Reducing Replication Bandwidth for Distributed Document DBs

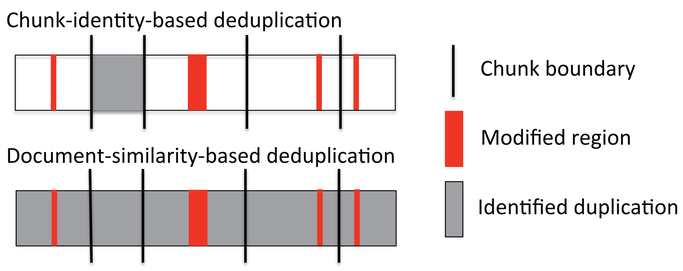
Lianghong Xu (Carnegie Mellon), Andrew Pavlo (Carnegie Mellon), Sudipta Sengupta (Microsoft Research),
Jin Li (Microsoft Research), Greg Ganger (Carnegie Mellon)

Problem and Motivation

- Document-oriented databases have emerged
  - › E.g., MongoDB, CouchDB, RavenDB, RethinkDB...
  - › "Document": self-describing semi-structured data
  - › Popular building block for web services
- Problem: network bandwidth for geo-replication
  - › Replicas synchronize by sending operation logs (oplogs)
  - › WAN bandwidth is expensive and upgraded slowly
  - › Limited bandwidth may degrade user performance
- Oplog deduplication is a promising approach

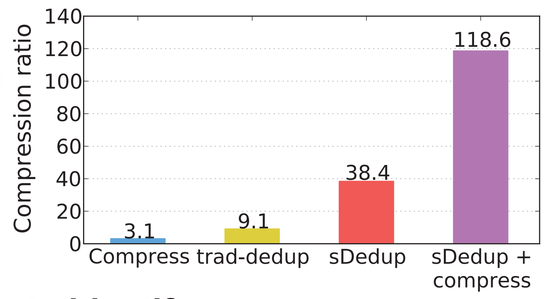
Why Traditional Dedup is Insufficient

- Key characteristics of document database workloads
  - › Most documents are small (< 100 KB)
  - › Changes are small (10s of Bs) and dispersed (~5/doc)
  - › Limited spatial chunk locality
  - › Decent temporal locality of document updates
- Traditional dedup (tradDedup) does not work well
  - › Need to index every unique chunk
  - › Too many chunks have small changes
  - › Decreasing chunk size increases indexing overhead



Why Deduplication?

- Why not just compress?
  - › Update batches are small
  - › Random unrelated docs



- Why not just "diff"
  - › Need application guidance to identify source
  - › Limited scope for available sources
- Deduplication finds and removes redundancy
  - › Relative to entire corpus w/o application guidance

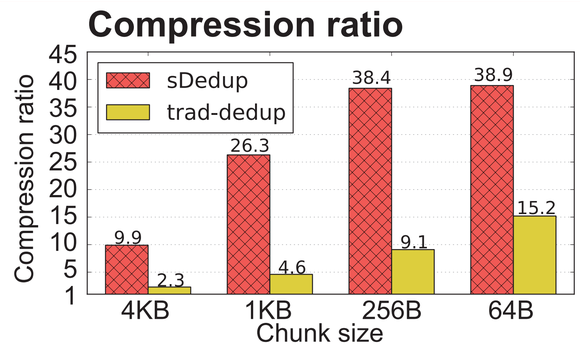
Similarity-based Dedup (sDedup)

- Deduplication workflow
  - › Use sampled chunk hashes to find similar docs
  - › Select one best match as source
  - › Delta compress
- Resource-efficient design
  - › ≤ 8 index entries per doc (vs. 1 per chunk in tradDedup)
  - › Compact key signature for Cuckoo hashing
  - › Small source document cache (90% hits)
- Easy integration into existing document DBMSs
  - › Use sDedup on each oplog entry
  - › Send deduplicated data to replicas

