Fast Crash Recovery in RAMCloud

Zhitian Xu
Overview

❖ Motivation
❖ Design
  ❖ System Structure
  ❖ Log-structured Storage
  ❖ Recovery
❖ Evaluation
❖ Concluding Remarks
Motivation
Motivation

- Driven by the needs of large-scale Web applications, DRAM has been widely used
- Advantages of DRAM: low cost, fast, enormous volumes
- Disadvantages of DRAM: volatile, hard to use effectively
Design
System Structure

- Hardware settings
  - thousands of off-the-shelf servers, each with as much DRAM as is cost-effective (24 to 64 GB today)

- Hardware properties
  - low latency
  - large scale
System Structure

- Data model
  - key-value store
  - tables with any number of objects
    - a 64-bit identifier
    - a variable-length byte array
    - a 64-bit version number
System Structure

- Each storage server has two components
- One distinguished server call the coordinator
- Client library maintains a cache
Replicas Management

- keep a single copy of each object in DRAM
- keep redundant copies on secondary storage such as disk or flash
Log-Structured Storage

- Purpose: transfer backup data to disk or flash as efficiently as possible
- The backups return immediately to master without writing to disk or flash
Log-Structured Storage

- DRAM and backup storage have the same log structure
- Divide log into segments
- A hash table maps from \(<\text{table identifier}, \text{object identifier}>\) pairs to current version of an object
Recovery

- When a RAMCloud storage server crashes, the objects in the DRAM must be reconstructed.
- If the period of unavailability is short and crashes happen infrequently, then crash recovery will be unnoticeable to application’s users.
- Goal: achieve 1-2 second recovery for servers with at least 64 GB memory.
**Recovery - Overall Approach**

- A simple mirrored approach where each master chooses 3 backups
- Disk bandwidth becomes a recovery bottleneck

![Diagram](image.png)
Recovery - Overall Approach

- Each master scatters its log data across all backups
- Send all log segments to a single recovery master
- Recovery master is a bottleneck
Recovery - Overall Approach

- Use multiple recovery masters
- Combine the disk bandwidth, network bandwidth and CPU cycles of backups and recovery masters
Recovery - Detect Failure

- Client notices if a server fails to respond
- RAMCloud checks its own servers
  - each server periodically send a ping RPC to another server at random and report failures to coordinators
- Timeout needs to be relatively short
Recovery - Scatter Log Segments

- Each RAMCloud master decides independently where to place each replica, using a combination of randomization and refinement
  - randomization: avoid cases in which all masters choose the same backups
  - refinement: provide a nearly optimal solution
- One of the replicas for each segment is the primary replica
Recovery - Three Phases

❖ Setup: find all replicas, select recovery masters and assign partition to each recovery master
❖ Replay: recovery masters incorporate the crashed master’s partitions into their own logs
❖ Cleanup: recovery masters start servicing requests, and crashed master’s log segments are freed from backup storage
Recovery - Setup

- Coordinator queries all backups to find the locations of crashed master’s replicas
Recovery - Setup

- Coordinator needs to determine whether the reported segment replicas form the entire log of the crashed master
- Method: make the log self-describing
Recovery - Setup

- Coordinator divides up the work of recovering the crashed master according to the will of the crashed master, which is uploaded to the coordinator periodically.
- Each master creates a will on his own.
- A will describes how the objects in a master should be partitioned during recovery.
Recovery - Setup

- A partition consists of some tablets
- Each tablet profile tracks the distribution of resource usage within a single
- If a bucket becomes too large, a child bucket is created to subdivide the bucket’s range
Recovery - Replay

- Take up the majority of the recovery time
- Each master receives two things from the setup stage
  - a list of locations of all the crashed master’s log segments
  - a list of tablets that the recovery master must recover
Contents of each segment are processed in six stages.

Different replay order doesn’t matter because version number ensures the correctness.
Recovery - Cleanup

- Recovery master notifies the coordinator when finishing recovery and then starts to service requests.
- When all recovery masters finish recovery, coordinator contacts the backups to free the storage of the crashed master’s segments.
Evaluation
Evaluation - Default Settings

- 60-node cluster
- One backup server on each machine with one disk
- Disk replication factor of 3
Evaluation - Partition Size

- With small objects, speed is limited by the cost of updating hash tables and tablet profiles
- With large objects, speed is limited by the network
- Partition should be limited to no more than 800 MB

![Graph showing recovery time vs partition size for different object sizes](image-url)
Evaluation - Scalability

- With 60 recovery masters, RAMCloud can recover 35 GB of data in 1.6 seconds, which is 26% long than it takes 2 recovery masters to recover 1.2 GB of data.
Closing Remarks

❖ Potential risks
  ❖ scalability
  ❖ short timeout to detect failure
  ❖ fragmented partitions lose locality
❖ Incomplete issues
  ❖ cold start
Questions?
Thank you!