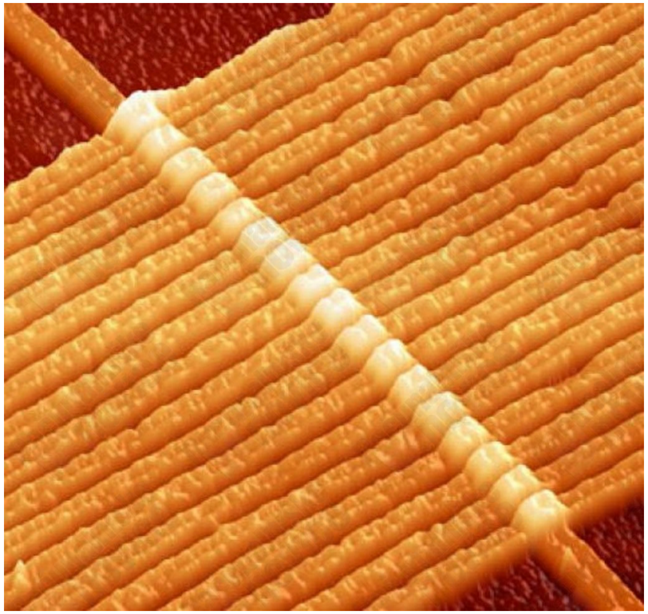
Consistent and Durable Data Structures for Non-Volatile Byte-Addressable Memory

Shivaram Venkataraman*†, Niraj Tolia‡, Parthasarathy Ranganathan* and Roy H. Campbell†

*HP Labs, Palo Alto, ‡Maginatics, and †University of Illinois, Urbana-Champaign

Non-Volatile Byte-Addressable Memory (NVBM)

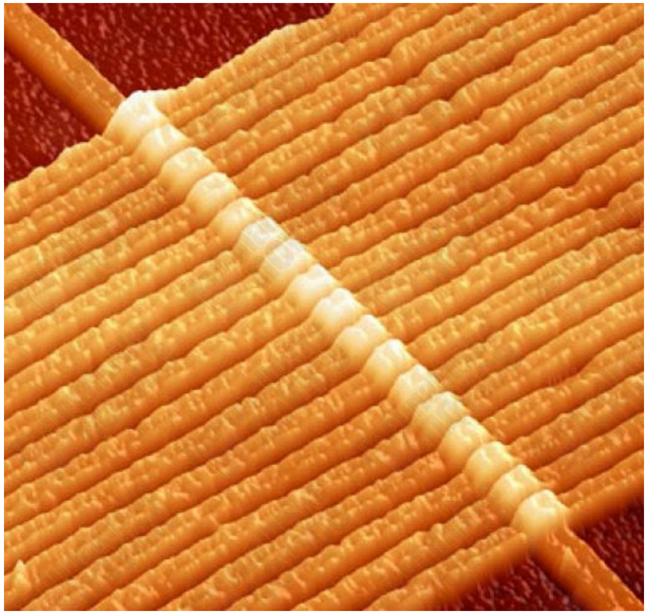


Memristor

Phase Change Memory

Memristor

Non-Volatile Byte-Addressable Memory (NVBM)



