Coerced Cache Eviction: Dealing with Misbehaving Disks through Discreet-Mode Journaling

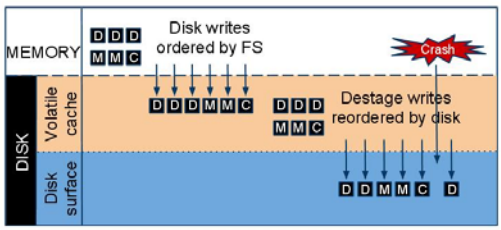
Abhishek Rajimwale, Vijay Chidambaram, Deepak Ramamurthi, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau

Introduction

What happens if the on-disk cache is not honest about writes? What if the cache ignores requests to flush its cache or behaves as a write-back cache when configured as write-through?

Anecdotal evidence from industry experts suggests that some manufacturers, in an effort to boost performance results on various benchmarks, ignore requests to force writes to disk and only push data to the media surface in the background. While this improves performance, file systems can no longer control the order in which writes go to the disk surface. If write-ordering constraints are not obeyed, an untimely crash can lead to inconsistent metadata, garbage data, and even unmountable file systems.

Effect on Journaling



When writes are reordered by the disk, it can lead to loss of file system integrity. In this example of journaling [1], because the commit block has been written to the disk surface, the file system assumes that the transaction succeeded, while in reality a disk block was lost.

Coerced Cache Eviction

We introduce a new method to flush writes to the disk surface despite a misbehaving disk cache. The idea is to generate a flush workload of requests to the disk. This flush workload is constructed such that it will replace some set of the current contents of the disk cache, forcing those items to be written back to the disk surface as desired.

The ideal workload should have a high probability of actually flushing the target set of blocks while inducing a negligible performance overhead. In order to design the CCE flush, we need to understand the underlying disk, in particular its cache.

Cache fingerprinting

We use a simple, self-designed micro-benchmark [2, 3, 4, 5] to fingerprint the behavior of the disk cache. We determine various parameters for a cache, most importantly its size and replacement policy. We use this information to design an efficient and effective CCE flush.

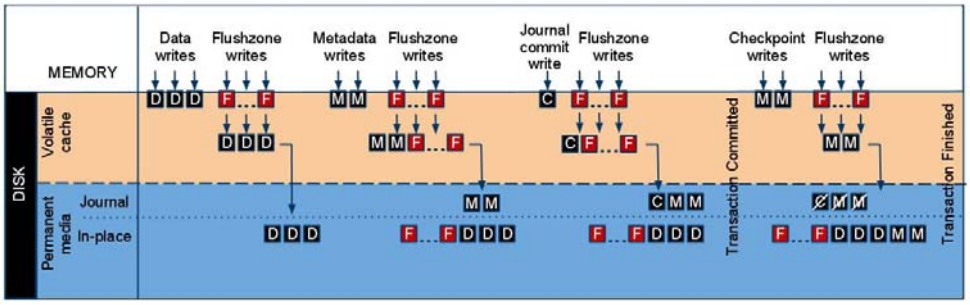
```
while(i < num_trials) {
  write(fd, random_location, 512);
  perform_CCE_flush();
  fsync();

  time_read = read(fd, random_location);
  if (time_read > CACHE_READ_TIME) {
      num_evictions++;
  }
  i++;
}
```

We vary three parameters in each flush workload:

Parameter	Variance in workloads
Num of writes in the workload	1 – 2048
Size of each write	1 – 128
Location of writes	Sequential / Random

We fingerprinted 9 SATA drives and found that cache behavior of disks from the same manufacturer is qualitatively similar across different models. Certain disks allow sequential streams to flush the cache effectively. Some disks identify large sequential streams and choose not to cache them. For these disks, random writes are required, and a large number of random writes are required to evict the cache with high probability.

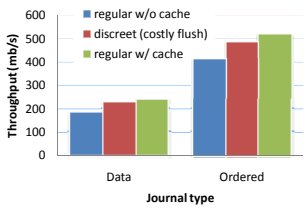
Discreet Mode Journaling



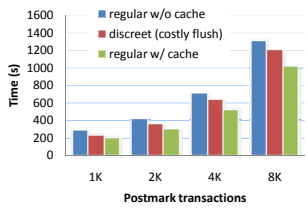
The CCE primitive can be used to guarantee data consistency and provide crash recovery in any file system that requires a specific ordering of write requests for correctness, even in the presence of misbehaving disks. We incorporated CCE into Linux ext3, a journaling file system. Journaling in ext3 requires block updates in a transaction to have a particular ordering. If a disk violates this ordering due to overly aggressive caching, then a crash can leave the file system in one of many possible inconsistent states. Discreet Mode Journaling adds CCE operations at each of the ordering points to ensure that the disk writes are durably performed.

Experimental Results

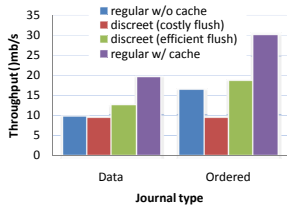
Filebench webserver benchmark



Postmark benchmark (Data Journaling)



Filebench varmail benchmark



We evaluated the performance of discreet mode journaling using several known benchmarks. With most workloads (OpenSSH, Filebench webserver, Postmark), discreet mode achieved noticeably better throughput than the only other safe alternative of disabling the cache.

For the I/O intensive Filebench Webserver benchmark, it performed comparably to the unsafe regular journaling cases, with only a modest 4 to 7 percent drop in throughput. For the Postmark benchmark, discreet mode journaling outperforms disabling the cache for various transaction sizes. The overhead of discreet mode journalling when compared to regular journalling is 25 percent.

The Filebench varmail benchmark repeatedly calls `fsync` causing the file system to flush the disk cache, causing four complete CCE flushes per `fsync`. To improve performance we remove one of the four CCE flushes by implementing a transactional checksum [6] for discreet mode journaling, and increase the transaction size .

References

[1] R. Hagmann. Reimplementing the Cedar File System Using Logging and Group Commit. In *Proceedings of the 11th ACM Symposium on Operating Systems Principles (SOSP '87), Austin, Texas, November 1987.*

[2] J. Schindler and G. R. Ganger. Automated disk drive characterization. Technical Report CMU-CS-99-176, Carnegie Mellon University, 1999.

[3] J. Schindler, J. L. Griffin, C. R. Lumb, and G. R. Ganger. Track-aligned Extents: Matching Access Patterns to Disk Drive Characteristics. In *Proceedings of the First USENIX Conference on File and Storage Technologies (FAST),* Monterey, CA, 2002.

[4] N. Talagala, R. H. Arpaci-Dusseau, and D. Patterson. Microbenchmark-based Extraction of Local and Global Disk Characteristics. Technical Report CSD-99-1063, University of California, Berkeley, 1999.

[5] B. L.Worthington, G. R. Ganger, Y. N. Patt, and J.Wilkes. On-Line Extraction of SCSI Disk Drive Parameters. In *Proceedings of the 1995 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '95), pages 146–156, Ottawa, Canada, May* 1995.

[6] V. Prabhakaran, L. N. Bairavasundaram, N. Agrawal, H. S. Gunawi, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. IRON File Systems. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles (SOSP '05), pages 206–220, Brighton, United Kingdom, October* 2005.