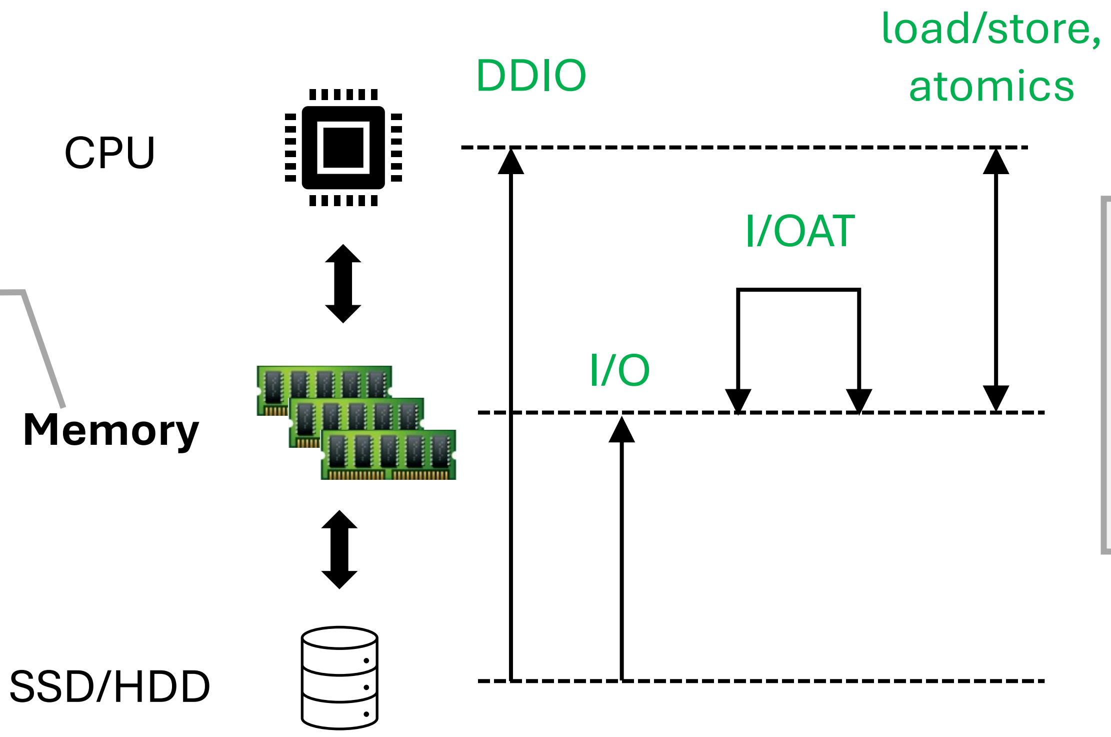
Declarative Memory Services

Jeronimo Castrillon
Jana Giceva
Yu Hua
Kimberly Keeton
Akhil Shekar
Kevin Skadron
Tianzheng Wang
Huanchen Zhang

“Memory” traditionally…

Properties:

- Single-node
- Byte-addressable
- Volatile DRAM
- Coherent
- ~100ns low latency
- High bandwidth
- Passive



Issues:

