Query Fresh: Log Shipping on Steroids

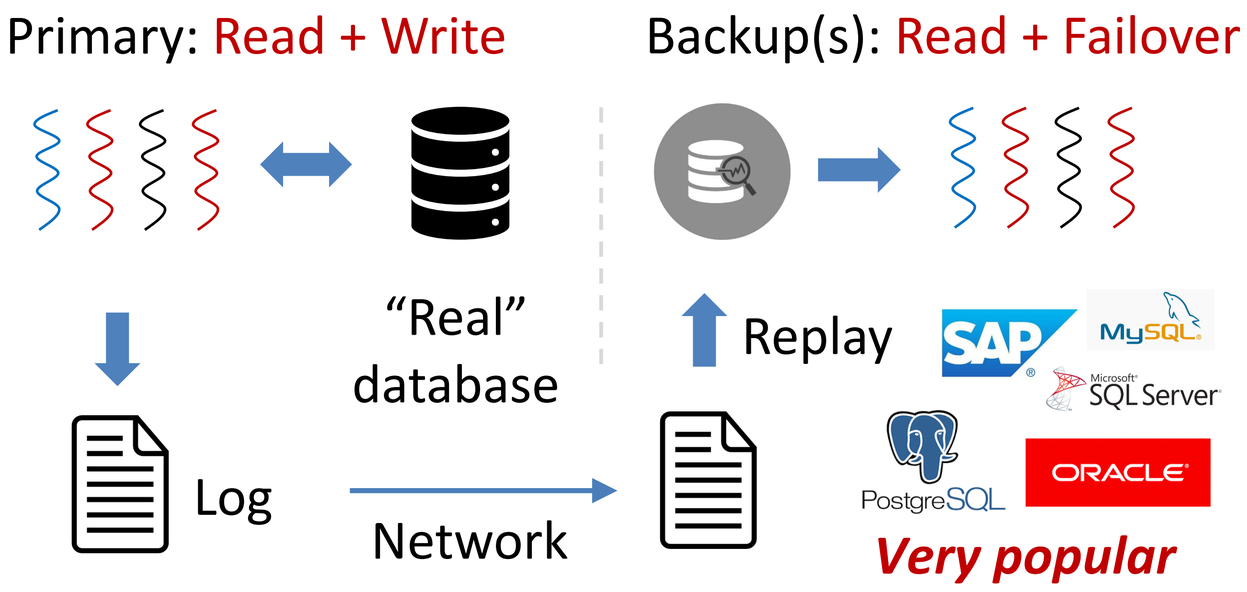
Tianzheng Wang* *Simon Fraser University*
Ryan Johnson, Ippokratis Pandis *Amazon Web Services*

Part of ERMIA: `https://github.com/ermia-db/ermia`

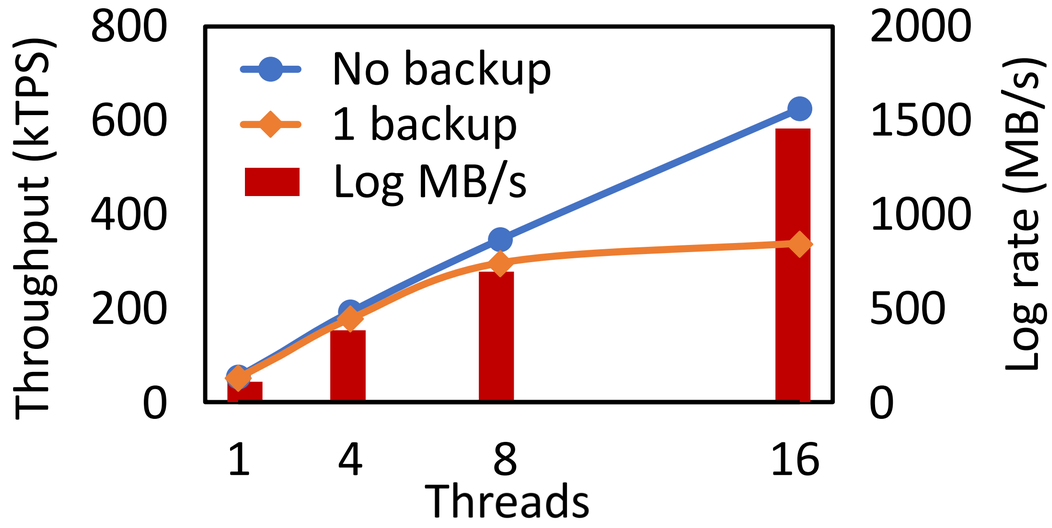
* Work performed while at University of Toronto

