Content-Based Block Caching

Charles B. Morrey III and Dirk Grunwald
University of Colorado, Boulder
Department of Computer Science

Overview

- Thesis
- Motivation
- System Design
- Results
- Related Work
- Conclusion

Thesis

Content-Based Block Caching attempts to maintain a single copy of any block in memory according to its contents. In the presence of repeated content, this mechanism increases the effective size of the buffer cache.

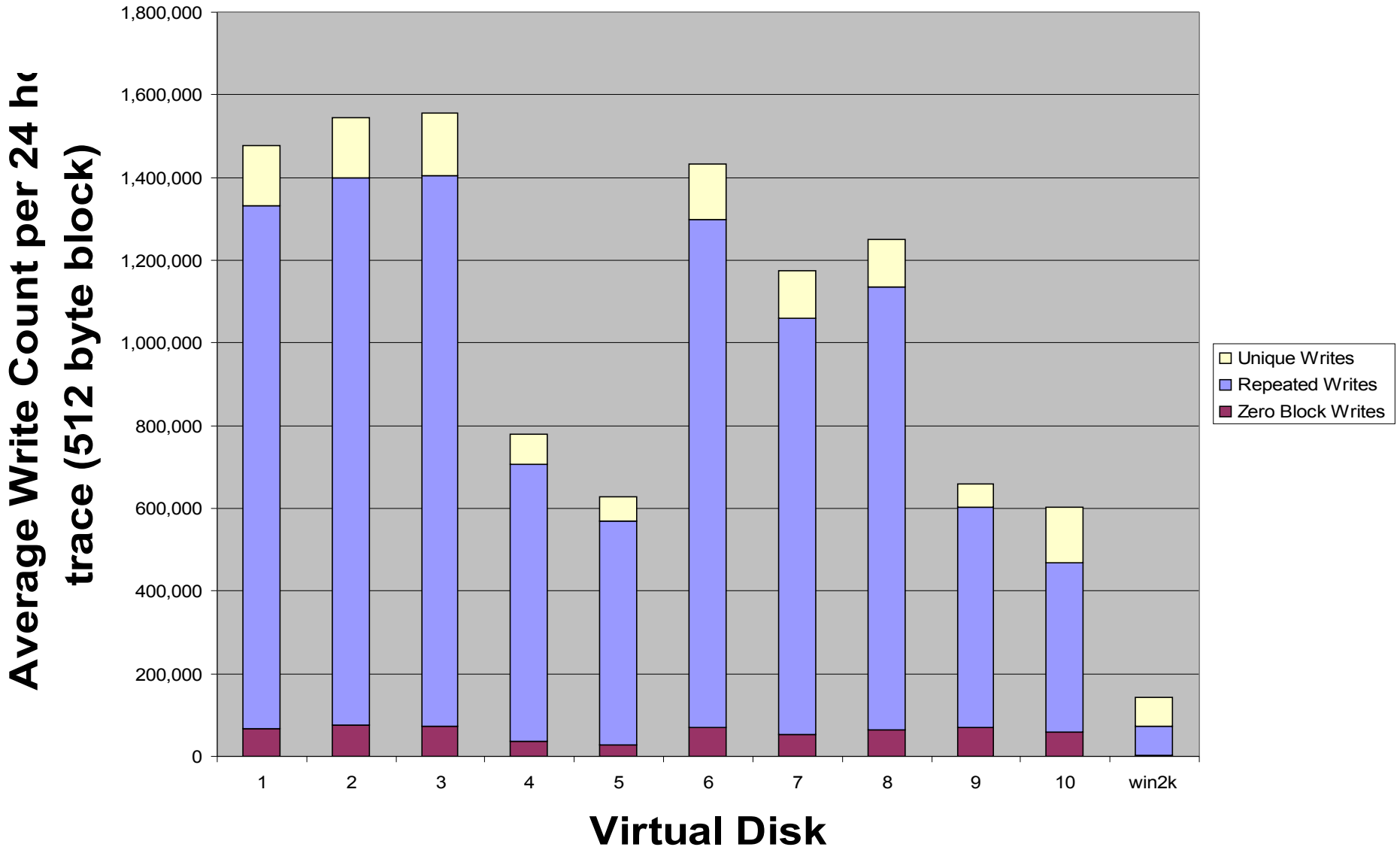
Motivation

- CPU and memory subsystems are experiencing performance growth much faster than disks.
  - Trade a bounded amount of computation and small amount of cache memory for a better cache hit rate (in the presence of repeated content)
  - Better cache hit rate should imply a shorter I/O stall time (hypothesis)

Testing For Repeated Content

- 11 Workstations configured to network boot an iSCSI disk which logged all writes
- 10 Mandrake Linux machines
- 1 Windows 2000 Machine inside VMWare Workstation 3.2 for Linux
- Test Systems used as student workstations for several weeks
  - Email, Web Browsing, Editing, Debugging, Compilation

Is There Repeated Content?



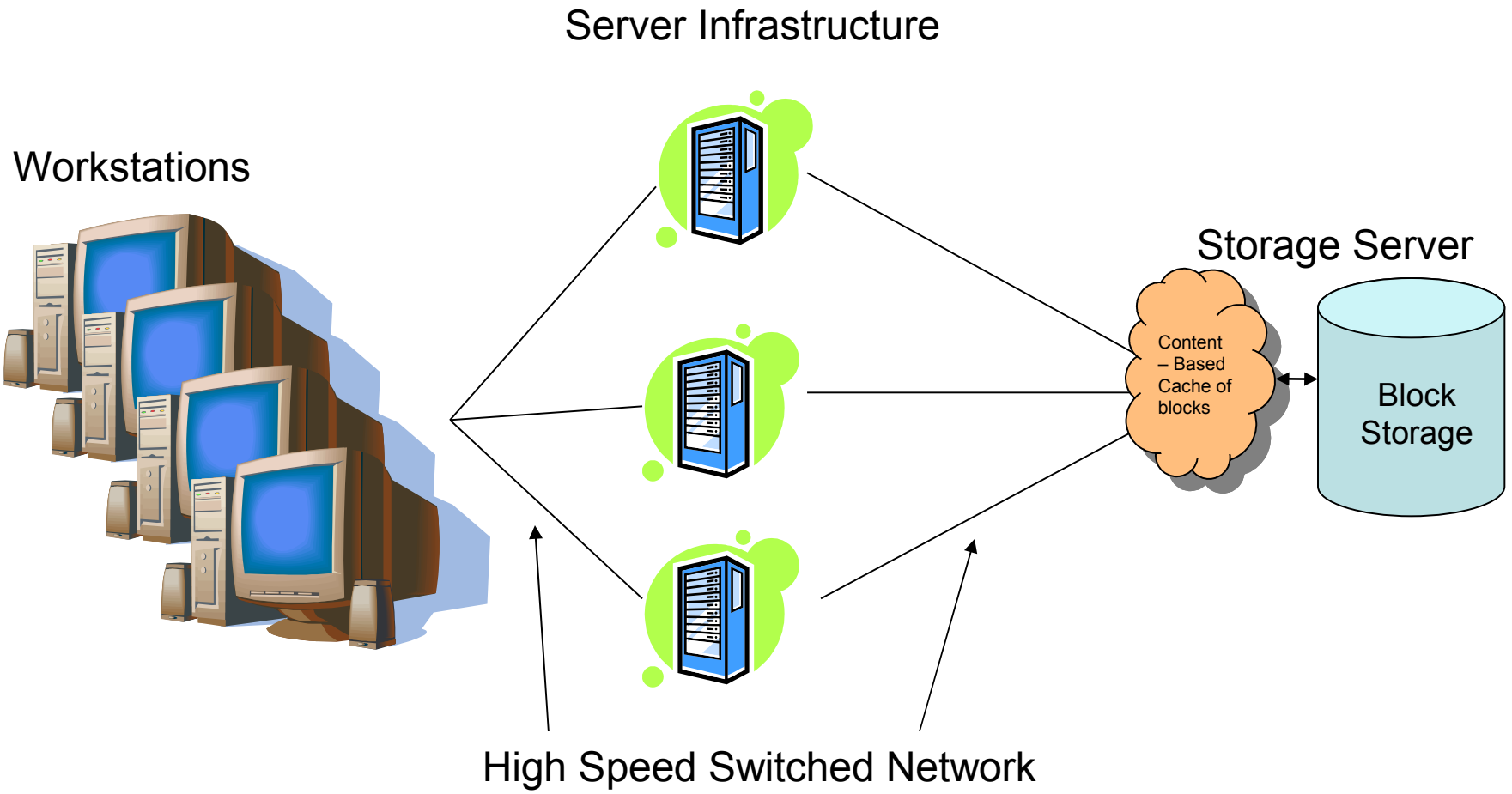
Research Artifacts

- Cache Simulator of CBBC
  - provides only hit rate results
- Disksim 3.0 modified to add CBBC
  - Provides accurate disk timing
- Several months of live system disk traces of Linux (ext3) and Windows (NTFS)

