CS4513 Project 1 Report

Author: Mengwen Li (mli2)

Design:

a). Script

To measure the time required to perform “rename()” operation, the following script was used:

```
#!/bin/bash
# Author: Mengwen Li (mli2)
# create DUMPSTER path first on test machine.
export DUMPSTER=/home/mengwen/Desktop/dumpster
totalRenameTime=0
# copy 20 version of test file.
for i in {1..20}
    do
        cp testRename "testRename$i"
    done
# remove these test files and time.
for i in {1..20}
    do
        sync
        # measure the first time.
        t1=$(date +%s%3N)
        ./rm "testRename$i"
        sync
        # measure the second time.
        t2=$(date +%s%3N)
        # calculate the time used.
        timeUse=$((t2-t1))
        # add to total time.
        totalRenameTime=$((totalRenameTime+timeUse))
        # print out the time used for one run.
        echo "$timeUse" milliseconds
    done
# print out the total time needed and the average time.
echo "total rename time in ms is " "$totalRenameTime" "milliseconds"
echo "average rename time in ms is" $((totalRenameTime/20)) "milliseconds"
# empty the dumpster.
./dump
```

Fig. 1. Script for measuring “rename”

The script first create the environment variable for “DUMPSTER”, then, it copies 20 version of the test file “testRename”. The stat for file “testRename” was shown in the following figure:

```
mengwen@Mengwen-PC:~/Desktop/cs4513/project1$ stat testRename
  File: ‘testRename’
  Size: 19314763        Blocks: 37736      IO Block: 4096   regular file
Device: 809h/2057d      Inode: 1050618     Links: 1
Access: (0777/-rwxrwxrwx)  Uid: ( 1000/ mengwen)   Gid: ( 1000/ mengwen)
Access: 2016-01-24 13:44:28.701361460 -0500
Modify: 2015-05-18 12:07:33.915872000 -0400
Change: 2016-01-24 13:44:06.453059069 -0500
 Birth: -
```

Fig. 2. Stat for file “testRename”

Then, in the for loop, the script calls “sync” to ensure that no disk operation is cashed, then, it records the starting time. After that, it calls function “rm” to remove the copied file. The script then calls “sync” again, record final time and update total time used. Finally, the script will print out the total time used and the average time. The script then call “./dump” to clear the dumpster.
To measure the time required to copy a large file across seperate partitions and calculate the throughput, the following script was used:

```
34 # measure copy a big file from another partition.
35 totalCopyTime=0
36 # get the size of the test file.
37 filename=/media/mengwen/OS/testCopy
38 filesize=$(stat -c%s "$filename")
39 # print out the size.
40 echo "Size of $filename = $filesize bytes."
41 # copy 20 version of test file.
42 for i in {1..20}
43     do
44         cp /media/mengwen/OS/testCopy "/media/mengwen/OS/testCopy$i"
45     done
46 # remove these test files and time.
47 for i in {1..20}
48     do
49         sync
50         # get the start time.
51         t1=$(date +%s%3N)
52         ./rm "/media/mengwen/OS/testCopy$i"
53         sync
54         # get the finish time.
55         t2=$(date +%s%3N)
56         # get the time used.
57         timeUse=$((t2-t1))
58         # add to the total time.
59         totalCopyTime=$((totalCopyTime+timeUse))
60         # print out the time used for one run.
61         echo "$timeUse" milliseconds
62     done
63 # print out the total time, average time and calculate the throughput.
64 echo "total copy time in ms is " "$totalCopyTime" "milliseconds"
65 echo "average copy time in ms is " $((totalCopyTime/20)) "milliseconds"
66 echo "throughput is " $(((filesize*20) / (totalCopyTime/1000))) "bytes/second"
67 # empty the dumpster.
68 ./dump
```

Fig. 3. Script for timing copy big file across partition

The script first determin the size of the file in bytes and print it out. Then, it make 20 copies of the file. After that, in the for loop, the script calls “sync” to ensure that no disk operation is cashed, then, it records the starting time. After that, it calls function “rm” to remove the copied file. The script then calls “sync” again, record final time and update total time used. Finally, the script will print out the total time used, the average time and the throughput it calculated in “bytes/second”. The script then call “./dump” to clear the dumpster.

To measure the time of copying a directory across partition, the following script was used:

```
# measure copy a big directory across partition.
# get the size of the directory.
filename=/media/mengwen/OS/testFolder
filesize=$(du -sb $filename | cut -f1)
echo "Size of $filename = $filesize bytes."

totalFolderTime=0
# copy 10 copies of test directory.
for i in {1..10}
    do
        cp -rf /media/mengwen/OS/testFolder "/media/mengwen/OS/testFolder$i"
    done
# remove these directories and time.
for i in {1..10}
    do
        sync
        # get the start time.
        t1=$(date +%s%3N)
        ./rm -r "/media/mengwen/OS/testFolder$i"
        sync
        # get the end time.
        t2=$(date +%s%3N)
        # get the time used.
        timeUse=$((t2-t1))
        # add to the total time.
        totalFolderTime=$((totalFolderTime+timeUse))
        # print out the time used for a single run.
        echo "$timeUse" milliseconds
    done
# print out the total time and the average time.
echo "total folder copy time in ms is " "$totalFolderTime" "milliseconds"
echo "average folder copy time in ms is " $((totalFolderTime/10)) "milliseconds"
# empty the dumpster.
./dump
```

