Working with Disaggregated Systems. What are the Challenges and Opportunities of RDMA and CXL?

Andreas Geyer[1] Daniel Ritter[2] Donghun Lee[3] Minseon Ahn[3] Johannes Pietrzyk[1]
Alexander Krause[1] Dirk Habich[1] Wolfgang Lehner[1]

Keywords: RDMA; CXL; Disaggregated Systems; Network; Memory

1 Motivation

The usage of disaggregated systems in large scale data-centers offers a lot of flexibility and easy scalability in comparison to the traditional statically configured scale-up and scale-out systems. Disaggregated architectures allow for the creation of software composable systems [Li17, Li18]. On the one hand, this allows seamless integration of specialized hardware like FPGAs or GPUs as well as a high degree of elasticity to scale a system with its workload by dynamically adding and removing resources via software. On the other hand, however, it also brings several challenges like an additional communication overhead for memory accesses, which is especially critical for In-Memory databases.

With the more traditional scale-up system approach, all communication happens on the same machine and is – depending on the interconnect and support by specific hardware components [Yu07] – reasonably fast. To access data from main-memory or storage, it is only necessary to retrieve it from the directly attached hardware. With growing systems, the multi-socket system became the de facto standard. This introduced the challenge of *non-uniform memory access (NUMA)* [Ps16]. The larger distance to hardware on the same machine but connected to another socket introduces a latency overhead when collecting data from this hardware. With growing NUMA distance to the desired hardware, the latency overhead grows for each communication. Therefore, such systems are designed to keep the NUMA distances as small as possible, which leads to the Near-Memory Processing (NMP) or compute paradigm [Ps16, Ki14, Pa10]. Modern software has already mostly adapted to the challenges NUMA entails, but it is still a research field of its own.

In comparison to these traditional system approaches, the usage of disaggregated systems in large scale data-centers offer a lot of flexibility and easy scalability. It allows for the

[1] Technische Universität Dresden, Database Systems Group, Nöthnitzer Straße 46, 01187 Dresden, Germany, {first.last}@tu-dresden.de
[2] SAP SE, Germany, daniel.ritter@sap.com
[3] SAP Labs, Korea, [dong.hun.lee|minseon.ahn]@sap.com

doi:10.18420/BTW2023-47

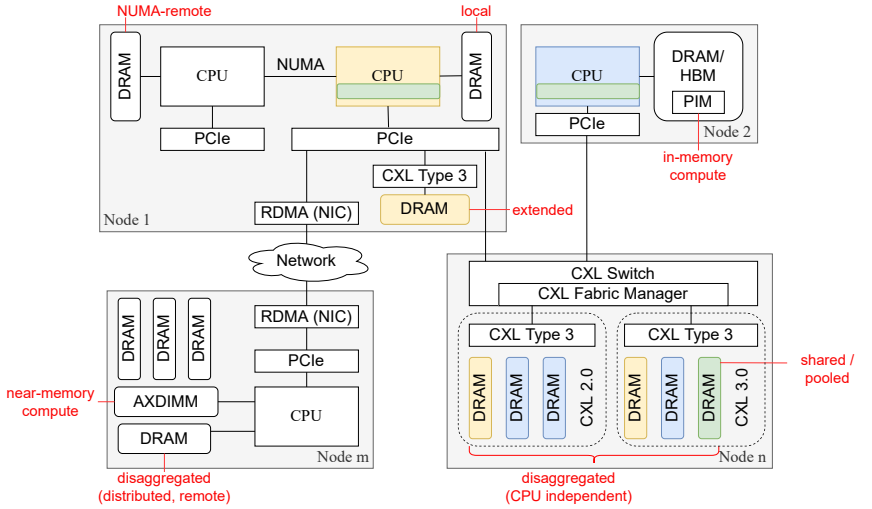

Fig. 1: Disaggregated Memory Approaches

creation of software composable systems [Wa22]. Therefore, the utilization of the available hardware can be a lot better than in scale-up or scale-out scenarios. If memory or CPU is not needed anymore, it is returned to the pool and can immediately be reassigned for other tasks. However, composing the hardware through any virtualization leads to physically distributed hardware and thus the inevitable necessity to cope with NUMA effects. Traditionally, performance optimization was achieved through either copying the data to the execution unit or moving the function to the data. This communication is a potential bottleneck for fast computation even though fast communication techniques like *Remote Direct Memory Access (RDMA)* and *Compute Express Link (CXL)* exist.

