RAMCloud and the Low-Latency Datacenter

John Ousterhout

Stanford University

Introduction

- Most important driver for innovation in computer systems:
  Rise of the datacenter
- Phase 1: large scale
- Phase 2: low latency
- RAMCloud: new class of storage for low-latency datacenters
- Potential impact: enable new class of applications

How Many Datacenters?

- **Suppose we capitalize IT at the same level as other infrastructure (power, water, highways, telecom):**
  - $1-10K per person?
  - 0.5-5 datacenter servers/person?

	U.S.	World
Servers	**0.15-1.5B**	**3.5-35B**
Datacenters	**1500-15,000**	**35,000-350,000**

(assumes 100,000 servers/datacenter)

- **Computing in 10 years:**
  - Most non-mobile computing (i.e. Intel processors) will be in datacenters
  - Devices provide user interfaces

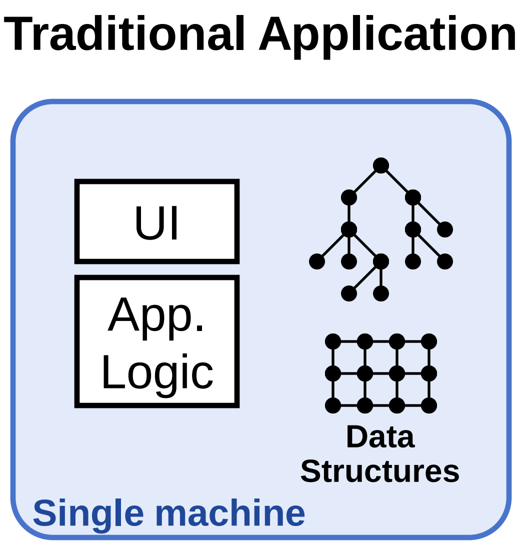
Evolution of Datacenters

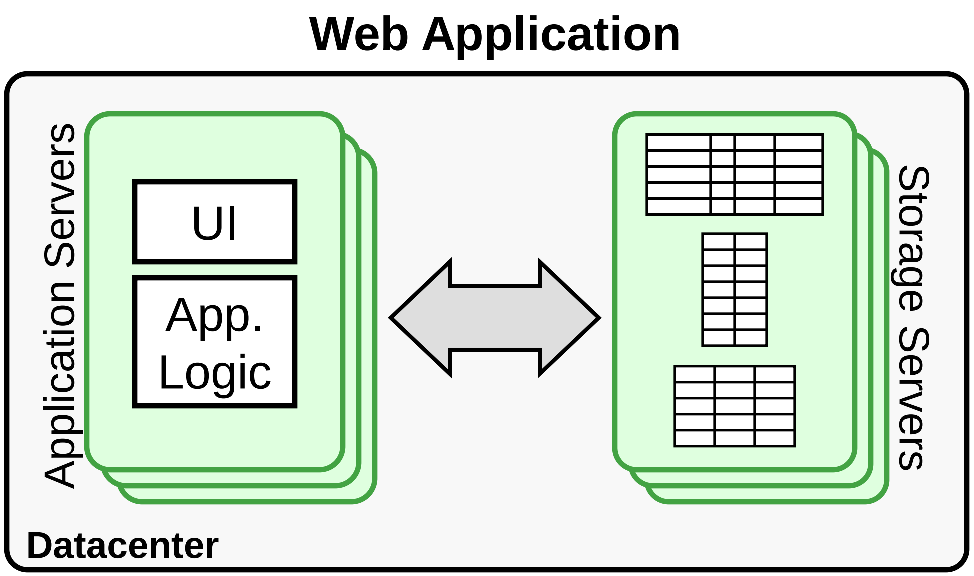
- **Phase 1: manage scale**
  - 10,000-100,000 servers within 50m radius
  - 1 PB DRAM
  - 100 PB disk storage
  - Challenge: how can one application harness thousands of servers?
    - Answer: MapReduce, etc.
- **But, communication latency high:**
  - 300-500µs round-trip times
  - Must process data sequentially to hide latency (e.g. MapReduce)
  - Interactive applications limited in functionality

Evolution of Datacenters, cont'd

- **Phase 2: low latency**
  - Speed-of-light limit: 1µs
  - Round-trip time achievable today: 5-10µs
  - Practical limit (5-10 years): 2-3µs

Why Does Latency Matter?



