Sherman: A Write-Optimized Distributed B+Tree Index on Disaggregated Memory, Qing Wang, Youyou Lu, Jiwu Shu∗ SIGMOD22

Reporter: Luo Tianyi

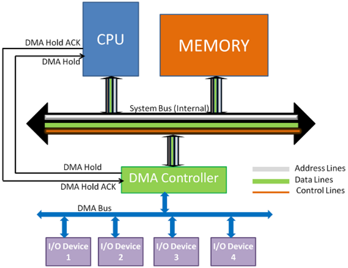
Koukou, HUST

May 21, 2023

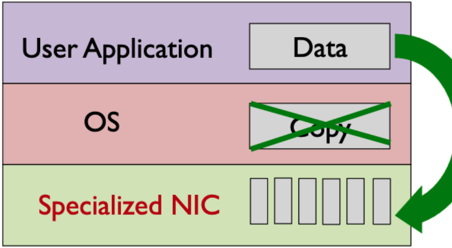
Contents

1. Background
    - RDMA
    - Disaggregated Memory
    - B+Tree Index Structure In DB
2. Bottleneck
    - Existing Approaches
    - Slow Write Operations
3. Design of Sherman
    - Overview
    - Combine
    - Hierarchical On-Chip Lock
    - Two Level Design
4. Performance
    - Exp Setup
    - Performance Test
    - Conclusion

Background : RDMA

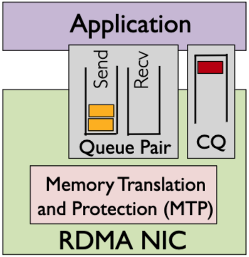


DMA Ideas

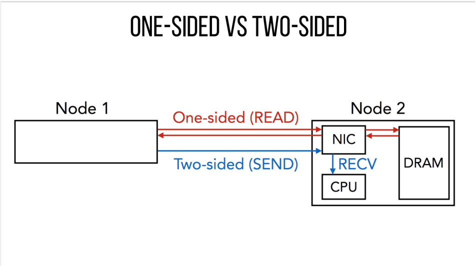


RDMA Ideas

- DMA->RDMA Reduce the number of copies, reduce CPU involvement, and support remote access.
- One-Sided and Two-Sided verbs
- Specialized NICs and communication protocols, developing to ultra-fast access

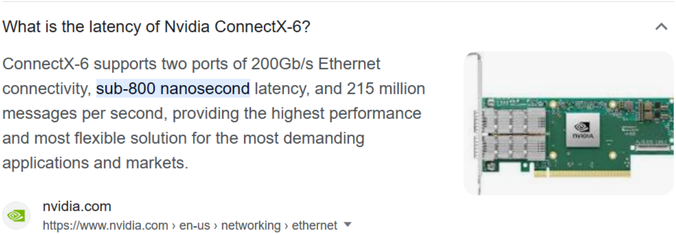


OS Bypass



Verbs

- Two-Sided: pass control messages
- One-Sided: Memory copy
- Verbs: int ibv_post_send(struct ibv_qp *qp, struct ibv_send_wr *wr, struct ibv_send_wr **bad_wr);



What is the latency of Nvidia ConnectX-6?

ConnectX-6 supports two ports of 200Gb/s Ethernet connectivity, sub-800 nanosecond latency, and 215 million messages per second, providing the highest performance and most flexible solution for the most demanding applications and markets.

nvidia.com
https://www.nvidia.com › en-us › networking › ethernet ▾

Figure: RDMA NICs Performance

- Microsecond level latency and high bandwidth
- Available for disggregated design

Background : Disaggregated Memory

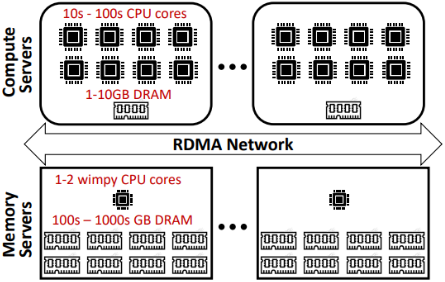


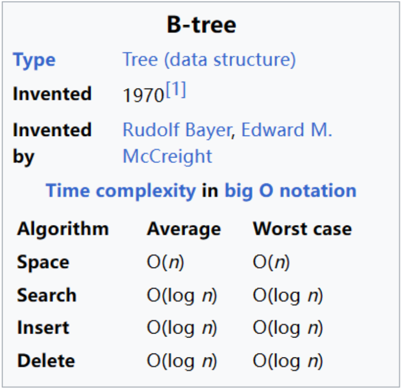
Figure 1: Architecture of Memory Disaggregation.

