CS 45: Operating Systems

CLab 12: File System Implemenation

DUE by next class

Write your answers in `README.md`. Create a `COLLAB.md` file to keep track of any outside resources you might use. Be sure to push to the repo after class (even if you are not done).

1 `vsfs.py`

1. Run the simulator with some different random seeds (say 17, 18, 19, 20), and see if you can figure out which operations must have taken place between each state change.
2. Now do the same, using different random seeds (say 21, 22, 23, 24), except run with the `-r` flag, thus making you guess the state change while being shown the operation. What can you conclude about the inode and data-block allocation algorithms, in terms of which blocks they prefer to allocate?
3. Now reduce the number of data blocks in the file system, to very low numbers (say two), and run the simulator for a hundred or so requests. What types of files end up in the file system in this highly-constrained layout? What types of operations would fail?
4. Now do the same, but with inodes. With very few inodes, what types of operations can succeed? Which will usually fail? What is the final state of the file system likely to be?

2 `fsck.py`

1. First, run `fsck.py -D`; this flag turns off any corruption, and thus you can use it to generate a random file system, and see if you can determine which files and directories are in there. So, go ahead and do that! Use the `-p` flag to see if you were right. Try this for a few different randomly-generated file systems by setting the seed (`-s`) to different values, like 1, 2, and 3.
2. Now, let's introduce a corruption. Run `fsck.py -S 1` to start. Can you see what inconsistency is introduced? How would you fix it in a real file system repair tool? Use `-c` to check if you were right.
3. Change the seed to `-S 3` or `-S 19`; which inconsistency do you see? Use `-c` to check your answer. What is different in these two cases?
4. Change the seed to `-S 5`; which inconsistency do you see? How hard would it be to fix this problem in an automatic way? Use `-c` to check your answer. Then, introduce a similar inconsistency with `-S 38`; is this harder/possible to detect? Finally, use `-S 642`; is this inconsistency detectable? If so, how would you fix the file system?
5. Change the seed to `-S 6` or `-S 13`; which inconsistency do you see? Use `-c` to check your answer. What is the difference across these two cases? What should the repair tool do when encountering such a situation?
6. Change the seed to `-S 9`; which inconsistency do you see? Use `-c` to check your answer. Which piece of information should a check-and-repair tool trust in this case?
7. Change the seed to `-S 15`; which inconsistency do you see? Use `-c` to check your answer. What can a repair tool do in this case? If no repair is possible, how much data is lost?
8. Change the seed to `-S 10`; which inconsistency do you see? Use `-c` to check your answer. Is there redundancy in the file system structure here that can help a repair?
9. Change the seed to `-S 16` and `-S 20`; which inconsistency do you see? Use `-c` to check your answer. How should the repair tool fix the problem?