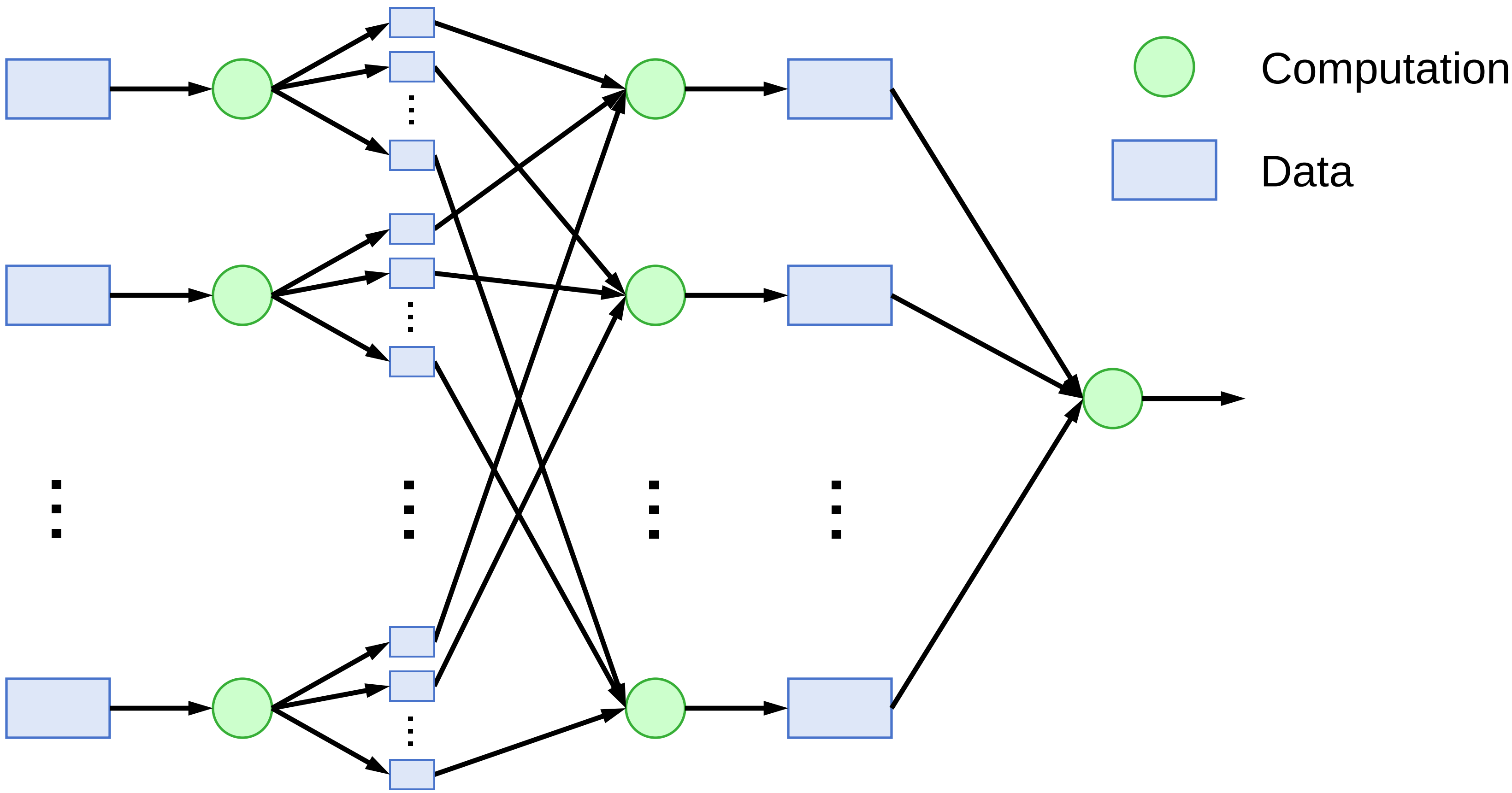


<< 1µs latency

0.5-10ms latency

- **Large-scale apps struggle with high latency**
  - Random access data rate has not scaled!
  - Facebook: can only make 100-150 internal requests per page