- Resource disggregation : Near zero computer power on memory side

Background : B+ Tree

Index Structure: A key technique to speed up massive data query, Determing the performance of DB system

B+ Tree Index: A widely-used tree index in DB systems, support range query...



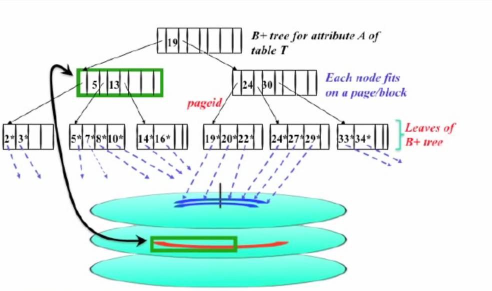
B-tree

Type	Tree (data structure)
Invented	1970[1]
Invented by	Rudolf Bayer, Edward M. McCreight

Time complexity in big O notation

Algorithm	**Average**	**Worst case**
Space	O(*n*)	O(*n*)
Search	O(log *n*)	O(log *n*)
Insert	O(log *n*)	O(log *n*)
Delete	O(log *n*)	O(log *n*)

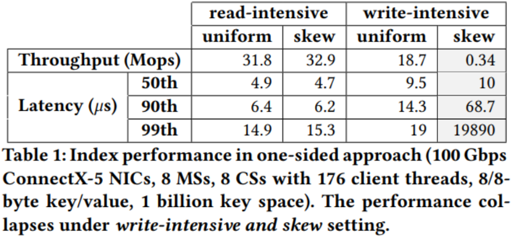
Basic structure of B+Tree:



- Fast read(query)
- Relatively slow write(insert)

Bottleneck : Existing Approaches

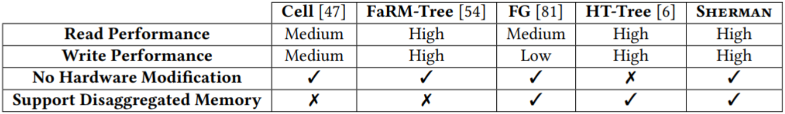
No previous design on disggregated memory.
(1) Using One-sided Verbs Purely, FG



		read-intensive		write-intensive	
		uniform	skew	uniform	skew
Throughput (Mops)		31.8	32.9	18.7	0.34
Latency (μs)	50th	4.9	4.7	9.5	10
	90th	6.4	6.2	14.3	68.7
	99th	14.9	15.3	19	19890

Table 1: Index performance in one-sided approach (100 Gbps ConnectX-5 NICs, 8 MSs, 8 CSs with 176 client threads, 8/8-byte key/value, 1 billion key space). The performance collapses under *write-intensive and skew* setting.

No previous design on disggregated memory.
(2) Extending RDMA Interfaces



	Cell [47]	**FaRM-Tree** [54]	**FG** [81]	**HT-Tree** [6]	**Sherman**
Read Performance	Medium	High	Medium	High	High
Write Performance	Medium	High	Low	High	High
No Hardware Modification	✓	✓	✓	✗	✓
Support Disaggregated Memory	✗	✗	✓	✓	✓

- Hard to deploy

Bottleneck : Slow Write Operations

Reasons? (1) Excessive Round Trips when modifying a Tree node.
What's RTT?
Round-trip time (RTT) is the duration in milliseconds (ms) it takes for a network request to go from a starting point to a destination.

A client thread needs 4 round-trips:

- Acquiring associated exclusive lock
- Reading the tree node,
- Writing back the modified tree node
- And finally releasing the lock

(2) Slow Synchronization Primitives

- Expensive in-NIC concurrency control

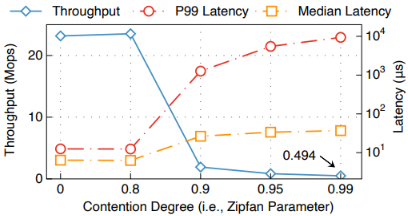


Figure 2: Performance of RDMA-based exclusive locks (ConnectX-5 NICs, 100 Gbps). The system experiences performance collapse under high-contention settings.

- Unnecessary retries.
- Lacking Fairness.

(3) Write Amplification

- Sorted layout: a lot to modify via RDMA_WRITE
- too large I/O size

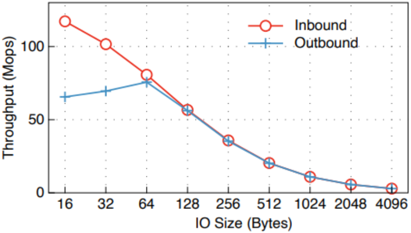


Figure 3: Performance of RDMA_WRITE (ConnectX-5 NICs, 100 Gbps). RDMA_WRITE prefers small IO size.

