CS450 Fall 2019
Programming Assignment 4
Jingwen Pickens
Anthony Pollizi

Design

Changes in Existing Functions and Header Files

In order to modify the file system to support small files whose data is stored in the inode structure itself rather than within data block pointers stored in the inode as in regular files, two new types of files were created. In order to define these new file types, the stat.h file was modified and two new file type numbers were defined, T_SDIR (4) & T_SFILE (5).

A new system call mkSFdir() was implemented in the sysfile.c file to facilitate the creation of small file directories. This system call behaves identically to the system call mkdir, except the directory it creates will be of type T_SDIR. Small file directories will not allow regular files (T_FILE) to be stored within them, only regular directories, other small directories, and small files can exist in a small file directory. Allowing other directory files to exist within a small file directory allows for the "." and ".." directories to be created within them, and allows for these directories to be used anywhere in the directory tree.

The create() system call was modified such that the code which handles T_DIR files would also handle T_SDIR files. In the create system call, the file type of the parent directory is checked and if it is a T_SDIR, then any T_FILE being created has its type changed to T_SFILE. T_SFILEs are created in the same way as T_FILEs by the same code in the create function.

The read() system call has a new clause which is executed when reading T_SFILEs. Instead of getting the data from the block pointers for such files, the data is extracted directly from the memory which would contain those pointers in a normal file.

The write() system call has a new clause which is executed when writing to T_SFILEs. Instead of writing the data into the blocks at the data block pointers for such files, the data is written directly to the memory which would contain those pointers in a normal file. Before the write call puts the data into the inode, it first checks to see whether the amount of bytes to write into the file plus the offset at which the file is being written will exceed the maximum size of small files, which is 52 bytes due to there being 13 4-byte pointers worth of space in the inode to store data. If the write would exceed the size of a small file, then the function simply prints an error and returns -1.

The ls.c file was modified to add cases for the ls command to print out the information in small files and small file directories.

The chdir() system call was modified to allow changing the current directory into small file directories.

Design Method

Handling write overflow

An alternative design that would allow for small files which are overflowed by a write system call to be converted into a regular file can be accomplished by allowing small files to be linked to regular directories.

Linking small files to regular directories

If the design were to allow for small files to be linked to regular directories, small file directories can be gotten rid of, as they serve no purpose and would unnecessarily complicate the design. Instead when a new T_FILE is created, it will instead be created as a T_SFILE. The newly created file will remain a T_SFILE until it is overflowed by write calls. Once a small file has a write call which would overflow it, the data from its inode will be temporarily stored in memory. Then the data blocks pointers would be created and subsequently filled into the inode, and the file type changed to T_FILE. The data to be written to the file would then be appended to the in memory previously held data, and then written to the data blocks referenced in the pointers.