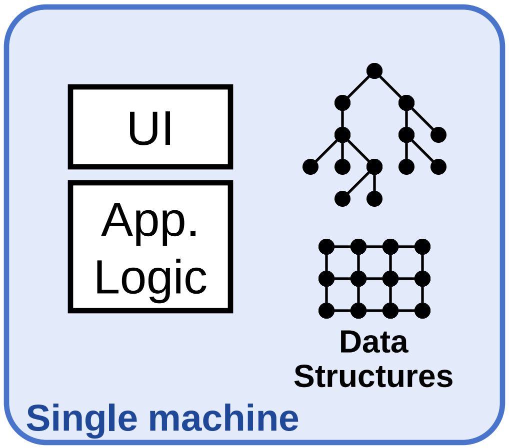
MapReduce



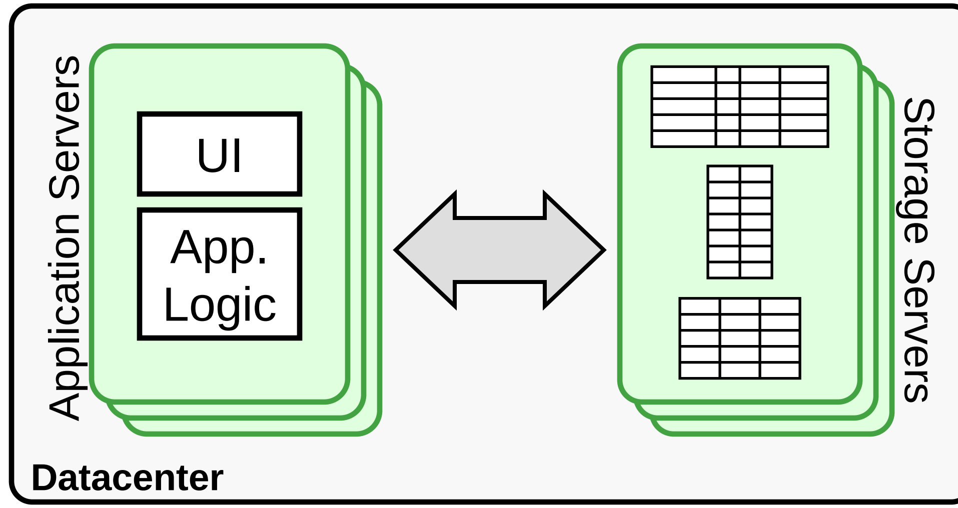
- ✓ **Sequential data access → high data access rate**
- ✗ **Not all applications fit this model**
- ✗ **Offline**

