Operating Systems

211 - Peter Pietzuch <prp@doc.ic.ac.uk>

Tutorial – Disk Management

1. A disk controller with enough memory can perform *read-ahead*, reading blocks on the current track into its memory before the CPU asks for them.

Should it also do *write-behind*, i.e. report back to the CPU that a block has been written once it is stored in the disk controller's memory?

Answer: Not in general. When the disk controller signals "operation complete" after a write to disk, the OS must be able to rely on that page staying written even in the event of an immediate crash (due e.g. to power failure). Reporting operation complete before the write to disk is finished violates this rule. A controller can do write-behind if its local memory is battery-backed for long enough and if at restart, it performs the writes that were reported completed before the crash.

2. Suppose that the current position of the disk arm is over cylinder 200. The disk request queue contains requests for sectors on the following cylinders:

400, 20, 19, 74, 899

In which order will the requests be handled under:

a. the FCFS disk head scheduling policy?
b. the SSTF policy?
c. the SCAN policy?
d. the C-SCAN policy?

Briefly describe the policies and their respective trade-offs.

Answer:

FCFS: 400, 20, 19, 74, 899
SSTF: 74,20,19,400,899
SCAN: 400,899,74,20,19 (assuming the initial direction is up)
CSCAN: 400,899,19,20,74 (assuming the initial direction is up)

- FCFS (first come first served) services disk requests in the order they arrive. It is simple, but it performs well only if the average queue length is close to one.

- SSTF (shortest seek time first) services whichever requests in the queue are closest to the current position of the disk head (it doesn't matter how ties are broken). It is only a bit more complex than FCFS, has good throughput, but tends to starve requests near the edge of the disk if the system is loaded.

- SCAN (the elevator algorithm) maintains a current seek direction and two queues: one for requests ahead of the disk head and those behind. It services the nearest request in the current direction; when no such remain, it switches direction. It has almost as good throughput as SSTF and doesn't starve any process.

- CSCAN also maintains two queues, but when the current one becomes empty it seeks to the request closest to the farthest edge of the disk and resumes seeking in the same direction from there. It is slightly slower than SCAN but does not discriminate in favour of blocks at the middle of the disk.

2. Disk requests come in to the disk drive for tracks 10, 22, 20, 2, 40, 6, and 38, in that order. A seek takes 5 ms per track moved. In all cases, the arm is initially at track 20. How much seek time is needed for

   a) First-come, first served
   b) Shortest seek time first
   c) Scan scheduling (initially moving upwards)

   Answer:

   - (10 + 12 + 2 + 18 + 38 + 34 + 32) = 146 tracks * 5 ms/track = 730 ms
   - (0 + 2 + 12 + 4 + 4 + 36 + 2) * 5 ms/track = 60 tracks * 5 ms/track = 300 ms
   - (0 + 2 + 16 + 2 + 30 + 4 + 4) * 5 ms/track = 58 tracks * 5 ms/track = 290 ms

3. Consider the following parameters describing a disk:

Parameter	Description
C	Number of cylinders
T	Number of tracks per cylinder (number of platters)
S	Number of sectors per track
ω	Rotational velocity (rotations per second)
B	Number of bytes per sector

In terms of these parameters, how many bytes of data are on each disk cylinder?

Suppose that you are designing a disk drive, and that you hope to reduce the expected rotational latency for requests from the disk. Which of the parameters above would you attempt to change, and in what way would you change them?

Suppose you wanted to reduce the disk's data transfer time - which parameters would you attempt to change?

Answer:

- *bytes/cylinder = T * S * B*
- *to reduce rotational latency, increase the rotational velocity*
- *to reduce transfer time, increase the rotational velocity, or increase the quantity SB, which represents the number of bytes per track*

4. A disk drive has C cylinders, T tracks per cylinder, S sectors per track, and B bytes per sector. The rotational velocity of the platters is ω rotations per second.

Consider s_1 and s_2, consecutive sectors on the same track of the disk. (Sector s_2 will pass under a read/write head immediately after s_1.) A read request for s_1 arrives at the disk and is serviced. Exactly d seconds ($0 < d < 1/\omega$) after the disk completes that request, a read request for sector s_2 arrives at the disk. (There are no intervening requests.)

How long will it take the disk to service the request for sector s_2?

