Cole Vikupitz

CIS 415

6 June 2017

Assignment 4

Textbook Questions

1. OSC 10.11: *Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 2150, and the previous request was at cylinder 1805. The queue of pending requests, in FIFO order, is:*

2069, 1212, 2296, 2800, 544, 1618, 356, 1523, 4965, 3681

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms?

a. *FCFS*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2069	(2150 - 2069) = 81	81
2069	1212	(2069 - 1212) = 857	(81 + 857) = 938
1212	2296	(2296 - 1212) = 1084	(938 + 1084) = 2022
2296	2800	(2800 - 2296) = 504	(2022 + 504) = 2526
2800	544	(2800 - 544) = 2256	(2526 + 2256) = 4782
544	1618	(1618 - 544) = 1074	(4782 + 1074) = 5856
1618	356	(1618 - 356) = 1262	(5856 + 1262) = 7118
356	1523	(1523 - 356) = 1167	(7118 + 1167) = 8285
1523	4965	(4965 - 1523) = 3442	(8285 + 3442) = 11727
4965	3681	(4965 - 3681) = 1284	(11727 + 1284) = **13011**

Total distance arm moves is **13,011 cylinders**.

b. *SSTF*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2069	(2150 - 2069) = 81	81
2069	2296	(2296 - 2069) = 227	(81 + 227) = 308
2296	2800	(2800 - 2296) = 504	(308 + 504) = 812
2800	3681	(3681 - 2800) = 881	(812 + 881) = 1693
3681	4965	(4965 - 3681) = 1284	(1693 + 1284) = 2977
4965	1618	(4965 - 1618) = 3347	(2297 + 3347) = 6324
1618	1523	(1618 - 1523) = 95	(6324 + 95) = 6419
1523	1212	(1523 - 1212) = 311	(6419 + 311) = 6730
1212	544	(1212 - 544) = 668	(6730 + 668) = 7398
544	356	(544 - 356) = 188	(7398 + 188) = **7586**

Total distance arm moves is **7,586 cylinders**.

c. *SCAN*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2296	(2296 - 2150) = 146	146
2296	2800	(2800 - 2296) = 504	(146 + 504) = 650
2800	3681	(3681 - 2800) = 881	(650 + 881) = 1531
3681	4965	(4965 - 3681) = 1284	(1531 + 1284) = 2815
4965	4999	(4999 - 4965) = 34	(2815 + 34) = 2849
4999	2069	(4999 - 2069) = 2930	(2849 + 2930) = 5779
2069	1618	(2069 - 1618) = 451	(5779 + 451) = 6230
1618	1523	(1618 - 1523) = 95	(6230 + 95) = 6325
1523	1212	(1523 - 1212) = 311	(6325 + 311) = 6636
1212	544	(1212 - 544) = 668	(6636 + 668) = 7304
544	356	(544 - 356) = 188	(7304 + 188) = **7492**

Total distance arm moves is **7,492 cylinders**.

d. *LOOK*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2296	(2296 - 2150) = 146	146
2296	2800	(2800 - 2296) = 504	(146 + 504) = 650
2800	3681	(3681 - 2800) = 881	(650 + 881) = 1531
3681	4965	(4965 - 3681) = 1284	(1531 + 1284) = 2815
4965	2069	(4965 - 2069) = 2896	(2815 + 2896) = 5711
2069	1618	(2069 - 1618) = 451	(5711 + 451) = 6162
1618	1523	(1618 - 1523) = 95	(6162 + 95) = 6257
1523	1212	(1523 - 1212) = 311	(6257 + 311) = 6568
1212	544	(1212 - 544) = 668	(6568 + 668) = 7236
544	356	(544 - 356) = 188	(7236 + 188) = **7424**

Total distance arm moves is **7,424 cylinders**.

e. *C-SCAN*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2296	(2296 - 2150) = 146	146
2296	2800	(2800 - 2296) = 504	(146 + 504) = 650
2800	3681	(3681 - 2800) = 881	(650 + 881) = 1531
3681	4965	(4965 - 3681) = 1284	(1531 + 1284) = 2815
4965	4999	(4999 - 4965) = 34	(2815 + 34) = 2849
4999	0	(4999 - 0) = 4999	(2849 + 4999) = 7848
0	356	(356 - 0) = 356	(7848 + 356) = 8204
356	544	(544 - 356) = 188	(8204 + 188) = 8392
544	1212	(1212 - 544) = 668	(8392 + 668) = 9060
1212	1523	(1523 - 1212) = 311	(9060 + 311) = 9371
1523	1618	(1618 - 1523) = 95	(9371 + 95) = 9466
1618	2069	(2069 - 1618) = 451	(9466 + 451) = **9917**