What? Hot standby solutions often give ***stale reads*** with ***no strong safety***
Why? Slow network, often-serial replay, and ***data redundancy*** (log + the "real" DB)
How? ***Append-only storage*** (fast replay) + RDMA over NVRAM (fast log shipping)

High availability by log shipping

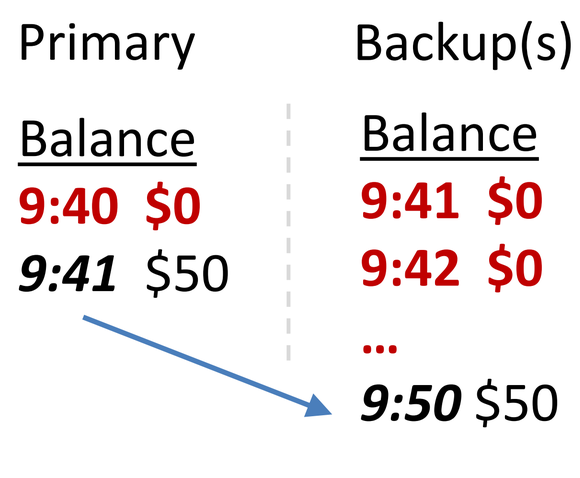


Infeasible: synchronous log shipping



Reality: asynchronous log shipping

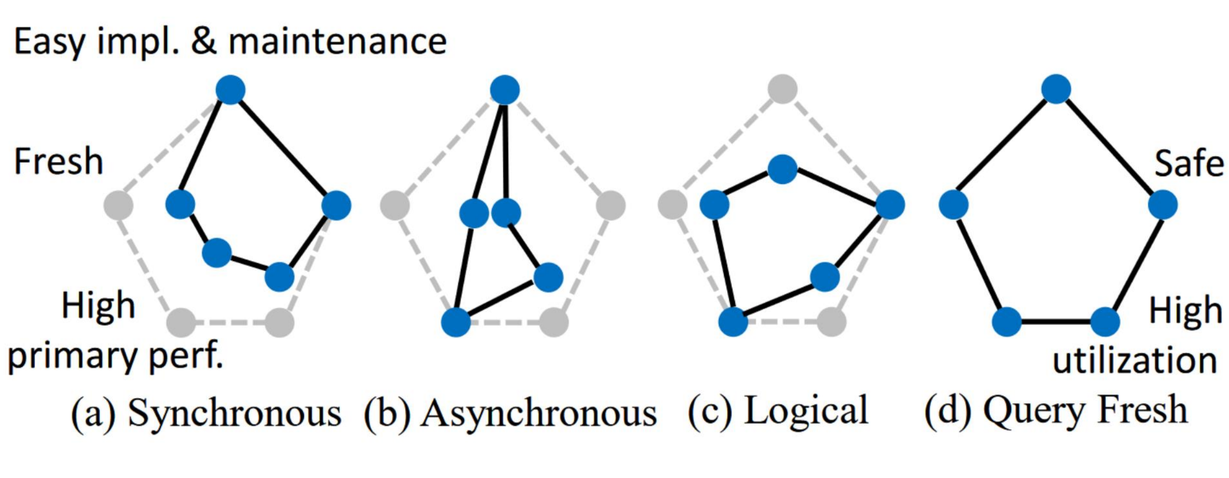
The freshness gap



Key reasons:

- Asynchronous log shipping due to slow network
- Heavyweight (serial) replay due to dual-copy architecture

Existing approaches vs. Query Fresh



Query Fresh = Append-only storage + Fast RDMA over NVRAM

Leveraging modern hardware

❶ **RDMA over fast network (e.g., InfiniBand)**
- Network no longer the slowest part
- Enables synchronous log shipping

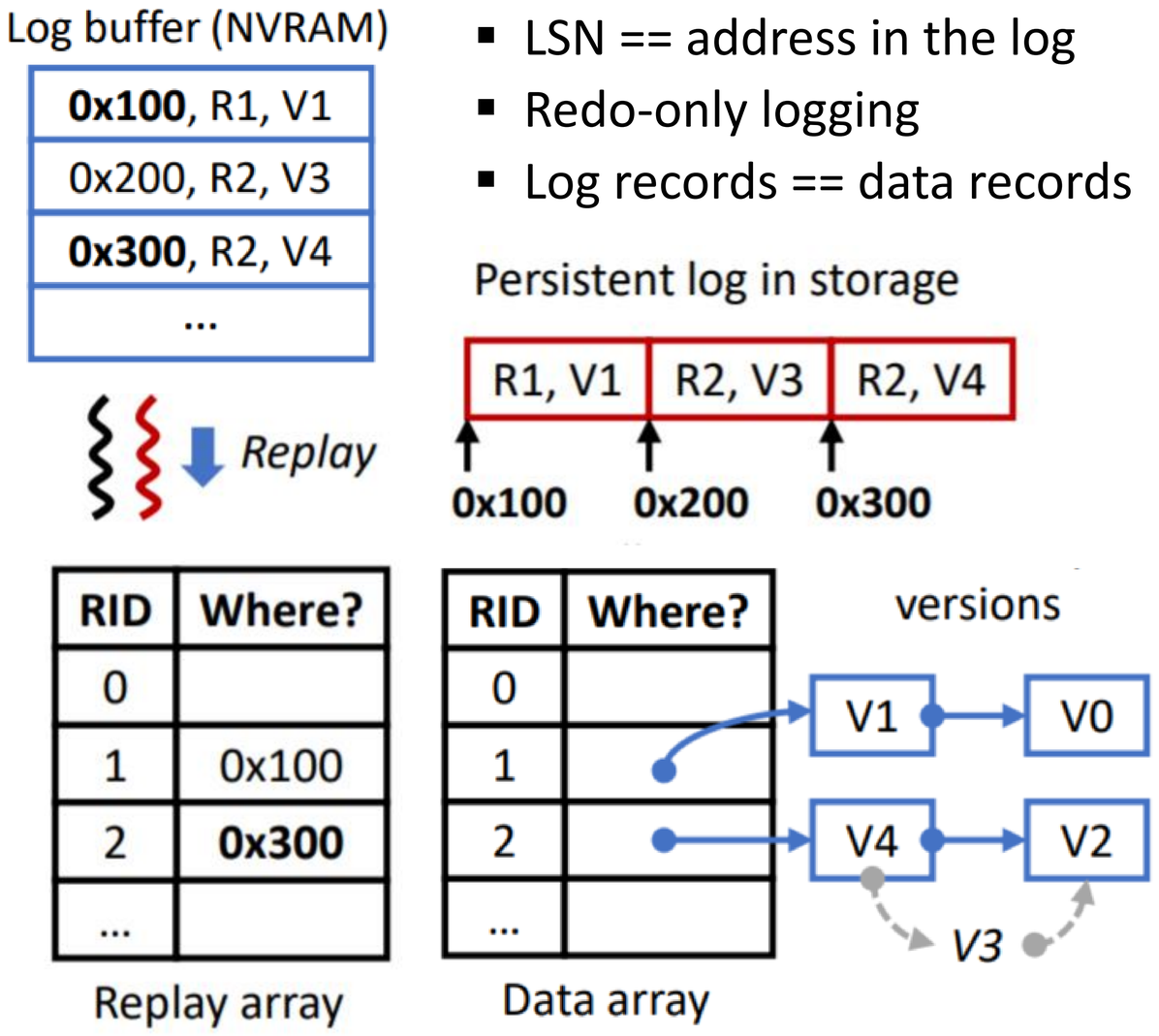
❷ **Log buffer in NVRAM (NV-DIMMs or 3D XPoint)**
- RDMA over persistent log buffers
- Fast persistence – no storage I/O on critical path

