Taming Memory with Disaggregation

Pankaj Mehra[1] and Tom Coughlin[2]
[1]Elephance Memory
[2]Coughlin Associates Inc. (corresponding author, tom@tomcoughlin.com)

Abstract:

With memory requirements growing to process increasing data for machine learning and other data-intensive applications, we need better ways to utilize installed memory. The CXL protocol enables creating pools of memory and accelerators, allowing memory disaggregation, and enabling composable virtual machines or containers which can be spun up or down on demand and make more efficient use of expensive memory. New software will make CXL memory pools even more useful by addressing needs such as enhanced security of data in disaggregated memory and state consistency preservation in the face of decoupled CPU and memory failures.

Memory Disaggregation in the Data Center

Data centers, esp. the large ones, are constantly seeking to optimize their resource utilization. With scale comes increasing pressure to get the most out one's hardware. The requirement to use compute resources more efficiently for instance led to widespread use of virtual machines running on servers and more recently to creating virtual machines or containers utilizing disaggregated (separated) storage and networking components. Disaggregation usually results in interconnected pools of computer resources such as processors, networks, and storage, which can then be re-aggregated using software to configure virtual machines or containers for running various processes. Software-based combination of pooled computer resources is also known as composable infrastructure.

Storage pooling today focuses on using NVMe running on fabrics (NVMe-oF), allowing arrays of SSDs in a storage pool that can then be assigned to provide storage for containers or virtual machines that can be spun up and spun down at will, resulting in much higher utilization of storage resources. New memory networking standards are now making it possible to disaggregate memory beyond today's direct connection to a CPU toward memory pools that can be shared on an interconnection network and allocated as part of a data center's composable infrastructure. Let's examine these developments that will help future data centers tame their memory needs.

In 2016, Rao and Porter [1] found memory disaggregation over traditional networks favorable for Apache Spark's memory-intensive and highly partitionable workloads. In 2017, Barroso, et al. [2] anticipated the changing access characteristics of data in data centers and encouraged software developers to address a gap in their stacks when it came to accessing data that was approximately one microsecond away. A form of disaggregating memory was possible even before Rao and Porter's work. Hardware proposals for standalone memory blades [4] anticipated many of the aspects of modern memory disaggregation fabrics.

In 2019 the Compute Express Link (CXL) Consortium was formed to create standards for disaggregating memory and creating memory pools indirectly connected to central processing units (CPUs). In November 2020 the CXL Consortium released its 2.0 specification [3]. The CXL 3.0 specification release is expected sometime in 2022. CXL runs on the PCIe bus and uses advances in serial link technology (such as high-speed SerDes), and the decades-old idea that a handful of serial links, each forming a lane of 4x to 16x wide-serial links, can serve as a system-expansion interconnect. CXL-enabled systems are expected by the end of 2022 or early 2023, based upon the latest PCIe specification, generation 5.

CXL makes protocol-layer enhancements to PCIe that make it especially apt for memory attachment. First, it allows long I/O packets and short cache-line grain accesses to share the same physical link by supporting arbitration at flow-digit (or, flit) level so that load-store operations and I/O Direct Memory Access (DMA) operations can share the same physical link without memory accesses incurring exorbitant latencies due to I/O Transport Layer packets crossing switch ports in front of memory data. Second, it specifies coherence protocols that allow caches and buffers to be coherently connected to processors inside a disaggregated heterogeneous system composed of both traditional elements such as general-purpose CPUs with their tightly coupled memory devices and novel elements such as far memory and domain-specific accelerators (FPGAs, GPUs and CGRAs with highly integrated SRAM or HBM DRAM). Figure 1 shows some CXL pooling approaches.

