Related Works

Mark Madler

1 KVS over RDMA

1.1 Kite: efficient and available release consistency for the datacenter

This is the real entry paper. This is a replicated KVS over RDMA with a "Linearizable variant of" **Release Consistency**. Some mention of Release Consistency as a "one sided barrier." Compares against Zookeeper and Derecho. Uses some monotonically increasing clocks to track versions.[11]

1.2 FaRM: Fast Remote Memory

Super similar to the entry paper in that is is a KVS for RDMA, but this one is I beleive either disaggregated or not. Farm could really be classified as a KVS or even a protocol. This design always replicates state info. Provides **strict serializability**, so Transaction based consistency. So FaRM provides lock-free reads. The authors mention a "shared address space." [6]

1.3 FaRMv2: Fast General Distributed Transactions with Opacity

Just like FaRM but with opacity. Also providing **strict serializability**. Idk if this paper is really something to look at or not (TODO)[26]

1.4 HERD: Using RDMA efficiently for key-value services

Herd. Fast because it uses a weird combination of UD and message things. This combination is essentiall UC (unreliable connection) requests places as writes to the server, then serviced by a server polling thread. The server replies over UD (unreliable datagram) (this is a 2-sided send). One of the main ideas of this paper is that writes can be faster than reads because there is (at least with unreliable connection) no need for the destination to respond. In the future it might be wise to refer back to this paper for info about the performance metrics from RDMA verbs.[18]

1.5 Sherman: A Write-Optimized Distributed B+Tree Index on Disaggregated Memory

This system is disaggregated. Node-granularity. "consistent". Optimizations are established to target mainly chained atomic operations and write-amplification. [31]

1.6 Using One-Sided RDMA Reads to Build a Fast, CPU-Efficient Key-Value Store (Pilaf)

Linearizable data store. Self-correcting. Note that this paper has a great analysis of IPoIB as well as RDMA vs verbs latency and thrup. [24]

1.7 Scythe: A Low-Latency RDMA-enabled Distributed Transaction System for Disaggregate Memory

KVS again. Seels to optimize concurrency control, timestamping, bandwidth. Uses Timestamp Oracle (TSO) hot-aware concurrecy control. Allows RPC. [22]

1.8 HCL: Distributing Parallel Data Structures in Extreme Scales

Strictly serializabile KVS. HCL stands for Hermes Containder Library. [5]

1.9 BCL: A Cross-Platform Distributed Data Structures Library

Confusingly uses either MPI, OpenSHMEM, GASNET-EX, and UPC++ as backends. Must be slow. [3]

1.10 Rolex

I don't think this one really works. Uses learned indexes. splits compute and memory. [21]

1.11 Exploiting Hybrid Index Scheme for RDMA-based Key-Value Stores*** (Hstore)

Seeks to solve issue of range-lookups. Compares against Sherman and Clover(LOOK INTO CLOVER). This does indeed outperform sherman on YCSB. Maybe problematic for LOCO. claims 54% improvement over sherman.[13]

1.12 Disaggregating Persistent Memory and Controlling Them Remotely: An Exploration of Passive Disaggregated Key-Value Stores

"Clover". Workes on Persistent memory. A bit older, not worth comparing against unless looking into PM. also maybe faster than Sherman but definitely not the fastest.[28]

1.13 AStore: Uniformed Adaptive Learned Index and Cache for RDMA-Enabled Key-Value Store

Supposedly outperforms **Sherman** by A LOT. ALso compares against "xstore" "Rolex" and "hybrid (not to be confused with the above hybrid index scheme which also independetly claims to be faster than Sherman)". The results also suggest that Rolex outperforms **Sherman**. claims up to 91% improvement over **Sherman**[25]

1.14 Maintaining Cache Consistency in RDMA-based Distributed Key-Value storage System

note really worth considering it seems. This is indeed a KVS but only seems to compare against scale store. [15]