Goal: Scale and Latency

Traditional Application



Web Application



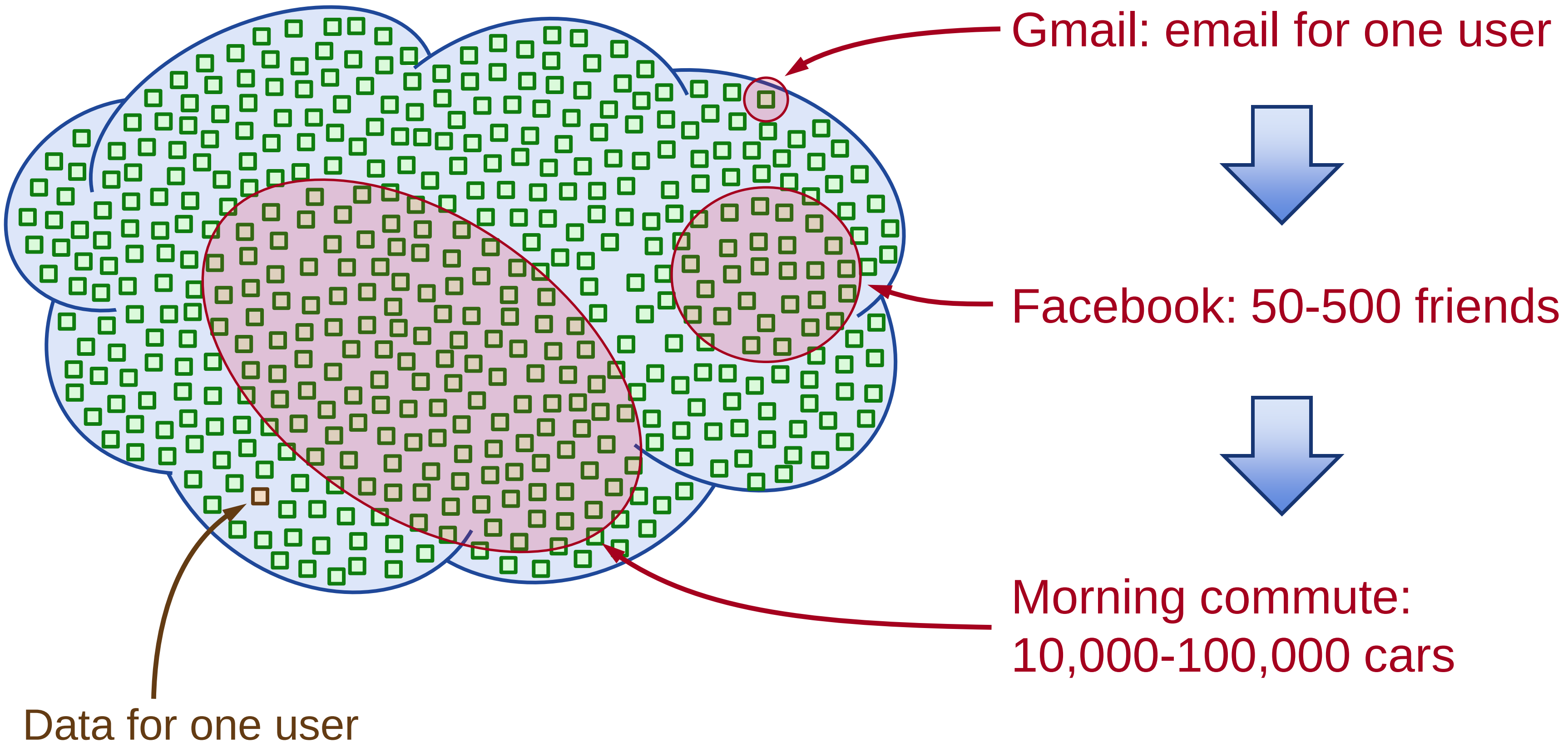
<< 1µs latency

~~0.5-10ms~~ latency
5-10µs

- **Enable new class of applications:**
  - Large-scale graph algorithms (machine learning?)
  - Collaboration at scale?

Large-Scale Collaboration

"Region of Consciousness"



Getting To Low Latency

- **Network switches (currently 10-30µs per switch):**
  - 10Gbit switches: 500ns per switch
  - Radical redesign: 30ns per switch
  - Must eliminate buffering
- **Software (currently 60µs total):**
  - Kernel bypass (direct NIC access from applications): 2µs
  - New protocols, threading architectures: 1µs
- **NIC (currently 2-32µs per transit):**
  - Optimize current architectures: 0.75µs per transit
  - New NIC CPU integration: 50ns per transit