Memristor

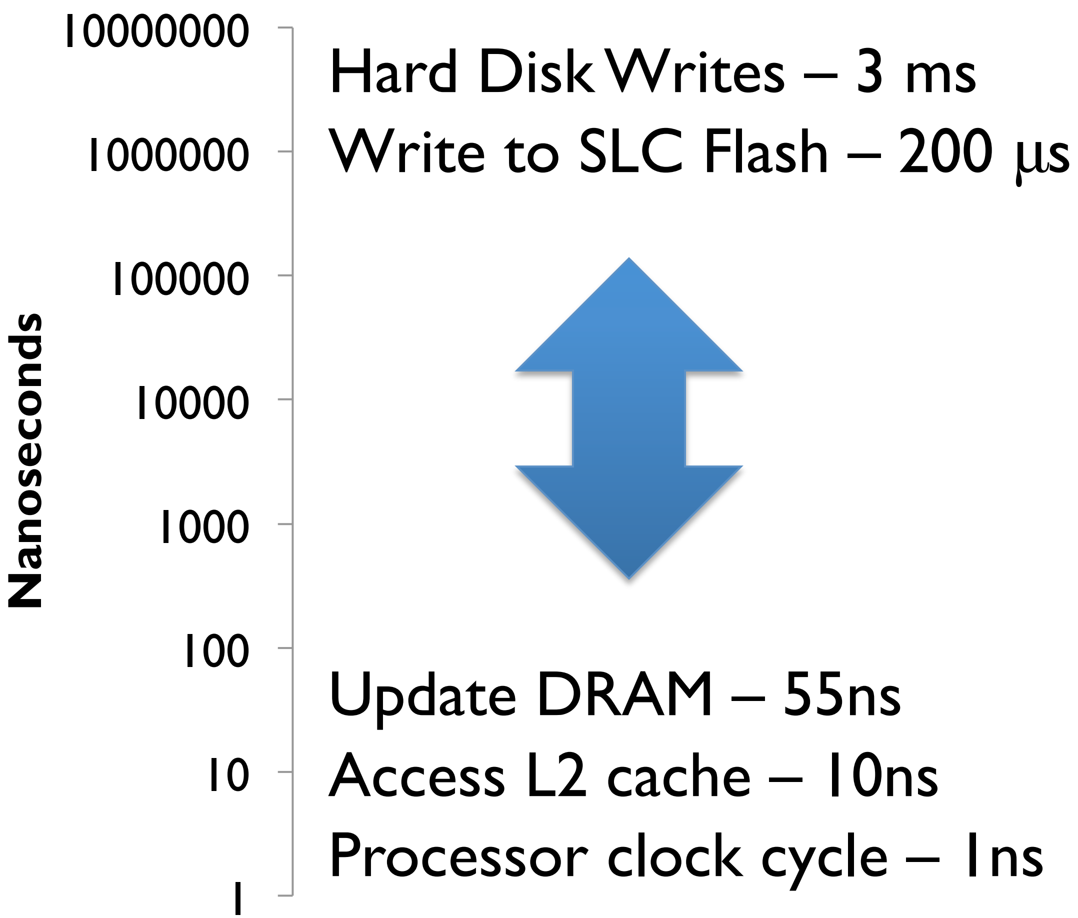
Non-Volatile

50-150 nanoseconds

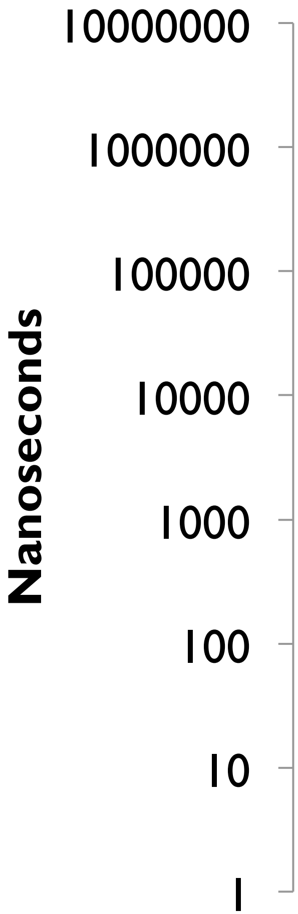
Scalable

Lower energy

Access Times

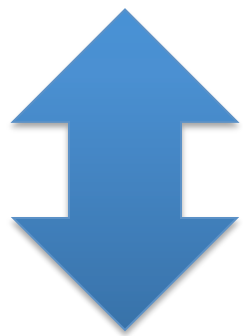


Access Times



Hard Disk Writes – 3 ms

Write to SLC Flash – 200 μs



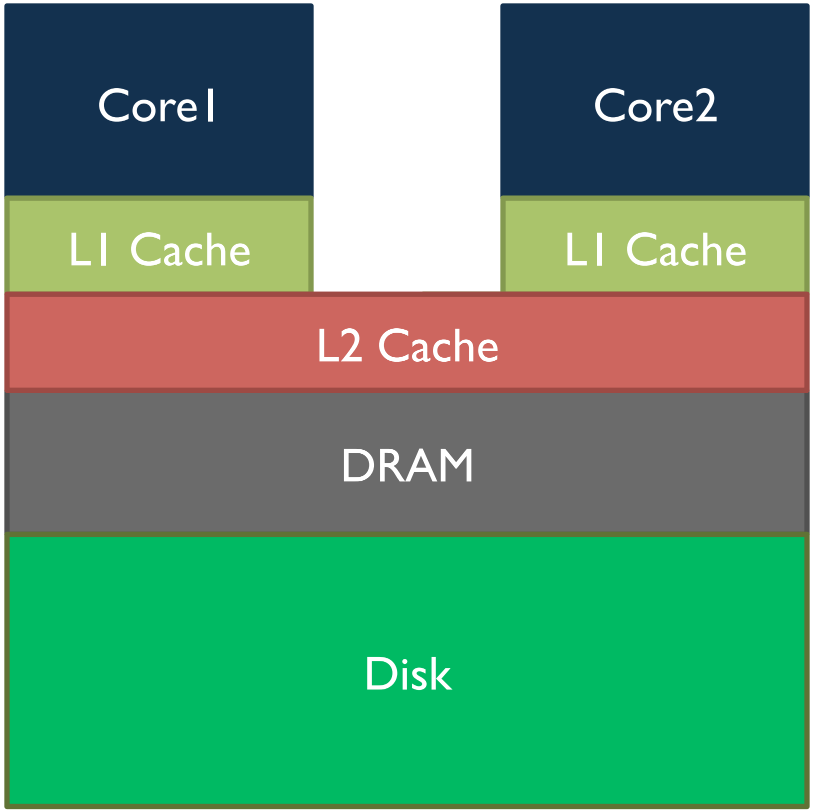
Writes to PCM / Memristor – 100-150 ns

Update DRAM – 55ns

Access L2 cache – 10ns

Processor clock cycle – 1ns

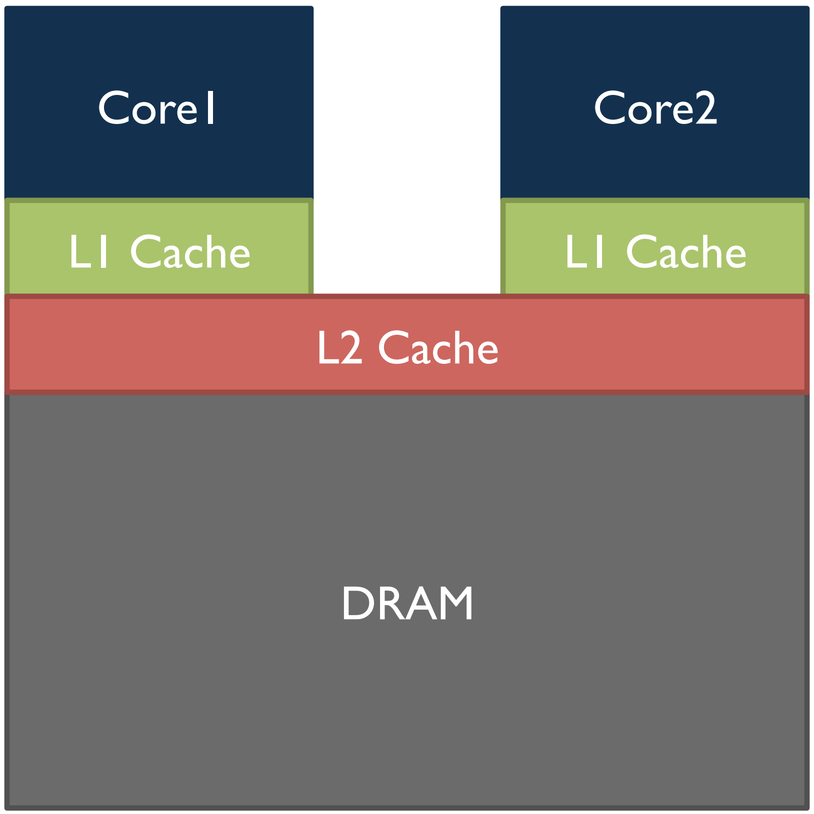