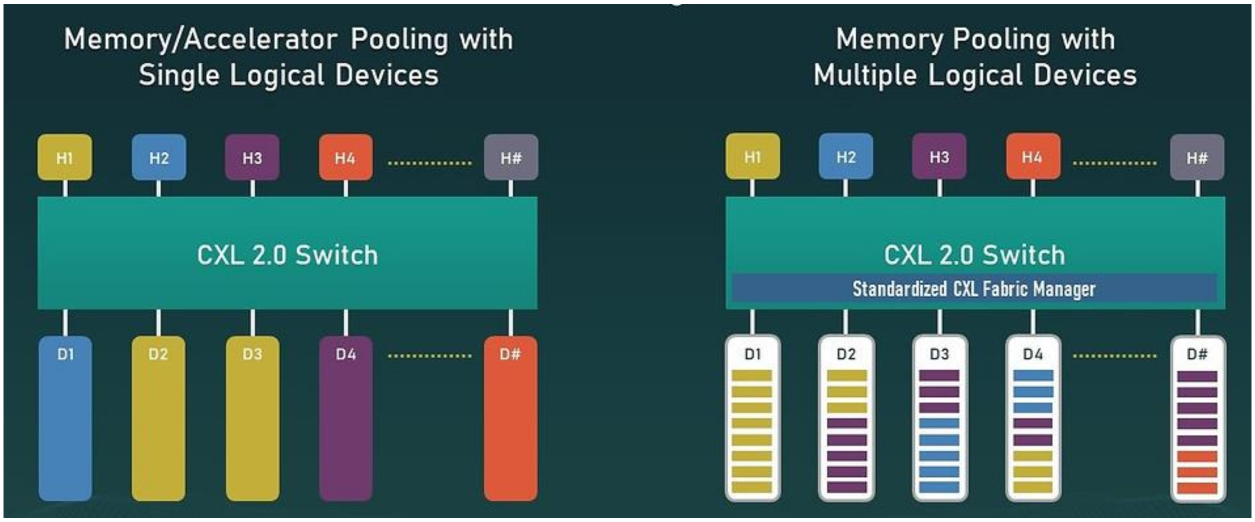

Memory/Accelerator Pooling with Single Logical Devices
H1
H2
H3
H4
H#
CXL 2.0 Switch
D1
D2
D3
D4
D#
Memory Pooling with Multiple Logical Devices
H1
H2
H3
H4
H#
CXL 2.0 Switch
Standardized CXL Fabric Manager
D1
D2
D3
D4
D#

Figure 1. CXL Memory/Accelerator Pooling Approaches (Image Courtesy of the CXL Consortium)

From In-server and Distributed Memory to Disaggregated Memory

Each generation of CXL will allow memory to be deployed farther from the CPU with increasing flexibility in terms of the capacity deployed, the dynamic configuration of host memory capacity, and the number of hosts able to share and efficiently access fabric-attached memory. The benefits of this are best understood in contrast with traditional *bespoke* deployment of dual-inline-memory-modules (or DIMMs) on the DDR buses of CPU sockets, each socket exposing 4, 6, or even 8 DDR channels, and allowing 2 (lately just 1 due to capacitive loading) DIMMs per channel.

Those CPUs were interconnected via a switched or point-to-point symmetric coherency fabric that allowed uniform or non-uniform latency of load-store access to each other's memory. The lanes of PCIe emanated from CPU sockets separately, often with 96 or 128 lanes per socket, were routed to I/O devices such as Network Interface Cards (NICs) or Solid State Disks (SSDs), with or without switches and retimers on the backplane or midplane. In other words, CPUs attached to memory one way and to I/O, another.

Due to the disaggregation of I/O, first providing access to storage over Fibre Channel and IP networks in the 1990s, and subsequently using the more expensive NICs and SmartNICs (Xsigo, Virtensys and Mellanox Multihost NICs) during 2000s and 2010s, PCIe was created to meet the need for system-expansion fabrics capable of supporting RDMA (Remote Direct Memory Access). Although CPUs and their application software also adopted RDMA for efficient inter-processor communication, the heavy software path of setting up and tearing down the memory registrations required for safe, zero-copy RDMA, and the heavy queue-pair based issue and completion paths of RDMA read and write operations remind one more of storage protocols (such as NVMe) than of memory access. By contrast, it is expected that even the higher CXL latencies (compared to DIMMs) will be an order of magnitude lower than the lower RDMA Read round-trip time (or, RTT).

Disaggregation-related Trends and their Implications

Some of the implications of memory disaggregation are similar to storage disaggregation in the late 1990s. When any resource decouples from a host server, it must be managed differently. Starting with power-up and boot, there are fewer ordering guarantees over the power-up sequence across disaggregated components. Due to independence of procurement and decommissioning of resources, and due to independent failures, there are fewer assurances of co-availability.

