# NVMe-oC Powered Memory Tiering: Unlocking Cost-Effective Data Management in DC

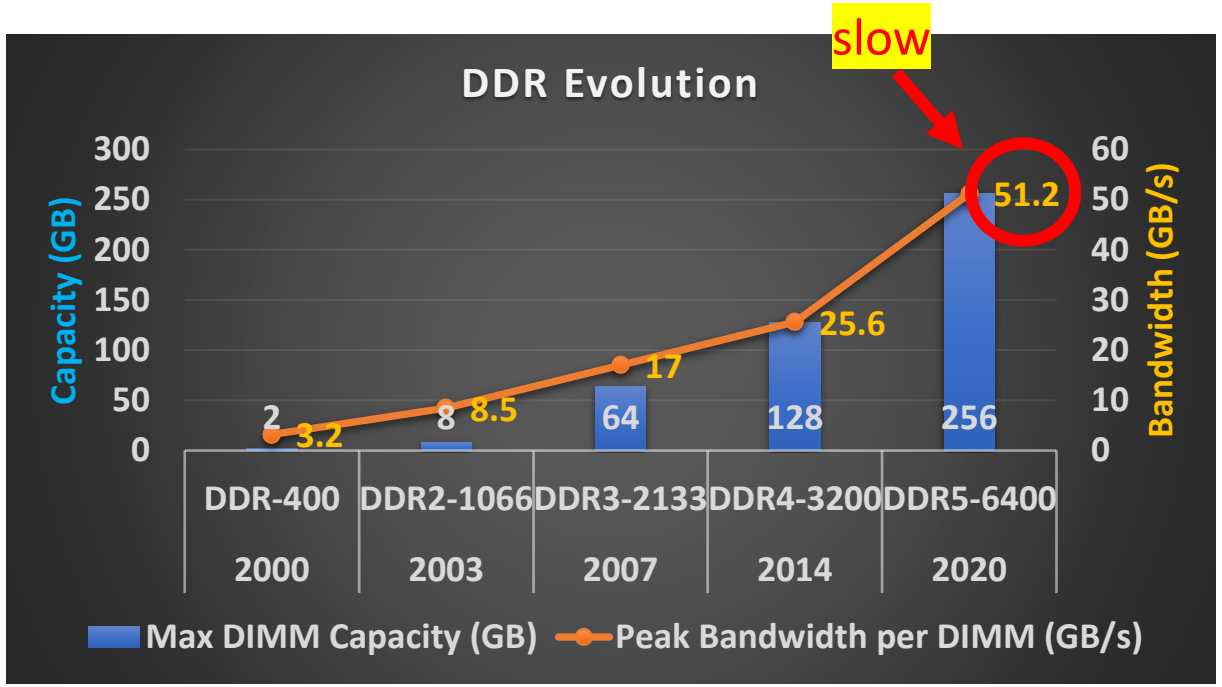
San Chang, Tai-Chun Kuo, Bearman Tsao, Po-Ting Yeh