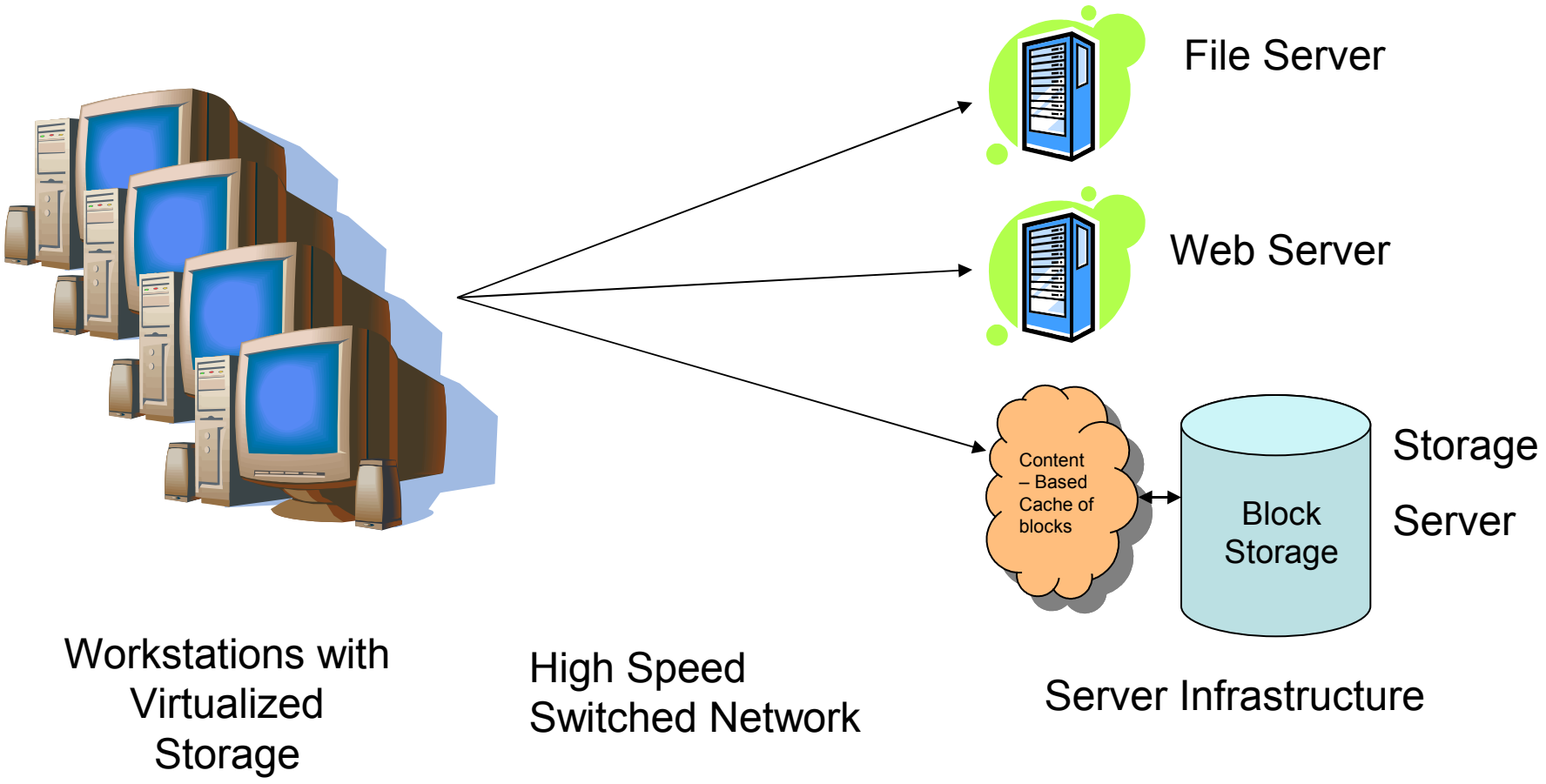
Overview

- Thesis
- Motivation
- **System Design**
- Results
- Related Work
- Conclusion

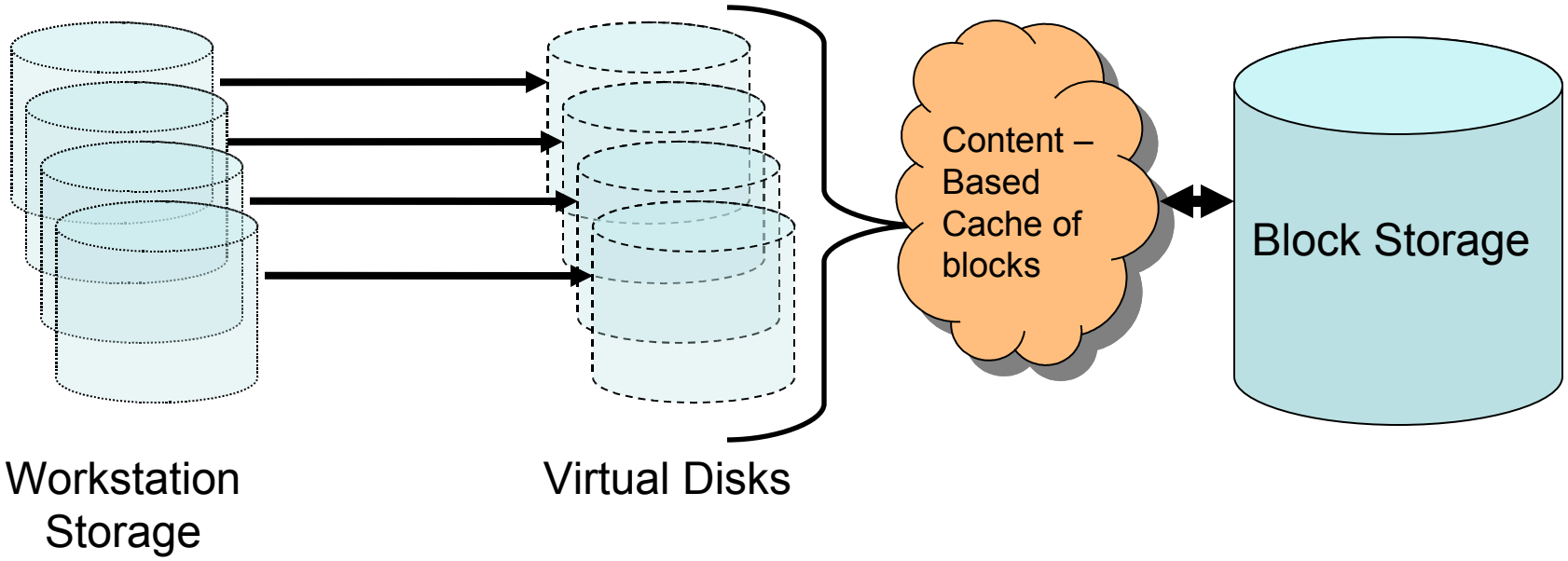
Storage Server Configuration



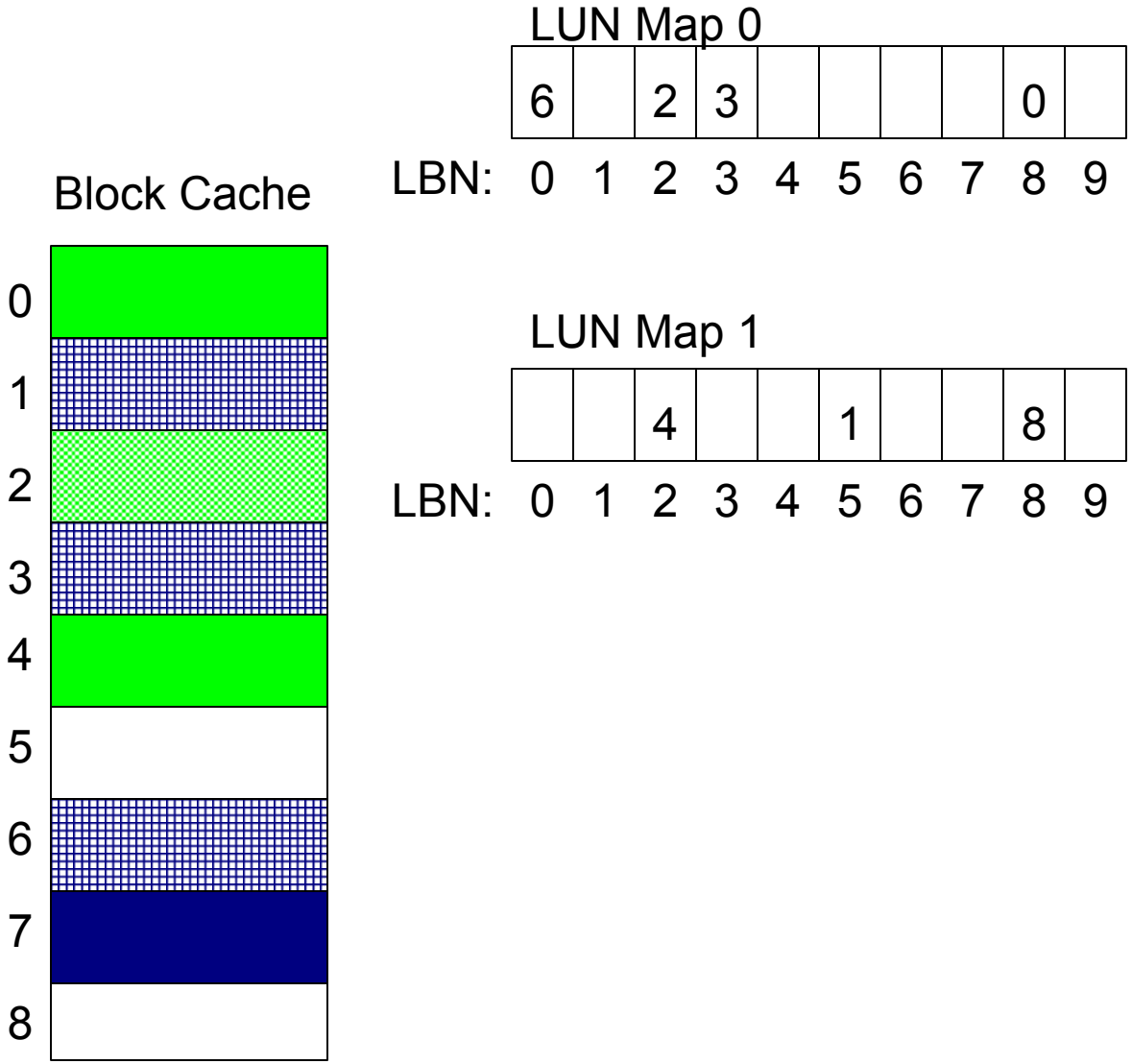
Another Configuration



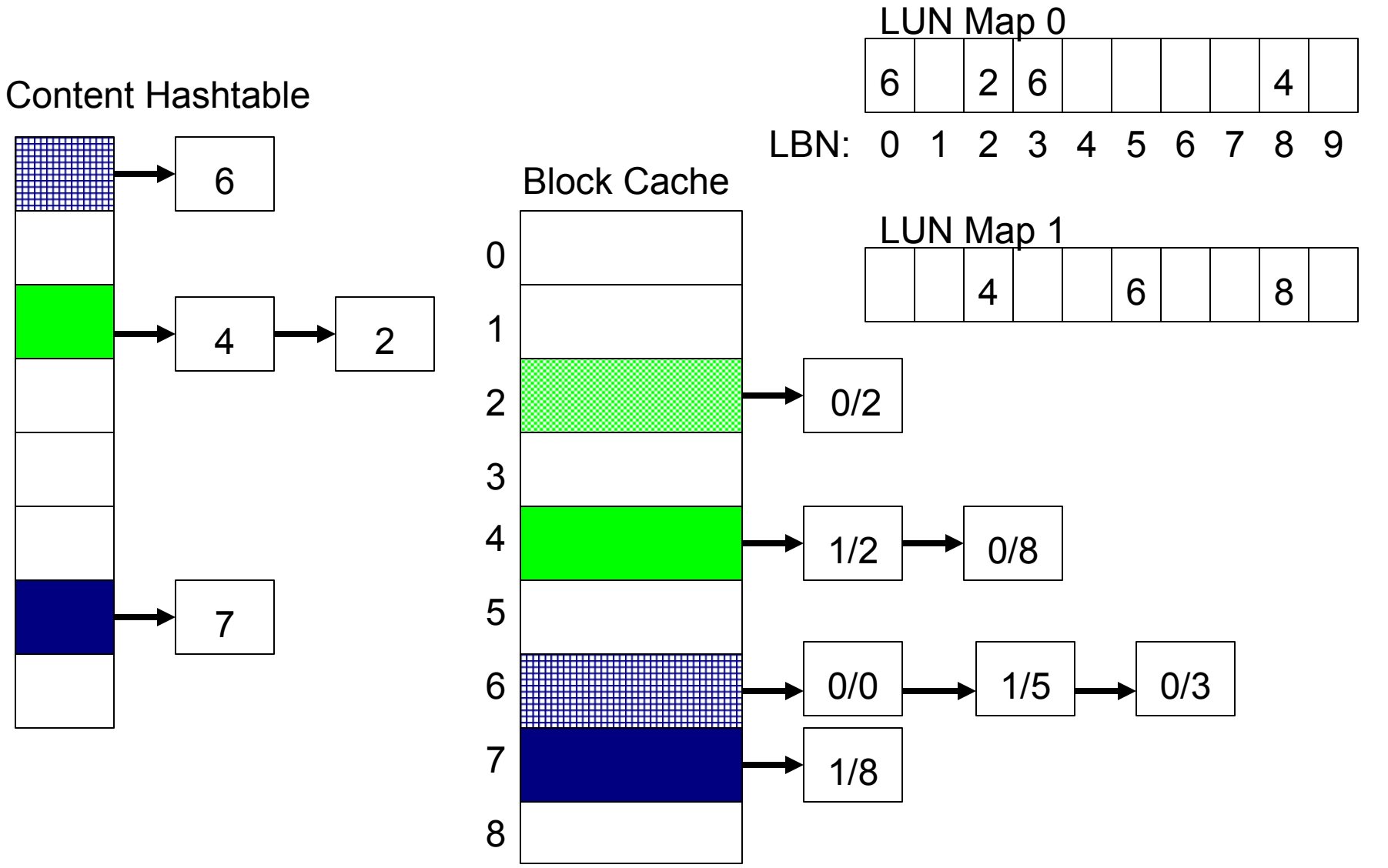
CBBC Integration



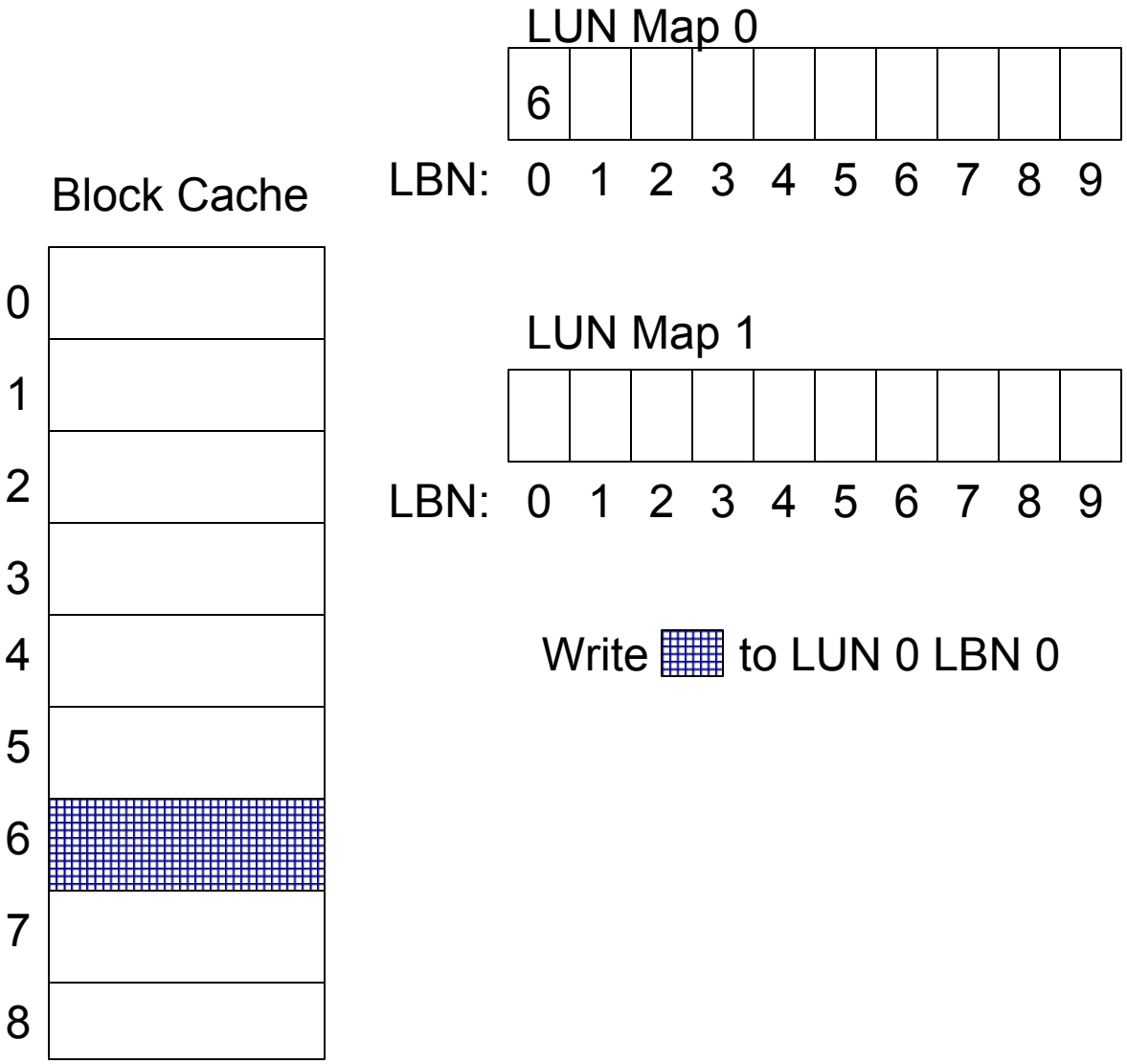
Offset-Based Block Cache



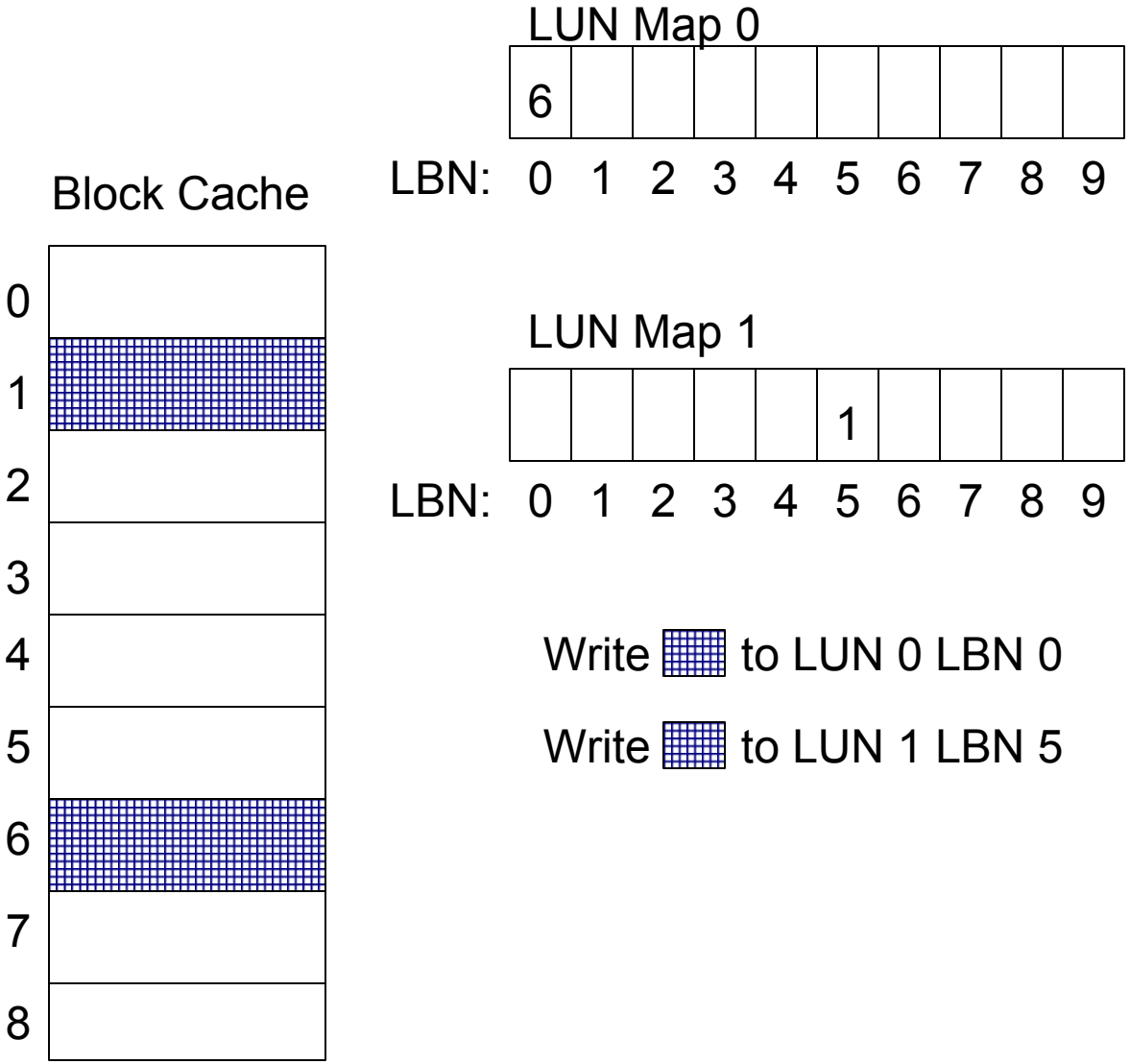
Content-Based Block Cache



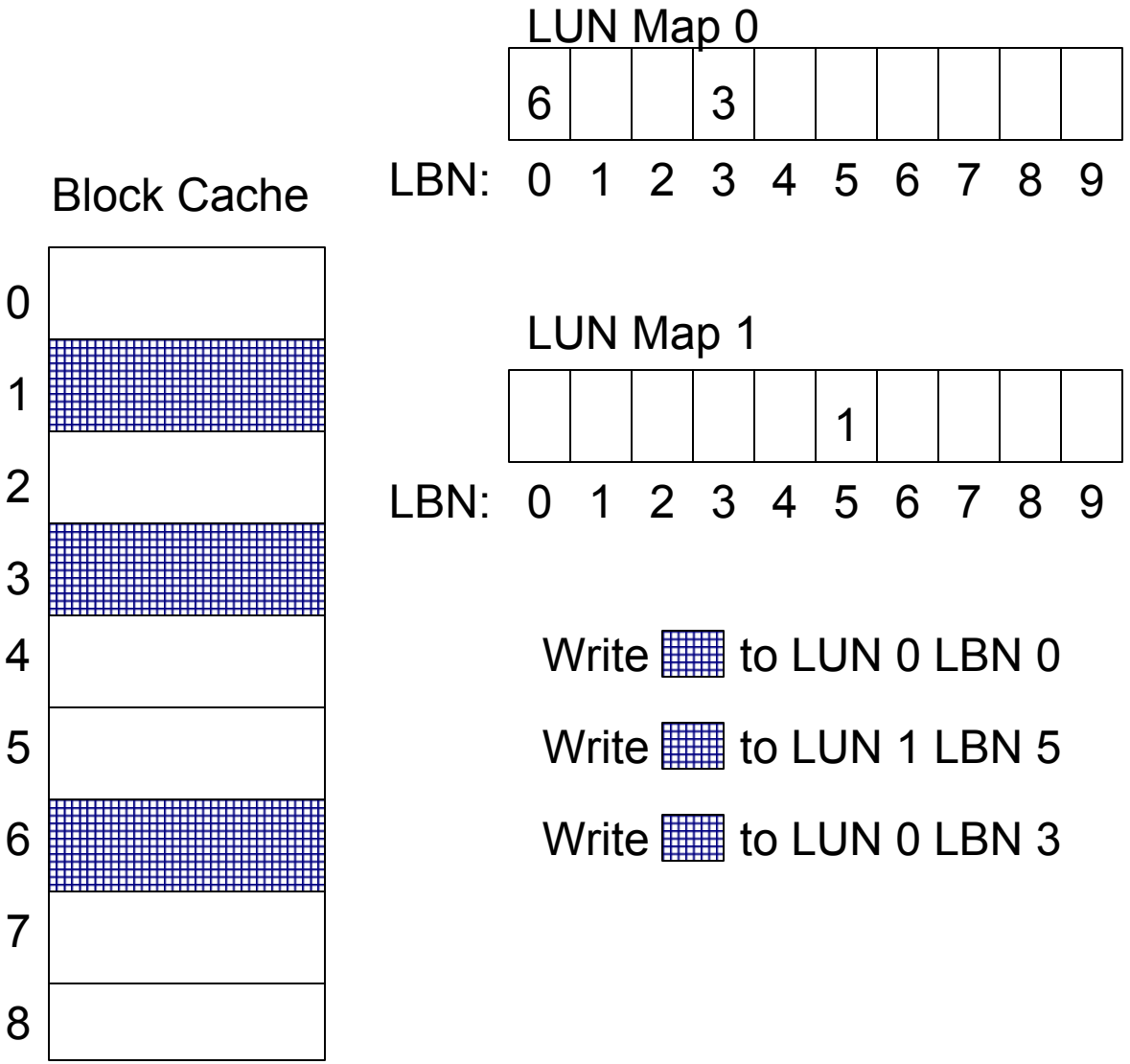
First Write Offset Cache



