Extended Model Answer

Operating Systems (Part IA) 2005 – Paper 1 Question 11 (SMH)

This question concerns the Unix V7 case study; in particular the first section on the filesystem.

The Unix V7 filesystem

A diagram here should show the layout on disk; for the V7 filesystem this is essentially the superblock followed by the inode table followed by the data blocks.

The superblock holds filesystem metadata such as logical block size, size of file system, mount status, number of inodes, start of free-inode list, start of free-block list, etc. Storage is managed by means of these free lists which contain inodes (respectively) blocks which have not been allocated. When a new inode/block is needed, the head of the relevant list is chosen. When inodes/blocks become free (e.g. after a deletion), they are returned to the relevant list.

Description of a Unix inode

A diagram here should show the hierarchical use of indirect blocks, etc.

Unix holds file (and directory) meta data in inodes. Each inode holds information such as the ower of the file, the time it was created (and modified, and accessed), the size of the file in bytes, the access permissions, and the location of the file data blocks on disk.

The addresses of the first n (e.g. 10) disk blocks are held directly in the inode, followed by the location of a single indirect block, followed by the location of a double indirect block, followed by the location of a triple indirect block.

Indirect blocks represent a trade-off between the speed of block location and the size of an mode. Since many files are small, having too many "direct" pointers in the mode would waste space. Using (the three kinds of) indirect blocks allows very large files to be supported, while not penalising small ones.

Largest file size estimate

Precise answer unimportant, but means of estimate should incorporate direct, indirect, etc blocks.

Assuming block size of 512 bytes and 4 byte block addresses, each indirect block holds 128 pointers. Hence we expect a max file size of approx $(10 + 128 + 128^2 + 128^3)$ blocks.

Improvements to UNIX V7 filesystem

2 marks each for plausible improvements (mine below are from the FFS).

Reliability could be improved by having multiple copies of important data structures (e.g. superblock, root inode).

Performance could be improved by changing the layout so that inodes and data blocks are not so far apart; it could also be improved by attempting to allocated contiguous blocks on disk to contiguous parts of a file (or to related files). Could also use a larger block size to reduce the impact of direct blocks, although would also then need to take into account internal fragmentation (e.g. via using explicit fragment blocks).