Achievable Round-Trip Latency

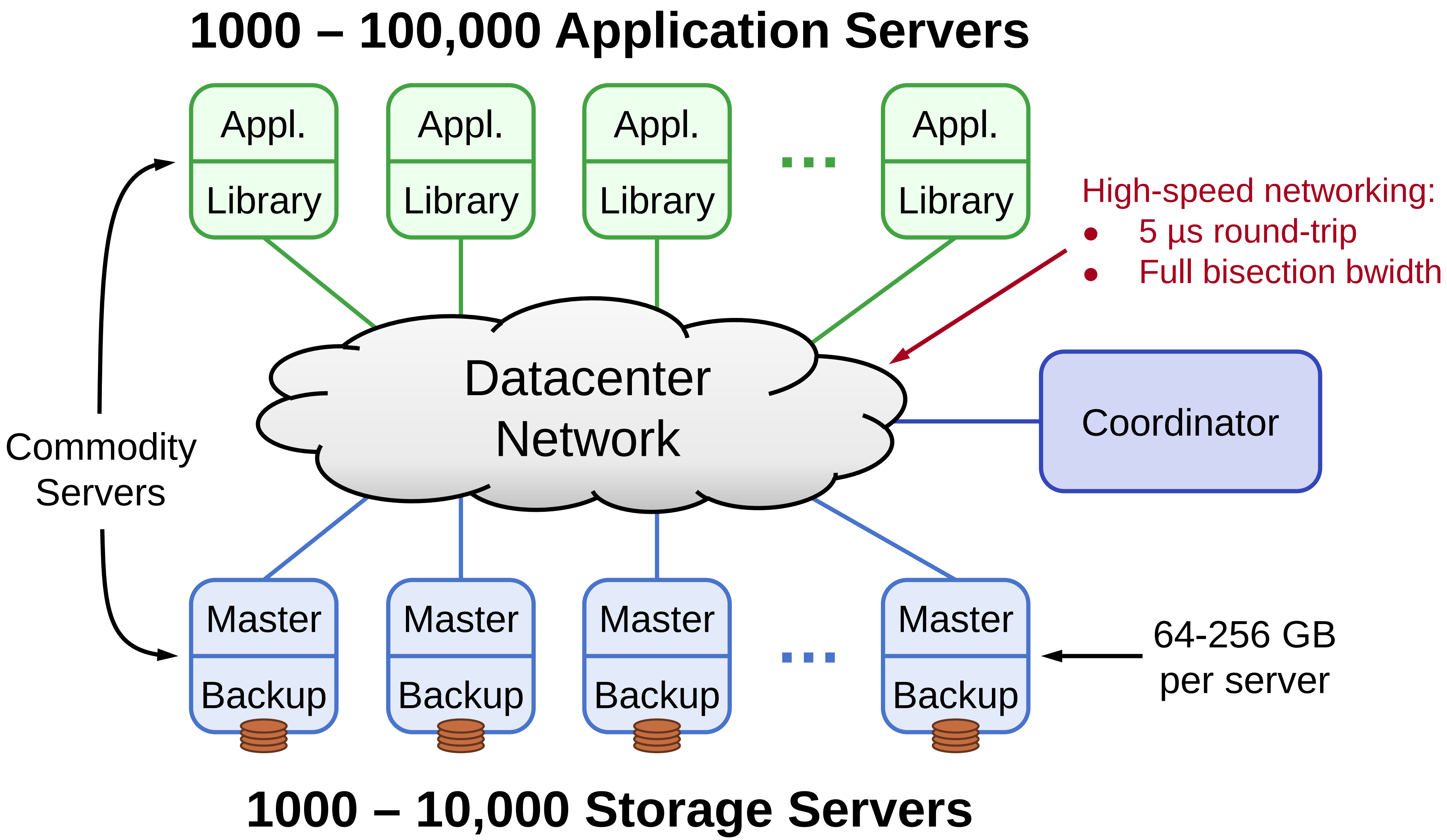
Component	Actual Today	Possible Today	5-10 Years
Switching fabric	100-300µs	5µs	0.2µs
Software	60µs	2µs	1µs
NIC	8-128µs	3µs	0.2µs
Propagation delay	1µs	1µs	1µs
Total	200-400µs	11µs	2.4µs

RAMCloud

Storage system for low-latency datacenters:

- **General-purpose**
- **All data always in DRAM (not a cache)**
- **Durable and available**
- **Scale: 1000+ servers, 100+ TB**
- **Low latency: 5-10µs remote access**

RAMCloud Architecture



Example Configurations

	2012	2015-2020
# servers	2000	2000
GB/server	128 GB	512 GB
Total capacity	256 TB	1 PB
Total server cost	$7M	$7M
$/GB	$27	$7

For $100K today:

- One year of Amazon customer orders (5 TB?)
- One year of United flight reservations (2 TB?)