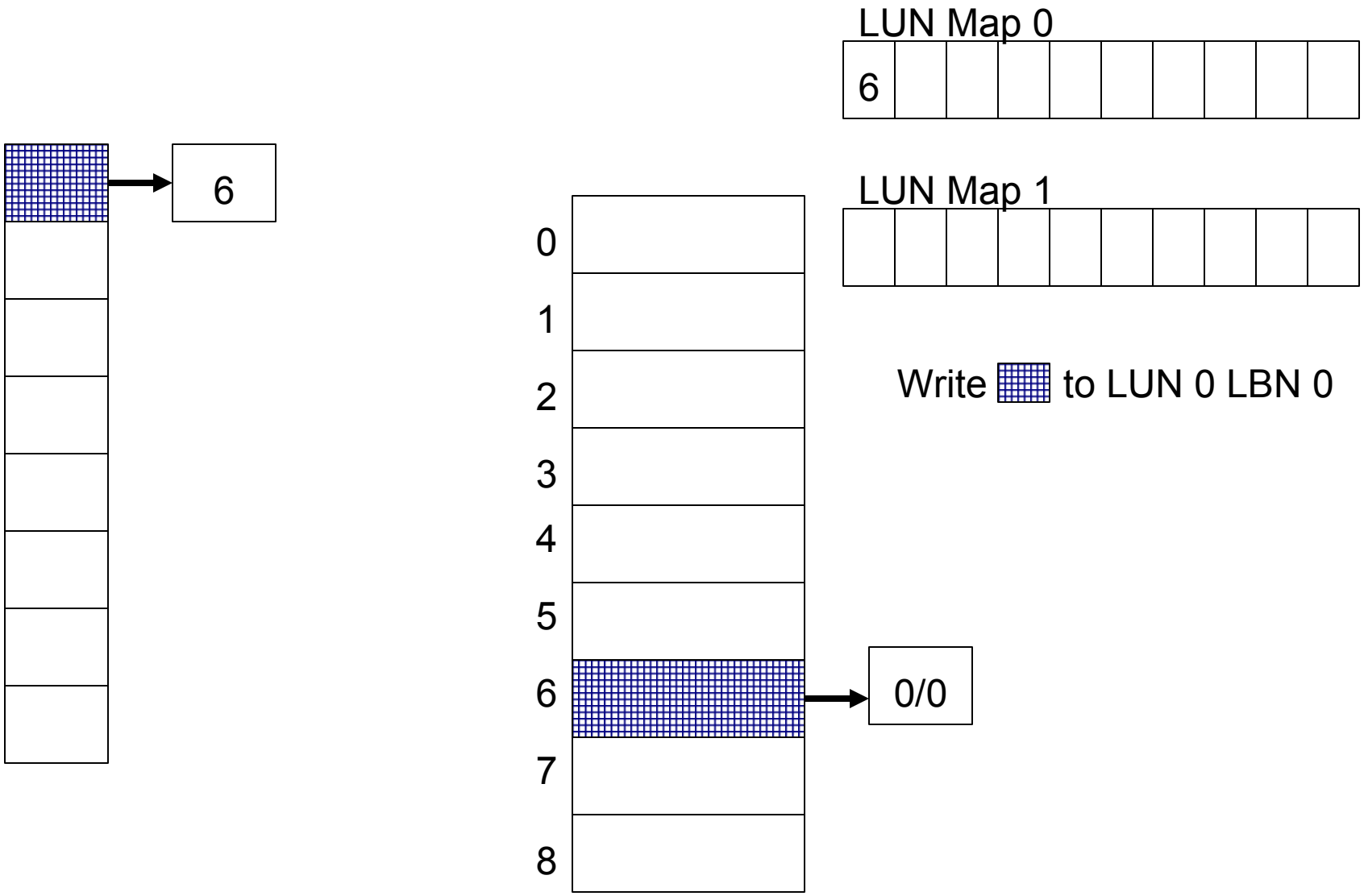
Second Write of Same Content



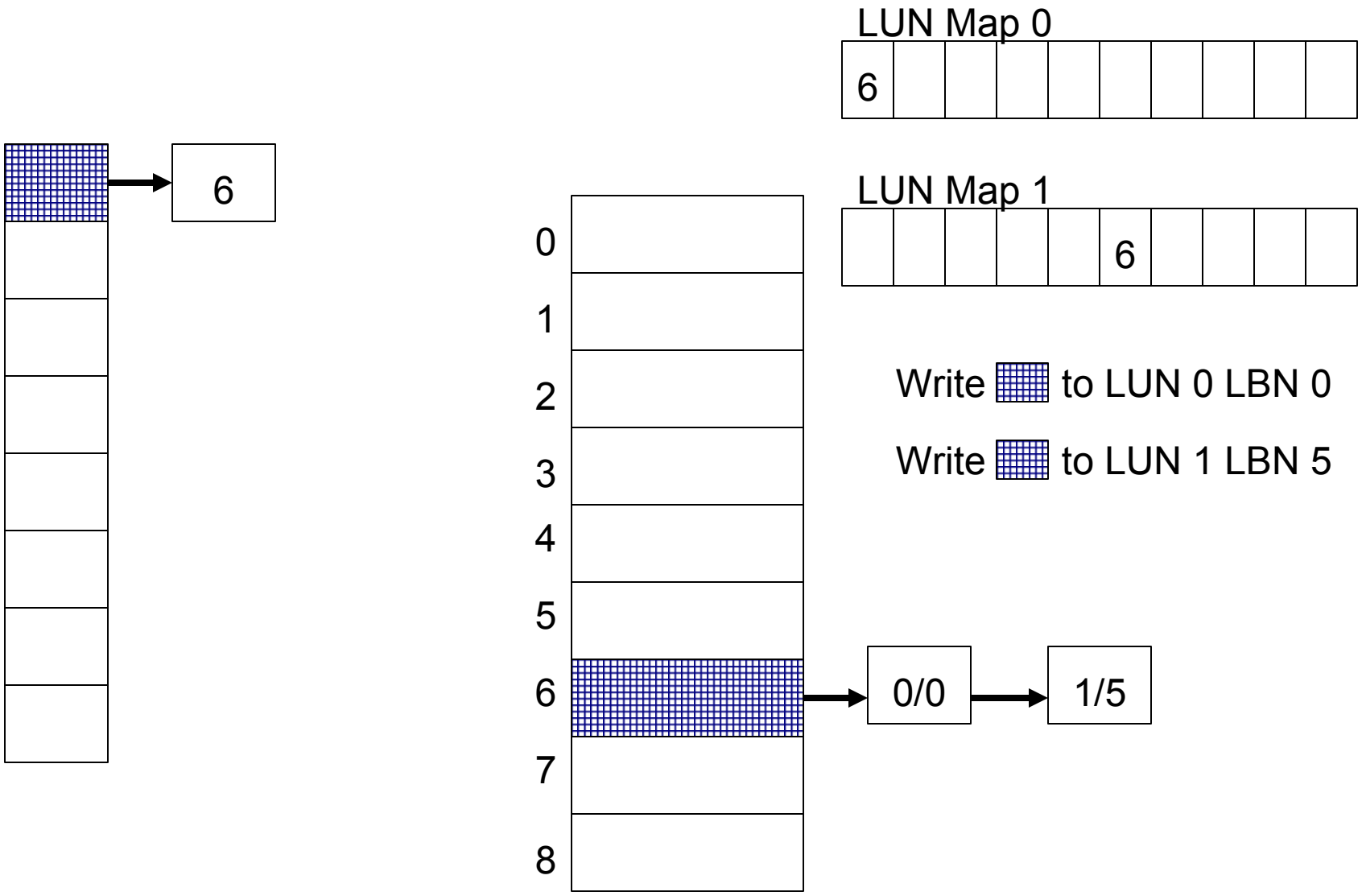
Third Write of Same Content



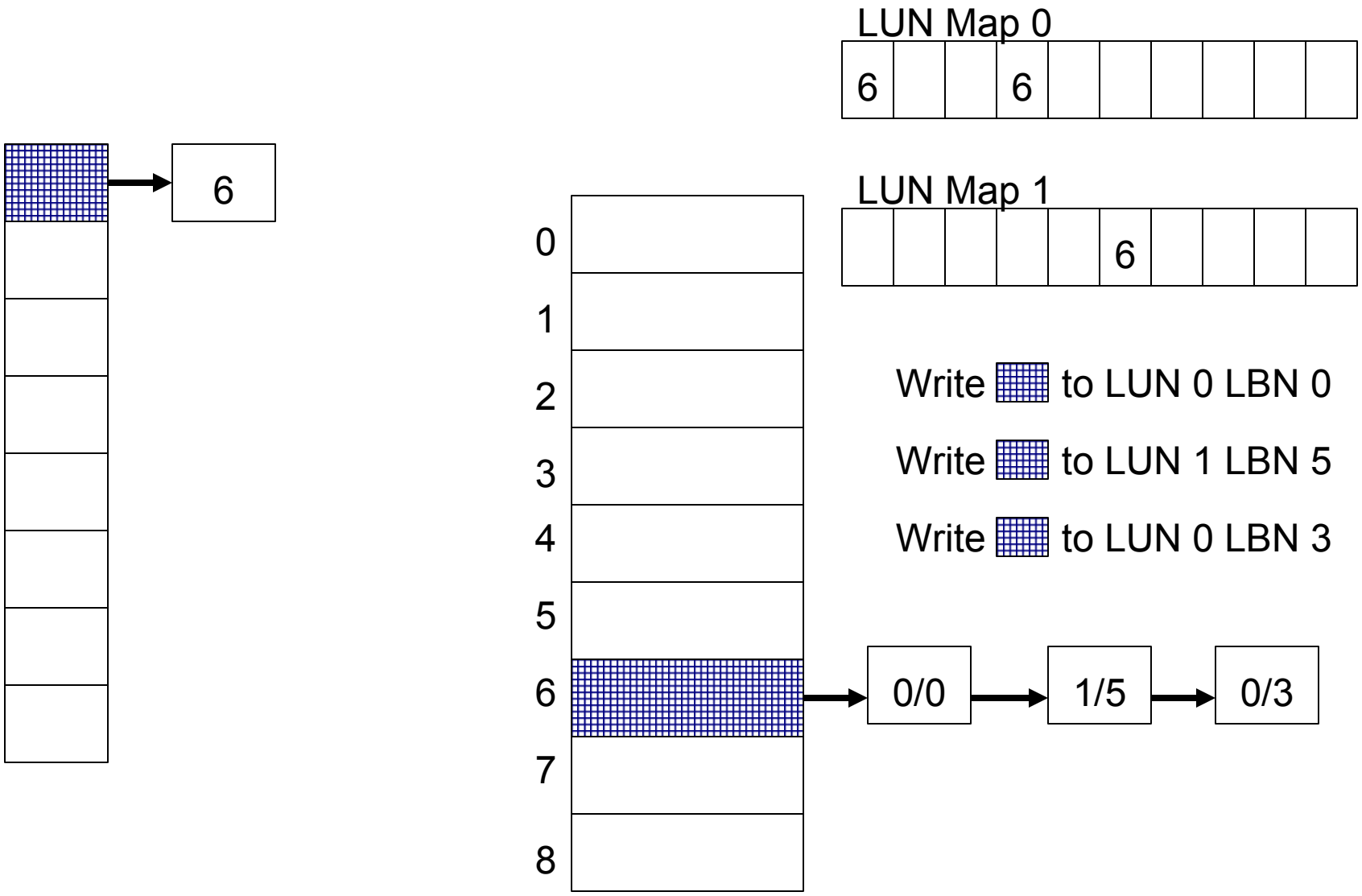
First Write CBBC



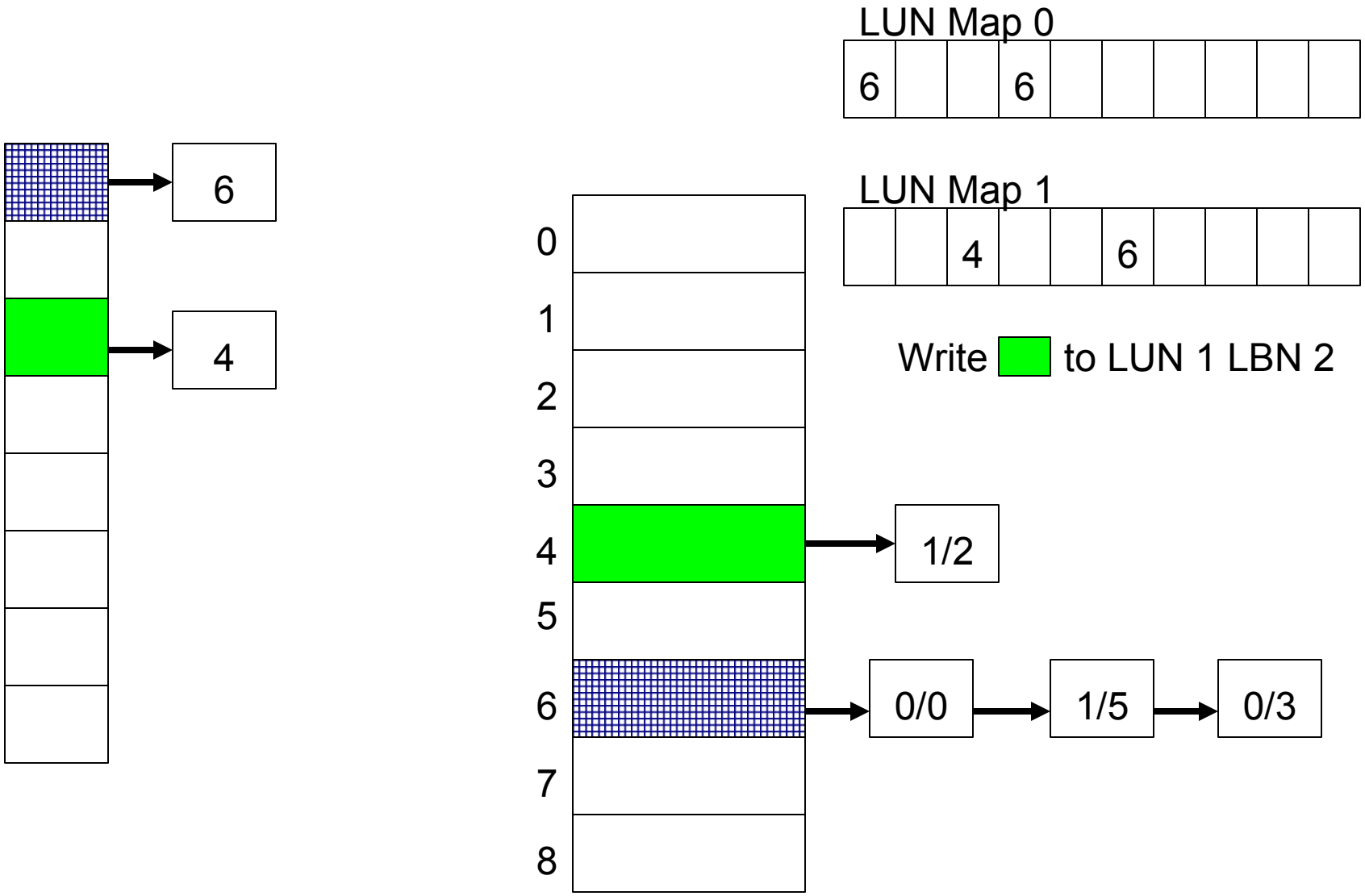
Second Write of Same Content



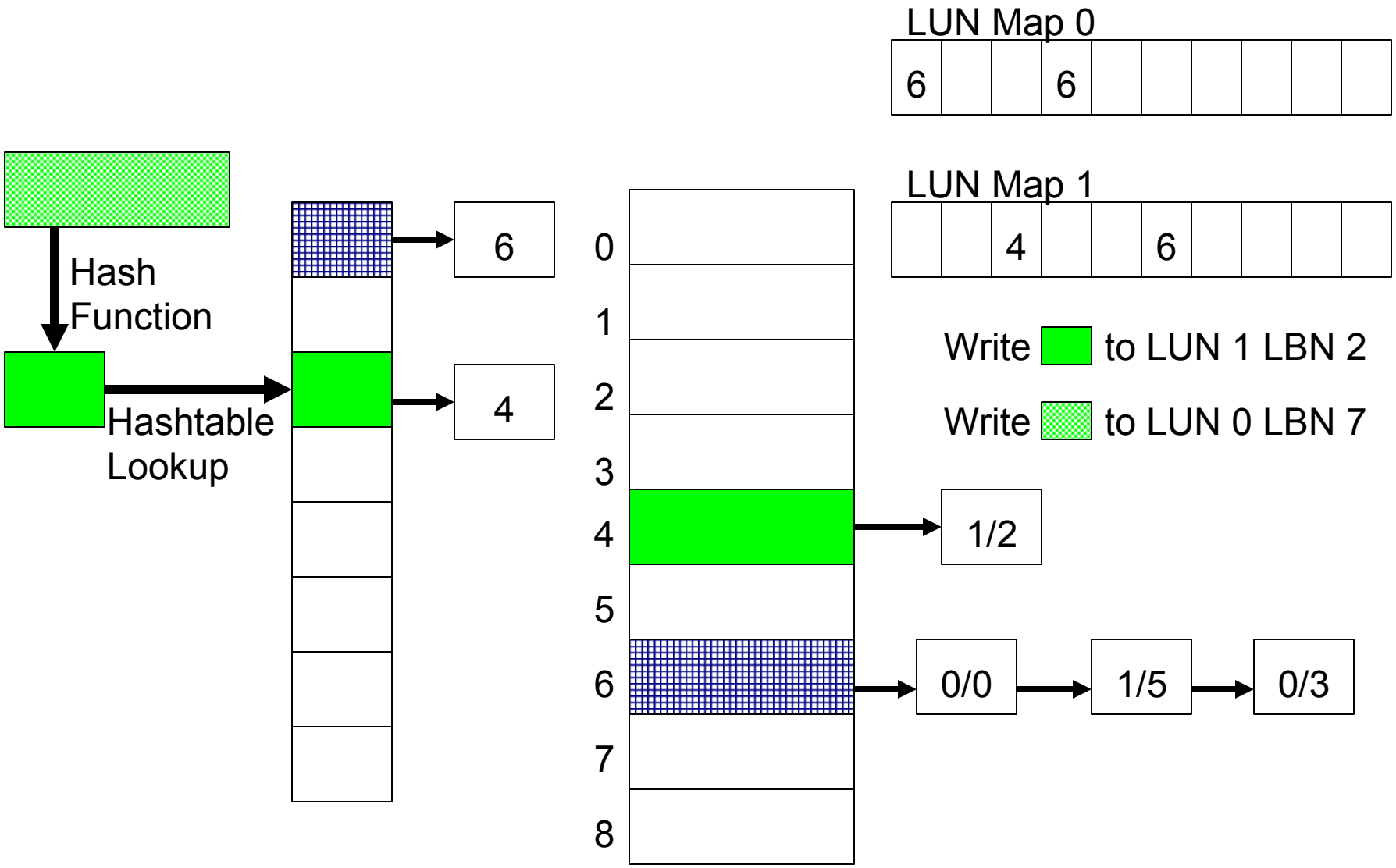
Third Write of Same Content



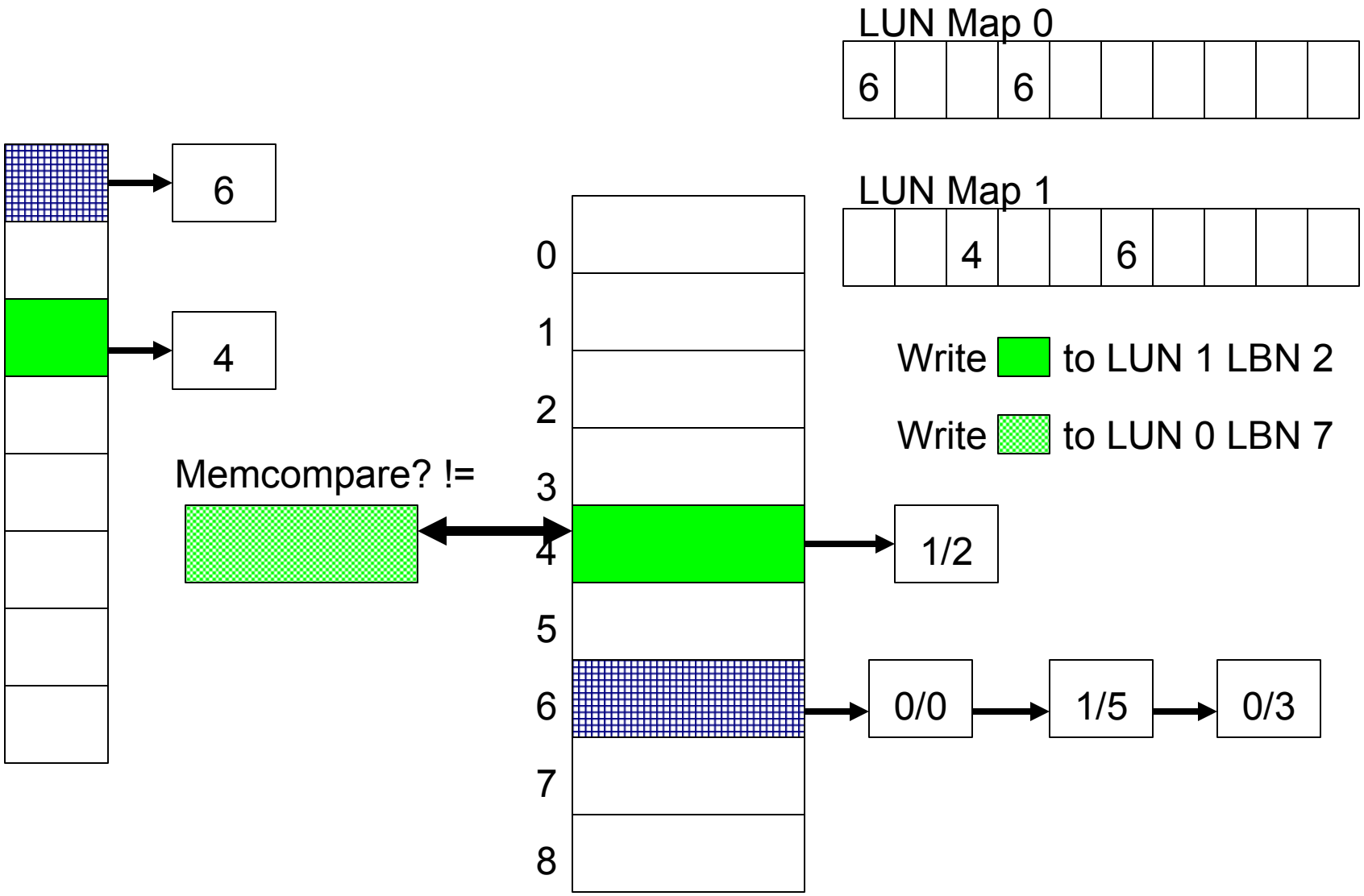
Content Collision Example



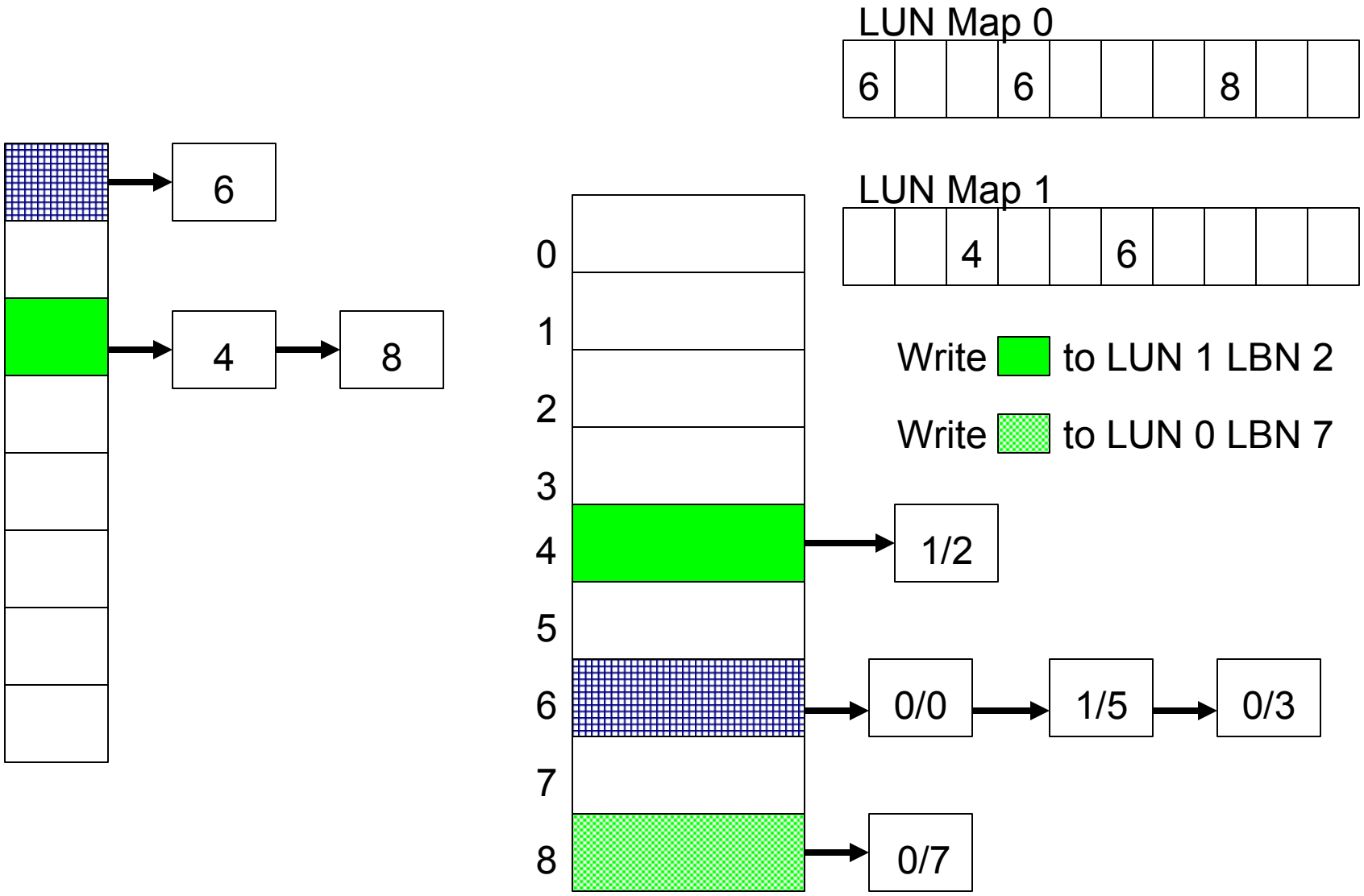
Content Collision Example



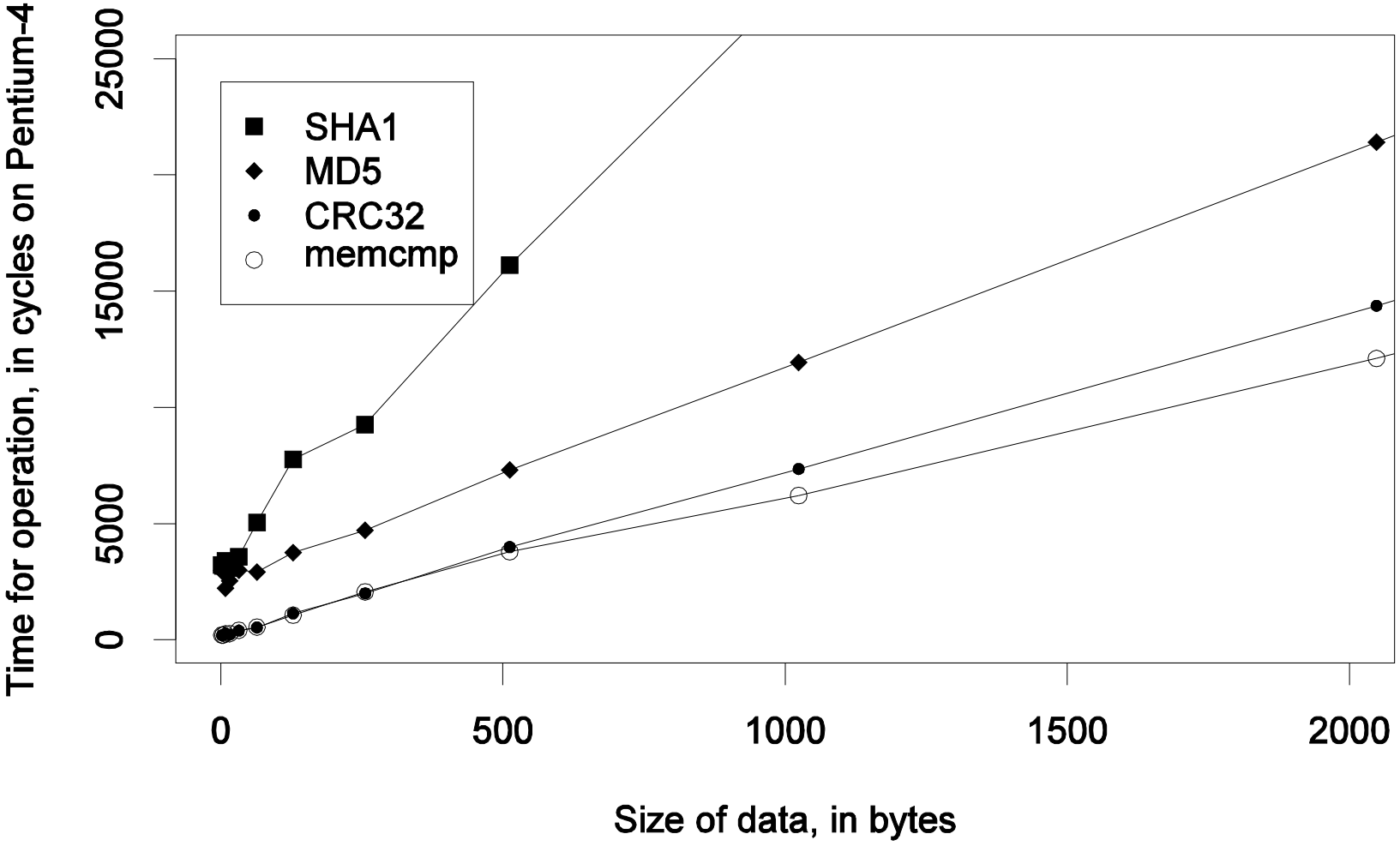
Content Collision Example