Data Stores - Disk



Traditional DB

File systems

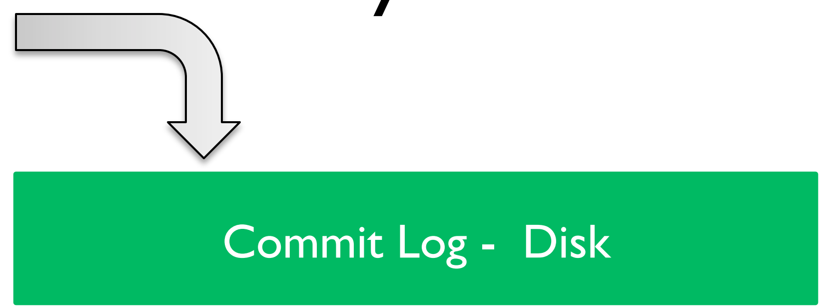
Data Stores - DRAM



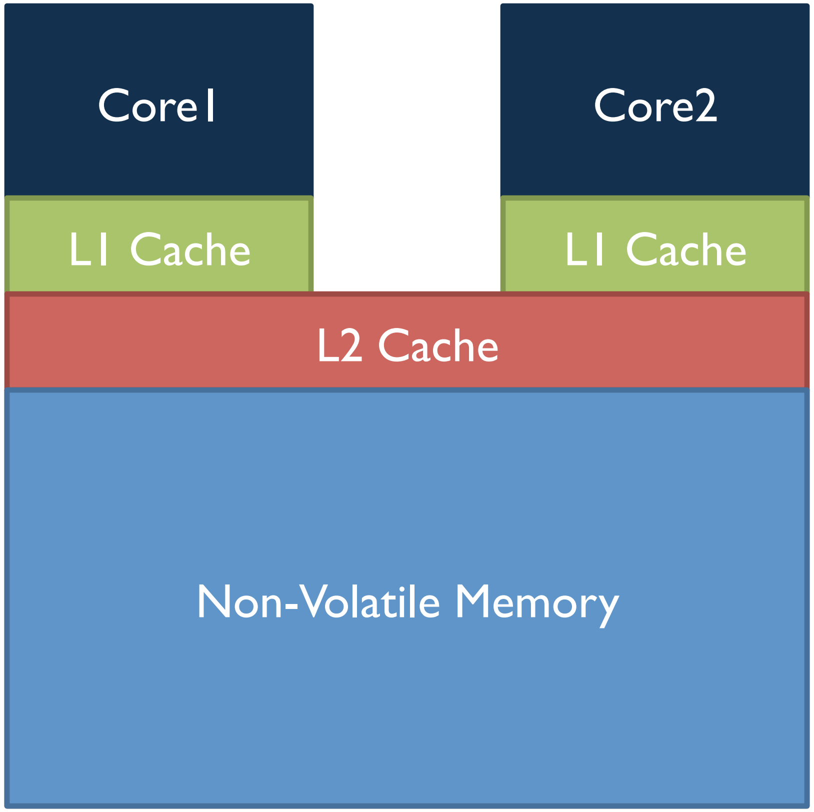
RAMCloud

memcached

Memory-based DB

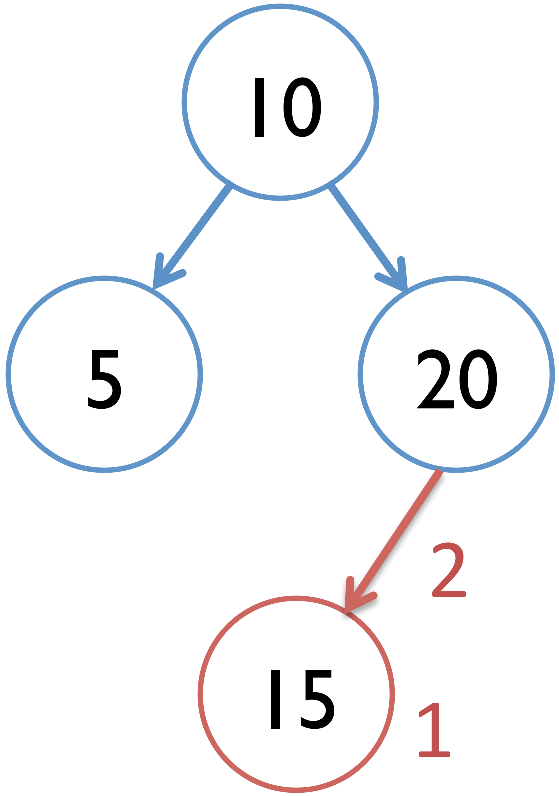


Data Stores - NVBM



Single-level store

Challenges



Consistency

Durability

Outline

- Motivation
- Consistent durable data structures
  - Consistent durable B-Tree
  - Tembo – Distributed Data Store Implementation
- Evaluation

Consistent Durable Data Structures

- Versioning for consistency across failures
- Restore to last consistent version on recovery
- Atomic change across versions
- No new processor extensions!

Versioning

- Totally ordered – Increasing natural numbers
- Every update creates a new version
- Last consistent version
  - Stored in a well-known location
  - Used by reader threads and for recovery

Consistent Durable B-Tree

Key
[start, end)