Answer:

seek time = 0
transfer time = $(1 / \omega)\ (1 / s)$
rotational latency = $1 / \omega - d$

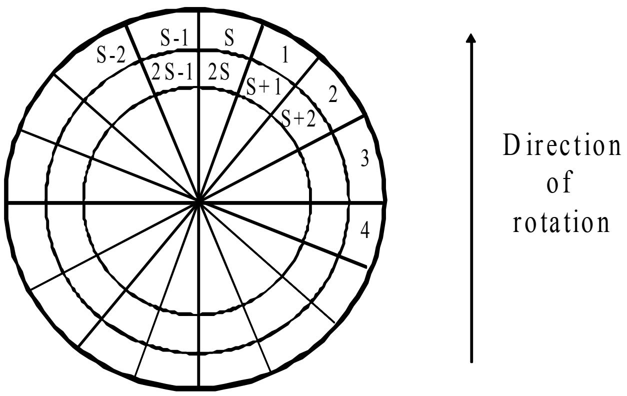
- Suppose that s_1 and s_2 are not laid out consecutively on the disk. Instead, there are k sectors between s_1 and s_2. Does this change your answer from part (a)? If so, for which value(s) of k will the time to service the read request for s_2 be minimized? (Only consider $0 \le k \le S - 2$.)

  Answer: The smallest integer k such that $(k / S)\ (1 / \omega) \ge d$, i.e. $k = \lceil dS\omega \rceil$

5. A disk drive has S sectors per track and C cylinders. For simplicity, we will assume that the disk has only one, single-sided platter, i.e., the number of tracks per cylinder is one. The platter spins at ω rotations per millisecond. The following function gives the relationship between seek distance d, in cylinders, and seek time, t_{seek}, in milliseconds:

$$t_{seek} = 0 \qquad d = 0$$
$$t_{seek} = 5 + 0.05d \qquad 0 < d \le C$$

The sectors are laid out and numbered sequentially, starting with the outer cylinder, as shown in the diagram below.



a) Suppose the disk read/write head is located over cylinder 10. The disk receives a request to read sector S.

   What is the expected service time for this request?

   Answer:

seek time = 5 + 0.05 * 10 = 5.5 ms
rotational latency = (1 / 2) (1 / ω)
transfer time = (1 / S) (1 / ω)
service time = seek time + rotational latency + transfer time

b) Exactly d milliseconds after completing the request for S, the disk receives a request for sector S + 1.

What is the expected service time for this request?

Answer:

seek time = 5 + 0.05 * 1 = 5.05 ms
rotational latency = $(1 / \omega) - (d + t_{seek})$

(This assumes that $d + t_{seek} \leq 1 / \omega$.)

transfer time = (1 / S) (1 / ω)
service time = seek time + rotational latency + transfer time

6. A disk drive has T = 1000 tracks per surface and S = 10 sectors per track. The platters spin at a rate of ω = 100 rotations per second. The following function relates seek distance, d, in cylinders, to the seek time (in milliseconds):

$$t_{seek} = 0.1d + 5$$

Sectors s_1 and s_2 are consecutive sectors on the same track of cylinder 100 (s_2 will pass under the read/write head immediately after s_1 does).

a) The read/write heads are initially located over cylinder zero. The disk receives a request for sector s_1. After servicing that request, it is idle for a time, and then receives a request for sector s_2.

What is the sum of the expected service times for these two requests?

Answer:

For the first request:

$t_{seek} = 0.1 * 100 + 5 = 15$ ms
$t_{rot} = (1/2) * 10 = 5$ ms
$t_{trans} = (1/10) * 10 = 1$ ms
$t_{service} = 21$ ms

For the second request:

$t_{seek} = 0$
t_{rot} and t_{trans} are the same as for the first request. This assumes that the delay between requests is unknown so that the disk is equally likely to be at any rotational position when the second request arrives.

$t_{service}$ = 27 ms

b) The read/write heads are initially located over cylinder zero. The disk receives a single request to read sectors s_1 and s_2.

What is the expected service time for this request?

Answer:

t_{seek} = 0.1 * 100 + 5 = 15 ms
t_{rot} = (1/2) * 10 = 5 ms
t_{trans} = (2/10) * 10 = 2 ms
$t_{service}$ = 22 ms