Content Collision Example



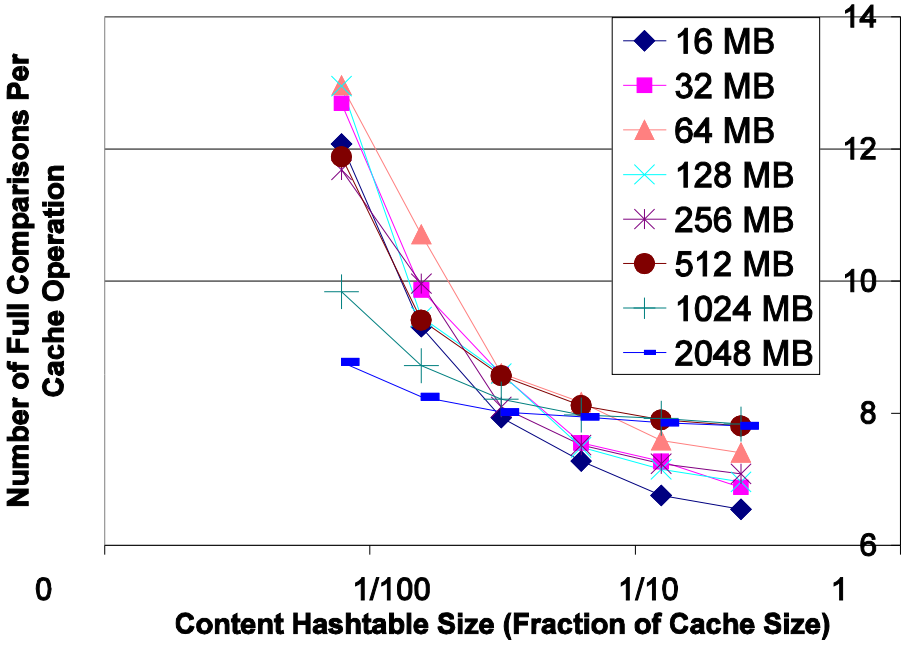
Overhead for Comparisons

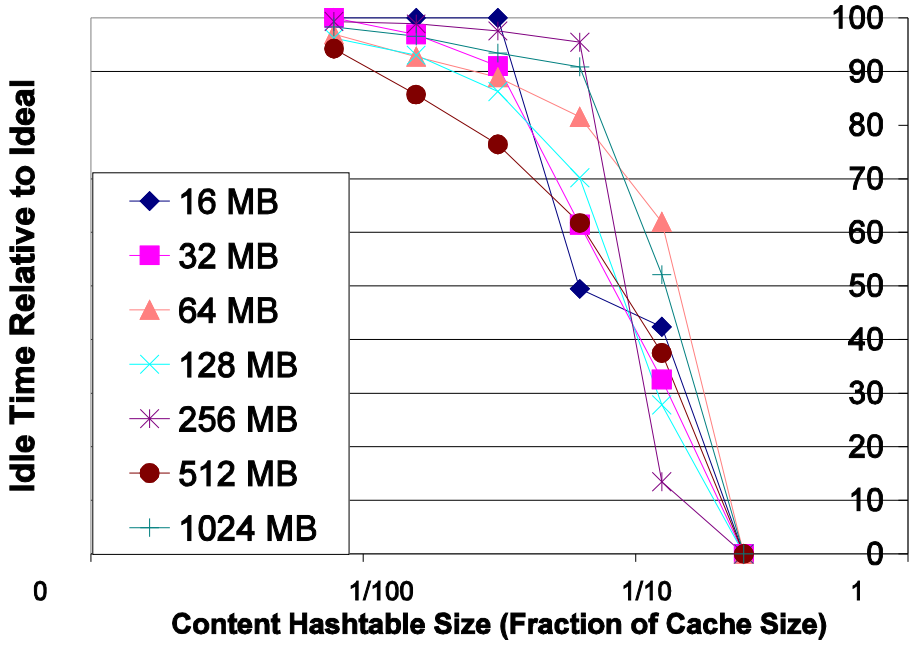


Disk Operation Cost

- Assuming Sequential Transfer Rate of 30MB/s 10 comparisons can be done for every block transferred
- Assuming a random disk access time of 10ms almost 7000 comparisons can be done for every block transferred
- Assume 10 to be conservative

How Big to Make Content Hashtable?

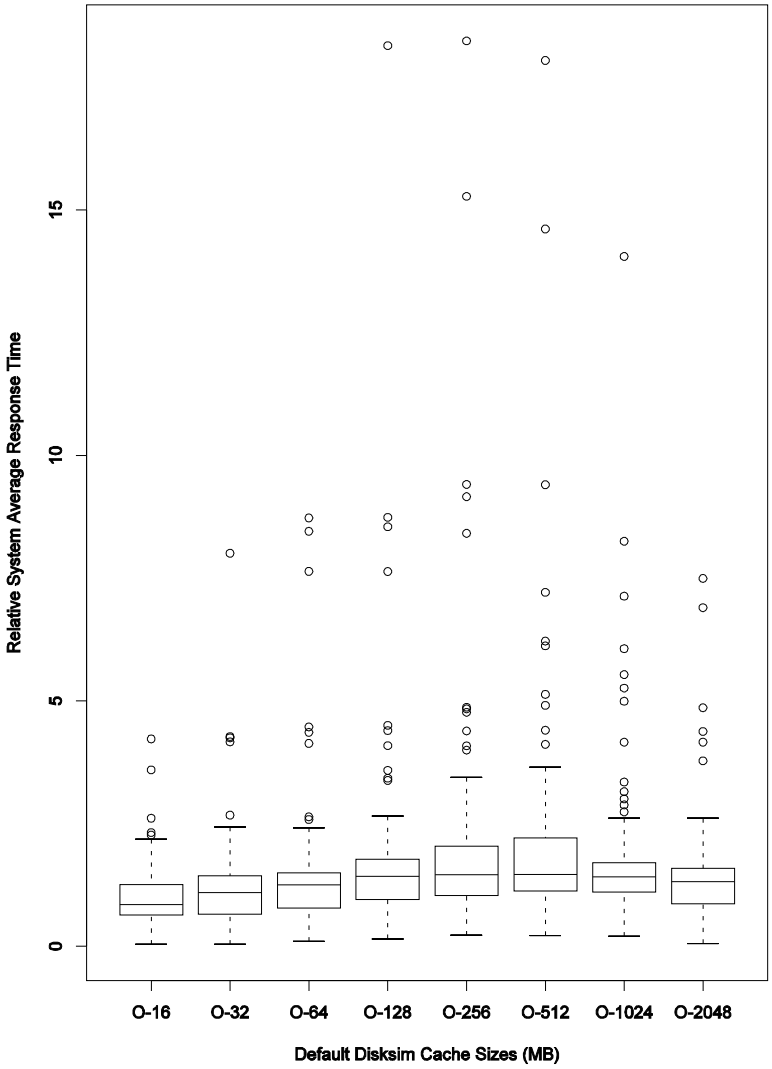


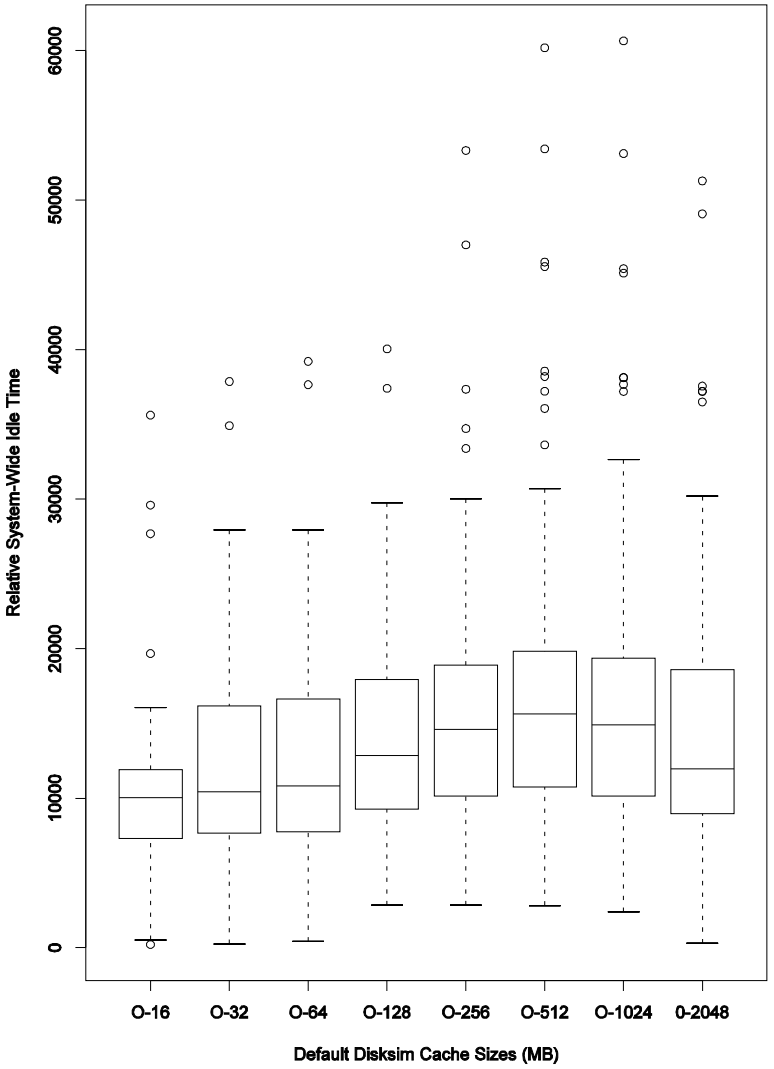


Overview

- Thesis
- Motivation
- System Design