2 Classification

From a database perspective, data and its locality has always been the center of attention. Correctly deciding on which data to prefetch from disk into memory was crucial for high performance. However, hardware in the cloud setting advanced over the years from scale-up and scale-out servers to disaggregated systems. Thus, imposing an even larger design space to consider for data placement. With our research, we want to investigate the implications of different hardware or memory extension techniques, based on RDMA and CXL. To achieve that, we introduce a classification of the different memory distances, based on their actual physical distance but taking the transport layer into account.

Figure 1 highlights the multitude of potential distances that can occur when scale-up servers meet hardware disaggregation. A single compute node can consist of multiple sockets, where a socket can then be further expanded both via RDMA, e.g. over InfiniBand, and

CXL-based switches. Hence, we first divide memory into *local* and *NUMA-remote*, which is meant to physically reside in the same server. *Extended memory* describes CXL memory devices, which are directly attached via CXL through the PCIe connector of the mainboard. Devices, that are attached to a socket through a CXL switch are called *disaggregated memory* (DM). With CXL 3.0 it will be possible to share this DM between multiple systems. When RDMA is used to connect two systems, we consider such memory to be *disaggregated remote memory* (DRM). The distinction between DM and DRM is made because the DM is managed from the host on the *active* side. DRM, however, is managed by another host, which is just exposing a part of its own memory to remote systems. Additionally, technologies like *high bandwidth memory* (HBM) [Ju17] and *processing in memory* (PIM) [GHI95] can be used and further increase the performance of the system.

3 Use Cases

Recent research has already outlined the importance of available memory for in-memory database systems [Ah22]. Without claiming completeness, we see a couple of main use cases for database centric systems that are necessary to consider when working with hardware disaggregation.

First, dynamic memory expansion for *In-Memory Database Management Systems* (IMDBMS) through CXL. Even when data keeps increasing and the DIMM slots in the server are already full, the customer can expand the memory space without upgrading their server nor memory devices for scale-up. For this, the IMDBMS may allocate the operational memory in the expanded memory without any change in data placement or may move the table data to the expanded memory. The latter one would be more beneficial as table data accesses consist of lots of sequential accesses than operational memory in general and the longer latency of expanded memory can be hidden by prefetching.

Second, integrating memory expansion for multiple sockets of a single server through CXL 2.0, since it allows the connection of multiple CXL devices through a CXL switch. Attaching only one socket in a server to a CXL switch will cause NUMA-like effects among the sockets. Providing a single memory pool, which can be accessed by every socket of the server will contribute to even access latency with better bandwidth. CXL 2.0 does not allow full sharing of a whole memory device, hence we would allocate dedicated regions per socket. A limiting factor of the effectiveness for this scenario is the hardware itself: currently, we would need the same amount of wired connections between the memory pool and each socket of the server. With more servers and hence more sockets in the joint, concurrency becomes an even more serious issue and dedicated memory areas are an important building block for multi-server memory pools. Further, the amount of required wired connections scales linearly to the amount of sockets of all servers.

Third, a shared memory pool among multiple servers, but through CXL 3.0 and hence the ability of sharing the pool as a whole. With the newest CXL standard, even the same memory

regions on a device can be shared among multiple hosts. This new degree of freedom requires precise rights management, i.e. read-/write-permissions and data ownership.

Fourth, with PIM and dedicated RDMA-connected servers enabling the offloading of some operators to the data. Due to computational power near the data this could reduce the data transfer significantly. Therefore, when grouping the data access like in [Ge23] it would be possible to reduce the latency and interleave data transfer and computation.

Based on these use cases, we see the possibility to further increase the systems complexity by linking these multiple servers also with InfiniBand. Such a setup would allow to limit the amount of CXL wires by replacing them with less InfiniBand cables. In this configuration, one server could serve as a memory managing unit, which exposes selected data regions to the other servers.

4 Call to Action

Based on our classification and the presented use cases, we identify a set of promising research directions. Traditionally, data was either shipped to the central processing unit (CPU) or the processing function or operator was moved to the data. We argue that with the rise of high performance interconnects like RDMA and CXL, there is no such black-and-white decision anymore. Our research focuses on investigating the performance implications of the different memory categories, based on our classification from Section 2, from the perspective of an IMDBMS. We already conducted initial experiments for *disaggregated remote memory* [Ge23] and want to extend our prototype by combining the different memory classes following our use case description of Section 3. The scope of our endeavor is not limited to experimenting with memory extensions, but also on how to include computational storage or Processing In memory (PIM) in such a system.