Live entry

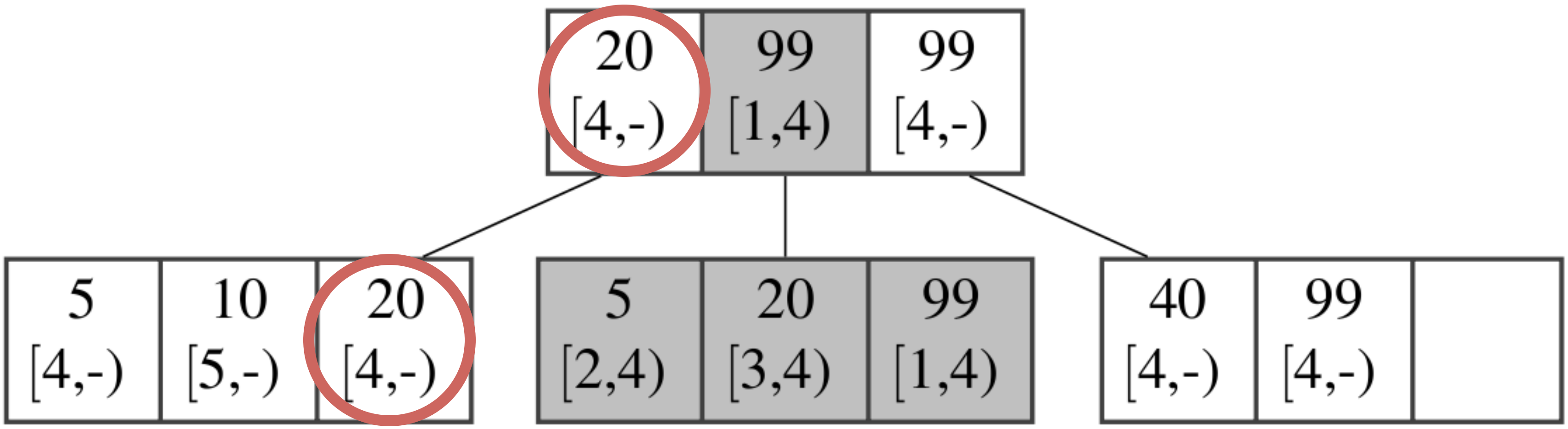
Deleted entry

B – Size of a B-Tree node

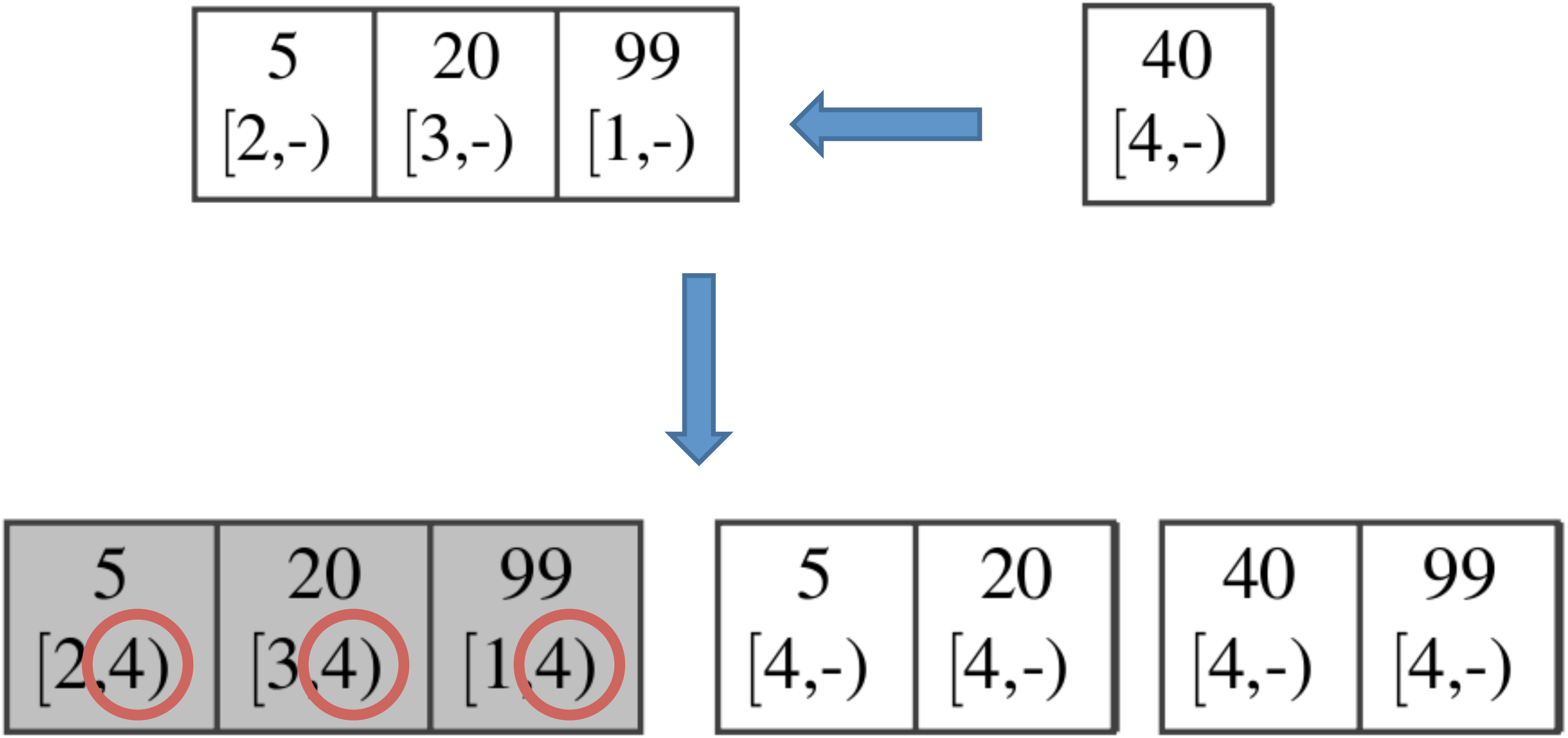
5	10	20	30	40	50	60	70
[1,-)	[2,-)	[3,-)	[4,6)	[5,7)	[8,-)	[9,-)	[10,-)

