OS MP4 Report

B08902055 李英華

June 12, 2021

1. Briefly explain how you solve each problem:

   (a) Problem 1: Large files

   - Calculation for `bigfile`:
     - Implement 2 doubly-indirect block, because (256 x 256 + 256 + 11) isn't enough. `MAXFILE` therefore becomes 10 + 256 + 2 x 256 x 256, and `NDIRECT` = 10.
     - Number of elements in `addrs[]` for both inode and dinode becomes `NDIRECT+3`.
   - `bmap()`:
     - Indices of `ip->addrs[]` for 2 doubly-indirect is `NDIRECT+1` and `NDIRECT+2`.
     - Calculate `bn/NINDIRECT` for accessing the right block of pointers, and `bn%NINDIRECT` (offset), for accessing the targeted block.
     - Allocates blocks only as needed: one for the targeted block of pointers, another for targeted doubly-indirect block.
   - `itrunc()`:
     Implement a 2-level loop for iteratively `bfree()` the doubly-indirect data blocks (inner loop), and then the block of pointers (outer loop).

   (b) Problem 2: Symbolic links to files

   - `sys_symlink()`:
     - `create()` inode for `path` and `writei()` the `target` to the inode (no matter whether the `target` actually exists).
     - Store both the length of `target` and `target` itself. This way, if I need to `readi()` the symbolic path later, I can know how many bytes to read in advance.
     - `iupdate()` to store the change, `iunlockput()` since `create()` return the locked inode.
   - `sys_open()`:
     - If `O_NOFOLLOW` is set, it will not go into the condition:
       ```
       if (ip->type == T_SYMLINK && !(omode & O_NOFOLLOW))
       ```
       and will be processed and opened as normal file.
     - For those enter the condition, they iterates in while loop:
       ```
       while (ip->type == T_SYMLINK)
       ```
       to `readi()` the path stored in symbolic link, and `namei()` to get next inode. The loop will break when the non-link file is reached or the depth of links ≥ 10.

  - Open must fail if the file doesn't exist, so we examine self-declared variable `sympath` after `readi(..., (uint64)sympath, ...)`:

    ```
    if ((ip = namei(sympath)) == 0){
        end_op();
        return -1;
    }
    ```

(c) Problem 3: Symbolic links to directories

- `namex()`: In the loop `while((path = skipelem(path, name)) != 0)`, we first check whether its type is `T_SYMLINK`. If true, than use the while loop `while (ip->type == T_SYMLINK)` same as the previous to recursively find the targeted directory so that `dirlookup()` can later look up entry in it.
- `sys_chdir()`:
  Add the while loop `while (ip->type == T_SYMLINK)` similar to that in `sys_open()` to recursively find the directory it links to.