1.15 FastStore: A High-Performance RDMA-Enabled Distributed Key-Value Store with Persistent Memory

not really worth considering. Again PM. COmpares against Clover and Sherman. Also is faster than Sherman with lower latency as well as higher throughput. [33]

1.16 LoLKV: The Logless, Linearizable, RDMA-based Key-Value Storage System

Weirdly does not compare against state of the art really. All of the comparisons are against older consensus protocols. LoLKV is worth looking at, but sincethere are no comparisons against Sherman I am more worried about the other papers that do. It seems like this paper is more concerned about the consensus protocols. [1]

2 DSM systems

2.1 Scaling out NUMA-Aware Applications with RDMA-Based Distributed Shared Memory: MAGI

Page-based DSM. [14]

2.2 Efficient Distributed Memory Management with RDMA and Caching

cache-line granularity DSM.[4]

2.3 Distributed Shared Object Memory

object based granularity, release consistency... too old for RDMA[12]

2.4 Gengar: An RDMA-based Distributed Hybrid Memory Pool

This is object based dsm over rdma but with non-volatile memory as well using Intel Optane. Seems to also use this lease assignment idea like in [8] but is not page based.[7]

2.5 TreadMarks: shared memory computing on networks of workstations

Was implemented over IP, lazy release consistency I think. Not sure of granularity yet.[2]

2.6 LITE Kernel RDMA Support for Datacenter Applications

This is page based DSM using the kernel. [29]

2.7 MENPS: A Decentralized Distributed Shared Memory Exploiting RDMA

- Page based DSM
- Special Diff merging and page sharing
- Combine write notices and logical leases (what is that?)[8]

2.8 Argo DSM

Page-based DSM again but directory coherence. This was maybe the first RDMA-based DSM paper, at least thats what the authors allude to.[20]

2.9 GiantVM: A Novel Distributed Hypervisor for Resource Aggregation with DSM-aware Optimizations

Page-based DSM again but also works over TCP and RDMA[17]

2.10 Scalable RDMA performance in PGAS languages

This paper is for PGAS languages. Has an address hash table similar to LOCO for remote lookups.[9]

2.11 misc RDMA coherence papers

2.12 Misc PGAS languages probably

3 Protocols over RDMA for Consistency

3.1 Notes on PGAS and "protocols"

It seems like there are not agreed upon semantics on what is a protocol. PGAS is a memory model. For myself and posterity, PGAS is a memory model where the address space is logicaly partitioned across nodes. This means that there can be some reasoning about locality likely from virtual memory addresses.

Some notes on MPI. MPI is a programming model which supports multiple communication backends. MPI can run over IP, RDMA, etc.

3.2 Odyssey: The Impact of Modern Hardware on Strongly-Consistent Replication Protocols

This paper is a summary of protocols used for RDMA communication. These protocols were used to enforce consistency and were tested with a series of KVSs. This paper is related to Kite (same authors) and Kite is one of the KVSs tested.[10]

3.3 Hermes: A Fast, Fault-Tolerant and Linearizable Replication Protocol

This paper [19] is one of the Protocols tested by the above paper Odyssey [10]. This protocol guarantees linearizablity and is designed to work on replicated store systems.

3.4 Hamband: RDMA Replicated Data Types

This paper [16] designed new RDMA data types that are replicated across nodes. This paper is sort of a protocol paper as it implements this protocol to keep replicated data through either relaxed or 'strong consistency'.