Lookup

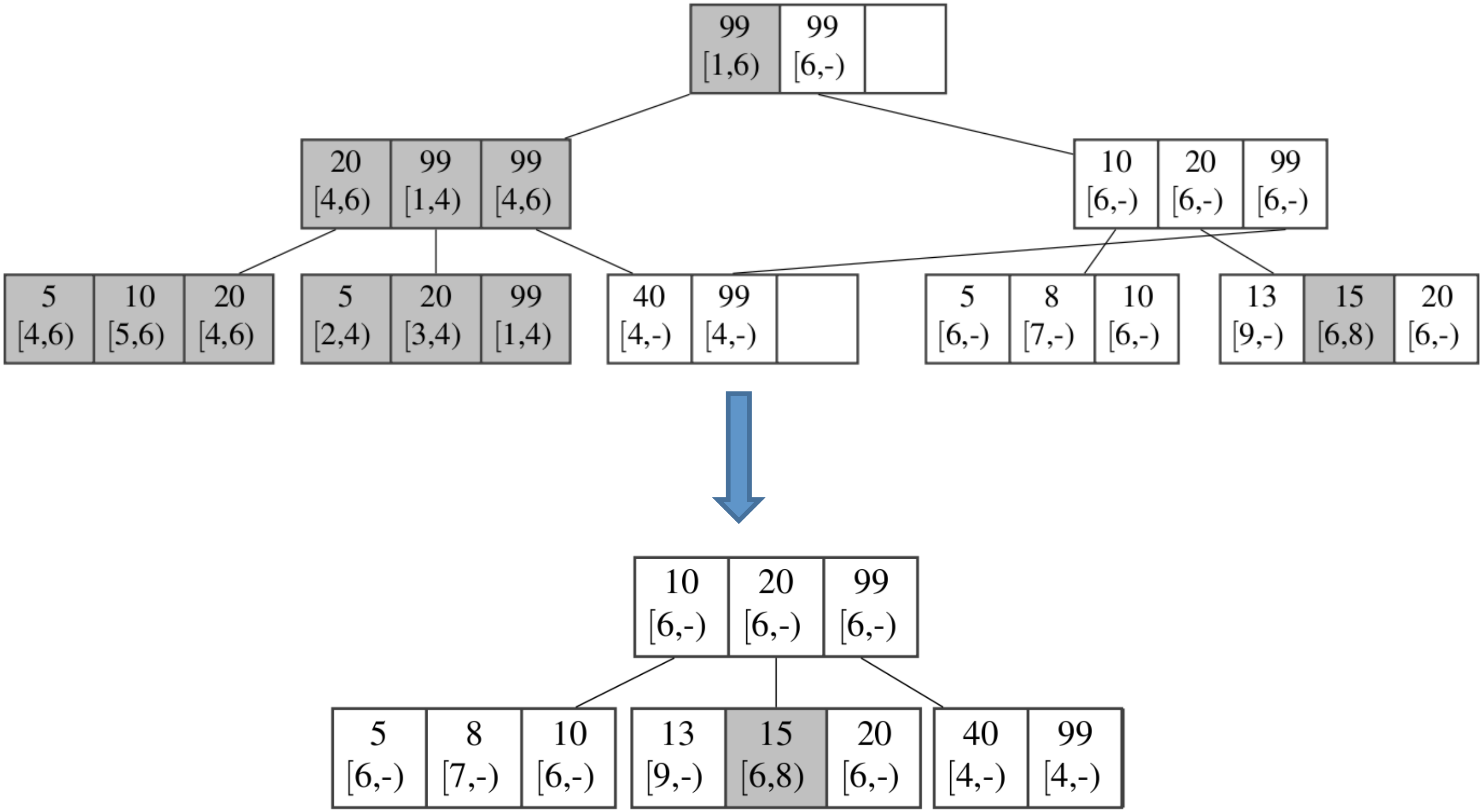
Find key 20 at version 5



Insert / Split



Garbage Collection



Tembo – Distributed Data Store Implementation

Based on open source key-value store

Widely used in production

In-memory dataset

Tembo – Distributed Data Store Implementation

Key Value Server

Consistent durable B-Tree

Single writer, shared reader

Consistent Hashing

Outline

- Motivation
- Consistent durable data structures
  - Consistent durable B-Tree
  - Tembo – Distributed Data Store Implementation
- Evaluation

