The Design and Implementation of a Log-Structured File System – Critique

- Vidit Jan (862394973)

Summary:

This article was published in 1991 to propose a file system which is able to capitalize on the trend of increasing memory sizes to alleviate the CPU speed bottleneck. The Sprite LFS framework is able to perform faster writes by sequentially writing the data, agnostic to the directory structure. The key aspect is that a larger memory will allow for larger file caches, absorbing a higher proportion of the read requests. Larger memory can also serve as write buffers, allowing the Sprite LFS to perform more efficient writes. The authors have used several concepts such as segments, segment summary, inode map, checkpoint region and segment cleaning to facilitate the new framework. The article also provides a comprehensive evaluation under several use cases, in addition to a comparison with the prevalent file systems.

Strengths:

Firstly, the idea itself is very innovative. Using file cache to create a 'log' of changes and writing only when the log is of a certain size (one segment) allows for very efficient writing. Also, using segment framework helps in effectively managing the system. To further improve the performance, a segment summary is also used. This is written as part of each segment, and enables quick lookup of the segment contents. Additionally, segment usage table provides a low overhead way to know the segment utilization, and select the appropriate segment to clean. The authors considered two types of segments i.e. 'hot' and 'cold'. This helps in understanding real world performance, and the cost-benefit based segment cleaning is able to perform much better than the greedy approach. In addition to the aforementioned, using checkpoints helps greatly with fault tolerance. Overall, the LFS system is able to perform much better than FFS without significant implementation complexity.

Weaknesses:

Although the article is very innovative, certain aspects were lacking detail. The authors described the concept of partial-segment writes. However, more detail here would have been helpful. Another such concept is using a hierarchy of logs. Also, the crash recovery and roll-forward segments were not intuitive. Considering only files of one size for evaluating was not representative of real world use case. Additionally, some discussion around specific application / user actions would have really added more context.

Other comments:

The LFS file system is truly innovative, and most of the article was intuitive. With the recent popularity gained by SSDs, this file system seems extremely suitable for wide scale adoption. However, since the SSDs are not constrained by mechanical movement, a refreshed comparison between various file systems would be really helpful.