Operating System
10610 CS342302
Prof. Pai H. Chou
TA:

Hsu PO-CHUN
Student ID: s105062803
Submitted: Dec. 18, 2017
Due Date: Dec. 17, 2017

Assignment 12

1 Problem 1

[20 points] (from Chapter 12) Explain why SSTF scheduling tends to favor middle cylinders over the innermost and outermost cylinders.

1.1 Answer

The SSTF algorithm is biased toward the middle cylinders in much the same way the SCAN algorithm is. Since, in the SSTF algorithm, the closest cylinder is always chosen, then all middle cylinder references will serviced on the way to the end of the disk.
The center of the disk is the location having the smallest average distance to all other tracks. Thus the disk head tends to move away from the edges of the disk.
Here is another way to think of it. The current location of the head divides the cylinders into two groups. If the head is not in the center of the disk and a new request arrives, the new request is more likely to be in the group that includes the center of the disk; thus, the head is more likely to move in that direction.

2 Problem 2

[20 points] 12.8 Requests are not usually uniformly distributed. For example, we can expect a cylinder containing the file-system metadata to be accessed more frequently than a cylinder containing only files. Suppose you know that 50 percent of the requests are for a small, fixed number of cylinders.

a. Would any of the scheduling algorithms discussed in this chapter be particularly good for this case? Explain your answer.

b. Propose a disk-scheduling algorithm that gives even better performance by taking advantage of this hot spot on the disk.

2.1 Answer

a. **SSTF** and **FCFS** would be favored under this circumstance.
Explain: The seek ordering could be arranged at least 50 percent to optimize the seeking time rather than just seeking back and forth. And it may reduce the partial unnecessary movement by arranging reference cylinders.

b. **META-SSTF**:
1. Put the hot data as close to the middle cylinder as possible.
2. Adding aging mechanism to prevent the reference data from starving.
3. Add a dynamic threshold,10ms to 10mins for instance, for the idle period.

4. Partition the disk to locate the hot region with higher seek possibility by the meta-data.
5. Seek to the hot region.

3 Problem 3

[40 points] 13.3 Consider the following I/O scenarios on a single-user PC:

a. A mouse used with a graphical user interface

b. A tape drive on a multitasking operating system (with no device preallocation available)

c. A disk drive containing user files

d. A graphics card with direct bus connection, accessible through memory-mapped I/O

For each of these scenarios, would you design the operating system to use buffering, spooling, caching, or a combination? Would you use polled I/O or interrupt-driven I/O? Give reasons for your choices.

3.1 Answer

a. A mouse used with a graphical user interface:
Spooling and caching are inappropriate. Buffering may be needed to record mouse movement during times when higher-priority operations are taking place. Interrupt driven I/O is most appropriate.

b. A tape drive on a multitasking operating system (no device preallocation is available)
Buffering may be needed to manage throughput difference between the tape drive and the source or destination of the I/O, Caching can be used to hold copies of data that resides on the tape, for faster access. Spooling could be used to stage data to the device when multiple users desire to read from or write to it. Interrupt driven I/O is likely to allow the best performance.

c. A disk drive containing user files Buffering can be used to hold data while in transit from user space to the disk, and visa versa. Caching can be used to hold disk-resident data for improved performance. Spooling is not necessary because disks are shared-access devices.
Interrupt-driven I/O is best for devices such as disks that transfer data at slow rates.

d. A graphics card with direct bus connection, accessible through memory-mapped I/O Buffering may be needed to control multiple access and for performance (double-buffering can be used to hold the next screen image while displaying the current one). Caching and spooling are not necessary due to the fast and shared-access natures of the device. Polling and interrupts are only useful for input and for I/O completion detection, neither of which is needed for a memory-mapped device.

4 Problem 4

[20 points] 13.7 Typically, at the completion of a device I/O, a single interrupt is raised and appropriately handled by the host processor. In certain settings, however, the code that is to be executed at the completion of the I/O can be broken into two separate pieces. The first piece executes immediately after the I/O completes and schedules a second interrupt for the remaining piece of code to be executed at a later time. What is the purpose of using this strategy in the design of interrupt handlers?

4.1 Answer

The most essential routines execute critically and the rest can be done while CPU idles.E.g.,the context switches for waking up the issued process go first,and the DMA data-cloning can execute later.