- **Results**
- Related Work
- Conclusion

Disksim System Performance





Overview

- Thesis
- Motivation
- System Design
- Results
- **Related Work**
- Conclusion

Hashing

- Bell Labs – Venti
  - Uses SHA-1 of block as address to store single copy of blocks.
  - Used as backup device
  - No write logging (discrete snapshots)
- Farsite – Microsoft (ICDCS ’02)
  - Big, Distributed File Server
  - Convergent Encryption to Coalesce Identical Files (Very Cool!)
    - Hash a File, Encrypt file with Hash, Encrypt Hash with Pub Key
- Low-Bandwidth File System
  - Semantic Block Chunking
  - Hash of block to save network traffic

Caching Related Work

- Compression caching
  - Compress available cache space to increase effective cache size
- Cache replacement policies
  - LRU to LFU spectrum
    - Orthogonal (content-caching uses a replacement policy)
  - Multi-Queue cache replacement algorithm
    - Keeps blocks around based on their access frequency and time
    - More effective for second level caches

Conclusion

- Content-Based Block Caching is a novel addition which provides several benefits by using content tracking to increase effective cache size

Future Work

- CIMStore (Come for my WIP tonight)
- http://systems.cs.colorado.edu/~cbmorrey

BACKUP SLIDES

Is There Repeated Content?

	Average Operations Per Day (or Trace)				
Disk #	Writes	Repeated Writes	Zero Block Writes	Reads	# of Days
1	1,476,573	1,265,093	66,781	105,227	13
2	1,545,183	1,322,575	75,710	71,898	14
3	1,556,154	1,332,916	72,537	56,449	14
4	780,208	670,227	36,983	82,446	12
5	628,096	540,547	29,431	89,734	3
6	1,432,974	1,227,173	70,582	92,678	14
7	1,174,403	1,006,937	53,712	92,761	8
8	1,251,825	1,071,016	63,778	170,958	5
9	657,768	532,934	69,530	168,617	26
10	603,793	407,582	59,448	418,060	45
win2k	142,932	69,280	2,895	3,087	16

Caching Related Work

- DEMOTE (Exclusive Caching) Wilkes et. al. (USENIX ’02)
  - Adds the DEMOTE operator to multi-level disk caches to improve cache coverage
- Cooperative Caching - Dahlin et. al. (OSDI ‘94)
  - Remote Client Memory to improve File System Performance
- Cooperative Caching - Voelker et. Al. (SIGMETRICS ’98)
  - Prefetching and Caching in a Globally Managed Memory System