Conceptually speaking, we want to build a database prototype, that can leverage different kinds of attached memory, based on a suitable abstraction layer. Our research aims to emit three main contributions: (1) define and confirm the different memory classes, based on our experiments with the combination of RDMA and CXL attached memory. (2) we will provide an analytical model on when to use which memory. That includes the decision of when to ship the data, e.g. because of computational limitations and when operators should be offloaded. Lastly, (3) we will identify a set of common access primitives, that can be leveraged to work with all memory classes through a dedicated API.

We would be delighted to present our ideas and the memory disaggregation classification at the workshop and discuss the presented ideas. The valuable feedback of the attendees will help us to further refine our classification both in terms of preciseness and applicability.

Acknowledgements. We would like to thank Marcel Weisgut from Hasso-Plattner-Institut, Potsdam for the fruitful discussions on CXL and his contributions to the memory disaggregation classification.

Bibliography

[Ah22] Ahn, Minseon; Chang, Andrew; Lee, Donghun; Gim, Jongmin; Kim, Jungmin; Jung, Jaemin; Rebholz, Oliver; Pham, Vincent; Malladi, Krishna T.; Ki, Yang-Seok: Enabling CXL Memory Expansion for In-Memory Database Management Systems. In: DaMoN. ACM, pp. 8:1–8:5, 2022.

[Ge23] Geyer, Andreas; Krause, Alexander; Habich, Dirk; Lehner, Wolfgang: Pipeline Group Optimization on Disaggregated Systems. In: CIDR. 2023. to appear.

[GHI95] Gokhale, M.; Holmes, B.; Iobst, K.: Processing in memory: the Terasys massively parallel PIM array. Computer, 28(4):23–31, 1995.

[Ju17] Jun, Hongshin; Cho, Jinhee; Lee, Kangseol; Son, Ho-Young; Kim, Kwiwook; Jin, Hanho; Kim, Keith: HBM (High Bandwidth Memory) DRAM Technology and Architecture. In: 2017 IEEE International Memory Workshop (IMW). pp. 1–4, 2017.

[Ki14] Kissinger, Thomas; Kiefer, Tim; Schlegel, Benjamin; Habich, Dirk; Molka, Daniel; Lehner, Wolfgang: ERIS: A NUMA-Aware In-Memory Storage Engine for Analytical Workload. In (Bordawekar, Rajesh; Lahiri, Tirthankar; Gedik, Bugra; Lang, Christian A., eds): International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures - ADMS 2014, Hangzhou, China, September 1, 2014. pp. 74–85, 2014.

[Li17] Li, Chung-Sheng; Franke, Hubertus; Parris, Colin; Abali, Bülent; Kesavan, Mukil; Chang, Victor I.: Composable architecture for rack scale big data computing. Future Gener. Comput. Syst., 67:180–193, 2017.

[Li18] Lin, An-Dee; Li, Chung-Sheng; Liao, Wanjiun; Franke, Hubertus: Capacity Optimization for Resource Pooling in Virtualized Data Centers with Composable Systems. IEEE Trans. Parallel Distributed Syst., 29(2):324–337, 2018.

[Pa10] Pandis, Ippokratis; Johnson, Ryan; Hardavellas, Nikos; Ailamaki, Anastasia: Data-Oriented Transaction Execution. Proc. VLDB Endow., 3(1):928–939, 2010.

[Ps16] Psaroudakis, Iraklis; Scheuer, Tobias; May, Norman; Sellami, Abdelkader; Ailamaki, Anastasia: Adaptive NUMA-aware data placement and task scheduling for analytical workloads in main-memory column-stores. Proc. VLDB Endow., 10(2):37–48, 2016.

[Wa22] Wang, Ruihong; Wang, Jianguo; Idreos, Stratos; Özsu, M. Tamer; Aref, Walid G.: The Case for Distributed Shared-Memory Databases with RDMA-Enabled Memory Disaggregation. CoRR, abs/2207.03027, 2022.

[Yu07] Yu, Hao; Moreira, José E.; Dube, Parijat; Chung, I-Hsin; Zhang, Li: Performance Studies of a WebSphere Application, Trade, in Scale-out and Scale-up Environments. In: 21th International Parallel and Distributed Processing Symposium (IPDPS 2007), Proceedings, 26-30 March 2007, Long Beach, California, USA. IEEE, pp. 1–8, 2007.