On the positive side, components that could not previously be independently scaled may now do so. Independent manageability required of the freshly disaggregated components creates opportunity for value-added services. For instance, storage arrays developed many new software-based capabilities not previously available in hard disk drives, such as snapshots, cloning and thin provisioning, to name a few. We likewise expect disaggregated memory nodes to evolve from devices into subsystems with a growing list of novel software-based capabilities.

Independent scaling of computation and memory is to be contrasted with homogeneous scale-out where the sins of bespoke memory deployment were compounded by eager overprovisioning and the inability to acquire more memory without the cost and latency of additional processors.

Moreover, the economic impact of bespoke memory deployment runs deep in today’s data centers. First, memory has now become the costliest element of a data center server’s bill of materials, accounting for as much as 50 percent of the overall cost compared with 25 percent in 2009 [4]. Due to this, as many as 5-7 server stock keeping units (SKUs) are commonly found in a 100,000-server cloud data center, mainly differing in their memory capacity. The use of these fixed SKUs can result in up to 34% of memory capacity remaining idle.

Second, due to the inability to dynamically grow memory capacity of a server to match demand, applications are forced to consider either tolerating *Out of Memory* errors or moving their data to larger instances, just when the footprint of their state is at its peak, neither of which is particularly palatable to modern DevOps.

Third, as if that wasn’t enough trouble, the capacity needs of applications vary wildly [4]. Speaking at the 5th International Symposium on Heterogeneous Integration, John Shalf, the CTO of Nuclear Energy Research Supercomputer (NERSC) at US Department of Energy, has observed that server workloads use less than 25% of their memory, 75% of the time [5]. So wasteful is bespoke deployment of memory in the data center that a resource that is procured by data center operators at approximately $4/GB is then rented out to cloud service operators at approximately $22-$30/GB *per year*, probably to make up for the losses in a poorly architected value chain.

In their 2022 ASPLOS paper, Microsoft Azure researchers [6] estimate that they can save approximately 10 percent of overall memory cost by placing just the cold pages (infrequently accessed provisioned memory) in a CXL-based far memory tier shared between 16 and 32 servers.

Industry's roadmap of memory disaggregation

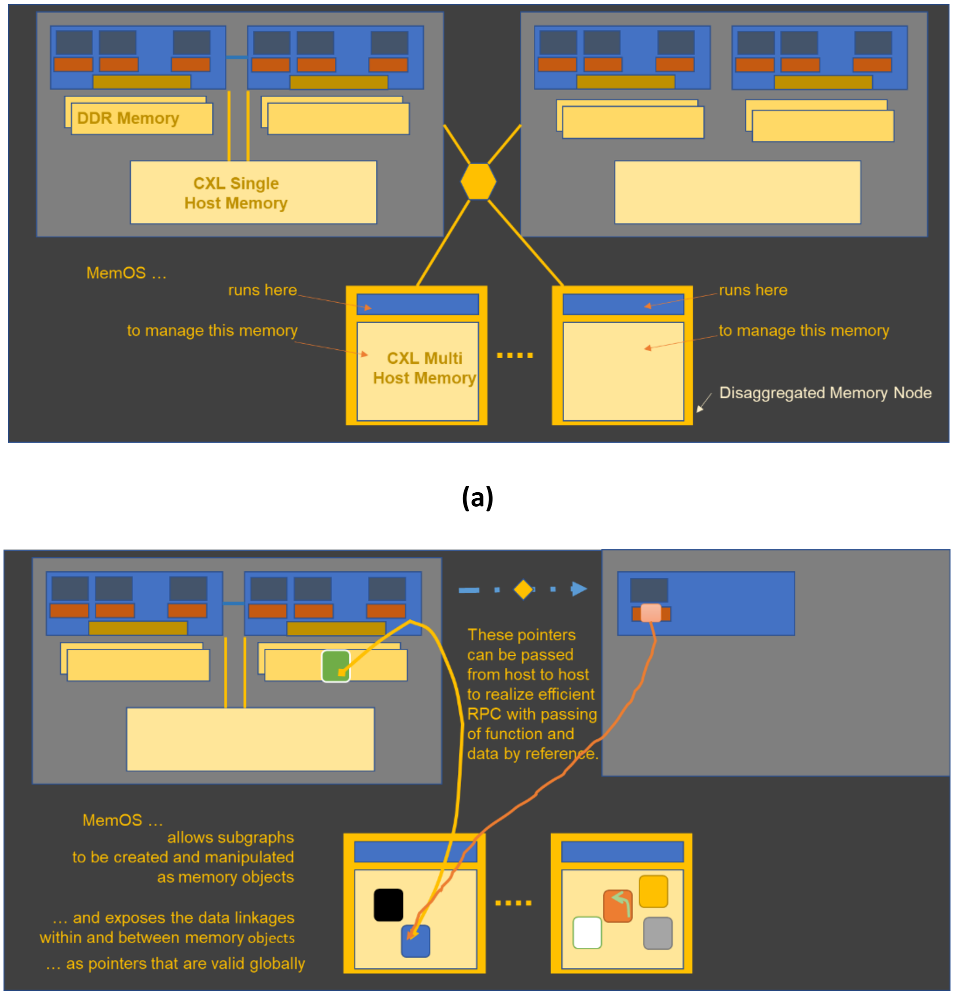
Given that the demand for memory keeps rising due to the growth of memory-intensive workloads, architects will need to get much more aggressive about leveraging memory as a far, fungible, and shared resource. There has been some recognition that bottom-up, hardware developments such as CXL are merely a first step in the right direction. Barroso, et al.’s guidance [2] is that software needs to evolve for more workloads (than just Spark) to take advantage of memory that is cost-effectively deployed but may incur higher latency.

