Matthew Koken

COEN 177

Assignment 5

ARC: A Self-Tuning, Low Overhead Replacement Cache

ARC is an Adaptive Replacement Cache designed for use in paging situations that have uniform page sizes. Being self-tuning, ARC revises how it chooses to evict pages from a cache based on the workload history. For each page request, ARC succeeds in having a constant time complexity, no worse than LRU. It also avoids cache congestion by single use pages, where other page replacement algorithms may unnecessarily evict more useful pages than low use pages. ARC is not a page replacement specific algorithm, it is a general algorithm that applies to general caching scenarios, and can be applied to situations such as page replacement and file caching.

ARC utilizes a two list method for caching. One list is used to hold pages for frequently and recently used pages, the other to hold recently used pages, such as single use pages. Pages that are likely to be used quite often and would likely sit at the front of a cache in methods such as LRU, taking space from other less used pages. The bottom of each list contains a ghost list, used to keep track of recent evictions – the ghosts of pages recently evicted that may still haunt the system. The ghosts do not hold the resource data, instead they hold only the metadata of the recent pages. Instead of being fully evicted on removal, the evicted pages are pushed onto the ghost lists in order to keep track of workload, and be brought back in should they be requested. Combining LRU and LFU for both lists, ARC can better balance the evictions of pages. By having two lists, important or often used pages are protected and can be balanced from encroaching on the pages needed by others. By using this, page faults can be reduced, improving upon LRU while also keeping efficiency and ease of implementation in mind.

Overall, ARC provides good improvements over LRU caching in terms of page faults without much increase in difficulty of implementation or complexity of runtime. Since the ghost lists are quite small and hold little data, the overhead increases are also minimal. As a general caching system, ARC will be beneficial over LRU in many applications. However, it does not seem to be a likely replacement for page replacement algorithms in virtual memory. Keeping constant time complexity, ARC cannot perform to the level needed for fast and efficient virtual memory paging. While the improvements on page faults and smart usage of storage is nice, they do not outweigh the still relatively high cost on each page request. Instead, it will fit applications where LRU was also applicable. Over LRU, ARC is likely to provide many benefits with caching for applications such as file system storage, on a storage device such as a hard drive or solid state drive. Here, ARC would be greatly beneficial, and currently sees use in storage controllers.

Works Cited

Nimrod Megiddo , Dharmendra S. Modha, ARC: A Self-Tuning, Low Overhead Replacement Cache, Proceedings of the 2nd USENIX Conference on File and Storage Technologies, March 31-31, 2003, San Francisco, CA