Fig. 4. Script for timing copy directory across partition

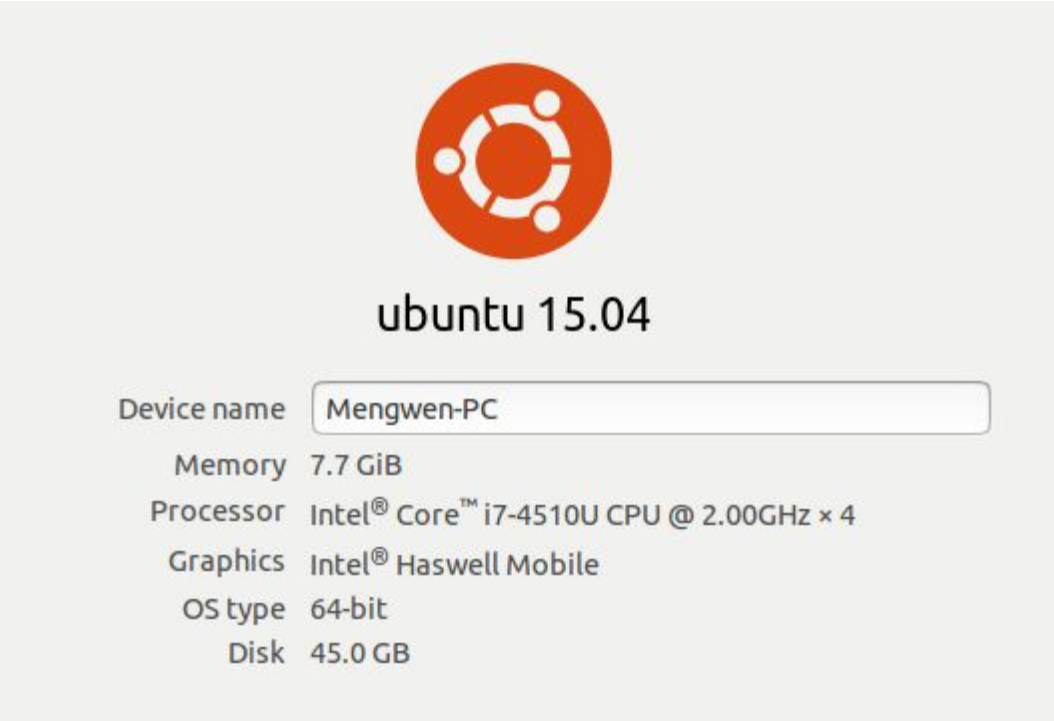
The script first determin the size of the directory in bytes and print it out. Then, it make 5 copies of the directory. After that, in the for loop, the script calls “sync” to ensure that no disk operation is cashed, then, it records the starting time. After that, it calls function “rm” with “-r” option to remove the copied directory. The script then calls “sync” again, record final time and update total time used. The script then call “./dump” to clear the dumpster.

b). How many runs I performed.

For timing of moving file in the same partition and copying file across partitions, 20 runs were performed to decied the final time needed. For timing copying directory across partition, 10 runs were performed to get the final time since the directory is large.

c). How you record you data.
The calculation results will be printed by the script to the terminal so I can looked into it.

d). What the system conditions were like.
The system condition was listed in the following figure:



e). Any other details you think are relevalent.
Since I have two operating systems on my laptop, I put the dumpster in the disk for one system and the large files and folders in the disk for another system. Also, the hard drive on my laptop is an SSD.

Results:

Moving file to dumpster on the same partition using “rename()”:

Run number and Statistics	Time used (ms)
1	14
2	9
3	6
4	20
5	14
6	6
7	13
8	20
9	8
10	7
11	15
12	7
13	6
14	7
15	10
16	7
17	19
18	14
19	12
20	14
Mean	11
Standard Deviation	4.77

Copying a large file across separate partitions:

Run number and Statistics	Time used (ms)
1	541
2	545
3	811
4	607
5	549
6	557
7	399
8	356
9	507
10	375
11	214
12	522
13	289
14	116
15	208
16	515
17	493
18	496
19	125
20	137
Mean	418
Standard Deviation	186.69
File Size (in bytes)	19314763
Throughput (in bytes/sec)	48286907

Copying a directory across seperate partitions:

Run number and Statistics	Time used (ms)
1	1366
2	1650
3	1037
4	1305
5	1737
6	993
7	1453
8	948
9	1148
10	1014
Mean	1265
Standard Deviation	283.10
Directory Size (in bytes)	57948385

Analysis:

From the result, we can tell that moving file to dumpster on the same partition using “rename()” takes much less time than moving a file to dumpster across partition. The reason is that using “rename()” system call only needs to modify the link of the file, so it needs less time, while moving file across partition can’t be done by simply create a new hard link, as a result, a file copying operation needs to be performed. So, moving a file across partition needs much more time than moving a file within the same partition. The conclusion corresponds to the data measured.
Anticipated directory transfer time: 57948385/48286907 = 1200.08(ms)
The anticipated result for transfering a large directory across partitions is roughly the same as the measured mean.
Since moving a directory across partiton needs the same operation with moving a file across partition, the throughput when moving a file can be used to estimate the time for moving a directory. The formula I use is Time = Size of the directory / throughput measured. The calculateion result corresponds to the measured data closly.