There are unique software requirements for disaggregated memory. The first of these is the friction of using rich data in disaggregated memory from independently scaled CPUs. The second is an enhanced need for leveraging hardware mechanisms to raise the level of security for data in CXL memory, which is technically located outside the CPU and may therefore outlive processors and processes. A related final issue is state consistency in the face of decoupled CPU and memory failures.

The principal difficulty of multiple hosts accessing data in disaggregated memory is that the virtual to physical address translation context of that data is a property of the process that is managed for the process by using microprocessor hardware mechanisms such as page table entries and memory management units.

New device-side software is drawing upon the analogy between memory and storage and building for disaggregated memory what services such as S3 did for cloud storage, a foundation based on self-contained objects [7]. In these new products Memory Objects rescue the translation context required by graph-structure data and compute and embed the necessary information in the form of a foreign object table that resides at a known location in every memory object.

Memory-efficient pointers take advantage of properly constructed objects (mostly intra-object pointers) to store unique 128-bit global object identifiers within the foreign object table for resolving extra-object pointers. Intra-object pointers can avoid the overhead by storing just the intra-object offsets. (Fig. 2a) Such techniques allow a MemOS (Memory Operating System) to expose global references (Fig. 2b) that can be used in describing computations and data that (a) can be placed flexibly within the disaggregated system and (b) can use the more efficient parameter passing by reference to communicate pointers to data between services [8] rather than the relatively inefficient parameter passing by value used in current Remote Procedure Call (RPC) mechanisms used by existing data-rich microservices.



(a)

(b)

Figure 2. MemOS Theory of Operations (Image Courtesy of Elephance Memory, Inc.)

New operating systems software for Disaggregated Memory Nodes knows how to keep out of the hardware data path except in the events of memory allocation, deallocation, or pointer dereferencing. However, there is also an enhanced need to protect the data held in far memory even after the failure of a process, operating system, or server hosting the computation that last wrote the data. MemOS will evolve to exploit *Architectural Capabilities* [9] which are hardware-enforced permission mechanisms that deliver spatial, temporal, and referential safety even to memory-unsafe languages.

Finally, much as the work on Sinfonia [10] did 15 years ago for network distributed memory, the software work for disaggregated memory needs to offer a safe way to mutate data held in far memory without risking consistency should failure occur at either end of the remote operation. Fresh research is currently in progress to address that issue.

Conclusion

Memory disaggregation is addressing a problem with high economic impact in data center servers. To realize the full potential of this new technology, software will evolve to exploit far and fungible memory through safe, portable and efficient mechanisms that enhance data sharing and respect data gravity.