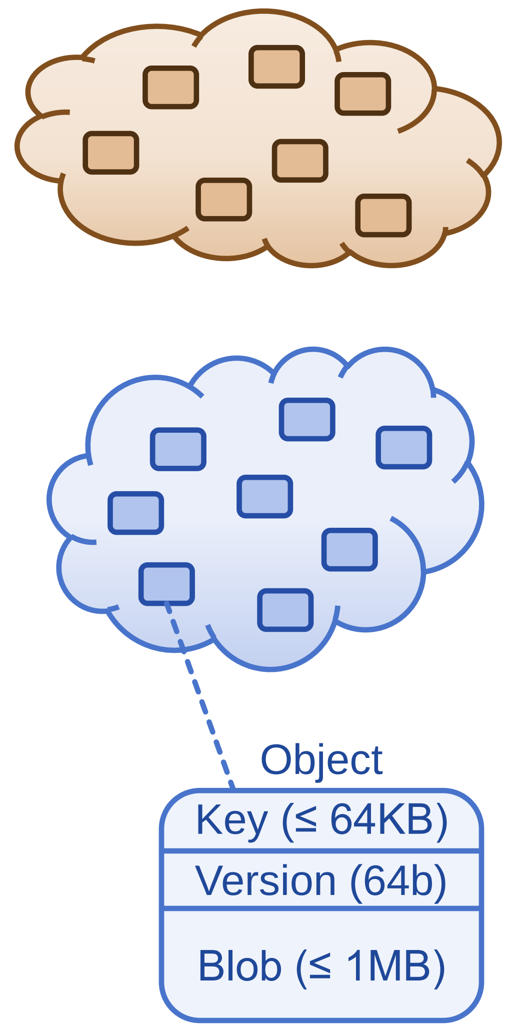
Data Model: Key-Value Store

- **Basic operations:**
  - `read(tableId, key)`
    `=> blob, version`
  - `write(tableId, key, blob)`
    `=> version`
  - `delete(tableId, key)`

(Only overwrite if version matches)

- **Other operations:**
  - `cwrite(tableId, key, blob, version)`
    `=> version`
  - Enumerate objects in table
  - Efficient multi-read, multi-write
  - Atomic increment
- **Not currently available:**
  - Atomic updates of multiple objects
  - Secondary indexes

Tables

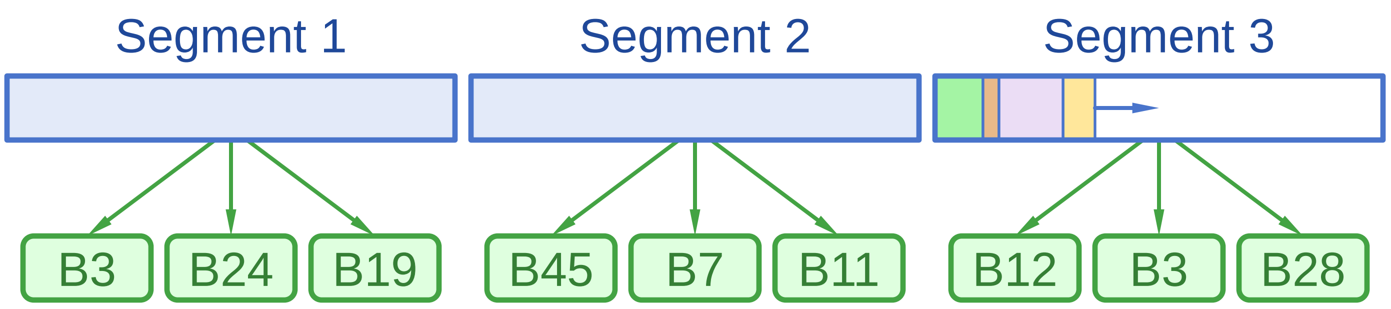


RAMCloud Performance

- **Using Infiniband networking (24 Gb/s, kernel bypass)**
  - Other networking also supported, but slower
- **Reads:**
  - **100B objects: 5µs**
  - 10KB objects: 10µs
  - Single-server throughput (100B objects): 700 Kops/sec.
  - Small-object multi-reads: 1-2M objects/sec.
- **Durable writes:**
  - **100B objects: 16µs**
  - 10KB objects: 40µs
  - Small-object multi-writes: 400-500K objects/sec.

