXL 结合作为新的内存介质以及作为 SSD 的下一代接口标准等。

通过专注于这些研究方向，我们可以进一步释放 CXL 的潜力，并推动其向未来计算系统的更高效、可扩展和安全的互连技术的发展。

6 结论

本文首先简介了当前计算和存储的发展遇到的各种挑战和瓶颈问题，即熟知的内存墙和 IO 墙等问题，随着互联技术的进一步发展，一些新型高速互联技术如 CXL 的涌现，为克服内存墙瓶颈带来了新的解决方案，本文首先介绍了 CXL 技术的详细内容，随后综述了一些在分布式内存系统上的应用案例。本文介绍的研究现状主要集中在学术界，包括提到了利用 CXL 实现内存分离的研究，并介绍了构建 CXL 内存池和优化云数据中心性能的相关工作。这些研究为进一步提高内存系统性能和利用率提供了重要的思路和方法。

未来，随着数据中心和云计算的快速发展，分布式内存系统的需求将不断增加。未来的研究可以着重探索以下方向：针对分布式内存系统的可扩展性和容错性进行研究，以满足大规模数据中心的需求；进一步优化 CXL 技术，提高其性能和可靠性，以支持更高速的数据传输和更复杂的应用场景；开发更智能的内存管理算法和策略，以提高内存利用率和性能，并减少能耗；结合人工智能和机器学习技术，探索智能化的内存资源分配和调度方法，提高系统的自适应能力。总之，分布式内存系统和 CXL 技术的结合会产生许多新的研究领域，其在未来工业界的应用潜力巨大。通过不断的创新和优化，可以进一步提高内存系统的性能、可靠性和可扩展性，推动数据中心、云计算、内存型存储系统等领域的发展。

参 考 文 献

[1] Sharma D D, Blankenship R, Berger D S. An Introduction to the Compute Express Link (CXL) Interconnect[J]. arXiv preprint arXiv:2306.11227, 2023.

[2] Maruf H A, Wang H, Dhanotia A, et al. TPP: Transparent page placement for CXL-enabled tiered-memory[C]//Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3. 2023: 742-755.

[3] Gouk D, Kwon M, Bae H, et al. Memory pooling with cxl[J]. IEEE Micro, 2023, 43(2): 48-57.

[4] Gouk D, Lee S, Kwon M, et al. Direct access,{High-Performance} memory disaggregation with {DirectCXL}[C]//2022 USENIX Annual Technical Conference (USENIX ATC 22). 2022: 287-294.

[5] Ha M, Ryu J, Choi J, et al. Dynamic Capacity Service for Improving CXL Pooled Memory Efficiency[J]. IEEE Micro, 2023, 43(2): 39-47.

[6] Berger D S, Ernst D, Li H, et al. Design Tradeoffs in CXL-Based Memory Pools for Public Cloud Platforms[J]. IEEE Micro, 2023, 43(2): 30-38.

[7] Li H, Berger D S, Hsu L, et al. Pond: CXL-based memory pooling systems for cloud platforms[C]//Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2. 2023: 574-587.

[8] Yang S P, Kim M, Nam S, et al. Overcoming the Memory Wall with {CXL-Enabled}{SSDs}[C]//2023 USENIX Annual Technical Conference (USENIX ATC 23). 2023: 601-617.

附录.

课堂汇报记录：

问题 1：课堂汇报所讲述的 CXL-SSD 与 CXL-ANNS 进行对比，二者的区别有哪些？

CXL-SSD 和 CXL-ANNS 是两个不同的概念和技术，它们在功能和应用方面有一些区别。下面是它们的主要区别：

1. CXL-SSD（Compute Express Link SSD）：CXL-SSD 是一种基于 Compute Express Link（CXL）技术的内存设备。CXL 是一种高速互连标准，旨在提供高带宽和低延迟的内存和设备连接。CXL-SSD 利用 CXL 接口连接到计算系统，以实现快速的数据传输和存储。CXL-SSD 主要用于高性能计算和数据中心应用，提供高速存储和数据处理能力。

2. CXL-ANNS（Approximate Nearest Neighbor Search）：CXL-ANNS 是基于 CXL 技术的近似最近邻搜索。近似最近邻搜索是一种用于在大规模数据集中查找最接近给定查询点的数据点的技术。CXL-ANNS 利用 CXL 接口和加速器等硬件资源，提供高效的近似最近邻搜索功能。它在人工智能、图像识别、推荐系统等领域具有广泛的应用。

综上所述，CXL-SSD 是一种基于 CXL 技术的内存设备，用于存储和数据处理，而 CXL-ANNS 是基于 CXL 技术的近似最近邻搜索技术，用于高效的数据检索和匹配。它们在应用场景和功能上有所不同，但都利用了 CXL 接口的高速互连能力来提供高性能和低延迟的数据处理能力

问题 2：研究中的“新瓶装旧酒”在本篇论文中体现在何处？

针对于 SSD 的研究，采用的 cache，不同的缓存替换和预取技术等就是所谓的“旧酒”，而新瓶就是引入 CXL 协议，通过基于 CXL 协议具有的高带宽、低延迟特性，可以将 Flash 拓展成为潜在的内存设备，为了解决引入 CXL 协议产生的粒度不匹配、延迟增加，设备耐久度等问题，本文便设想通过缓存替换，预取数据和增加 MSHR 寄存器来解决上述产生的问题。