CSC369 A1a Proposal

Lingjing Zou
Yifan Liu

1. A simple diagram of the layout of your disk image. The diagram will indicate where the different components of your file system will be stored.

Super block	Inode Bitmap	Block Bitmap	Inodes	…...
Data Blocks	…...			

2. A description of how you are partitioning space. This will include how you are keeping track of free inodes and data blocks, and where those data structures are stored, how you determine which blocks are used for inodes. Note that the size of the disk image and the number of inodes are parameters to mkfs.

Ans. We have four main parts which are super block, Inode Bitmap, Block Bitmap, Inodes, and Data Blocks. We keep track of inodes and blocks in two bitmaps which will indicate blocks are used or free to use. We will reserve blocks after bitmap only for inodes as number of inodes will be given when we want to make up our file system and we can store the number of inodes and data blocks in super block struct when we have the size of the disk images nd number of inodes when mkfs is performed.

3. A description of how you will store the information about the extents that belong to a file. For example if the data blocks for file A are blocks 2,3,4 and 8,9,10, then the extents are described as (2,3) and (8,3). Where is this information about the extents stored?

Ans. We have an array attribute called *blocks* in our inode struct and we will store the pointer to the extents of the file (blocks occupied for this file). We can have another attribute about the extents about file in the form of (a, b).

4. Describe how to allocate disk blocks to a file when it is extended. In other words, how do you identify available extents and allocate it to a file? What are the steps involved?

Ans. We will first find the inode for this file, then we will try to find free blocks in data block sections by checking blocks bitmap and allocated and extend the blocks for this file. In this process we need to allocated the blocks as close as possible to minimize the time to read the file. After we extend the file successfully, we need to update the attributes in inode of the files, the number of free blocks in super block struct and also the blocks bitmap.

5. Describe how to free disk blocks when a file is truncated (reduced in size) or deleted.

Ans. When we truncate or delete a file, we need to find the inode for this file which contain important information about this file. We will need to free all data blocks are used for this file and then free this inode block, then we also need to update located directory info about this file. Lastly, all the inode and data blocks are free for this file needed to be updated in inode and data block bitmaps.

6. Describe the algorithm to seek to a specific byte in a file.

Ans. First we will look through the array of pointers in the inode, and find out which block of data stores the byte we need, since the file was stored sequentially. We then go to that block and locate the byte that we are looking for.

7. Describe how to allocate and free inodes.

Ans. We allocate a new inode by looking up the inode bitmap to find the first zero value of the bitmap, and we allocate our new inode at that free location in inode table. When free an inode, we just change the corresponding bit of inode bitmap. In both cases, and update the free_inodes_count in the superblock.

8. Describe how to allocate and free directory entries within the data block(s) that represent the directory.

Ans. We treat the directory entries as a normal file, but we store a struct of directory entry, each directory entry struct will contain the information of inode number, file name and the length of the struct itself.

9. Describe how to lookup a file given a full path to it, starting from the root directory.

Ans. First we go to the root directory and find the file with the name we want and get the inode number of the file, then we look up the inode and go its data blocks. Repeating the steps above until we reach the file we want.