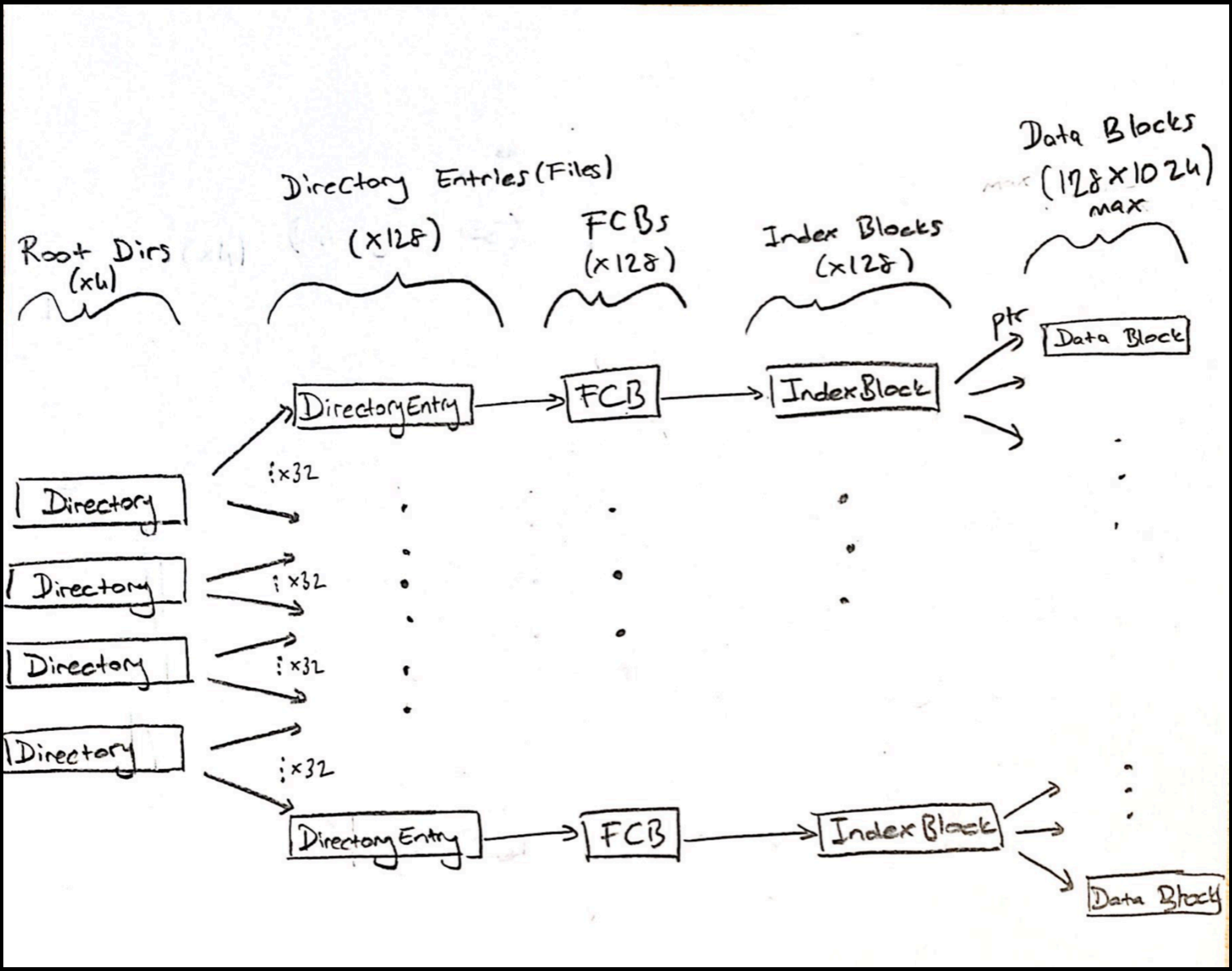
CS342 Project 4 Report

Ahmet Kaan Uğuralp

Structure Of Entries:

You can see a sketch of the structure of my implementation below:



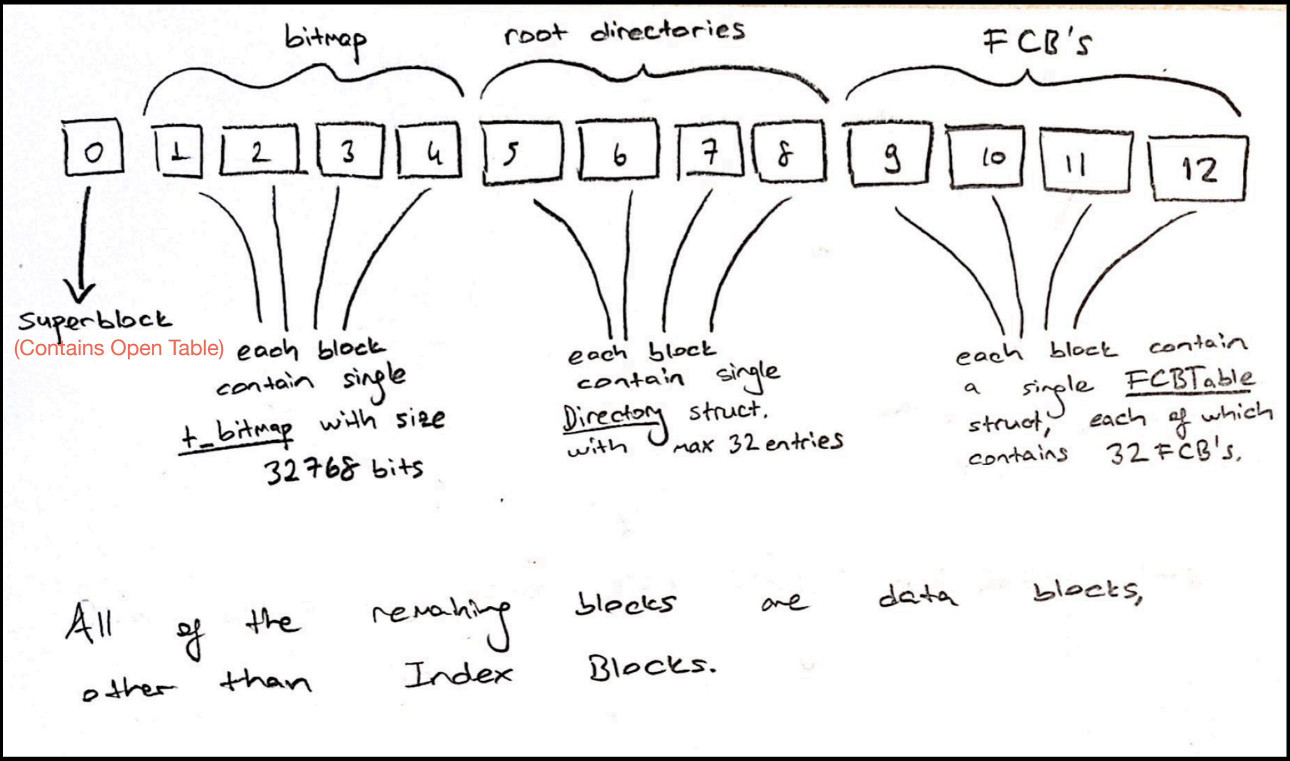
There are four root directories and each of these directories contain max 32 directory entries (files). Each of these directory entries store indexes to their corresponding FCB's to locate them, along with their file name, file size and mode.

Inside each FCB, index of their corresponding IndexBlock are being stored, along with their last read offset (index of the last read item), last item offset (index of the last inserted item), used block count which indicates how many data blocks does the index blocks point to and used flag. These Index Blocks contain an unsigned int array with size 1024 therefore, each index block can point to 1024 blocks max. This means:

$$maxfilesize = 1024x4096 = 4MBs$$
$$maxdisksize = 4MBx128 = 512MBs$$

All of the FCBs are being stored in four FCBTable structs, which contains a FCB array with size 32.

Here is another sketch where the first 13 blocks are shown (metadata):



each block contain single t_bitmap with size 32768 bits

each block contain single Directory struct, with max 32 entries

each block contain a single FCBTable struct, each of which contains 32 FCB's.