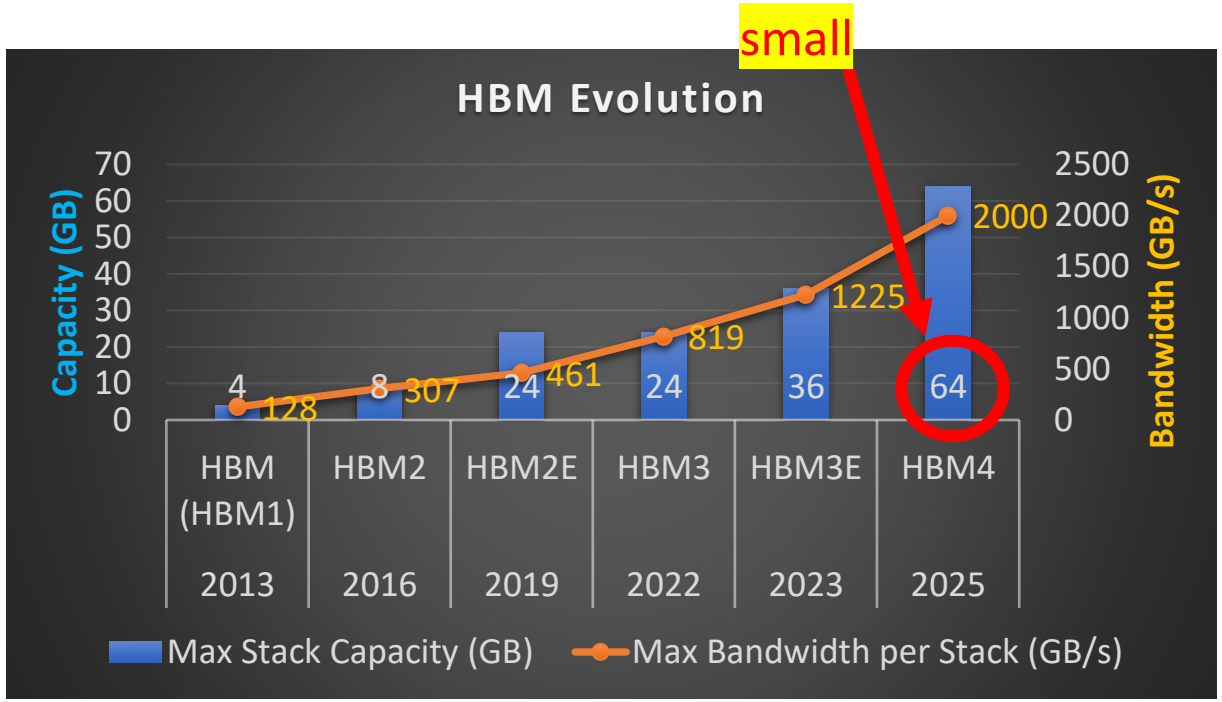
san@wolleytech.com

Wolley Inc.

# Memory Is Not Scaling Fast Enough

- DRAM bandwidth grows slowly; HBM capacity gains are incremental
- Emerging workloads such as AI demand memory systems far beyond today's limits