Single-copy + quick replay = fresh reads

❶ **Append-only storage: Log == Database**
- Index: key ➔ permanent record ID (RID)
- Indirection array: RID ➔ record address

❷ **Fast replay: simply set up indirection arrays**
- No record creation, no index ops (except inserts)
- Parallel and reuse existing recovery machinery

Append-only storage for log shipping

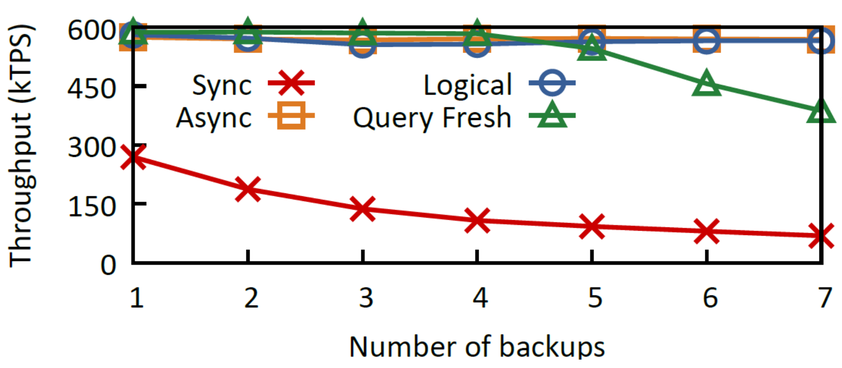


- LSN == address in the log
- Redo-only logging
- Log records == data records

Safe, fast primary and fresh backups

8 x 16-core (2-socket) Intel E5-2650, 64GB RAM
56Gbps InfiniBand RDMA and 10Gbps Ethernet TCP

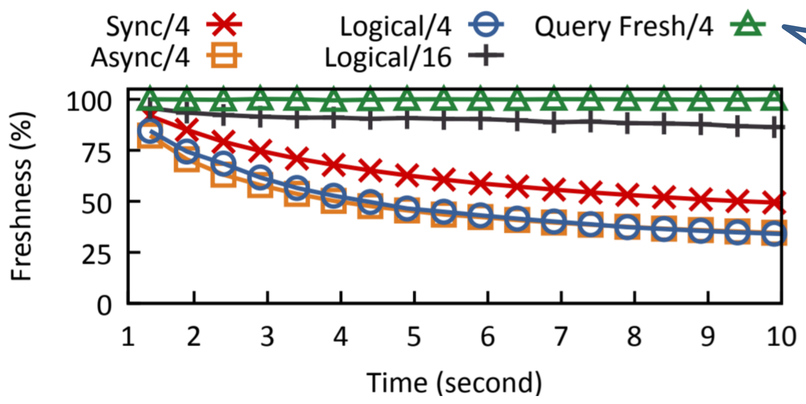
Primary: full TPC-C



Network saturated

56Gbps BW / 11.37Gbps logs ~= 4.9 backups

Backups: Read-only StockLevel and OrderStatus



4 replay threads only

12 query threads / 16 total threads = 75% utilization