3.5 Evaluation of RDMA opportunities in an Object-Oriented DSM

Interesting result is that it proves that invalidation protocols are better suited for distributed systems. [30]

4 table i found

TABLE 1
Categories of RDMA-Based Storage Systems and Software Techniques

System Types	Related Works
Key-value Store	HERD [6] ccKVS [7] FaSST [8] Pilaf [9] RFP [10] HydraDB [11] C-Hint [12] DrTM [13] FaRM [14] Nessie [15] RStore [16] ScaleTX [17] Cell [18] Catfish [19] NAM-Tree [20] NVDS [21] FlatStore [22] RDMP-KV [23] RACE [24] RAMCloud [25] Sherman [32]
File System	CephFS [33] GlusterFS [34] Crail [35] NVFS [36] Octopus [5] Orion [37] FileMR [38] Assise [39] DeltaFS [40] GekkoFS [41] DAOS [42] PolarFS [43] Lustre [44] GPFS [45] BeeGFS [46] PVFS2 [47]
Distributed Memory	FaRM [14] RackOut [48] Grappa [49] InfiniSwap [50] Hotpot [51] Clover [52] AsymNVM [53] Kona [54] CoRM [55]
Databases	NAM-DB [56], [57] Chiller [58] PolarDB Serverless [59] D-RDMA [60] Zamanian *et al.* [61] Li *et al.* [62] HyPer [63] Barthels *et al.* [64] I-Store [65] L5 [66] Liu *et al.* [67]
Smart NICs	FlexNIC [68] KV-Direct [69] Lynx [70] StRoM [71] LineFS [72] Xenic [73] 1RMA [28] D-RDMA [60] HyperLoop [74]
Core Modules	**Related Works**
Communication Mode	DrTM-H [75] Cell [18] Catfish [19] Storm [76] DaRPC [77] HERD [6] FaSST [8] RF-RPC [78] ScaleRPC [17] Storm [76] Octopus [5] FlatStore [22] LITE [79] eRPC [80] Accelio [81] Mercury [82] X-RDMA [29] FLOCK [83] DFI [84] HatRPC [85]
Concurrency Control	DrTM [13] FaRM [14] Cell [18] NAM-Tree [20] Pilaf [9] RACE [24]
Fault Tolerance	HydraDB [11] Mojim [86] Orion [37] Tailwind [87] HyperLoop [74] DARE [88] APUS [89] Derecho [90] Odyssey [91] INEC [92] Aguilera *et al.* [93] Zamanian *et al.* [61]
Caching Resource Management	GAM [94] Aguilera *et al.* [95] DrTM [13] HydraDB [11] C-Hint [12] XStore [96] RACE [24] Kumar *et al.* [97] HERD [6] FaSST [8] FaRM [14] LITE [79] ScaleRPC [17] X-RDMA [29] FLOCK [83]

[23]

5 Loosely Related but Evaluated

5.1 CoRM: Compactable Remote Memory over RDMA

page based I think (re-read this)[27]

5.2 Rcmp: Reconstructing RDMA-Based Memory Disaggregation via CXL

page based and uses CXL, not comparable[32]

References

[1] Alquraan, A., Udayashankar, S., Marathe, V., Wong, B., and Al-Kiswany, S. LoLKV: The logless, linearizable, RDMA-based Key-Value storage system. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)* (Santa Clara, CA, 2024), USENIX Association, pp. 41–54.

[2] Amza, C., Cox, A., Dwarkadas, S., Keleher, P., Lu, H., Rajamony, R., Yu, W., and Zwaenepoel, W. Treadmarks: shared memory computing on networks of workstations. *Computer 29*, 2 (1996), 18–28.

[3] Brock, B., Buluç, A., and Yelick, K. Bcl: A cross-platform distributed data structures library. In *Proceedings of the 48th International Conference on Parallel Processing* (New York, NY, USA, 2019), ICPP '19, Association for Computing Machinery.

[4] Cai, Q., Guo, W., Zhang, H., Agrawal, D., Chen, G., Ooi, B. C., Tan, K.-L., Teo, Y. M., and Wang, S. Efficient distributed memory management with rdma and caching. *Proc. VLDB Endow. 11*, 11 (July 2018), 1604–1617.