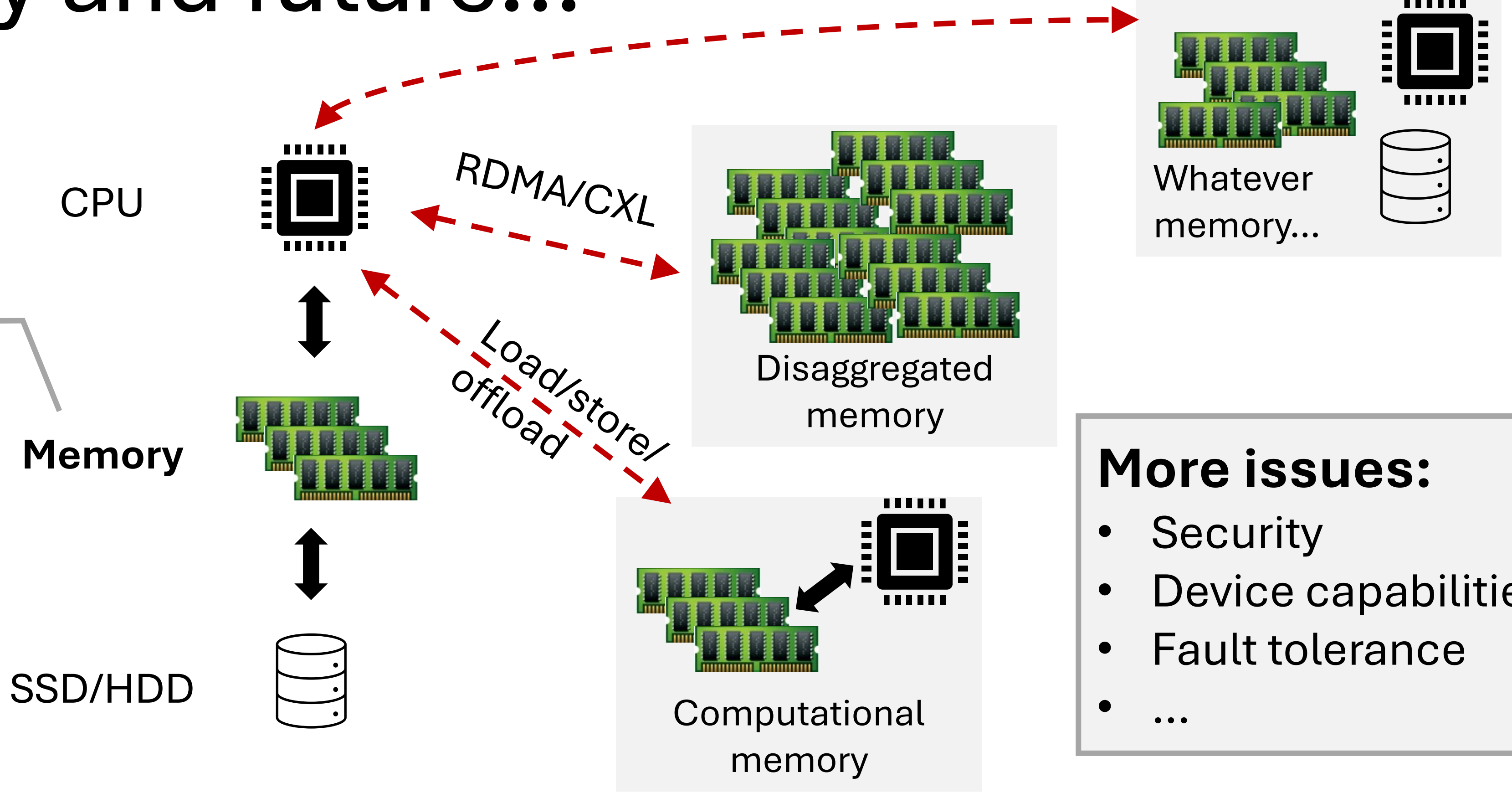
- NUMA-awareness
- Allocator performance
- Cache-conscious
- …

Relative tractable primitives and tools + imperative programming

Life was ok.

“Memory” today and future…

Uncertainties:
- Coherent?
- Volatile?
- Passive or active?
- Various latency and bandwidth profiles