All of the remaining blocks are data blocks, other than Index Blocks.

Here is a short summary of what actions the fundamental functions perform:

When a new file is being added to system (sfs_create):
1- Current file amount attribute stored in Superblock gets incremented.
2- An unused directory entry location is found by iterating through the directories (if needed) and the name of the file is inserted into directory entry.
3- An unused FCB is found by iterating through all four FCB Tables and used flag will be set, and a new index block will be allocated into a block that is not in use (found from bitmap). Location (index) of this index block will then be stored inside the FCB of the newly created file.
4- Bitmap is updated accordingly.

When a file is being opened (sfs_open):
1- Check is made if there are already 16 open files.
2- Corresponding directory entry is found by iterating through each directory entry and comparing the file names with the file name inside the parameter. Mode of the directory entry is saved into the corresponding directory entry of the file.
3- Open table of the superblock is updated. Name and entry index of the file that is being opened is saved inside the open table.

When new data is being appended (sfs_append):

1- Check if the mode of the file is append.

2- Check to see if the file has any data blocks pointed by the index block of the file. If index block does not contain any pointer (index) to a data block, a new free block is found and index of this block is added to the index block.

2- If n + last item offset exceeds block size, find and add a the index of the new block into the index block.

3- Data gets appended into the data blocks.

When a file is being deleted (sfs_delete):

1- Check if the file exists.

2- Current file amount attribute stored in Superblock gets decremented.

3- Directory entry location is found by iterating through the directories and directory entry in that location is deleted.

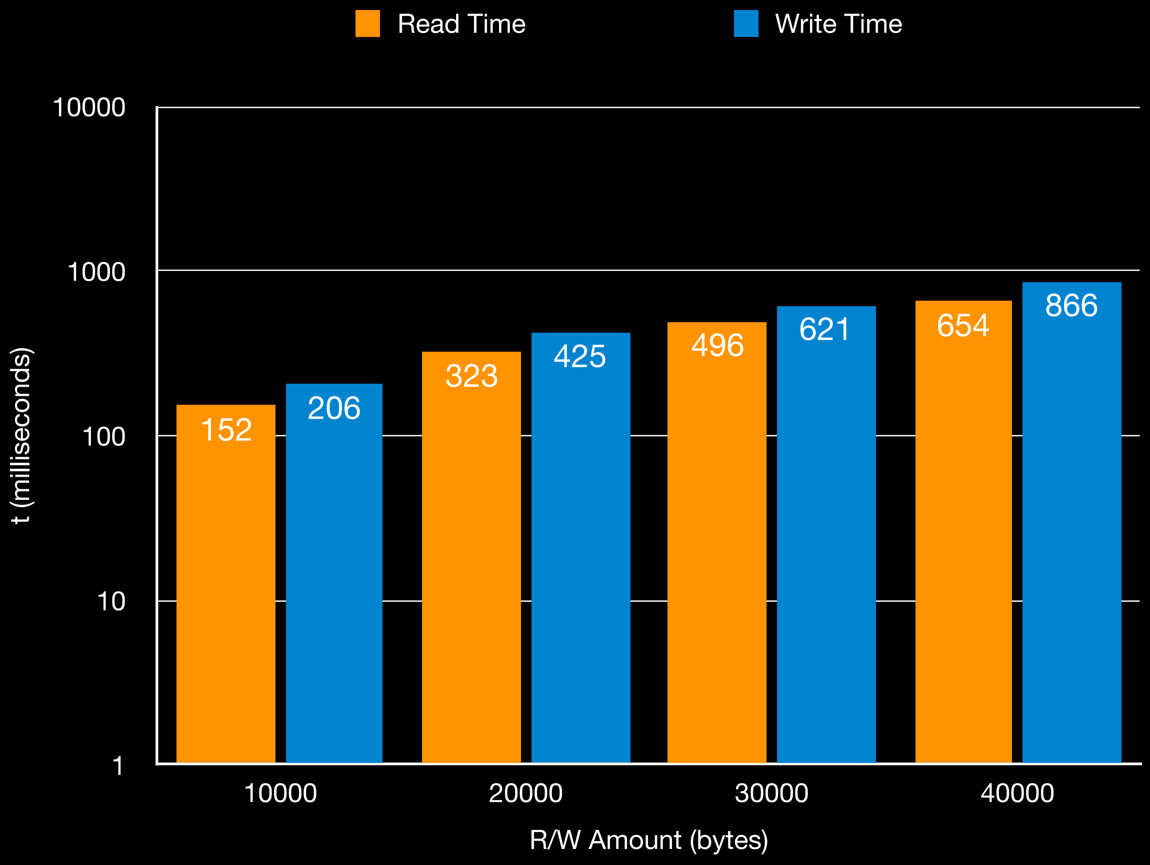
4- Index table of the file is found and all of the blocks in that index table is set as free (0) from the bitmap so that other files can overwrite them. Index of the index table itself is also set as free.