[5] Devarajan, H., Kougkas, A., Bateman, K., and Sun, X.-H. Hcl: Distributing parallel data structures in extreme scales. In *2020 IEEE International Conference on Cluster Computing (CLUSTER)* (2020), pp. 248–258.

[6] Dragojević, A., Narayanan, D., Hodson, O., and Castro, M. Farm: fast remote memory. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation* (USA, 2014), NSDI'14, USENIX Association, p. 401–414.

[7] Duan, Z., Liu, H., Lu, H., Liao, X., Jin, H., Zhang, Y., and He, B. Gengar: An rdma-based distributed hybrid memory pool. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)* (2021), pp. 92–103.

[8] Endo, W., Sato, S., and Taura, K. Menps: A decentralized distributed shared memory exploiting rdma. In *2020 IEEE/ACM Fourth Annual Workshop on Emerging Parallel and Distributed Runtime Systems and Middleware (IPDRM)* (2020), pp. 9–16.

[9] Farreras, M., Almasi, G., Cascaval, C., and Cortes, T. Scalable rdma performance in pgas languages. pp. 1–12.

[10] Gavrielatos, V., Katsarakis, A., and Nagarajan, V. Odyssey: the impact of modern hardware on strongly-consistent replication protocols. In *Proceedings of the Sixteenth European Conference on Computer Systems* (New York, NY, USA, 2021), EuroSys '21, Association for Computing Machinery, p. 245–260.

[11] Gavrielatos, V., Katsarakis, A., Nagarajan, V., Grot, B., and Joshi, A. Kite: efficient and available release consistency for the datacenter. In *Proceedings of the 25th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (New York, NY, USA, 2020), PPoPP '20, Association for Computing Machinery, p. 1–16.

[12] Guedes, P., and Castro, M. Distributed shared object memory. In *Proceedings of IEEE 4th Workshop on Workstation Operating Systems. WWOS-III* (1993), pp. 142–149.

[13] Han, S., Zhang, M., Jiang, D., and Xiong, J. Exploiting hybrid index scheme for rdma-based key-value stores. In *Proceedings of the 16th ACM International Conference on Systems and Storage* (New York, NY, USA, 2023), SYSTOR '23, Association for Computing Machinery, p. 49–59.

[14] Hong, Y., Zheng, Y., Yang, F., Zang, B.-Y., Guan, H.-B., and Chen, H.-B. Scaling out numa-aware applications with rdma-based distributed shared memory. *Journal of Computer Science and Technology 34* (2019), 94–112.

[15] Hou, Z., Zheng, Q., Song, Y., and Nie, X. Maintaining cache consistency in rdma-based distributed key-value storage system. In *2024 7th International Conference on Data Science and Information Technology (DSIT)* (2024), pp. 1–6.

[16] Houshmand, F., Saberlatibari, J., and Lesani, M. Hamband: Rdma replicated data types. In *Proceedings of the 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation* (New York, NY, USA, 2022), PLDI 2022, Association for Computing Machinery, p. 348–363.

[17] Jia, X., Zhang, J., Yu, B., Qian, X., Qi, Z., and Guan, H. Giantvm: A novel distributed hypervisor for resource aggregation with dsm-aware optimizations. *ACM Trans. Archit. Code Optim. 19*, 2 (Mar. 2022).

[18] Kalia, A., Kaminsky, M., and Andersen, D. G. Using rdma efficiently for key-value services. In *Proceedings of the 2014 ACM Conference on SIGCOMM* (New York, NY, USA, 2014), SIGCOMM '14, Association for Computing Machinery, p. 295–306.

[19] Katsarakis, A., Gavrielatos, V., Katebzadeh, M. S., Joshi, A., Dragojevic, A., Grot, B., and Nagarajan, V. Hermes: A fast, fault-tolerant and linearizable replication protocol. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (New York, NY, USA, 2020), ASPLOS '20, Association for Computing Machinery, p. 201–217.