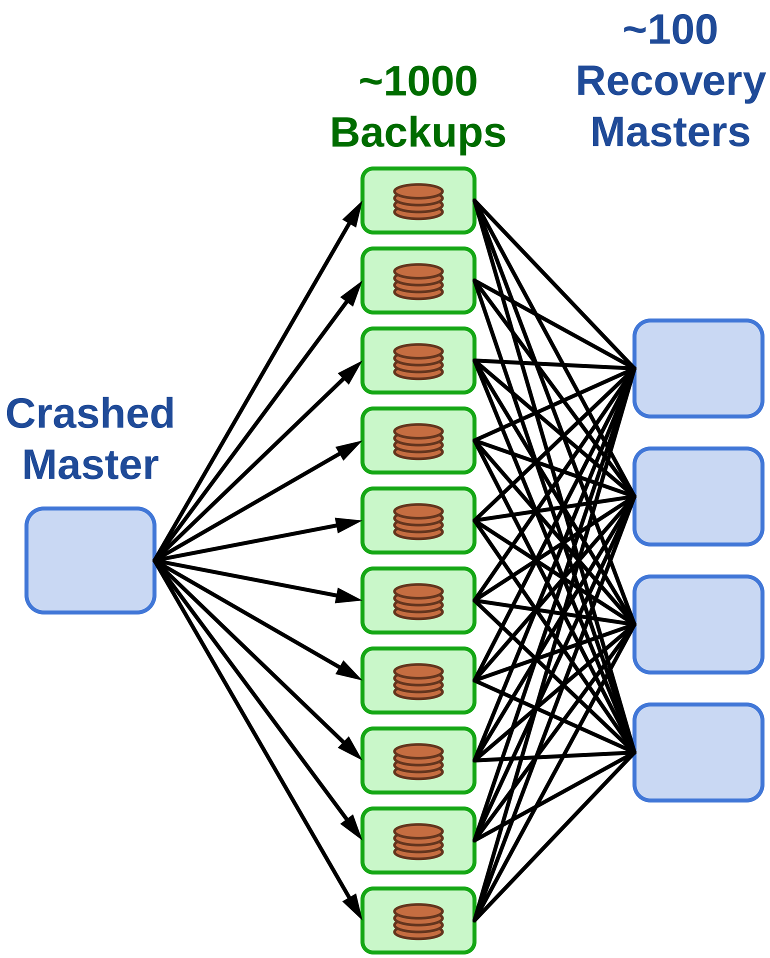
Data Durability

- **Objects (eventually) backed up on disk or flash**
- **Logging approach:**
  - Each master stores its objects in a log
  - Log divided into segments
  - Segments replicated on multiple backups
  - Segment replicas scattered across entire cluster
- **For efficiency, updates buffered on backups**
  - Assume nonvolatile buffers (flushed during power failures)