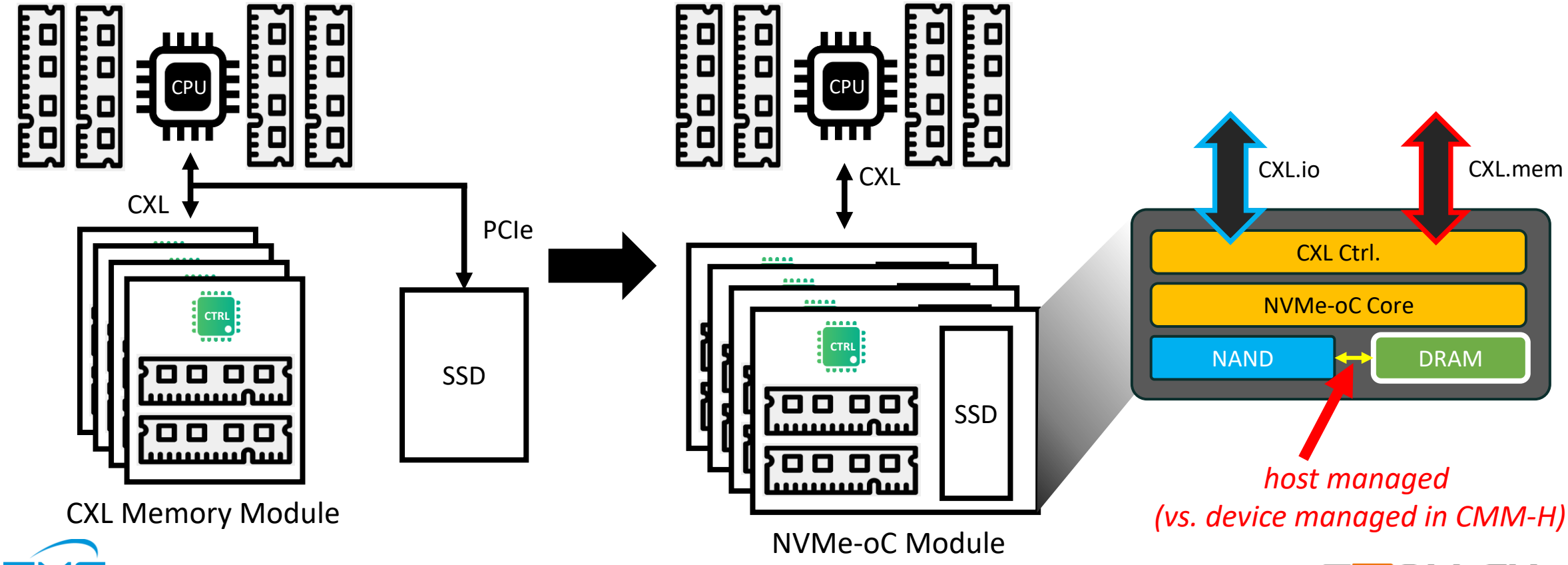

FMS the Future of Memory and Storage



VOLLEY

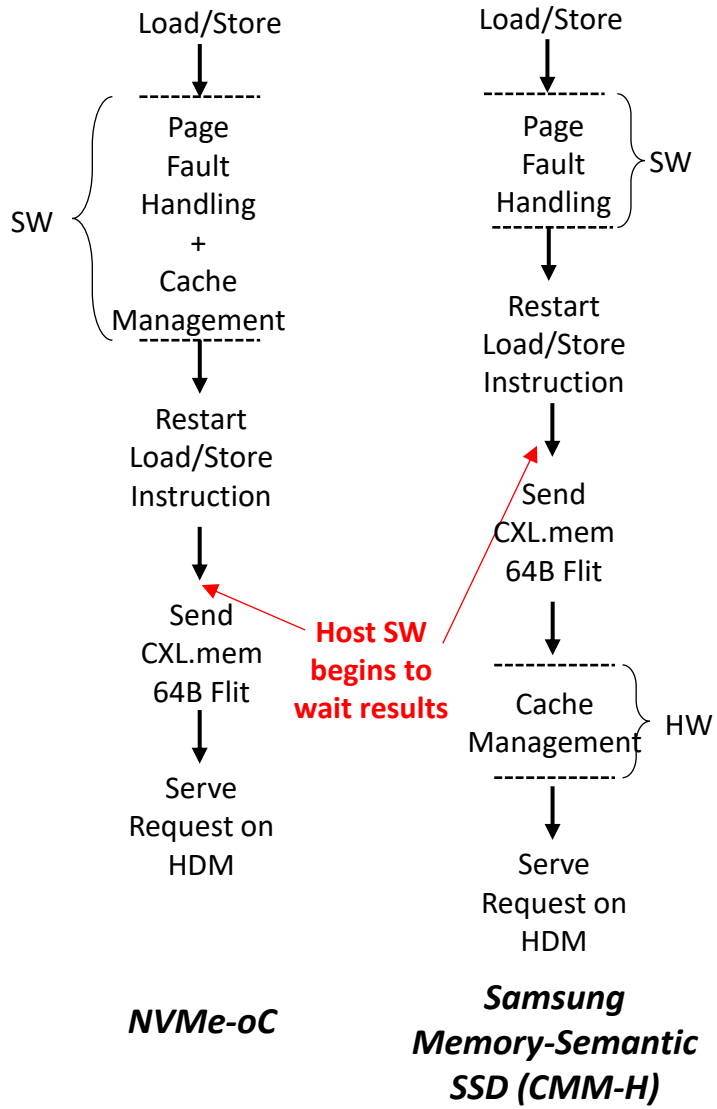
# NVMe-over-CXL: Unlocking Both Bandwidth and Capacity

*"NVMe-oC, with DAX-tiering, unifies DRAM, CXL, and SSD into a seamless and scalable memory architecture"*



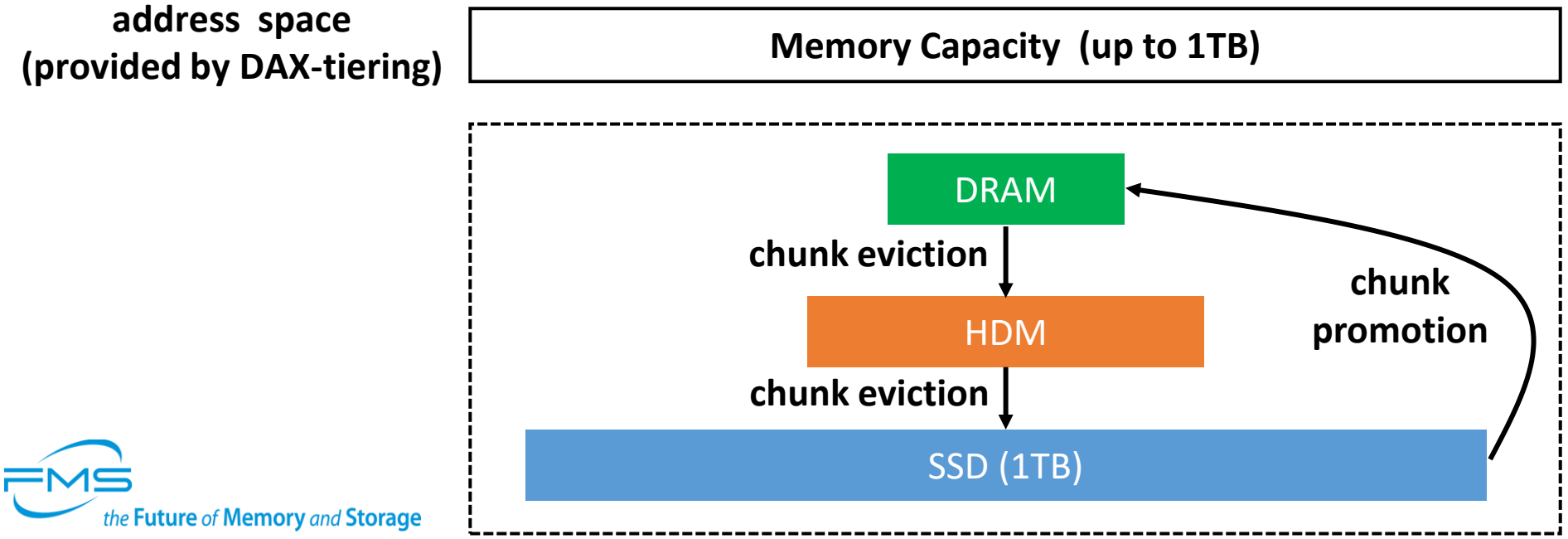
# DAX-Tiering : DRAM + HDM + SSD with Auto-Tiering

