Functionality Composition Across Layers in a Storage System

Florentina I. Popovici, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau
Computer Sciences Department, University of Wisconsin, Madison
{popovici, dusseau, remzi}@cs.wisc.edu

1 Problem Statement

To ease development, modern systems are often built in layers. However, it is common for the same functionality to be duplicated at several layers. Let us consider, as an example, a Web server that is deployed in a virtualized hosting environment. In this setup, caching, prefetching, and scheduling decisions are made at several layers: application layer (web server), kernel, virtual machine layer, RAID system, and finally at the disk.

This duplication results in suboptimal system behavior. For example, in caching, the same block can be cached simultaneously at the kernel level and RAID, thus resulting in an underutilization of the available resources. The desired situation is exclusive caching. Similar problems can occur in the case of other functionalities. Other studies ([1], [2]) have also noticed this problem, though, they concentrated mainly on caching, and they considered the presence of only two layers.

Because of its complexity and influence on the overall performance of the storage system we focus on IO scheduling, and then plan to extend observations to other functionalities, such as caching, prefetching, and layout.

We are pursuing the answers to the following questions:

- How is the performance of the system influenced by a certain combination of scheduling decisions? For example, if a layer implements the classic C-LOOK algorithm, and another layer underneath implements C-LOOK also, what is the final performance of the system? Would the answer change if the algorithm is SATF?
- What is the best place to implement an algorithm? For example, if the SATF algorithm can be placed at several layers, where is the best place to have it?
- How are QoS schedulers influenced by a hierarchy of layers? How 'good' is the decision of a QoS scheduler at an upper layer, when requests could be reordered at the lower layers?
- If a layer makes a bad scheduling decision, can a scheduler at a layer underneath 'fix' the bad decisions?
- What are the parameters that influence the performance of an algorithm? For example, how sensitive are schedulers to queue length variations, and request sizes.
- How would workload characteristics influence the answers to the above?

2 Approach

We study the answers to the above questions by building a simulator environment. The simulator is able to simulate the composition of a configurable number of layers, each implementing a specified scheduling policy.

We plan on exploring combinations of policies, such as: FCFS (no reordering), SSTF (reorders requests according to the shortest seek time relative to the block distance), C-LOOK (elevator-like variant of SSTF, that reorders requests only in one direction), SATF (reorders requests according to the shortest access time), anticipatory scheduling (reorders requests while considering the process that issued the requests and what is the expected process think time and access time).

To this date, we have implemented a scaled down version of the simulator, that supports a subset of the schedulers we plan to study. We have a set of preliminary results that uncover the answer to some of the questions. For example, our results show that SATF schedulers should be placed as close to the disk as possible, and that queue space should rather be given to a scheduler at one layer, rather than split it between two schedulers at two layers.

References

[1] Z. Chen, Y. Zhang, Y. Zhou, H. Scott, and B. Schiefer. Empirical evaluation of Multi-level Buffer Cache Collaboration for Storage System. In *Proceedings of the 2005 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '05)*, Banff, Canada, June 2005.

[2] T. M. Wong and J. Wilkes. My Cache or Yours? Making Storage More Exclusive. In *Proceedings of the USENIX Annual Technical Conference (USENIX '02)*, Monterey, California, June 2002.