THE DATA ACCESS CONTINUUM*

* A SEQUENCE WHERE ADJACENT ELEMENTS ARE NOT PERCEPTIBLY DIFFERENT FROM EACH OTHER, BUT THE EXTREMES ARE QUITE DISTINCT

Paolo Faraboschi
Fellow and VP, Hewlett Packard Labs
DATA ACCESS HIERARCHY CA. 2014 – TWO NEW ENTRIES

PERFORMANCE

1-10 ns
50ns
50-100ns
100-500ns
1-10us
ms

1-10 MBs
10-100GBs
1TBs
1-10TBs
10-100TBs

CAPACITY (and byte/$)

SRAM (caches)
Massive b/w
HBM DRAM
DDR DRAM
Universal Memory?
NVM
Storage class memory
SSDs
DISKs

10-100ns

DATA ACCESS HIERARCHY IN 2019 – GETTING WORSE!

- **SRAM (caches)**: Small capacity, Persistence, High speed
- **HBM DRAM**: < DRAM speed, Flash capacity
- **MRAM**: Flash capacity, DRAM speed
- **DDR DRAM**: 3DXP DIMM + DRAM DIMMS
- **3DXP NAND**: Low Latency NAND
- **NAND**: Latency optimized
- **Dimms**: Capacity optimized
- **QLC**: OLC NAND
- **NAND**: Combo SSDs
- **DISKs**: 1-10TBs, 10-100TBs

CAPACITY (and bytes/$)

PERFORMANCE

- 1-10 ns
- 50ns
- 50-100ns
- 100-500ns
- 1-10us
- ms

**MBs**: 10-100GBs, 1TBs

**1-10TBs**: 500 ns?
DATA ACCESS INTERCONNECTS

30-100ns
200-500 GB/s
Memory semantics (ld/st)
Invisible to software, CC
Point-to-point
Dedicated resources

Inside the server

Cluster

1

Capacity Memory/NVM

Compute

High Bandwidth Memory

Switch

2

Capacity Memory/NVM

Compute

High Bandwidth Memory

Switch

3

100-300ns
100-400 GB/s
Invisible to software, not CC
Exposed to HV / container
Memory semantics (ld/st)
Switched (low latency)
Shared resources

300ns-1,000ns
25-100 GB/s
Software-exposed (libfabrics)
Messaging semantics
(send/receive or put/get)
Switched (high throughput)
TRADITIONAL DATA STRUCTURES ARE ILL-SUITED

- How to design data structures and algorithms for the “data continuum”?
- Memory tiers provide ample capacity, but are further away
- Concurrent cached accesses without coherence may cause stale data
- Near memory data structures
  - Assume small, homogeneous access time to memory
- NUMA-aware data structures
  - Discriminate only between local memory, rely on hardware cache coherence
  - Simplifying assumptions (e.g., two levels, cache coherence) are short lived
- Traditional distributed data structures
  - Assume memory is operated on by local processor
  - Remote processors access memory via two-sided accesses (RPCs)
- Need to design new “distance-avoiding” data structures for far memory
**FAR MEMORY DATA STRUCTURES**

1. *Far data: truth in far memory*
   - Core content of data structure in far memory, (possibly) persistent

2. *Caches at data-structure clients*
   - (Ephemeral) data structure-specific caches

3. *Algorithm for operations*
   - Executed by clients to access data structure, using hardware (not RPCs)

- **Goal:** $O(1)$ far memory accesses
  - Target: single round trip per operation

- **Idea:** trade far accesses for near accesses
  - Optimize pointer chasing in far memory
  - Cache some data intelligently at clients
  - Efficiently support data sharing with caching

---

M. Aguilera, K. Keeton, S. Novakovic, S. Singhal
*Designing Far Memory Data Structures: Think Outside the Box*
HotOS 2019
**SIMPLE HARDWARE EXTENSIONS CAN HELP**

### Indirect Addressing

- **load**(addr, len) { return *tmp; }
- **store**(addr, val, len) { *tmp = val; }
- **add**(addr, val, len) { *tmp += val; }

### Notifications

- Callback triggered when data changes, avoids remote probing
- Useful for invalidating or updating cached data or metadata at clients

- **notify0**(addr, len): signal change in \([addr, addr + len]\)
- **notifye**(addr, val, len): signal when addr set to val
- **notify0d**(addr, len): signal change in \([addr, addr + len]\), return data
EXAMPLE: HASH TABLE TREE

#1 Far memory:
- tree of hash tables

#2 Near memory:
- tree only

#3 Use local tree to find hash table
- Go to hash bucket in far memory

Advantages
- Quick split of hash table
- One far access per read
• The “data continuum” space is fragmenting, driven by economics
• New media and new access protocols increase heterogeneity
• Simple assumptions (e.g., coherence or 2-level mem) are short lived

• Far memory needs new data structures
• New hardware primitives can help
• Goal: one “far access” per operation

• Think outside the box!