`fscachesim`: A quick-and-dirty client-array cache simulator

Ted Wong

This manual describes how to use `fscachesim`, a quick-and-dirty simulator I wrote to model the behavior of (large) client and array caches that implement various experimental management protocols. In particular, the simulator models protocols that use the DEMOTE operation proposed by myself and John Wilkes [WW02] for transferring read-only disk blocks from client to array caches.

1 Command summary

`fscachesim` *option-flags... array-type client-size array-size trace-files...*

1.1 Option flags

`-b` ***block-size***
Specify the block size in bytes. The default is 4096.

`-c` ***warmup-count***
Specify the warm-up I/O count. After processing the warm-up I/O count, `fscachesim` will reset all cache statistics, but retain the cache contents. The default is 0.

`-d`
Enable demotion of blocks from the client to the array.

`-m`
Read MAMBO-format trace files [UAS97].

`-w` ***warmup-time***
Specify the warm-up time in seconds. After the warm-up time elapses, `fscachesim` will reset all cache statistics, but retain the cache contents. The default is 0.

1.2 Parameters

array-type
Specify the array type. `fscachesim` supports a number of cache models:

LRU: Cache blocks in traditional LRU order.

MRULRU: Cache blocks read from disk in MRU order, and blocks demoted from clients in LRU order.

RSEGEXP: Cache blocks using the adaptive caching models described in the USENIX 2002 paper [WW02], in which we divide the ejection queue into ten exponential size segments. Each cache segment is twice the size of the previous segment, and the largest segment is at the end of the queue. We maintain *ghost caches*, which measure how often read requests would hit in the actual cache if we only cached either read or demoted blocks. We then insert blocks read from disk (and blocks demoted from clients) into a segment in the actual cache based on the hit rate in the read (or demote) ghost.

$$in\text{-}seg_r = \frac{hits_r}{hits_r + hits_d} segs \quad \text{and} \quad in\text{-}seg_d = \frac{hits_d}{hits_r + hits_d} segs$$

NSEGEXP: Similar to RSEGEXP, except that we normalize the insertion segment number by the maximum of the two potential numbers.

{N,R}SEGUNI: Similar to {N,R}SEGEXP, except with uniform size segments.

client-size
Specify the per-client cache size in MB. All clients have the same cache size.

array-size
Specify the array cache size in MB.

trace-files...
Specify the trace files. `fscachesim` creates a simulated client for each input trace file.

2 Input trace file format

`fscachesim` uses plain-text input trace files. Each line in a file corresponds to a single I/O request, and has the following format (where *time-request-issued* is in seconds, and *offset-into-object* and *length-of-request* are in bytes):

time-request-issued object-ID offset-into-object length-of-request

`fscachesim` can also use MAMBO input trace files [UAS97].

3 Example usage

`fscachesim -w 32768 -d LRU 64 64 random`
Simulate a 64 MB client and a 64 MB array system. The client will demote blocks, and the array will cache read and demoted blocks in LRU order. The client will replay I/O requests from the file named '`random`'. `fscachesim` will reset the cache statistics after 32768 seconds of simulated time (as measured by the value of *time-request-issued* on the requests).

`fscachesim -w 32768 -d MRULRU 64 64 random`
Simulate a 64 MB client and a 64 MB array. The client will demote blocks, and the array will cache read blocks in MRU order, and demoted blocks in LRU order. The client will replay I/O requests from '`random`'. `fscachesim` will reset the cache statistics after 32768 seconds of simulated time (as measured by the value of *time-request-issued* on the requests).

`fscachesim RSEGEXP 256 2048 db2-1 db2-2`
Simulate two 256 MB clients and a 2 GB (2048 MB) array. The client will discard blocks that it ejects from its cache. The array will insert read blocks into the cache based on the read ghost cache score. One client will replay I/O requests from '`db2-1`', and the other client will replay requests from '`db2-2`'.

`fscachesim -d RSEGEXP 256 2048 httpd-1 httpd-2`
Simulate two 256 MB clients and a 2 GB (2048 MB) array. The clients will demote blocks, and the array will insert read and demoted blocks into the cache based on the read and demote ghost cache scores. One client will replay I/O requests from '`httpd-1`', and the other client will replay requests from '`httpd-2`'.

References

[UAS97] Mustafa Uysal, Anurag Acharya, and Joel Saltz. Requirements of I/O systems for parallel machines: An application-driven study. Technical Report CS-TR-3802, Dept. of Computer Science, University of Maryland, College Park, MD, May 1997.

[WW02] Theodore M. Wong and John Wilkes. My cache or yours? Making storage more exclusive, In Proc. of the 2002 USENIX Annual Technical Conference (June 2002)