Paper summary

The paper discusses the shortcomings of the 'old' 512-byte Unix file system for applications requiring high throughput such as VLSI, image processing. The 'new' Berkeley Fast File System, released with the 4.2 Berkeley Distribution, addresses this problem by improving reliability(atomicity in critical file systems) and throughput(block size from 512 to 1024 bytes). The only modification in the underlying implementation is done and retains the interface of the old system. To avoid internal fragmentation, the concept of 'fragment' is introduced. For minimizing seek latency, data blocks are in the same cylinder. A global policy decides the location of new files to avoid crowding a single cylinder. Some enhancements like Long file names, File locking, symbolic links, and quotas are also suggested at the end.

Strengths

1. Easy to follow, like the UNIX paper.

2. It tries to address most of the edge cases.

Weaknesses

1. Very abrupt ending, no conclusion.

2. The design was "rotationally" optimized. It won't work on memories like flash and, in hindsight, SSD.

3. Since the paper focuses on saving memory, the assumption of having 5-10% free space available is a bit vague.

Comments for author

1. The decision for no deadlock detection could have been more justified in the file locking section.

2. Benchmarks for different scenarios like frequent deleting data, large file size, or a large number of files could have been better than the older file system.

3. The complexity of saving space worth the overhead?