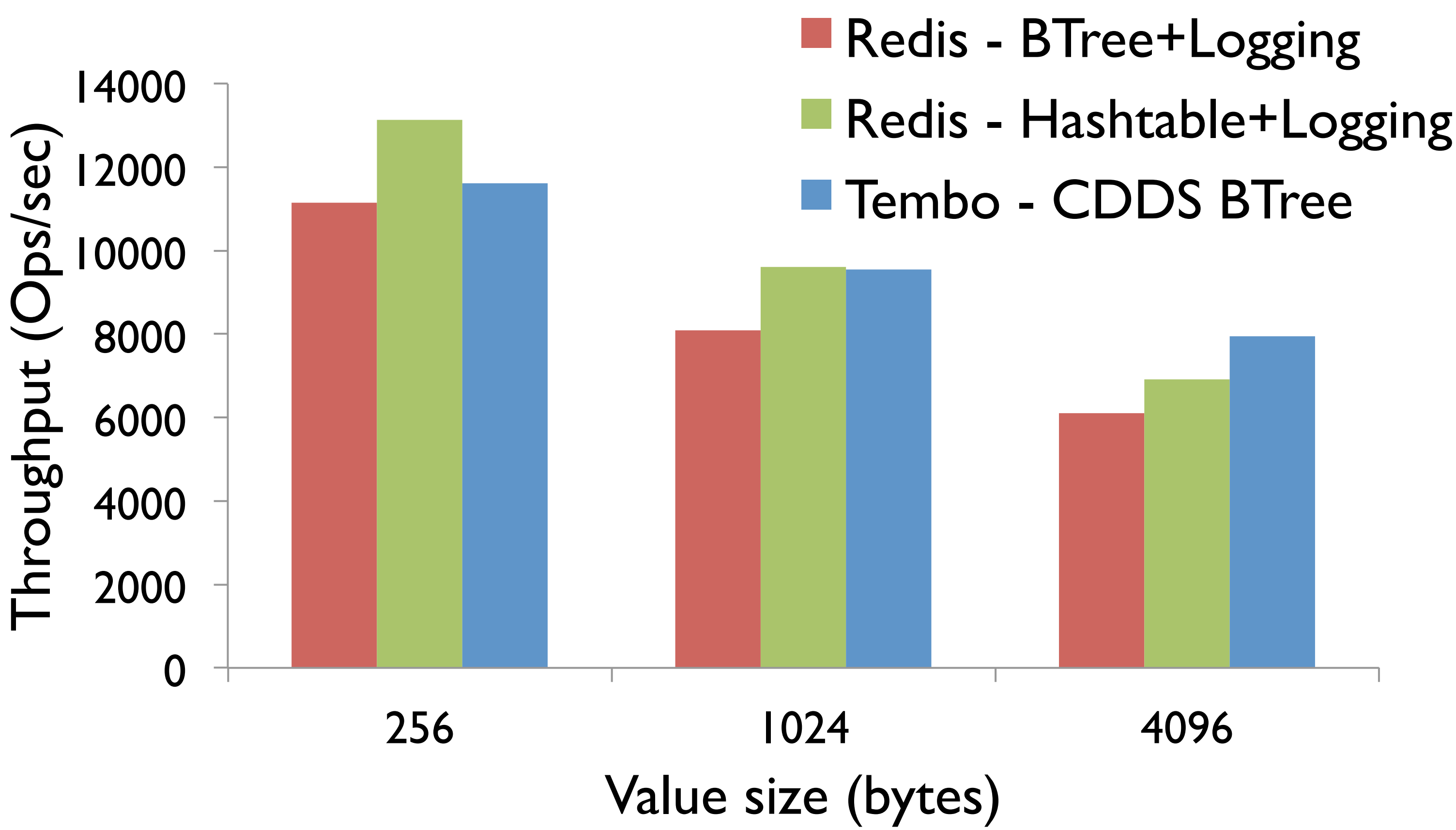
Ease of Integration

	Lines of Code
Original STX B-Tree	2110
CDDS Modifications	1902 (90%)
Redis (v2.0.0-rc4)	18539
Tembo Modifications	321 (1.7%)