More issues:
- Security
- Device capabilities
- Fault tolerance
- …

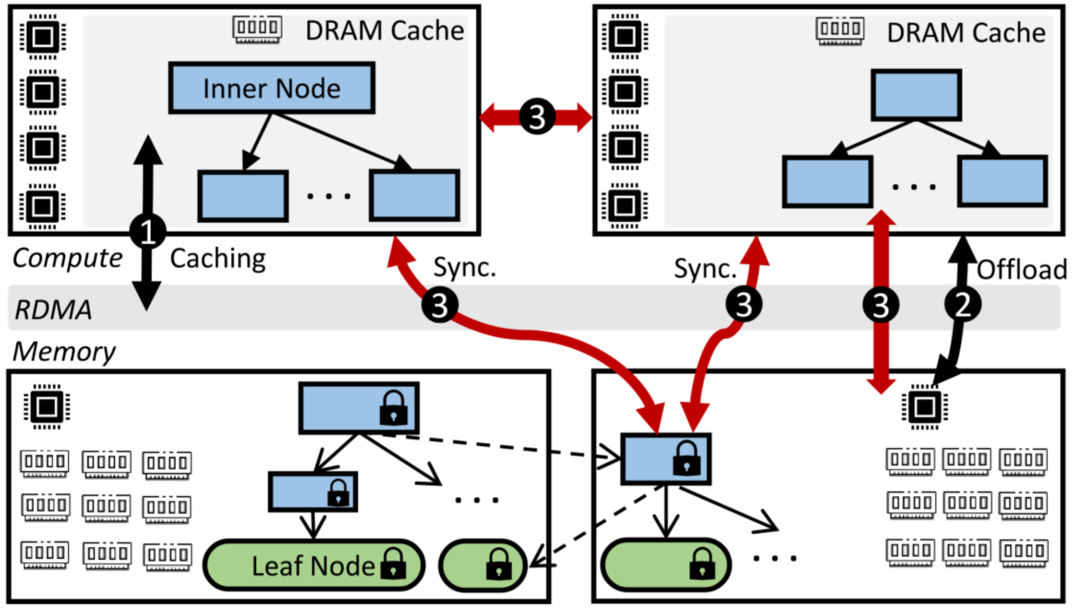
Intractable primitives → highly complex, imperative programming

Life is hard.

Case Study:

Adapting a B+-Tree for disaggregated memory

(1) Longer latency, should cache:
- Which B+-tree nodes to cache?
- Is there coherence between compute servers?



(2) Memory has CPU, should offload:
- How much CPU do I have?
- What operations to offload?

(3) Data placement + replication:
- Who can access which data?
- How to partition?

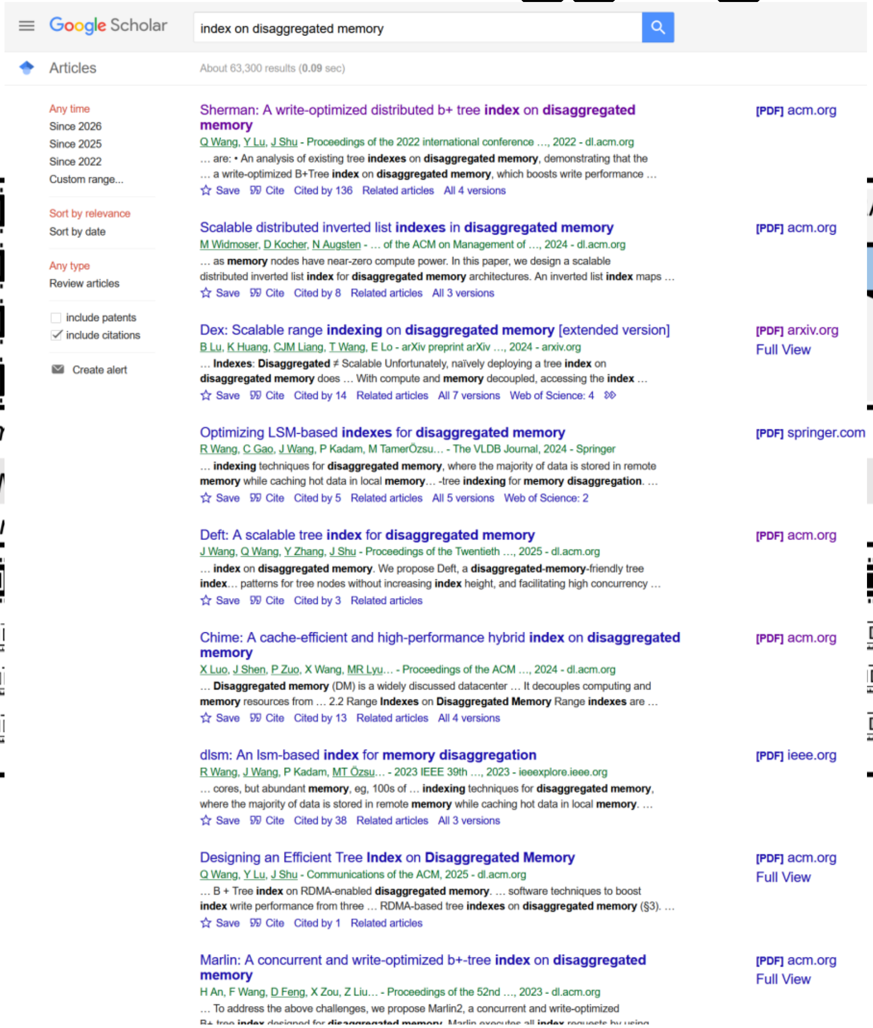
* DEX: Scalable Range Indexing on Disaggregated Memory, VLDB 2024

