Calvin Huynh, 861015448

Homework 3 for CS153 (Spring 2015)

Due: on ilearn by the end of 5th June 2015

Instructions:

***** Be brief in your answers. You will be graded for correctness, not on the length of your answers.

***** Make sure to write legibly. Incomprehensible writing will be assumed to be incorrect.

I. Consider that requests to read the following set of logical block numbers are enqueued to be serviced from a disk that has 60 logical blocks laid out sequentially from block 0 to block 59.

{1, 4, 13, 29, 35, 38, 49}

Assume that the seek time in moving the disk arm head from logical block i to block j is proportional to |i – j|. Given that the arm head is currently positioned at block 34 and is in the midst of moving from left to right, what is the sequence in which the enqueued blocks will be read with the i) SSTF, ii) SCAN, and iii) C-SCAN algorithms? (3 points)

i)SSTF (shortest seek time first)

[1]--[4]--------[13]----------[29]-----[**35**]--[38]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[**38**]----------[49]
[1]--[4]--------[13]----------[**29**]-----[35]--[38]----------[49]
[1]--[4]--------[**13**]----------[29]-----[35]--[38]----------[49]
[1]--[**4**]--------[13]----------[29]-----[35]--[38]----------[49]
[**1**]--[4]--------[13]----------[29]-----[35]--[38]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[38]----------[**49**]

Sequence: 35 → 38 → 29 → 13 → 4 → 1 → 49

ii) SCAN

[1]--[4]--------[13]----------[29]-----[**35**]--[38]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[**38**]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[38]----------[**49**]
[1]--[4]--------[13]----------[**29**]-----[35]--[38]----------[49]
[1]--[4]--------[**13**]----------[29]-----[35]--[38]----------[49]
[1]--[**4**]--------[13]----------[29]-----[35]--[38]----------[49]
[**1**]--[4]--------[13]----------[29]-----[35]--[38]----------[49]

Sequence: 35 → 38 → 49 → 29 → 13 → 4 → 1

iii) C-SCAN

[1]--[4]--------[13]----------[29]-----[**35**]--[38]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[**38**]----------[49]
[1]--[4]--------[13]----------[29]-----[35]--[38]----------[**49**]
[**1**]--[4]--------[13]----------[29]-----[35]--[38]----------[49]
[1]--[**4**]--------[13]----------[29]-----[35]--[38]----------[49]
[1]--[4]--------[**13**]----------[29]-----[35]--[38]----------[49]
[1]--[4]--------[13]----------[**29**]-----[35]--[38]----------[49]

Sequence: 35 → 38 → 49 → 1 → 4 → 13 → 29

II. Consider a file system that uses a structure similar to an i-node with the following differences. If the file size is less than 200 bytes, the data is stored directly in the i-node. If it is larger, there are 10 direct links (point to a data block), 2 single-indirect links, 2-double indirect links and 1 triple indirect link. What is the largest file size that can be indexed in this system? Assume the block size is 512 bytes and the pointer size is 4 bytes. (3 points)
10 direct links = 10 * 512
512/4 = 128 pointers
2 indirect = 512 * (128)*2
2 double indirect links = 512 *(128^2)*2
1 triple indirect = 512 * (128^3)
512*10 + 512 * 2(128) + 512 *2(128^2) + 512(128^3) = 1090655232 bytes

III. On the original UNIX file system, we are reading the file "/one/two/three." All the directories fit in a single block each.

