

COS214 Tutorial 6

Roberto Togneri, 2000

SOLUTIONS

1. (a) What is "device independence"?
- (b) In which of the four I/O software layers is each of the following done.
 - (i) Computing the track, sector, and head for a disk read.
 - (ii) Maintaining a cache of recently used blocks.
 - (iii) Writing commands to the device registers.
 - (iv) Checking to see if the user has permission to use the device.
 - (v) Converting binary integers to ASCII for printing.

Answer

- (a) Device independence means that files and devices are accessed the same way, independent of their physical nature. Systems that have one set of calls for writing on a file, but a different set of calls for writing on the console (terminal) do NOT exhibit device independence.
- (b)
 - (i) device driver
 - (ii) device-independent software
 - (iii) device driver
 - (iv) device-independent software
 - (v) user-level software

2. A user process is accessing data from a large file and performs 0.5 seconds of computation for each 128K of data read. The disk rotation speed is 5000 rpm, average seek time is 2 msec, and data transfer rate is 10 MB/sec. Memory to memory data transfer rate is 100 nsec per 4-byte word. Calculate the CPU time used by the user process for every 5 minutes wall-clock time, and the amount of memory used by kernel buffers, assuming:
 - (a) no buffering
 - (b) single buffer
 - (c) double bufferAssume the process is the only active one in the system and the file is stored contiguously on the disk.

Answer

- (a) The time taken for each I/O request cycle is $T + C$, where $C = 0.5$ sec and T is the time to satisfy the I/O request. $T_s = 2$ msec, $T_r = 0.5(60/5000) = 6$ msec, $T_d = (128 \times 1024 / 10 \times 1024 \times 1024) = 12.5$ msec, $\rightarrow T = 20.5$ msec
 $\therefore T + C = 520.5$ msec
 \rightarrow CPU time used in 5 minutes = $C / (T + C) \times 300$ seconds
 $\rightarrow = (500/520.5)(300) = 288.2$ seconds = 4.8 minutes.
 \rightarrow the kernel does not allocate any buffers

- (b) With a single buffer, the kernel driver can read the next 128K into the system buffer while the user process is accessing the current 128K of data. At the next I/O request the system buffer is copied to the user buffer:
Each I/O request cycle is $\max[T, C] + M$, where $C = 500$ msec, $T = 20.5$ msec and M is the memory-to-memory copy of 128K of data from system to user buffer, $M = (128K = 32,768 \text{ words}) \times 100 \text{ nsec/word} = 3.28$ msec
 $\therefore \max[T, C] + M = C + M = 503.28$ msec.
 \rightarrow CPU time used in 5 minutes = $C / (C + M) \times 300$ seconds
 $\rightarrow = (500/503.28)(300) = 298$ seconds = 4.97 minutes.
 \rightarrow the kernel allocates a buffer of 128K
- (c) With a double buffer, the kernel driver can read the next 128K into one system buffer while the other system buffer is available immediately to be copied to the user buffer. Each I/O request cycle is $\max[T, C+M] = C + M = 503.28$ msec and the CPU time in 5 minutes is 4.97 minutes. However the kernel needs to allocate two buffers of 128K each, 256K total. **Think!** So why use double buffering?

3. A local area network (LAN) is used as follows. The user issues a system call to write to the network. The OS then copies the data to a kernel buffer. Then it copies the data to the network controller board. When all the bytes are safely inside the controller, they are sent over the network at a rate of 10 Mbits/sec. When the last bit arrives, the destination CPU is interrupted, and the kernel copies the new data to a kernel buffer to inspect it. Once it has figured out which user they are for, the kernel copies the data to the user space. If we assume that each interrupt and its associated processing takes 1 msec, that packets are 1024 bytes (ignore the headers), and that copying a byte takes 1 μ sec, what is the maximum rate at which one process can pump data to another? Assume that the sender is blocked until the work is finished at the receiving side and an acknowledgment comes back. For simplicity, assume that the time to get the acknowledgment back is so small it can be ignored.

Answer

A packet must be copied four times during this process, which takes $4 \times 1024 \times 1 \mu\text{sec} = 4.1$ msec. There are also two interrupts (system call to send data, and interrupt when data is ready), which account for 2 msec. Finally, the transmission time is $(1024 \times 8) / (10 \times 10^6) = 0.82$ msec, for a total of 6.92 msec per 1024 bytes. The maximum data rate is thus 147,977 bytes/sec or 1.18 Mbits/sec, or about 12 percent of the nominal 10 Mbits/sec network capacity. (If we include protocol overhead the figures get even worse).

4. (a) A floppy disk has 40 cylinders (an old one anyway, most now have 80). A seek takes 6 msec per cylinder moved. If no attempt is made to put the blocks of a file close to each other, two blocks that are logically consecutive (i.e. follow each other in the file) will be about 13 cylinders apart on average. If, however, the operating system makes an attempt to cluster related blocks, the mean interblock distance can be reduced to 2 cylinders (for example). How long does it take to read a 100 block file in both cases, if the rotational latency is 100 msec and the transfer time is 25 msec per block?
- (b) Why are output files for the printer normally spooled on disk before being printed, instead of being printed directly from the application program?

Answer

- (a) The time per block is built up of three components: seek time, rotational latency and transfer time. In all cases the rotational latency plus transfer time is the same, 125 msec. Only the seek time differs. For 13 cylinders it is 78 msec; for 2 cylinders it is 12 msec. Thus for randomly placed files the total is 203 msec, and for clustered files it is 137msec.
- (b) If the printer were assigned as soon as the output appeared, a process could tie up the printer by printing a few characters and then going to sleep for a week.

5. Disk requests come in to the disk driver for cylinders 10, 22, 20, 2, 40, 6 and 38, in that order (Initially, the arm is at cylinder 20). A seek takes 6 msec per cylinder moved. How much seek time is needed for
- (a) First-Come, First Served (FCFS).
(b) Shortest Seek First (SSF).
(c) SCAN elevator algorithm (initially moving upwards)

Answer

Initially at cylinder 20 and moving up

- (a) (10,22,20,2,40,6,38) $\rightarrow 10+12+2+18+38+34+32 = 146$ cylinders = 876 msec.
(b) (20,22,10,6,2,38,40) $\rightarrow 0+2+12+4+4+36+2 = 60$ cylinders = 360 msec.
(c) (20,22,38,40,10,6,2) $\rightarrow 0+2+16+2+30+4+4 = 58$ cylinders = 348 msec.

6. A personal computer salesman visiting a university in Tasmania remarked during a sales pitch that his company had devoted substantial effort to making their version of UNIX very fast. As an example, he noted that their disk driver used the elevator algorithm and also queued multiple requests within a cylinder in sector order. A student, Harriet Hacker, was impressed and bought one. She took it home and wrote a program to randomly read 10,000 blocks spread across the disk. To her amazement, the performance measured was identical to first come first served. Was the salesman lying?

Answer

Not necessarily. A UNIX program that reads individual 10,000 blocks randomly issues the requests one at a time, blocking after each one is issued until after it is completed. Thus the disk driver sees only one request at a time; it has no opportunity to do anything but process them in the order of arrival. Harriet should have started up many processes at the same time to see if the elevator algorithm worked.