Case Study:

Adapting a B+-Tree for disaggregated memory

(1) Longer latency, should cache:

- Which B+-tree nodes to cache?
- Is there coherence between compute servers?



Google Scholar

index on disaggregated memory

Articles

About 63,300 results (**0.09** sec)

Any time
Since 2026
Since 2025
Since 2022
Custom range...

Sort by relevance
Sort by date

Any type
Review articles

include patents
✓ include citations

Create alert

Sherman: A write-optimized distributed b+ tree **index** on **disaggregated memory** — [PDF] acm.org
Q Wang, Y Lu, J Shu - Proceedings of the 2022 international conference ..., 2022 - dl.acm.org
... are: • An analysis of existing tree **indexes** on **disaggregated memory**, demonstrating that the ... a write-optimized B+Tree **index** on **disaggregated memory**, which boosts write performance ...
☆ Save 99 Cite Cited by 136 Related articles All 4 versions

Scalable distributed inverted list **indexes** in **disaggregated memory** — [PDF] acm.org
M Widmoser, D Kocher, N Augsten - ... of the ACM on Management of ..., 2024 - dl.acm.org
... as **memory** nodes have near-zero compute power. In this paper, we design a scalable distributed inverted list **index** for **disaggregated memory** architectures. An inverted list **index** maps ...
☆ Save 99 Cite Cited by 8 Related articles All 3 versions

Dex: Scalable range **indexing** on **disaggregated memory** [extended version] — [PDF] arxiv.org Full View
B Lu, K Huang, CJM Liang, T Wang, E Lo - arXiv preprint arXiv ..., 2024 - arxiv.org
... **Indexes**: **Disaggregated** ≠ Scalable Unfortunately, naïvely deploying a tree **index** on **disaggregated memory** does ... With compute and **memory** decoupled, accessing the **index** ...
☆ Save 99 Cite Cited by 14 Related articles All 7 versions Web of Science: 4 ≫

Optimizing LSM-based **indexes** for **disaggregated memory** — [PDF] springer.com
R Wang, C Gao, J Wang, P Kadam, M TamerÖzsu... - The VLDB Journal, 2024 - Springer
... **indexing** techniques for **disaggregated memory**, where the majority of data is stored in remote **memory** while caching hot data in local **memory**... -tree **indexing** for **memory disaggregation**. ...
☆ Save 99 Cite Cited by 5 Related articles All 5 versions Web of Science: 2

Deft: A scalable tree **index** for **disaggregated memory** — [PDF] acm.org
J Wang, Q Wang, Y Zhang, J Shu - Proceedings of the Twentieth ..., 2025 - dl.acm.org
... **index** on **disaggregated memory**. We propose Deft, a **disaggregated-memory**-friendly tree **index**... patterns for tree nodes without increasing **index** height, and facilitating high concurrency ...
☆ Save 99 Cite Cited by 3 Related articles