- DAX (2024): Combines SSD and HDM, enabling mmap access and internal HDM-to-SSD data movement
- DAX-Tiering (2025): Adds DRAM as the top layer, dynamically promoting/demoting data across DRAM → HDM → SSD for higher capacity and better performance balance



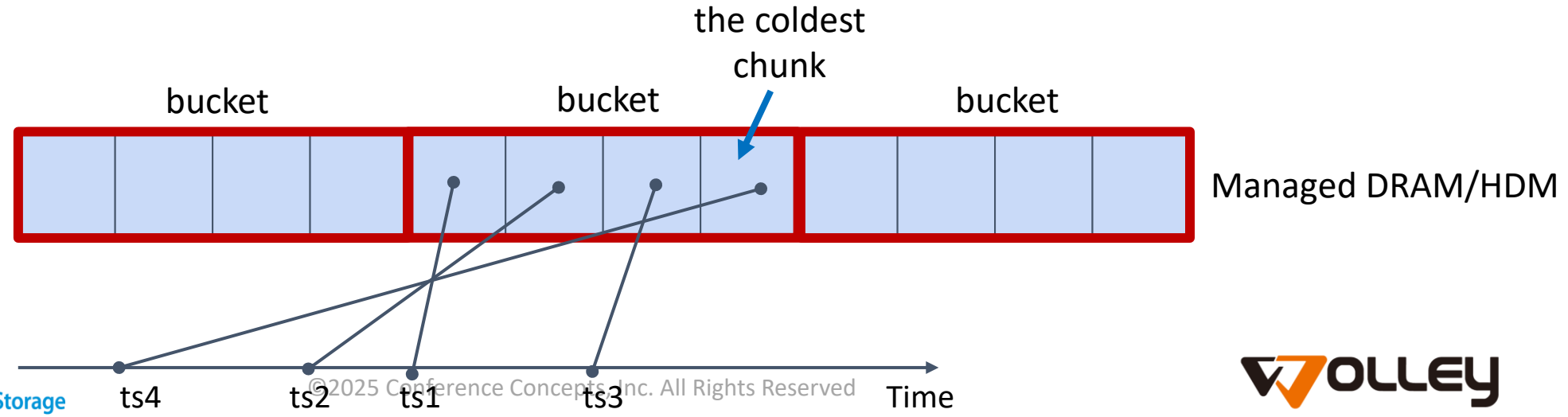
# Exclusive DRAM-HDM Cache with a Write-back SSD Backing Store

- The driver exposes a single up-to-1 TB virtual memory (limited by SSD)
  - DRAM <-> HDM: exclusive cache pair (no duplicates)
  - HDM <-> SSD: write-back strategy, not an inclusive cache



# TRACE: Time-based Recency Access Classifier for Eviction

- Detects data temperature using PTE access bits over time
  - Classifies hot and cold pages without application changes
  - Background kernel thread periodically scans and updates metadata
- Evicts cold pages across memory tiers with precision
  - “DRAM to HDM” and “HDM to SSD”
  - Targets the coldest chunks based on temporal recency
- Achieves high efficiency with minimal overhead
  - Only ~1% memory for metadata & minor runtime disruption