Design: Overview

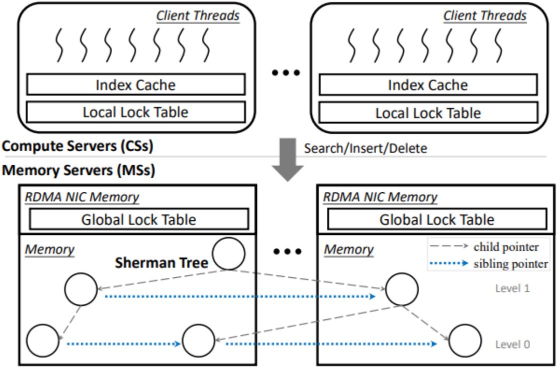


Figure 5: SHERMAN's architecture and interactions.

- Concurrency Control:
  Before modifying a tree node, the client thread must acquire the associated exclusive lock
  Use a HOCL approach.
- Cache Mechanism:
  Case 1:
  Internal nodes visited before.
  Case 2:
  The first 2 levels of nodes

Design : Combine(Reduce RTTS)

Abstract: Combine several commands together to reduce round trips.

Case 1:

Since a tree node and its lock co-locate at the same Memory node, transmit them together.

Case 2:

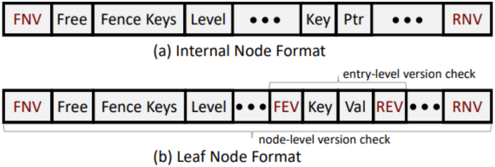
When split and the generated sibling node is in the same Memory node, transmit them together.

Design : Hierarchical On-Chip Lock(Deal with concurrency control)

HOCL leverages on-chip memory of NICs to avoid PCIe transactions at MS-side; it also maintains local locks at CS-side, to form a hierarchical structure, reduce retries.

- Memory Side: On-chip(NICs) lock table
- Compute Side: A local lock table to coordinate conflicting lock requests within the same Compute nodes.

Design : Two Level Design(Deal with the write amplification)



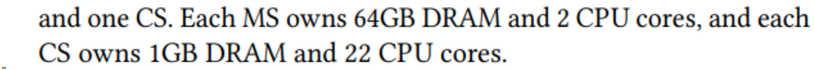
(a) Internal Node Format

(b) Leaf Node Format

Use a version check to control:
When not splitting a node, only transmit a 4-bit ”version”

Performance : Exp Setup

(1) Hardware



and one CS. Each MS owns 64GB DRAM and 2 CPU cores, and each CS owns 1GB DRAM and 22 CPU cores.

MS: Memory Server CS: Compute Server

(2) Workload

There are two types of key popularity: uniform and skewed. In uniform workloads, all keys have the same probability of being accessed. Skewed workloads follow a Zipfian access distribution, common in production environments.

(3) Evaluation Indicators

Bandwidth and Latency.

Performance : Performance Test

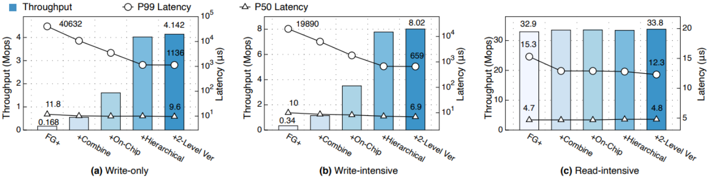


Figure 10: Contributions of techniques to performance (skewed workloads, skewness=0.99).

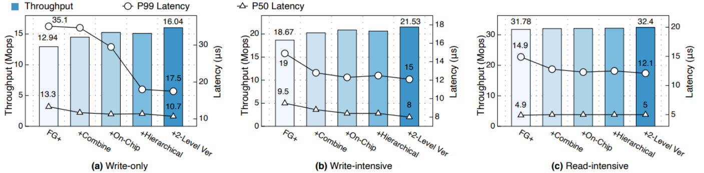


Figure 11: Contributions of techniques to performance (uniform workloads).

Performance : Conclusion

The paper proposed and evaluated Sherman, an RDMA-based B+Tree(and the first) index on disaggregated memory. Sherman introduces a set of techniques to boost write performance and outperforms existing solutions, demonstrates that combining RDMA hardware features and RDMA-friendly software designs can enable a high-performance index on disaggregated memory.

Thank You !

Luo Tianyi