5- Delete FCB of the corresponding entry.

Experiments:

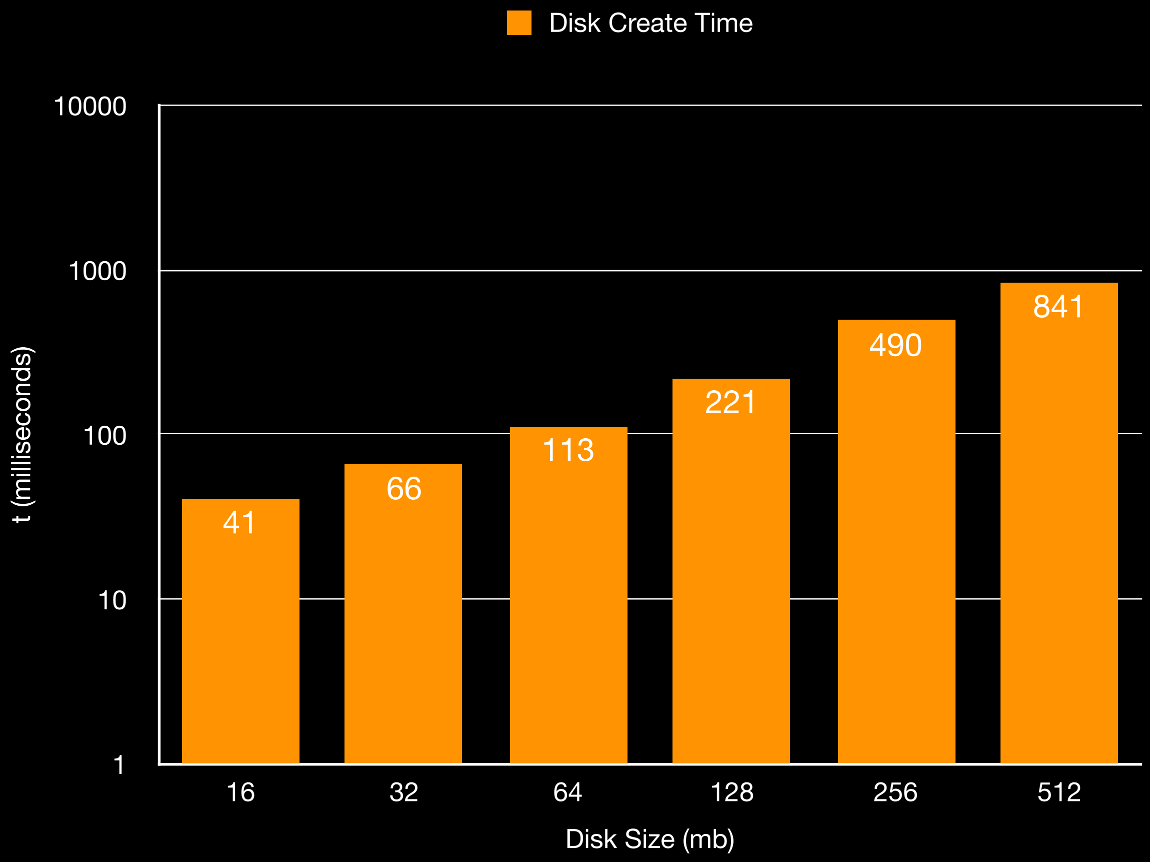
Read and Write Times Based on Read and Write Amount (Virtual disk size = $2^{24} = 16MB$):