(1) Describe and count the number of disk reads involved in reading the file. (2 points)
(2) How does FFS improve performance over the basic UNIX file system? (1 point)
(3) How does LFS improve performance? (1 point)

1)

- open the directory "/"
- Search for the entry and read the location of "one" in the directory entry
- open directory "one," search for the entry and read the location of "two"
- open directory "two," search for the entry and read the location of "three"
- open file "three"

3 reads total
2) It improves disk utilization and decreased response time by localizing files to cylinder groups
3) It improve write performance by treating disk as a log

IV. Consider a disk on which the rate at which it can read sequentially laid out data is 100 MBps. Average seek time on the disk is 5 milliseconds. If the block size used by the file system is 64 KB and all blocks are randomly distributed across the disk, how long will it take for an application to read a file of size 1 GB? (2 points)

$\frac{1GB}{64KB} = 15625\ blocks\ of\ 64\ KB$

Converting MB to KB...

$\frac{100\ MB}{1\ sec} \cdot \frac{1000\ KB}{1\ MB} = \frac{100000\ KB}{1\ sec}$

Setting up a proportion, we can find out how fast it takes to read 64 KB.

$\frac{100000\ KB}{1\ sec} = \frac{64\ KB}{x}$

$\frac{1\ sec}{100000\ KB} = \frac{x}{64\ KB}$

0.00064 sec = 0.64 milliseconds = x

It takes 0.64 milliseconds to read 64 KB. Since there are 15625 blocks, there are 15625 reads. IN addition, there will be 15625 seeks that take 5 milliseconds each. Adding both times will yield the total time the application takes to read 1GB files.

$5 * 15625 + 0.64 * 15625 = 88125\ milliseconds = 88.125\ seconds$

It takes 88.125 seconds to read 1 GB files.

V. Consider the following two disk types:

i) Each unit of Disk1 offers 200 MBps of read/write bandwidth and fails within three years with a 10% probability.

ii) Each unit of Disk2 offers 800 MBps of read/write bandwidth and fails within three years with a 1% probability.

If you were to use a RAID configuration, what is the minimum number of units of Disk1 that would be necessary to ensure that both the read/write bandwidth and the probability of losing data within three years are at least as good as those with 1 unit of Disk2? Explain which RAID configuration you used. Assume that every disk fails independently and only large files are read/written. To get points, you can at show any setup you can come up with and compute the bandwidth and probability of losing data. (2 + 2 bonus points)

Using 4 mirrored Disk1's using RAID 1 in a RAID0 array.

Because we allow mirroring without parity and block-level striping, writing is not limited to the performance of the weakest drive.

reading = 4(200) = 800 MBps

writing = (4)(200) = 800 MBps with one striping per drive1

failure rate = (0.1)^4 = 0.0001

VI. The following is a brief description of the NT File System (NTFS), due to Rajkumar:
"NTFS works with disk volumes. NTFS allocates the volume in clusters, where a cluster is one or more (generally, a power of two) contiguous sectors. The clusters size is defined when the disk is formatted. NTFS uses logical cluster numbers as disk addresses. Multiplying the cluster number by the cluster size yields a physical disk address.

Each volume is organized into four regions. The first few sectors contain the partition boot sector. Next is the Master File Table (MFT), essentially a master directory. The MFT entries contain information on each file on the volume. Following the MFT is a system area that contains a duplicate of portions of the MFT (to provide redundancy), log files used for NTFS for recoverability, a cluster bit map, and an attribute definition table. The rest of the volume is data area.

The master file table is the main object NTFS uses to access files. It consists of an array of variable length records. The first 16 records describe the MFT itself, followed by a record for each file or directory. If the file is small enough (less than 1500 bytes) it is written directly in the MFT entry. Otherwise, the MFT entry contains index pointers to the clusters that contain the actual data."
(1) What is the purpose of the cluster bit map? (1 point)
(2) What type of allocation does NTFS use (contiguous, linked, indexed, or something else?)
(1 point)
(3) What is the idea behind allowing small files to be written directly in the MFT? (1 point)
(4) What is the purpose of using clusters rather than blocks as the unit of allocation? (1 point)

1) It records which clusters of a file are and are not compressed
2) Contiguous
3) To create as little clusters as possible
4) Cluster sizes are not predetermined, but are determined by the disk volumes to avoid fragmentation