Disk Scheduling

Carrying out disk accesses in the order they are received will not always produce optimal performance.

Seek time is the reason for differences in performance.

For a single disk there will be a number of I/O requests.

If requests are selected randomly, we will expect poor performance.

Can use priority scheme.

Can reduce average access time by sending requests to disk controller in certain order.
First-in, first-out (FIFO)
- process request sequentially
- "fair" to all processes
- approaches random scheduling in performance if there are many processes

Request order: 55 58 39 18 90 160 150 38 184

Total distance head moves: 498
SSTF Scheduling

SSTF: shortest seek (service) time first
- select the disk I/O request that requires the least movement of the disk arm from its current position
- guarantees minimum average seek time, but can lead to starvation

Request order: 55 58 39 18 90 160 150 38 184
Actual order: 90 58 55 39 38 18 150 160 184

Total distance head moves: 248
SCAN Scheduling

SCAN: “elevator algorithm”
- arm moves in one direction only, satisfying all outstanding requests until it reaches the last track in that direction
- then direction is reversed

Request order: 55 58 39 18 90 160 150 38 184
Actual order: 150 160 184 90 58 55 39 38 18

Total distance head moves: 250
C-SCAN Scheduling

C-SCAN:
- restricts scanning to one direction only
- when the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again
- more uniform waiting times
- "fairer" than SCAN

Request order: 55 58 39 18 90 160 150 38 184
Actual order: 150 160 184 18 38 39 55 58 90

Total distance head moves: 312
Other Variations

N-step-SCAN
- Segments the disk request queue into subqueues of length N
- Subqueues are processed one at a time, using SCAN
- New requests added to other queue when queue is processed

FSCAN
- Two queues
- One queue is empty for new requests
### I/O Scheduling

#### Comparison

<table>
<thead>
<tr>
<th>(a) FIFO</th>
<th>(b) SSTF</th>
<th>(c) SCAN</th>
<th>(d) C-SCAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>(starting at track 100)</td>
<td>(starting at track 100)</td>
<td>(starting at track 100, in the direction of increasing track number)</td>
<td>(starting at track 100, in the direction of increasing track number)</td>
</tr>
<tr>
<td><strong>Next track accessed</strong></td>
<td><strong>Number of tracks traversed</strong></td>
<td><strong>Next track accessed</strong></td>
<td><strong>Number of tracks traversed</strong></td>
</tr>
<tr>
<td>55</td>
<td>45</td>
<td>90</td>
<td>10</td>
</tr>
<tr>
<td>58</td>
<td>3</td>
<td>58</td>
<td>32</td>
</tr>
<tr>
<td>39</td>
<td>19</td>
<td>55</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>21</td>
<td>39</td>
<td>16</td>
</tr>
<tr>
<td>90</td>
<td>72</td>
<td>38</td>
<td>1</td>
</tr>
<tr>
<td>160</td>
<td>70</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>150</td>
<td>10</td>
<td>150</td>
<td>132</td>
</tr>
<tr>
<td>38</td>
<td>112</td>
<td>160</td>
<td>10</td>
</tr>
<tr>
<td>184</td>
<td>146</td>
<td>184</td>
<td>24</td>
</tr>
<tr>
<td><strong>Average seek length</strong></td>
<td><strong>Average seek length</strong></td>
<td><strong>Average seek length</strong></td>
<td><strong>Average seek length</strong></td>
</tr>
<tr>
<td>55.3</td>
<td>27.5</td>
<td>27.8</td>
<td>35.8</td>
</tr>
</tbody>
</table>
Operating systems are the best place to manage the scheduling of disk accesses.

Problem: high-level interfaces like ATA and SCSI provide the OS with logical block addresses, not physical disk addresses.
Host-Ordered vs Drive-Ordered

FIGURE 6.19 Example showing OS versus disk schedule accesses, labeled host-ordered versus drive-ordered. The former takes three revolutions to complete the four reads, while the latter completes them in just three-fourths of a revolution (from Anderson [2003]). Copyright © 2009 Elsevier, Inc. All rights reserved.