University of Texas –Fall 2012

CS 439 H
Homework # 5 – Due Nov 12 @ 11:59PM

Problem 1

Consider a disk containing 1024 tracks, numbered from 0 to 1023, and where the number of sectors n per track t is given by the formula:

$n(t) = 1024$ if $t < 512$, and
$n(t) = 1024 + 50t$ otherwise

For a sector size of 512 bytes, and given that the disk has 4 double-sided platters, compute the capacity of the disk in bytes.

The capacity of the disk is computed by finding first the number of sectors per platter. This is given by $1024 * 512 + \Sigma_{t = 512, 1023} (1024 + 50t) = 1024 * 512 + 1024 * 512 + \Sigma_{t = 512, 1023} 50t$.

The above expression evaluates to: 20696576 sectors.

The size of the disk is thus 20696576 * 4 (plattters) * 2 (double-sized) * 512 (bytes/sector) = 80.846 GB.

Note: The equation cannot really represent a plausible disk. Often, you will find that approximations are made because it is not easy to represent an actual disk with a closed formula.

Problem 2

A file system has a disk block of 8192 bytes and uses an indexing scheme where an index node contains 10 direct indexes, 1 indirect index, 1 level-2 indirect index, and 1 level-3 indrect index. Compute the amount of disk blocks required for the inode to represent a 1-GB file on a 64GB disk.

The problem does not specifically give the maximum disk size, so we have to resort to assumptions about the size so that we can compute the number of bytes that are necessary to represent an index node. In this problem, we have a 64GB disk so this should give us an indication of the reasonable size. Consider that a disk block is 8KB (note, the disk block size is different from the sector size, because the former is defined by the file system software while the latter is defined by hardware). It is safe to assume that 4 bytes will be sufficient to represent the inode index as this number can represents files that can fill the disk.

The direct indices will only represent a file with 80KB. The first index consumes 1 disk block (8KB) to represent a file of size 2048 pointers * 8KB = 16MB. The second level of indexing consumes 1 + 2048 disk blocks to represent a file of up to 16MB * 2048 = 32GB. Thus, we will not need the third level index. To represent 1GB, we will roughly need 63 2nd level indexes each pointing to a disk block of inodes. So, in total, we need 1 block for the first level index and 1

block for the second level indexes and 63 blocks for the indexes pointed to by the second level block, i.e. 65 blocks total (66 if you want to count the block containing the inode).

Problem 3

A "lease" is a lock on a file for a specified duration. When a process acquires a lease on a particular file, the operating system grants a lock to the process if it is not already held by another process. After the duration of the lease expires, the operating system revokes the lock and notifies the process via an upcall (signal). Describe a situation where the use of leases is superior to traditional locks.

Leases are superior to traditional locks in a situation when the process that acquires the lock fails. In this case, a lease would expire allowing the rest of the processes to access the file. It is also useful in a situation where a process acquires the lock but fails to release it (because of a bug). In this case, the lease expires and allows other processes access to the file.

Problem 4

You are a member of a design team to build a backup tool for a UNIX system. One of your colleagues who did not attend CS 439H suggests that the tool can traverse the directory structure, marking all files that have been modified since the last backup by inspecting their timestamps. Then, all such files are then dumped on tape. Another colleague pleads with you to stop this insanity and design a better approach. Identify what is the problem with the proposed approach and outline a better decision mechanism to identify the files that need to be backed up. (hint: man cp).

It is possible to create new files in UNIX with a time stamp that equals that of an existing file (by cp –p src dest). In this case, the scheme outlined in the problem can fail to realize that a file has been added and needs to be backed up. To fix this problem, several approaches are possible. For example, a backup system can maintain a copy of the directory tree and uses it in addition to the proposed scheme to have correct backup operation, potentially augmented with a checksum computed either at a file level or a block level to quickly detect changes in the data itself. Another solution is to use cp –upr / /dev/tape0 where tape0 is a device, or using the command dd.

Submission Instructions

To submit your homework, typeset your answers into a document using whichever editor you prefer. Make sure that your code works! Send a note including a PDF file of your solution with the subject: UT439H:HW5. This will be read by a machine so please be precise. Please send the file to mootaz@us.ibm.com.