Chime: A cache-efficient and high-performance hybrid **index** on **disaggregated memory** — [PDF] acm.org
X Luo, J Shen, P Zuo, X Wang, MR Lyu... - Proceedings of the ACM ..., 2024 - dl.acm.org
... **Disaggregated memory** (DM) is a widely discussed datacenter ... It decouples computing and **memory** resources from ... 2.2 Range **Indexes** on **Disaggregated Memory** Range **indexes** are ...
☆ Save 99 Cite Cited by 13 Related articles All 4 versions

dlsm: An lsm-based **index** for **memory disaggregation** — [PDF] ieee.org
R Wang, J Wang, P Kadam, MT Özsu... - 2023 IEEE 39th ..., 2023 - ieeexplore.ieee.org
... cores, but abundant **memory**, eg, 100s of ... **indexing** techniques for **disaggregated memory**, where the majority of data is stored in remote **memory** while caching hot data in local **memory**. ...
☆ Save 99 Cite Cited by 38 Related articles All 3 versions

Designing an Efficient Tree **Index** on **Disaggregated Memory** — [PDF] acm.org Full View
Q Wang, Y Lu, J Shu - Communications of the ACM, 2025 - dl.acm.org
... B + Tree **index** on RDMA-enabled **disaggregated memory**. ... software techniques to boost **index** write performance from three ... RDMA-based tree **indexes** on **disaggregated memory** (§3). ...
☆ Save 99 Cite Cited by 1 Related articles

Marlin: A concurrent and write-optimized b+-tree **index** on **disaggregated memory** — [PDF] acm.org Full View
H An, F Wang, D Feng, X Zou, Z Liu... - Proceedings of the 52nd ..., 2023 - dl.acm.org
... To address the above challenges, we propose Marlin2, a concurrent and write-optimized B+ tree **index** designed for **disaggregated memory**. Marlin executes all **index** requests by using

(2) Memory has CPU, should offload:

- How much CPU do I have?
- What operations to offload?

(3) Data placement + replication:

- Who can access which data?
- How to partition?

Hand-coded decisions
Unsustainable (*more cases in paper*).

Would be nice to be more *declarative*

- Decouple device-specific logic from high-level design
  - "I want this function to be offloaded, if possible"
  - "Latency to access this memory block should not exceed 5ms"
- Simplify programming for today and future, unknown architectures
  - Same DBMS design, any hardware
- Better cross-device optimizations

How to get there?

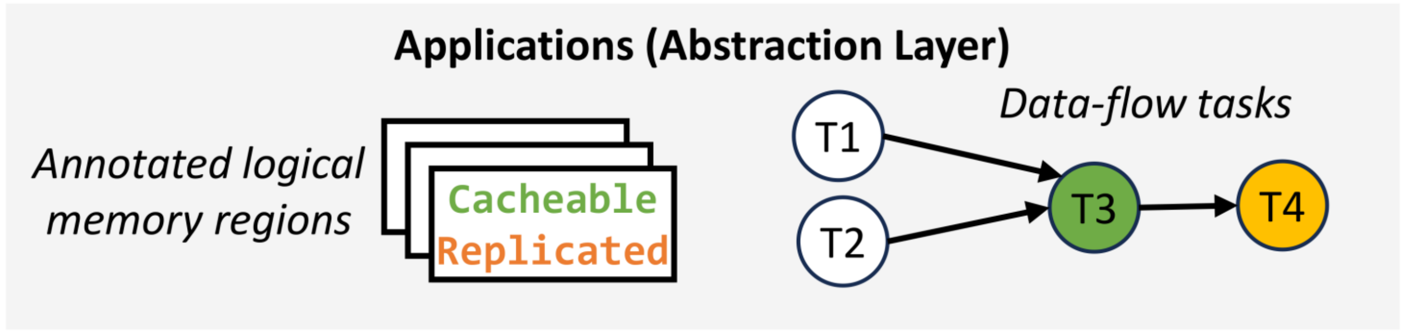
Vision: Declarative Memory Services

Three-layer design:

- Abstraction Layer
  - Developers work with “logical memory regions” and data flows
  - Annotate with desired properties
- Calibration Layer
  - Discover and index device capabilities
  - Expose device primitives and APIs
- Memory Services Layer
  - A set of generic “memory services” that well use memory devices
  - Jointly optimize for the application based on annotations

Caveat: yet to implement, this is pure vision!