[20] Kaxiras, S., Klaftenegger, D., Norgren, M., Ros, A., and Sagonas, K. Turning centralized coherence and distributed critical-section execution on their head: A new approach for scalable distributed shared memory. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing* (New York, NY, USA, 2015), HPDC '15, Association for Computing Machinery, p. 3–14.

[21] Li, P., Hua, Y., Zuo, P., Chen, Z., and Sheng, J. Rolex: a scalable rdma-oriented learned key-value store for disaggregated memory systems. In *Proceedings of the 21st USENIX Conference on File and Storage Technologies* (USA, 2023), FAST'23, USENIX Association.

[22] Lu, K., Zhao, S., Shan, H., Wei, Q., Li, G., Wan, J., Yao, T., Wu, H., and Wang, D. Scythe: A low-latency rdma-enabled distributed transaction system for disaggregated memory. *ACM Trans. Archit. Code Optim. 21*, 3 (Sept. 2024).

[23] Ma, S., Ma, T., Chen, K., and Wu, Y. A survey of storage systems in the rdma era. *IEEE Transactions on Parallel and Distributed Systems 33*, 12 (2022), 4395–4409.

[24] Mitchell, C., Geng, Y., and Li, J. Using one-sided rdma reads to build a fast, cpu-efficient key-value store. In *Proceedings of the*

2013 USENIX Conference on Annual Technical Conference (USA, 2013), USENIX ATC'13, USENIX Association, p. 103–114.

[25] Qiao, P., Zhang, Z., Li, Y., Yuan, Y., Wang, S., Wang, G., and Yu, J. X. Astore: Uniformed adaptive learned index and cache for rdma-enabled key-value store. *IEEE Transactions on Knowledge and Data Engineering 36*, 7 (2024), 2877–2894.

[26] Shamis, A., Renzelmann, M., Novakovic, S., Chatzopoulos, G., Dragojević, A., Narayanan, D., and Castro, M. Fast general distributed transactions with opacity. In *Proceedings of the 2019 International Conference on Management of Data* (New York, NY, USA, 2019), SIGMOD '19, Association for Computing Machinery, p. 433–448.

[27] Taranov, K., Di Girolamo, S., and Hoefler, T. Corm: Compactable remote memory over rdma. In *Proceedings of the 2021 International Conference on Management of Data* (New York, NY, USA, 2021), SIGMOD '21, Association for Computing Machinery, p. 1811–1824.

[28] Tsai, S.-Y., Shan, Y., and Zhang, Y. Disaggregating persistent memory and controlling them remotely: an exploration of passive disaggregated key-value stores. In *Proceedings of the 2020 USENIX Conference on Usenix Annual Technical Conference* (USA, 2020), USENIX ATC'20, USENIX Association.

[29] Tsai, S.-Y., and Zhang, Y. Lite kernel rdma support for datacenter applications. In *Proceedings of the 26th Symposium on Operating Systems Principles* (New York, NY, USA, 2017), SOSP '17, Association for Computing Machinery, p. 306–324.

[30] Veldema, R., and Philippsen, M. Evaluation of rdma opportunities in an object-oriented dsm. pp. 217–231.

[31] Wang, Q., Lu, Y., and Shu, J. Sherman: A write-optimized distributed b+tree index on disaggregated memory. In *Proceedings of the 2022 International Conference on Management of Data* (New York, NY, USA, 2022), SIGMOD '22, Association for Computing Machinery, p. 1033–1048.

[32] Wang, Z., Guo, Y., Lu, K., Wan, J., Wang, D., Yao, T., and Wu, H. Rcmp: Reconstructing rdma-based memory disaggregation via cxl. *ACM Trans. Archit. Code Optim. 21*, 1 (Jan. 2024).

[33] Xiong, Z., Jiang, D., and Xiong, J. Faststore: A high-performance rdma-enabled distributed key-value store with persistent memory. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)* (2023), pp. 406–417.