OS File Systems Homework

Taylor B. Morris
December 13, 2016

1. One benefit of contiguous allocation is that read requests are significantly faster for requests of sequential blocks as compared to requests of non-sequential blocks. Additional benefits is that it would be more memory efficient to keep track of all the blocks of a file over separated blocks, as the system then only has to track one range per file, instead of several. Contiguous allocation is rarely use because files are commonly deleted and resized, meaning that a contiguous allocation would end up wasting space - you would need a contiguous block of exactly the right size in order for the contiguous allocation to work and not be wasting space - i.e. say you have 200 blocks of memory free in a row, but have 190 blocks of information to store there. Now, you only have 10 blocks of memory free in that location. As that free value gets smaller, nothing will be able to fill the free space. Repeated several times throughout the drive, you could end up with hundreds of blocks of wasted memory. Additionally, contiguous allocation requires there to actually be contiguous blocks, making it difficult to store information as the drive gets full.

2. File Types:

   (Type 1) Regular file: The normal file time. .txt files are an example of this, as are most user files and executables.

   (Type 2) Directory: the file that holds the directory structure. Contains information about what files are in that directory.

   (Type 3) Character device: files which offer non-buffered access to hardware

   (Type 4) Block device: files which offer buffered access to hardware

   (Type 5) Named pipe: a file which makes one program's output another programs input - has two "ends" which can be opened from, a read end and a write end.

   (Type 6) Socket: a file which acts as a network connection.

   (Type 7) Symbolic link: a soft link to another file. If this link is deleted, the file it points to won't be, unlike a "hard-link"

   Data blocks associated with directory files are as follows

Bytes	0-3	4-5	6-6	7-7	8-(8+N-1)
Field	Inode	Size of Entry	Name length	file type	name

3. Assuming typical inode structure with 12 direct pointers, 1 indirect, 1 double indirect and 1 triple indirect. First, the 12 direct pointers give us $12 * 2^{10} = 12288$ bytes. Then, the indirect pointer gives us $\frac{1024^2}{4} = 262144$ bytes. Next, the double indirect pointer gives $\frac{1024^3}{4^2} = 67108864$ bytes. Finally, the triple indirect gives $\frac{1024^4}{4^3} = 17179869184$. That leaves us with a total of 17247252480 bytes.

4. After the sequence of operations, the following is true:

(a) The superblock would have to be modified to update the number of free blocks and inodes.

(b) The block bitmap would have to be modified to indicate the now used blocks

(c) Since we create a new file, the file has an inode assigned to it, and, thus, the inode bitmap must be modified to mark the inode as used.

(d) The Inode table needs to have one of its inodes modified to act as the inode of the file.

(e) Data blocks will have to be modified to store the "Small string." texts

Assuming a character takes 1 byte, the file will write 130 bytes of data. So, 1 data block will need to be allocated, to make space for the information. Additionally, 1 inode will be allocated to hold the file information. However, no directory blocks will need to be allocated, as the directories affected occupy only 1 data block both before and after the file's modificiations.

5. If data for very small ordinary files was kept in the inode's area reserved for block pointers, the system would need some way of determining whether or not the area was being used for block pointers or the file's data. This would require sending it through some sort of check in the processor. During this check, however, the drive would still be running, causing it to have to wait an entire rotation before the data could be read again. Due to this extra rotation, this would likely slow down the file system.

6. 5 inodes need to be read: home/, csdept/, jmayo/, cs4411/, hw.txt. Including the inodes, 9 disk blocks will need to be read in order to get every inode. A minimum of 4 directory data blocks would need to be read.

7. The maximum number of keys a tree of degree 2 can contain in a single node is 2(2-1) or 2 keys. Every node other than the root must contain at least 1 key. An internal node can have at most 4 children. Note: images for tree separate.

8. The maximum height is determined by the equation $h \leq \log_t \frac{n+1}{2}$ where h is height, t is the minimum degree, and n is the number of keys. To find the number of keys, we multiply the number of nodes by the maximum number of keys per node, or $n = 2 * 10^6(2 * 10^2 - 1)$. Since a partial height doesn't make sense, we take the ceiling of the log function to get a maximum height of 5.