Evaluation - Setup

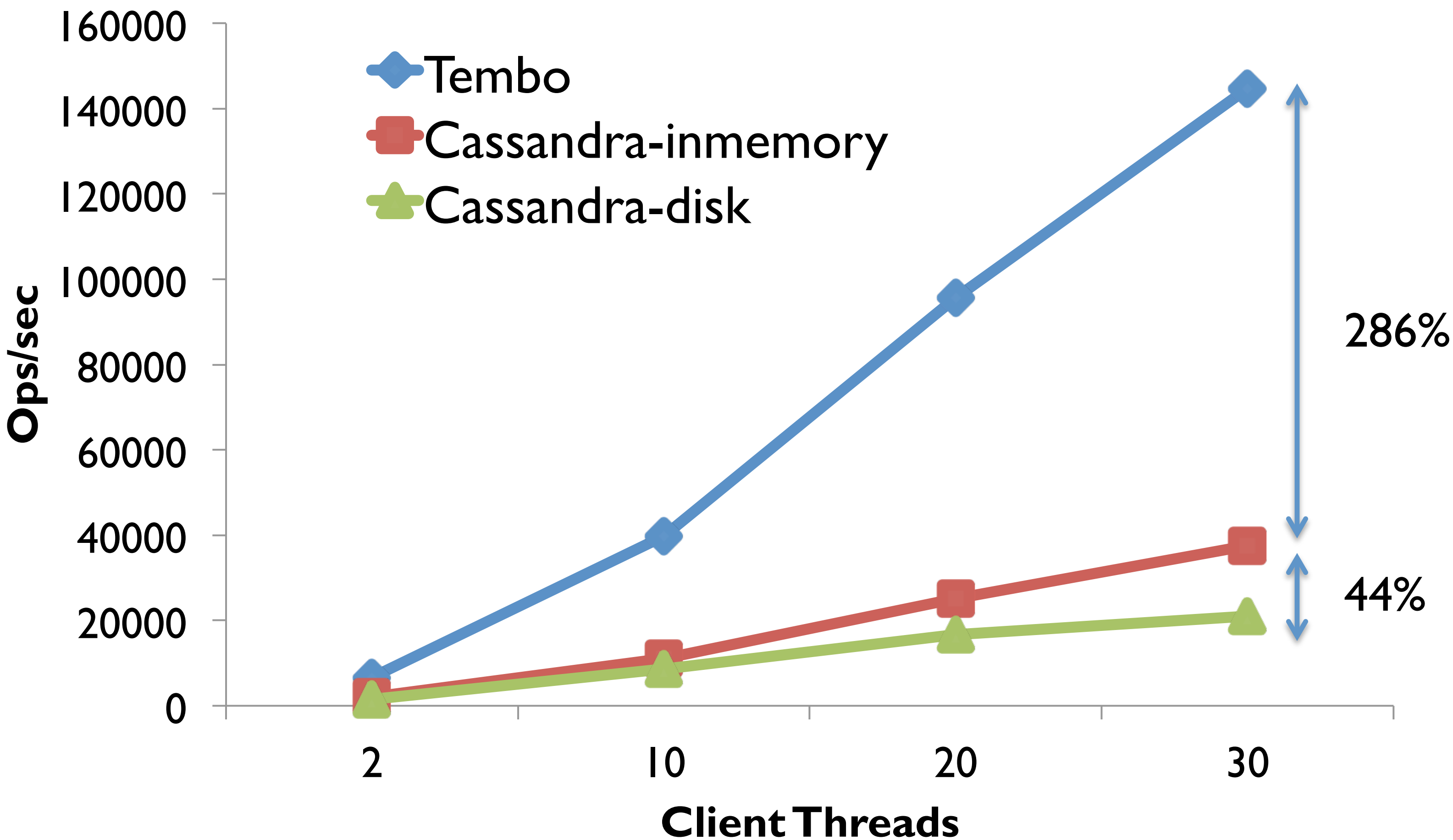
- API Microbenchmarks
  - Compare with Berkeley DB
  - Tembo: Versioning vs. write-ahead logging
- End-to-End Comparison
  - NoSQL systems – Cassandra
  - Yahoo Cloud Serving Benchmark
- 15 node test cluster
  - 13 servers, 2 clients
  - 720 GB RAM, 120 cores

Durability - Logging vs. Versioning



2M insert operations, two client threads

Yahoo Cloud Serving Benchmark



Furthermore

- Algorithms for deletion
- Analysis for space usage and height of B-Tree
- Durability techniques for current processors

Related Work

- Multi-version data structures
  - Used in transaction time databases
- NVBM based systems
  - BPFS – File system (SOSP 2009)
  - NV-Heaps – Transaction Interface (ASPLOS 2011)
- In-memory data stores
  - H-Store – MIT, Brown University, Yale University
  - RAMCloud – Stanford University

Work-in-progress

- Robust reliability testing
- Support for transaction-like operations
- Integration of versioning and wear-leveling

Conclusion

- Changes in storage media
  - Rethink software stack
- Consistent Durable Data Structures
  - Single-level store
  - Durability through versioning
  - Up to 286% faster than memory-backed systems