Rosenblum, M. and Ousterhout, J.
The Design and Implementation of a Log-Structured File System
ACM Transactions on Computer Systems, Vol. 10, No. 1, February 1992, pp. 26-52.

1. What is the technology motivation for a log-structured file system? What is the workload motivation? Why do existing file systems perform poorly under these circumstances?
2. How does one read a file in LFS? How does LFS know where the inode map is on disk? How does LFS avoid performing extra disk reads?
3. A log-structured file system must have free space to write the log. How would a threaded log work? Why is it a bad idea? How would compaction with a single log work? Why is it a bad idea?
4. How can these two ideas for free space management be combined? How should segment size be chosen?
5. What happens during segment cleaning? How can the cleaner know whether or not a particular block is live? (Why is it possible for there to be multiple segment summary blocks per segment?) Why is a version number useful?
6. How is write cost defined for a log structured file system, if "u" is the utilization of segments being cleaned? What is write cost for FFS? What does Figure 3 show?
7. There are a number of policy questions regarding segment cleaning.
   We'll investigate which existing segments should be cleaned and how should the live data be grouped? Which segment does a simple greedy policy choose to clean? What was the idea behind the "Hot-and-Cold" policy? Why does "hot-and-cold" not work well with the greedy policy? Why is it less beneficial to clean a "hot" segment?
8. What is the idea of the cost-benefit policy? According to Figure 6, at what segment utilization are "hot" segments cleaned? At what utilization are "cold" segments cleaned?
9. When system crashes, updates to disk may be lost. How does FFS (or ext2) handle a system crash? (We'll talk more later about ext3 and other journaling file systems.) LFS uses checkpoints (consistent states of the fs) with a roll-forward (to update since last checkpoint). What are the two steps for creating a checkpoint? Why are two checkpoint regions needed? What are the pros/cons of having a longer time between checkpoints?
10. What needs to be updated during roll forward?
11. How does LFS perform on small-file operations (create and delete)?
12. For large files, when does LFS perform better than SunOS? The same? Does it ever perform worse? What is logical versus temporal locality?
13. Conclusions?