MFS: A Network-Aware Mobile File System

Benjamin Atkin and Kenneth P. Birman
Cornell University, Ithaca, NY
{batkin,ken}@cs.cornell.edu

Abstract

An application running on a host in a wireless network must tolerate a high degree of bandwidth variability if it is to make extensive use of the network. We have designed MFS, a file system for mobile clients [2], which provides support for applications to exchange data over a variable-bandwidth wireless link. MFS is also an example of "modeless adaptation" [1], a more general technique for adapting application network usage to bandwidth variations. Most existing network-aware adaptation schemes [4, 3, 5] are primarily intended for low bandwidth environments, and their adaptation to bandwidth variations is too coarse-grained to make efficient use of all available bandwidth. In contrast, modeless adaptation in MFS makes use of filesystem-specific semantics to adapt gracefully to degradations in bandwidth.

RPC-level adaptation

Client-server applications such as a file system present many opportunities for fine-grained adaptation. An MFS client may initiate concurrent communication on behalf of several processes, and accesses to files result in multiple types of RPCs, such as cache validations and writing back changes. MFS adapts to insufficient bandwidth by allocating bandwidth preferentially among simultaneous RPCs. "Interactive" RPCs, such as fetching files, receive higher priority than "background" RPCs such as writing back changes. This ensures that a process which fetches a file at the same time as another file is being written back to the server will not have to compete for bandwidth.

This approach differs from adaptation in mobile file systems such as Coda [5], which react to low bandwidth by making file updates asynchronous, and only writing them back to the server after a delay. This frees a process writing to files from having to wait for the updates to be transmitted to the server, but can generate interference with later RPCs when the updates are written back. In MFS the lower priority of writes relative to other RPCs ensures writes are only transmitted when there is surplus bandwidth. Our experiments have demonstrated that adding priorities to MFS with and without asynchronous writes improves performance over equivalent schemes which do not use priorities.

MFS also performs prefetching using low-priority RPCs. This ensures prefetches will incur minimal interference with "foreground" activity. When MFS has good prefetching hints for a workload, prefetching results in a considerable improvement, and even when the hints are ineffective, there is only a moderate overhead.

Selective cache consistency

A dangerous side-effect of delayed writebacks is that consistency can be significantly reduced: one client can access a file for which another client holds a delayed update. However, it is possible to improve cache consistency without seriously impeding performance, by making use of information about how files are shared among clients. In MFS, since accesses to files are performed through a central file server, the server can determine which files are "shared" and which are "private". Delaying updates to files which are not accessed by other clients is acceptable. In contrast, stronger consistency for shared files can be achieved by a combination of synchronous invalidation when a shared file is modified, followed by asynchronous writeback. If a shared-but-invalid file is accessed by another client, the server pulls outstanding changes from the client holding the delayed update, before supplying the file to the new client. This division of files by access pattern allows MFS to restrict more expensive cache consistency to files which require it.

References

[1] B. Atkin and K. P. Birman. Evaluation of an adaptive transport protocol. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom 2003)*, San Francisco, California, Apr. 2003.

[2] B. Atkin and K. P. Birman. MFS: an adaptive distributed file system for mobile hosts. Submitted for publication, Mar. 2003.

[3] A. D. Joseph, J. A. Tauber, and M. F. Kaashoek. Mobile computing with the Rover Toolkit. *IEEE Transactions on Computers: Special issue on Mobile Computing*, 46(3):337–352, Mar. 1997.

[4] B. D. Noble and M. Satyanarayanan. Experience with adaptive mobile applications in Odyssey. *Mobile Networks and Applications*, 4(4), 1999.

[5] M. Satyanarayanan. The evolution of Coda. *ACM Transactions on Computer Systems*, 20(2):85–124, May 2002.