# Experiment Setup

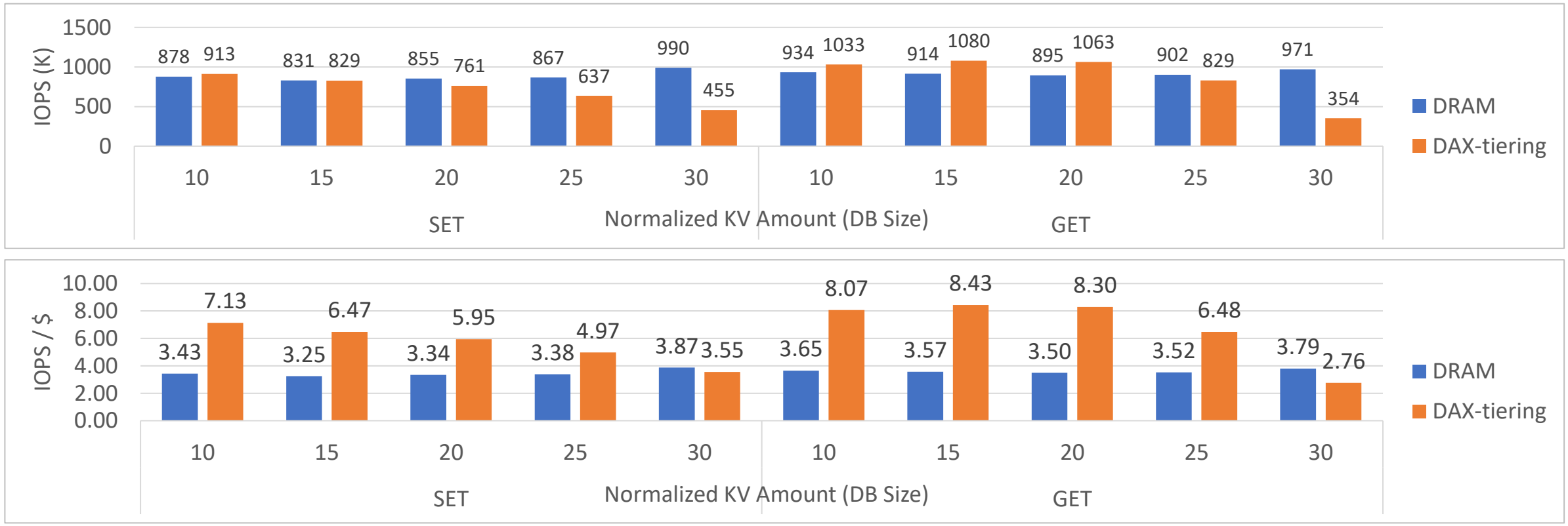
| Item | Description |
|---|---|
| CPU | Intel Xeon (Model 173), 144 cores (72 cores per socket × 2 sockets) |
| Memory | 2 DDR5 RDIMM, 6400 MT/s, 32GB each (total bandwidth: 100GB/s) |
| NVMe-oC | PCIe Gen3 x8 (8GB/s), 16GB HDM + 64GB SSD |
| OS | Fedora Linux 40 (Workstation Edition) |
| kernel | 6.8.5-301.fc40.x86_64 |

***Memory Configurations***

| | | |
|---|---|---|
| DRAM | 64GB DRAM | ***Capacity: 64GB*** |
| DAX-tiering | 16GB DRAM + 16GB HDM + 64GB SSD | ***Capacity: 64GB*** |

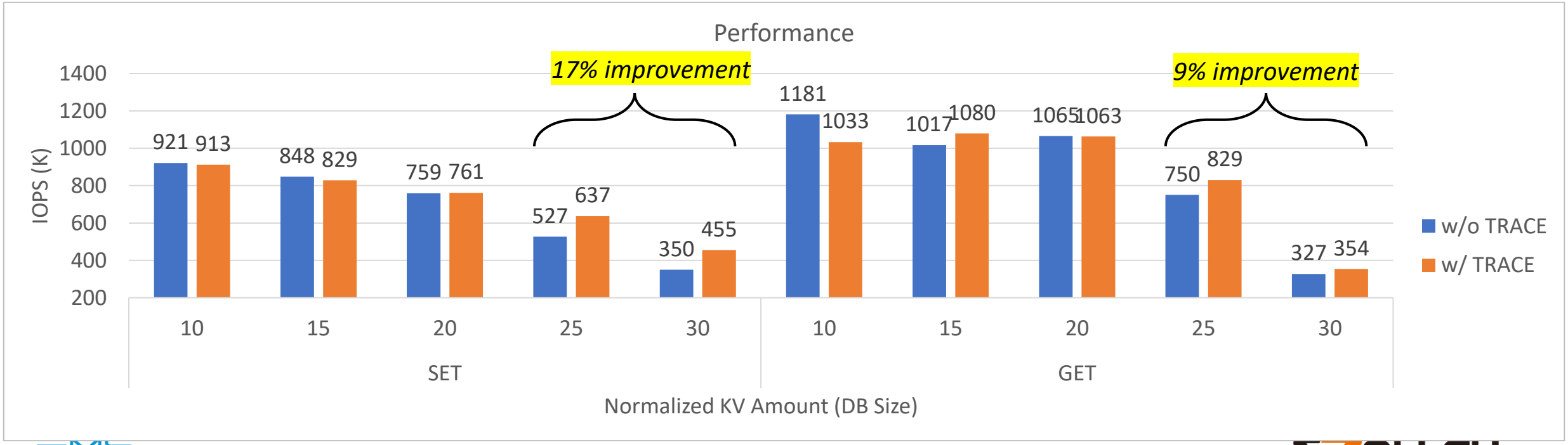
# Redis Performance & Performance-per-dollar (16 server instances, 64GB in total)