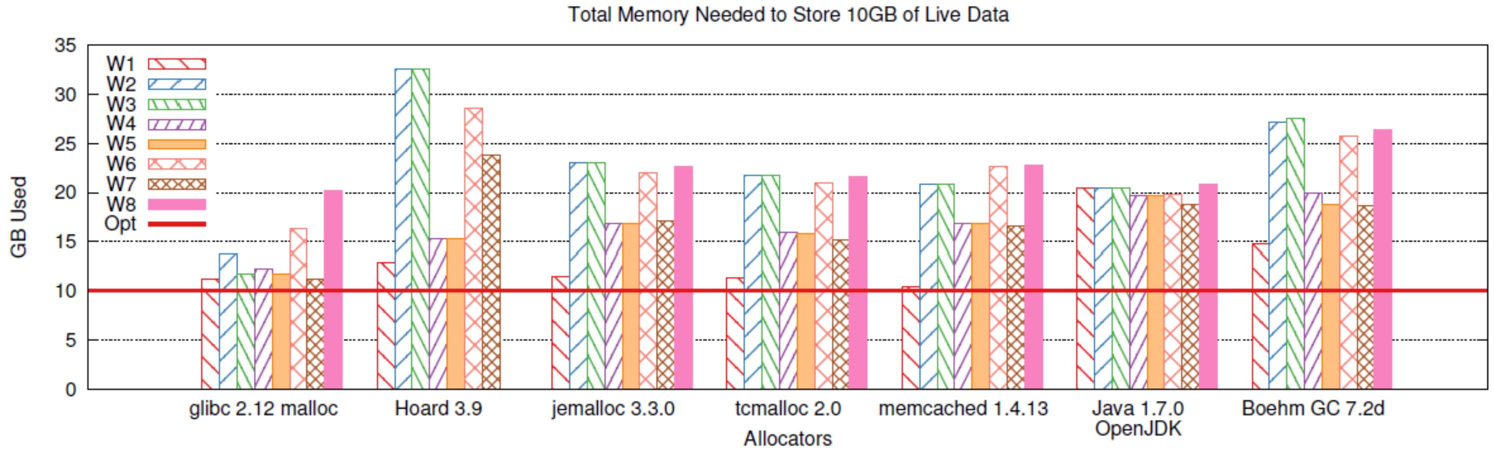
1-2 Second Crash Recovery

- **Each master scatters segment replicas across entire cluster**
- **On crash:**
  - Coordinator partitions dead master's tablets.
  - Partitions assigned to different recovery masters
  - Backups read disks in parallel
  - Log data shuffled from backups to recovery masters
  - Recovery masters replay log entries, incorporate objects into their logs
- **Fast recovery:**
  - 300 MB/s per recovery master
  - Recover 40 GB in 1.8 seconds (80 nodes, 160 SSDs)



Log-Structured Memory

- **Don't use malloc for memory management**
  - Wastes 50% of memory



- **Instead, structure memory as a log**
  - Allocate by appending
  - Log cleaning to reclaim free space
  - Control over pointers allows incremental cleaning
- **Can run efficiently at 80-90% memory utilization**

New Projects

- **Data model:**
  - Can RAMCloud support higher-level features?
    - Secondary indexes
    - Multi-object transactions
    - Graph operations
  - Impact on scale/latency?
- **System architecture for datacenter RPC:**
  - Goals: large scale, low latency
  - Current implementation works, but:
    - Too slow (2µs software overhead per RPC)
    - Sacrifices throughput for latency
    - Too much state per connection (1M connections/server in future?)

Threats to Latency

- **Layering**
  - Great for software structuring
  - Bad for latency
  - E.g. RAMCloud threading structure costs 200-300ns/RPC
  - Virtualization is potential problem
- **Buffering**
  - Network buffers are the enemy of latency
    - TCP will fill them, no matter how large
    - Facebook measured 10’s of ms RPC delay because of buffering
  - Need new networking architectures with no buffers
  - Substitute switching bandwidth for buffers

Conclusion

- **Datacenter revolution < half over:**
  - Scale is here
  - Low latency is coming
- **Next steps:**
  - New networking architectures
  - New storage systems
- **Ultimate result:**
  - Amazing new applications

Extra Slides

The Datacenter Opportunity

- **Exciting combination of features:**
  - Concentrated compute power (~100,000 machines)
  - Large amounts of storage:
    - 1 Pbyte DRAM
    - 100 Pbytes disk
  - Potential for fast communication:
    - Low latency (speed-of-light delay < 1µs)
    - High bandwidth
  - Homogeneous
- **Controlled environment enables experimentation:**
  - E.g. new network protocols
- **Huge Petri dish for innovation over next decade**