R/W Amount (bytes)	Read Time (ms)	Write Time (ms)
10000	152	206
20000	323	425
30000	496	621
40000	654	866

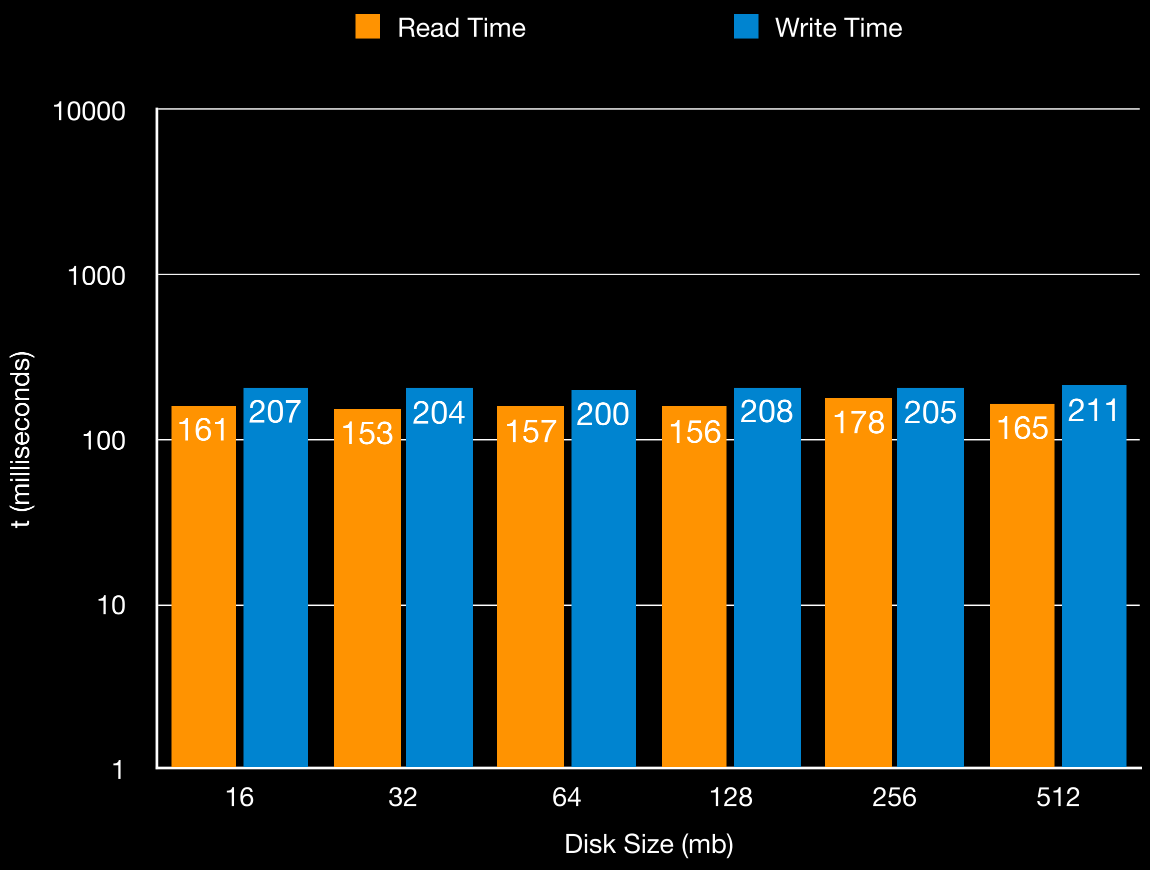


We can conclude here that write command takes longer to execute than read.

Disk Creation (create_format_vdisk) Times Based on Virtual Disk Sizes:



Effect of Virtual Disk Size on Read and Write Times (R/W Size=10000 bytes):



Based on the results we got from the plot above, we can see that the disk size has no effect on read and write times.