Total distance arm moves is **9,917 cylinders**.

f. *C-LOOK*

Current Cylinder	**Next Cylinder**	**Distance**	**Total Distance**
2150	2296	(2296 - 2150) = 146	146
2296	2800	(2800 - 2296) = 504	(146 + 504) = 650
2800	3681	(3681 - 2800) = 881	(650 + 881) = 1531
3681	4965	(4965 - 3681) = 1284	(1531 + 1284) = 2815
4965	356	(4965 - 356) = 4609	(2815 + 4609) = 7424
356	544	(544 - 356) = 188	(7424 + 188) = 7612
544	1212	(1212 - 544) = 668	(7612 + 668) = 8300
1212	1523	(1523 - 1212) = 311	(8300 + 311) = 8611
1523	1618	(1618 - 1523) = 95	(8611 + 95) = 8706
1618	2069	(2069 - 1618) = 451	(8706 + 451) = **9157**

Total distance arm moves is **9,157 cylinders**.

2. *Consider a file system that uses inodes to represent files. Disk blocks are 1 kB in size, and a pointer to a disk block requires 4 bytes. Each inode has 8 direct block pointers, as well as one each of single, double, and triple indirect block pointers.*

n	2	3	5	10	20	30	40
2^n	4	8	32	1,024	1,048,576	1,073,741,824	1,099,511,627,776
Label				kB	MB	GB	TB

a. *What is the maximum size of a disk (in bytes) for which one can use this file system?*

A pointer to a disk block stores the block number, which is an unsigned integer, and the pointer itself is 4 byte long. This means that a block pointer can store any number in the range [0, 4,294,967,295]. This means that there are 4,294,967,296 blocks available, each 1,024 bytes, meaning the maximum disk size is **2^{10} * 4,294,967,296 bytes (~4 TB).**

b. *What is the maximum size of a file (in bytes) that can be stored in this file system?*

We have 8 direct block pointers, each pointing to 2^{10} bytes, giving $8*2^{10}$ bytes. Each indirect block pointer can hold up to 256 direct block pointers (1024 / 4), giving $256*2^{10}$ bytes. Each double indirect block pointer can hold up to 256^2 direct block pointers, giving 256^2*2^{10} bytes. Finally, each triple indirect block pointer can hold up to 256^3 direct block pointers, giving 256^3*2^{10} bytes.

$2^{10} * (8 + 256 + 256^2 + 256^3) = 2^{10} * (8 + 256 + 65{,}536 + 16{,}777{,}216)$

= **2^{10} * 16,843,016 bytes (~16 GB)**

3. *The processor for which you are designing your application has L1i and L1d virtual caches.*

a. *What type of data does each cache hold?*

The L1i is an instruction cache, it stores recently executed instructions. The L1d is a data cache, it stores any data that a program may use.

b. *Describe in detail the activities of the cache + memory system when executing the instruction.*

LOAD virtual address, register

First, the instruction must be fetched from memory, so it searches for it in the L1i cache, then goes to main memory if not found. The instruction is then executed, and the data is fetched from the L1d cache, or main memory if not found. The instruction is then put into the L1i cache for later use.

c. *Assume that the above instruction is executed many times in a loop, and that the instruction itself is in the cache. Also assume that memory access costs τ μs, and cache access costs τ /15 μs. What cache hit rate p for "virtual address" is required for the memory system to run 5 times faster than with no caching at all? Show your work.*

d. *Suppose we have a memory system that has a main memory, a single-level cache, and demand paging virtual memory. The three level of the memory system have the following access times:*

Cache	2ns
Main memory	100ns
Paging disk	10ms

i. *The cache has a 95% hit rate. What is the effective memory access time if we consider only the cache and main memory and ignore page faults and disk access times?*

Since we're ignoring page faults and disk access times, the formula should be something like:
EAT = (Hit Rate * Cache Access Time) + (1 - Hit Rate * Main Memory Access Time)
EAT = (0.95 * 2ns) + (0.05 * 100ns) = 1.9ns + 5ns = **6.9 nanoseconds**

ii. *Now recalculate the effective memory access time assuming the same cache hit rate (95%) plus a page fault rate of 0.001% (i.e., 99.999% of the memory accesses succeed without producing a page fault).*

EAT = (Hit Rate * Main Memory Access Time) + (1 – Page Fault Rate * Main Memory Access Time) + (Page Fault Rate * Paging Disk Access Time)
EAT = (0.95 * 2ns) + (1 – 0.00001 * 100ns) + (0.00001 * 10000000ns)
= 1.9ns + 99.999ns + 100ns = **201.899 nanoseconds**