- Delivers near-DRAM performance at 50% lower memory cost
- Boosts performance-per-dollar by up to 2.3×
- Sustains performance as workload exceeds DRAM+HDM capacity



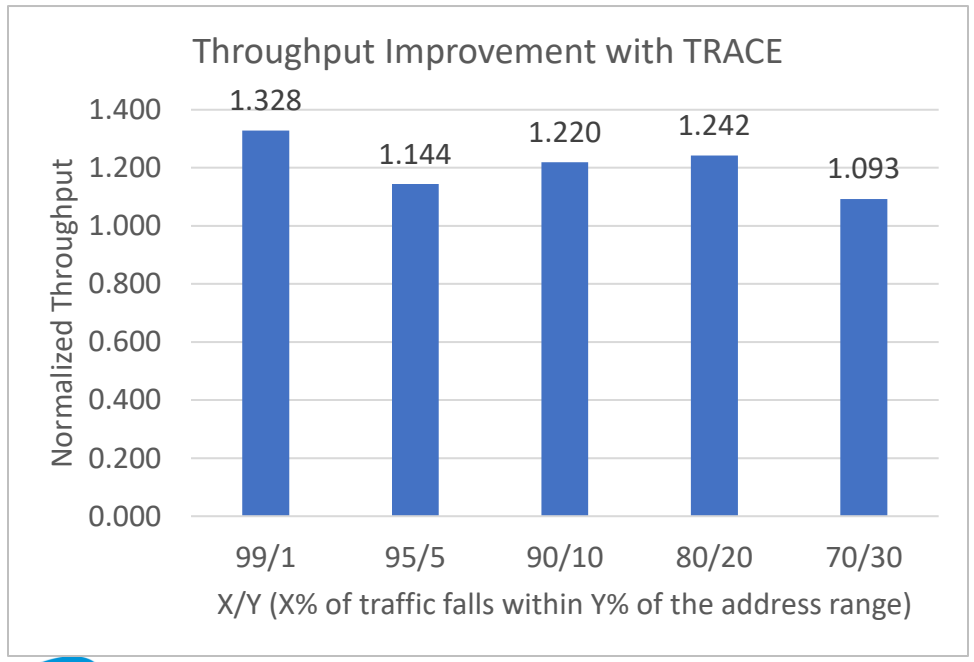
# Redis Performance w/ and w/o TRACE

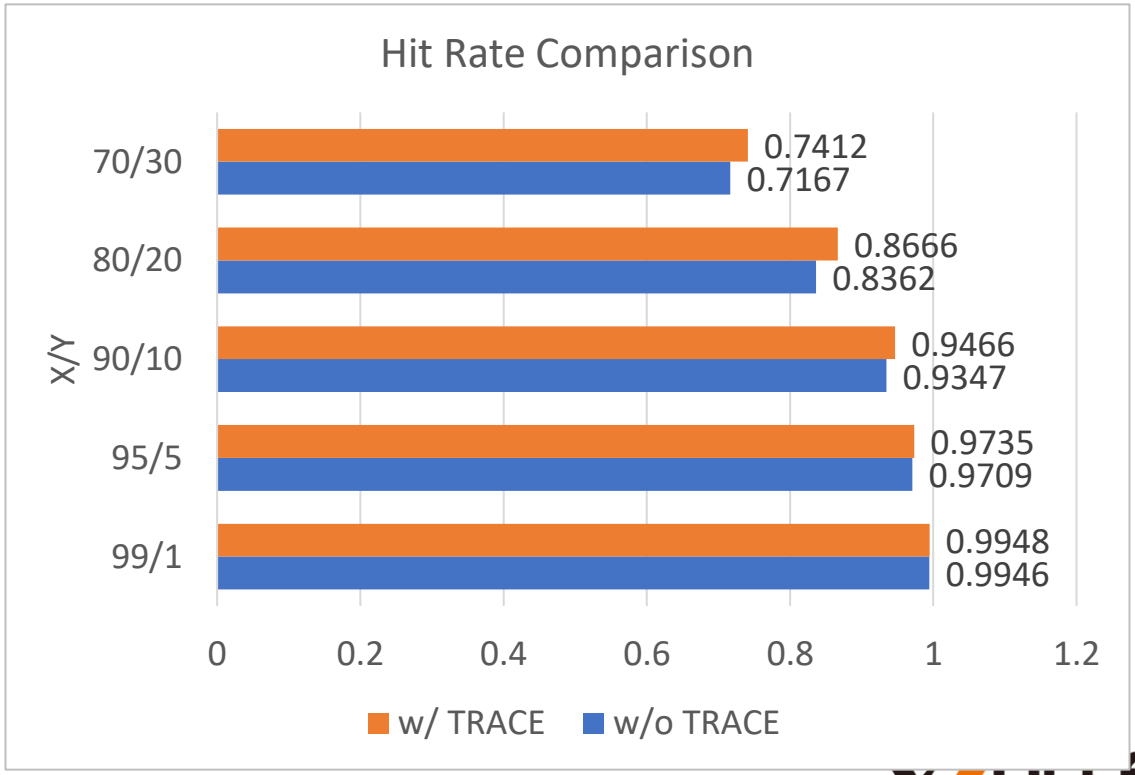
- Slight overhead at small DB sizes (≤15), with minor IOPS drop due to TRACE management
- As DB working set spans DRAM, HDM, and SSD, TRACE unlocks multi-tier cache benefits, boosting SET by ~17% and GET by ~9%



# TRACE Performance using Synthetic Workload

- NVMe-oC with TRACE enabled improves the throughput by up to 1.33× and averages ~1.2× across workloads
- TRACE consistently improves cache hit rate across all locality distributions



# NVMe-oC: One Architecture, Many Configurations

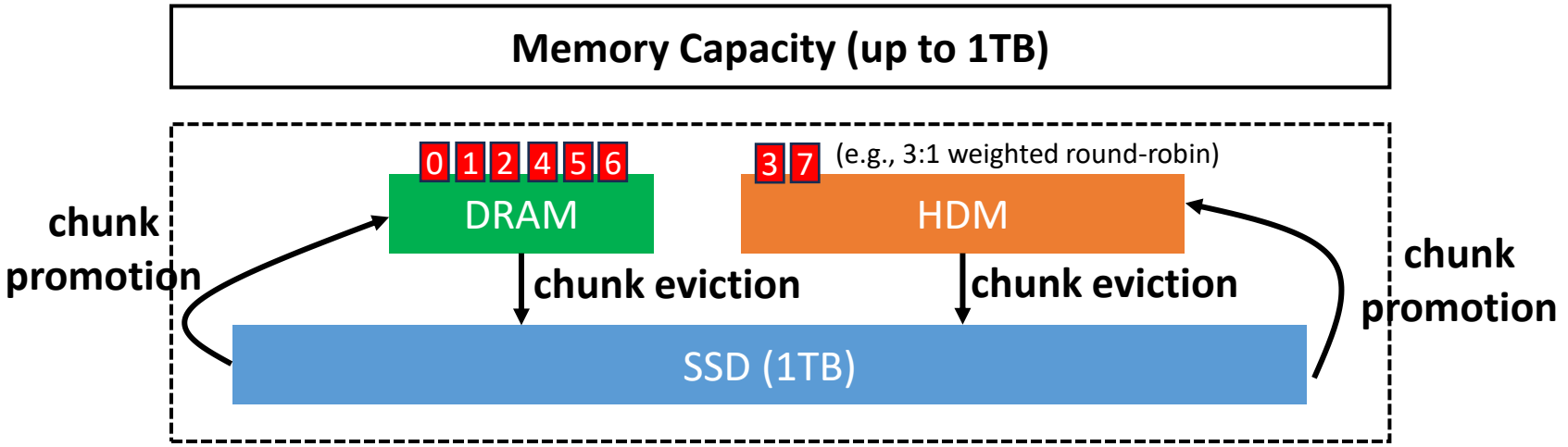
- Host-managed caching enables configurable DRAM:SSD ratios
  - DRAM includes host DRAM and device DRAM (HDM)
- Flexible ratios optimize for workload needs
  - 1:2 for latency-sensitive applications such as inference
  - 1:4 for read-heavy applications like Redis
  - 1:8 for high-locality applications such as cache systems

| Memory Type / Tier | Estimated $ / GB | Typical Latency | Ideal Workload |
|---|---|---|---|
| DDR5/LPDDR5 | $4 ~ $5 | 100ns | ultra-hot data |
| SSD | $0.02 ~ $0.05 | 60us | cold / backup data |
| NVMe-oC (1:2) | $2.0 ~ $2.6 | Cache hit ≈ 250 ns<br>Cache miss ≈ 60 µs | Latency-sensitive apps |
| NVMe-oC (1:4) | $1.0 ~ $1.3 | Cache hit ≈ 250 ns<br>Cache miss ≈ 60 µs | read-heavy DBs (e.g., Redis) |
| NVMe-oC (1:8) | $0.52 ~ $0.68 | Cache hit ≈ 250 ns<br>Cache miss ≈ 60 µs | high-locality caches |