References

[1] P. S. Rao and G. Porter, "Is Memory Disaggregation Feasible? A Case Study with Spark SQL," in *Symposium on Architectures for Networking and Communications Systems (ANCS '16)*, New York, NY, 2016.

[2] L. Barroso, M. Marty, D. Patterson and P. Ranganathan, "Attack of the killer microseconds," *Commun. ACM,* pp. 48-54, April 2017.

[3] CXL Consortium, "CXL 2.0 Specification," 10 Nov 2020. [Online]. Available: https://www.computeexpresslink.org/download-the-specification. [Accessed 20 June 2022].

[4] K. T. Lim, J. . Chang, T. . Mudge, P. . Ranganathan, S. K. Reinhardt and T. F. Wenisch, "Disaggregated memory for expansion and sharing in blade servers," *ACM Sigarch Computer Architecture News,* vol. 37, no. 3, pp. 267-278, 2009.

[5] J. Shalf, G. Michelogiannakis, B. Austin, T. Groves, M. Ghobadi, L. Dennison, T. Gray, Y. Shen, M. Y. Teh, M. Glick and K. Bergman, "Photonic Memory Disaggregation in Datacenters," in *Adv Photonics Congress*, 2020.

[6] D. S. B. S. N. L. H. D. E. P. Z. M. S. I. A. M. D. H. M. F. R. B. Huaicheng Li, "First-generation Memory Disaggregation for Cloud Platforms," https://arxiv.org/abs/2203.00241, 2022.

[7] D. Bittman, P. Alvaro, P. Mehra, D. D. Long and E. L. Miller, "Twizzler: a data-centric OS for non-volatile memory," *ACM Transactions on Storage,* vol. 17, no. 2, pp. 1-31, May 2021.

[8] D. Bittman, R. Soule, E. L. Miller, V. Shrivastav, P. Mehra, M. Boisvert, A. Silberschatz and P. Alvaro, "Don't Let RPCs Constrain Your API," in *HotNets '21: Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks*, 2021.

[9] J. . Woodruff, R. N. M. Watson, D. . Chisnall, S. W. Moore, J. . Anderson, B. . Davis, B. . Laurie, P. G. Neumann, R. . Norton and M. . Roe, "The CHERI capability model: revisiting RISC in an age of risk," *ACM Sigarch Computer Architecture News,* vol. 42, no. 3, pp. 457-468, 2014.

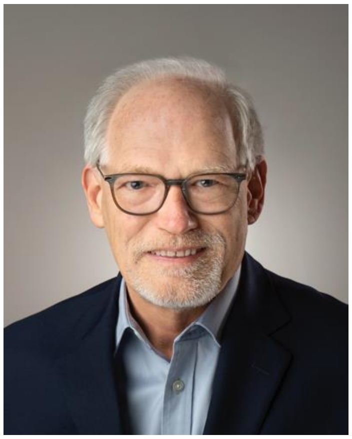
[10] M. Aguilera, A. Merchant, M. Shah, A. Veitch and C. Karamanolis, "Sinfonia: a new paradigm for building scalable distributed systems," in *SOSP '07: Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, 2007.

About the authors:

Pankaj Mehra

Pankaj Mehra, Founder of Elephance Memory, has held executive technology and management positions in Memory and Storage industries since 2013. He was previously Senior Fellow at SanDisk and Western Digital, and Distinguished Technologist at Hewlett-Packard. Pankaj has held faculty and visiting positions at IIT Delhi, UC Santa Cruz, and IBM TJ Watson Research Center. He is an author/inventor with more than 100 books, papers, and patents.

Thomas M. Coughlin



Tom Coughlin, President, Coughlin Associates, consults, publishes books and market and technology reports, puts on digital storage-oriented events and is a regular storage and memory contributor for forbes.com and M&E organization websites. Dr. Coughlin is an IEEE Fellow, Past-President of IEEE-USA, Past Director of IEEE Region 6 and Past Chair of the Santa Clara Valley IEEE Section, Chair of the Consultants Network of Silicon Valley and is also active with SNIA and SMPTE. For more information on Tom Coughlin and his activities go to www.tomcoughlin.com.