Declarative Abstraction layer



Applications (Abstraction Layer)
Annotated logical memory regions
Cacheable
Replicated
Data-flow tasks
T1
T2
T3
T4

B+-tree node definition:

```
struct InternalNode {
  KV kv_pairs[MAX_KV];
  int key_count;
  …
};
```

Declare desired properties

Previously:

```
InternalNode *n = allocate(…)

// hand-made decision to cache it
cache.insert(n);
```

Now with DMS:

```
[cacheable, coherent, latency < 10µs]
InternalNode *n = allocate(…)

// placed in coherent, compute-side memory, by DMS
```
~~cache.insert(n);~~

Data flows work similarly:

- Properties attached to tasks, enforced by DMS runtime

Physical design and logical functionality decoupled

Calibration Layer

- Discovers and track device capabilities, provide APIs
- Key component: device catalogue
  - A table that evolves with hardware changs

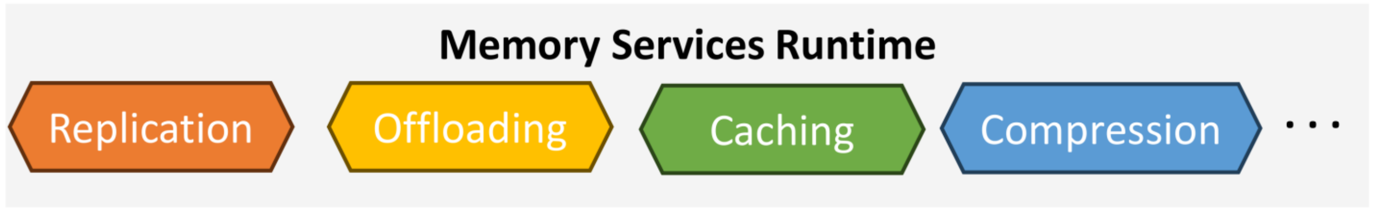
Device	Capabilities	APIs	Characteristics
Local DRAM	Coherence Byte-addressable	dram-load, dram-store, dram-dsa, atomics…	… x GBps within socket, under y load…
CXL DRAM	Partial coherence Byte-addressable	cxl-load, cxl-store…	… 300ns best - 1us worst latency…
Membrane (computational memory)	Compute Byte-addressable	pim-load, pim-store, pim-offload…	… x ns latency with host…

Implemented and maintained by DMS developers

Challenging

Memory Services Layer

- Use device catalogue APIs to build services



- DEX example:
  - Services needed: data placement and caching
  - Upon allocation: place data based on annotated desired properties
  - Runtime: lightweight metadata tracking for caching
- Customized policies possible
  - “Please don’t evict parent node before child node”
  - “Please use this encoding scheme for such and such data”

Research Challenges and Agenda

- Device Characterization
  - Beyond simple stats: e.g., latency behaviour under varying load levels
  - Self-evolving the device catalogue with new hardware
- Properties → Services: When to pick which implementation?
- SLA Guarantees
  - Memory services monitor metrics, and migrate between services to meet SLO
  - How to deal with conflicting SLAs?
    - E.g., tenants prioritizing throughput vs. latency
- DMS Deployment
  - DMS requires non-trivial information (global and local server) to work
- Correctness and Debugging
  - DMS-based programs are declarative
  - How to verify their correctness and debug them? Tools for exploring why an SLO was missed?

Summary

- Memory is heterogeneous: complexity arises with more features
  - Current approach to leveraging memory devices is unsustainable
  - Hand-crafted with low-level primitives
  - Getting worse as hardware evolves

- **Declarative Memory Services**
  - Developers specify logical functionality
  - Calibration layer discovers and characterises devices
  - Memory services provide physical implementations and optimizations

Thank you!