- https://www.dramexchange.com/
- https://memory.net/memory-prices/

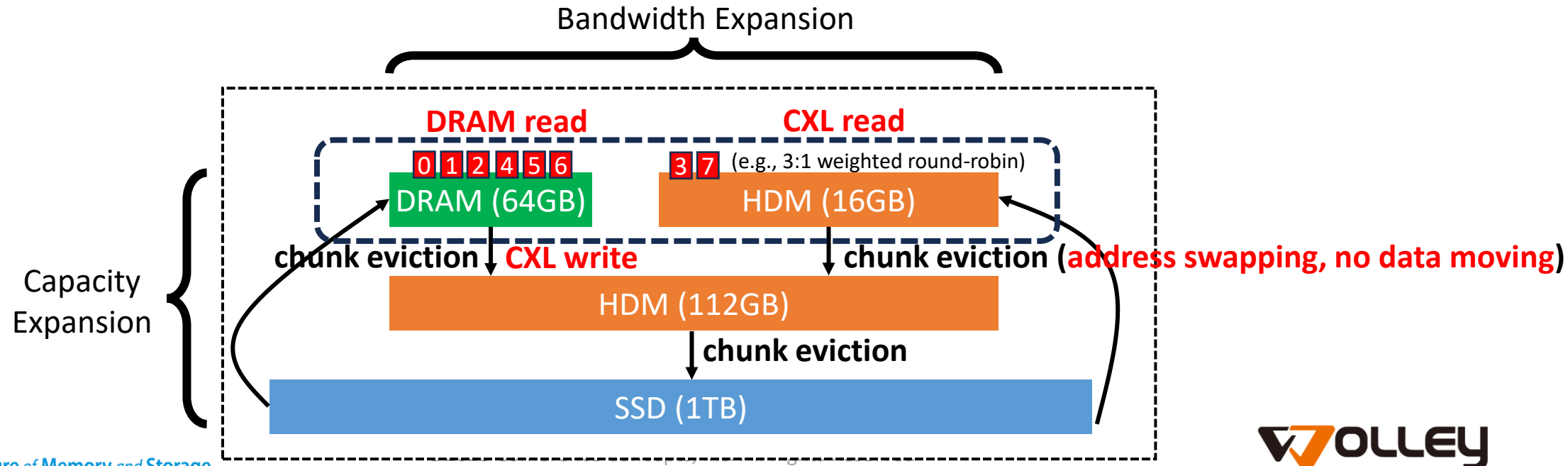
# Host-defined Memory Architecture (1) – Interleaved DRAM-HDM Composite Cache

- Chunks are placed in a weighted round-robin pattern across
  - DRAM (low latency, high BW)
  - HDM (higher latency, moderate BW).
- Interleaving DRAM and HDM at allocation time delivers additive bandwidth and cheaper capacity, but at the cost of higher latency variance



# Host-defined Memory Architecture (2) – Multi-Tier Composite Cache for Balanced Performance

- Simultaneously expands both memory bandwidth and capacity
  - *Fully utilizes CXL full-duplex link*: foreground reads target upper-tier HDM while background writes stream into lower-tier HDM
  - *Zero-copy HDM demotion*: address remapping between HDM tiers without physical data movement



# Takeaway

- Cost-Effective Memory Scaling with NVMe-oC
  - Combines DRAM, HDM, and SSD into a unified memory space
  - Delivers near-DRAM performance at 50% lower memory cost
- Intelligent Hot/Cold Tiering with TRACE
  - Hot data stays in DRAM, cold data flows to HDM and SSD
  - Enables real-time hot/cold detection with only ~1% metadata overhead
- Flexible Host-defined Caching for Any Workload
  - Configurable DRAM:SSD ratios (1:2, 1:4, 1:8) per workload need
  - Leverages CXL full-duplex bandwidth and zero-copy HDM demotion
  - Adapts to bandwidth-sensitive or latency-sensitive applications across tiers

San Chang, san@wolleytech.com