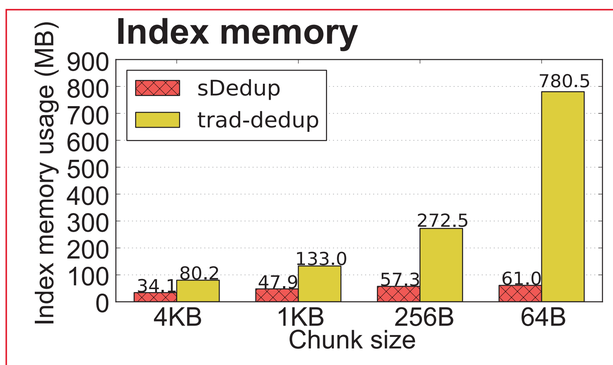
Evaluation

- MongoDB v2.7, Wikipedia dataset

Compression ratio



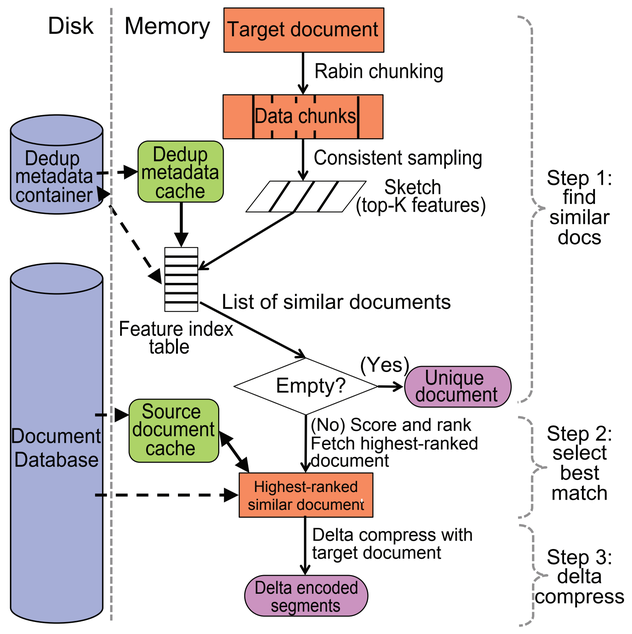
Index memory



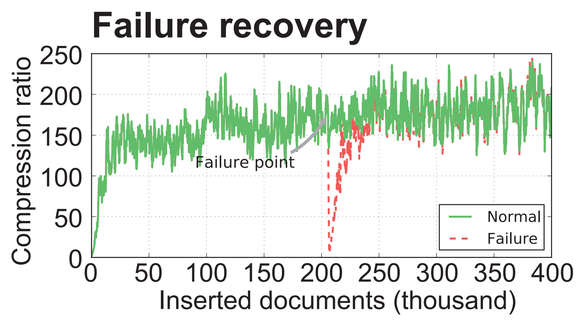
Scaling with sharding

# Shards	1	3	5	9
Comp. Ratio	38.4	38.2	38.1	37.9

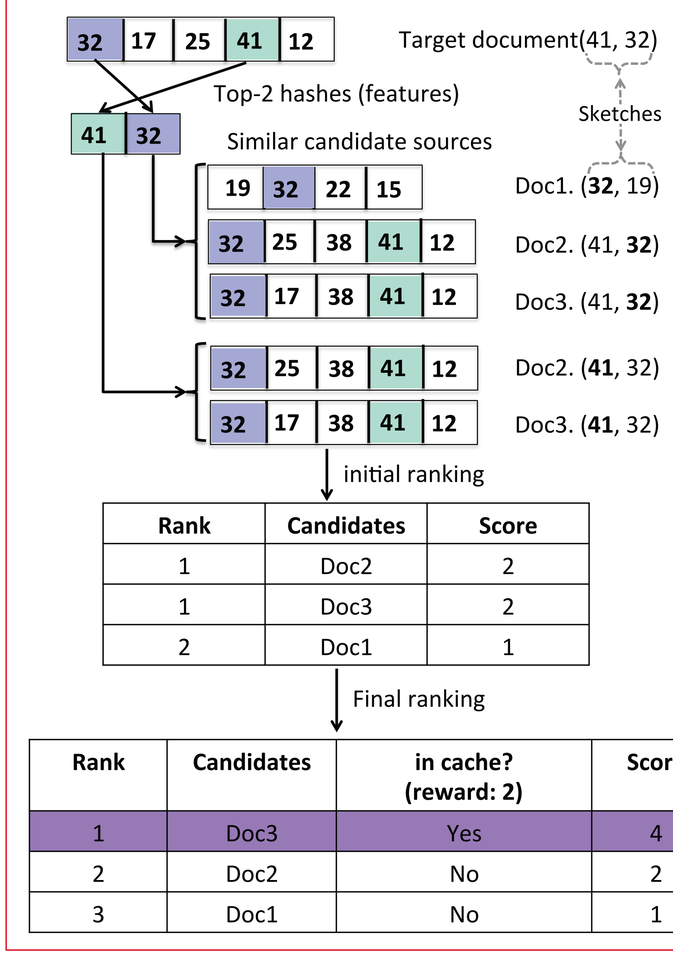
sDedup workflow



Failure recovery



Find similar docs + select best match



Rank	Candidates	Score
1	Doc2	2
1	Doc3	2
2	Doc1	1

Final ranking

Rank	Candidates	in cache? (reward: 2)	Score
1	Doc3	Yes	4
2	Doc2	No	2
3	Doc1	No	1

Carnegie Mellon University

Parallel Data Laboratory