Building a Distributed Database on a CXL Pod: Synchronizing Data Accesses without Networks

Yibo Huang (ybhuang@cs.utexas.edu), Vijay Chidambaram, Dixin Tang, Emmett Witchel
The University of Texas at Austin

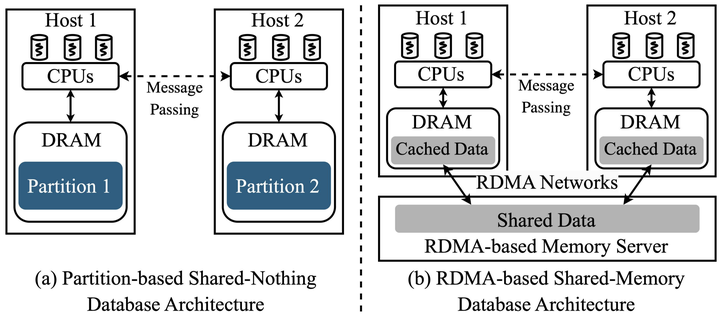
The University of Texas at Austin
Department of Computer Science
College of Natural Sciences

Semiconductor Research Corporation
PRISM

Motivation

How to scale a single-node DB to multiple nodes?

Existing work synchronize data accesses via networks, leading to numerous message exchanges and inevitable overhead



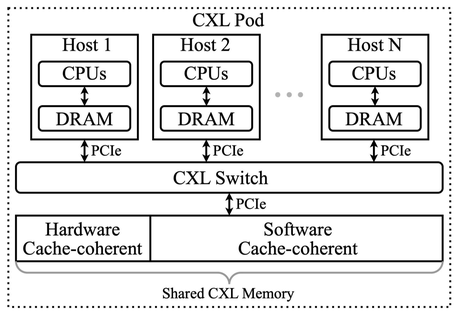
(a) Partition-based Shared-Nothing Database Architecture

(b) RDMA-based Shared-Memory Database Architecture

- (a) Partition-based Shared-nothing Architecture
  - sends messages for synchronization (e.g., 2PC)
  - bad for non-partitionable workloads
- (b) RDMA-based Shared-memory Architecture
  - uses one-sided RDMA
  - suffers from the high latency of RDMA

A New Opportunity: CXL Pod

- A collection of hosts connected to a shared CXL memory device
- Hardware-based cache coherence across hosts
- Better performance than RDMA



Sync over Memory >>> Sync over Network

- Sync over memory: shared data structure + atomic operations
- Sync over network: message passing with multiple roundtrips

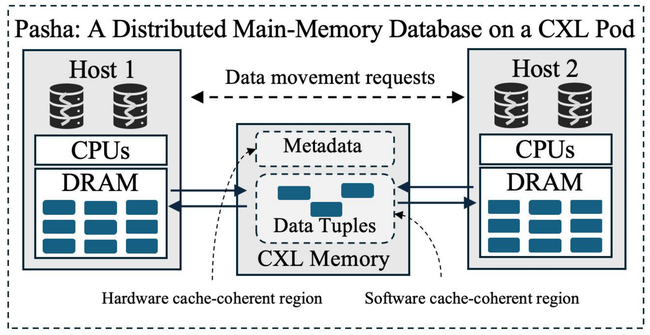
But CXL Memory has Limitations...

1. Limited hardware cache-coherence across hosts
2. 2-4× higher latency than local DRAM

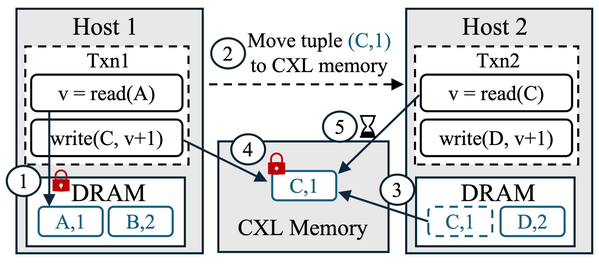
Key Insight

We only need to maintain CAT in CXL memory
(CAT: Cross-host Active Tuples)

Design & Example



- Initially, data is partitioned and stored in local DRAM
- Tuples are moved in upon requests and moved out based on policies
- Use software cache-coherence for data to reduce data movement
- Adopt 2PL and enhance it to efficiently coordinate concurrent accesses

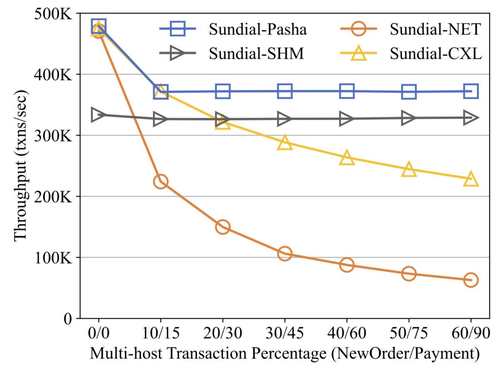


Evaluation

- Experiment Setup
  - Use VMs and a CXL 1.1 memory module for CXL pod emulation
  - Use SR-IOV to emulate Ethernet

Component	Description
OS (Kernel)	Ubuntu 22.04.2 LTS (Linux kernel v5.19)
CPU (Intel SPR)	2× Intel® Xeon 8460H CPUs @2.2 GHz [42] 40 cores and 105 MB LLC per CPU
RAM	8× DDR5-4800 channels on each socket (16 in total) 1× DDR5-4800 CXL memory with PCIe 5.0 ×8
NIC	ConnectX-6, 100 Gbps (MCX653106A-ECAT)

- Prototype and Baselines
  - Sundial-Pasha - Pasha prototype based on Sundial
  - Sundial-NET - Sundial using network as a transport
  - Sundial-CXL - Sundial using CXL as a transport
  - Sundial-SHM - Sundial adopting a shared-memory architecture
- TPC-C with different percentages of distributed transactions



- YCSB (RW = 1:1, uniform distribution) with different percentages of distributed transactions