操作系统原理第 12 章作业

姓名：马福泉　学号：23336179　截止日期：2025 年 6 月 18 日

完成日期：2025 年 6 月 4 日

Question 1：12.1 除了 FCFS，就没有其他的磁盘调度算法是真正公平的(可能会出现饥饿)。

a.说明为什么这个断言是真。

b.描述一个方法，修改像 SCAN 这样的算法以确保它公平。

c.说明为什么在分时系统中公平是一个重要的目标。

d.给出三个以上例子，在这些情况下操作系统在服务请求时“不公平”很重要。

Answer 1：（a）FCFS 是按照请求到达的顺序进行服务，因此每个请求都会在它到达后按照顺序被处理，不会出现某些请求被无限期推迟的情况,因此，FCFS 是公平的。

其他常见的磁盘调度算法包括：

SSTF：选择离当前磁头位置最近的请求。这可能导致远离磁头的请求长时间得不到服务（饥饿）。

SCAN ：磁头沿一个方向移动，服务路径上的请求，到达一端后反向移动。如果一个请求刚刚被磁头“错过”，可能需要等待磁头往返一次才能被服务，极端情况下等待时间较长。

C-SCAN：类似 SCAN，但只在一个方向服务请求，返回时不服务。返回时的请求需要等待更长时间。

LOOK 和 C-LOOK：SCAN 和 C-SCAN 的改进版，只在有请求的方向移动。类似的问题，只是减少了不必要的移动。

（b）限制请求的等待时间：可以为每个请求设置一个最大等待时间阈值。如果某个请求等待时间超过阈值，则优先服务该请求。

使用 FSCAN：FSCAN 是 SCAN 的变种，它将请求队列分为两个子队列。在一个扫描周期中只服务一个队列的请求，新到达的请求放入另一个队列。这样可以防止新请求不断抢占服务，确保当前队列的请求都能被服务。

（c）用户体验：分时系统需要为每个用户提供及时的响应。如果某些用户的请求长时间得不到处理，会导致用户等待时间过长。

资源分配的合理性：公平的调度算法可以确保每个用户都能在合理的时间内获得资源，避免某些用户独占资源，从而提高系统的整体利用率。

多任务处理：分时系统通常需要同时处理多个任务，公平的调度算法可以确保每个任务都能获得公平的处理机会，提高系统的并发处理能力。

（d）实时系统：在实时操作系统中，某些高优先级的任务（如航空控制系统）的 I/O 请求必须优先处理，即使这意味着低优先级任务可能被延迟。不公平的调度可以确保关键任务的及时完成。

高优先级进程：操作系统可能赋予某些进程更高的 I/O 优先级（如系统守护进程或关键服务）。这些进程的 I/O 请求会被优先处理，以确保系统的正常运行。

缓存预热：操作系统可能优先处理那些有助于提高缓存命中率的 I/O 请求（如顺序读取），而暂时忽略随机读取请求。这种“不公平”可以提高整体 I/O 效率。

防止死锁：资源紧张时，操作系统可能优先处理某些 I/O 请求以释放资源，避免系统陷入死锁状态。

Question 2：

12.2 假设一个磁盘驱动器有 5000 个柱面，从 0~4999。驱动器正在为柱面 143 的一个请求提供服务，且前面的一个服务请求是在柱面 125。按 FIFO 顺序，即将到来的服务队列是 86，1470，913，1774，948，1509，1022，1750，130 从现在磁头位置开始，按照下面的磁盘调度算法，要满足队列中的服务要求磁头总的移动距离是多少?

a. FCFS

b. SSTF

C. SCAN

d. LOOK

e.C-SCAN

f. C-LOOK

Answer 2：

a. FCFS:

143 → 86: |143 - 86| = 57

86 → 1470: |1470 - 86| = 1384

1470 → 913: |1470 - 913| = 557

913 → 1774: |1774 - 913| = 861

1774 → 948: |1774 - 948| = 826

948 → 1509: |1509 - 948| = 561

1509 → 1022: |1509 - 1022| = 487

1022 → 1750: |1750 - 1022| = 728

1750 → 130: |1750 - 130| = 1620

b. 总移动距离 = 57 + 1384 + 557 + 861 + 826 + 561 + 487 + 728 + 1620 = 7081

SSTN

当前位置：143

最近：130 (|143-130|=13), 86 (|143-86|=57) → 选择 130 移动：13

当前位置：130

最近：86 (|130-86|=44), 1470 (|1470-130|=1340) → 选择 86 移动：44

当前位置：86

最近：913 (|913-86|=827), 其他更大 → 选择 913 移动：827

当前位置：913

最近：948 (|948-913|=35), 1022 (|1022-913|=109) → 选择 948 移动：35

当前位置：948

最近：1022 (|1022-948|=74), 1470 (|1470-948|=522) → 选择 1022 移动：74

当前位置：1022

最近：1470 (|1470-1022|=448), 其他更大 → 选择 1470 移动：448

当前位置：1470

最近：1509 (|1509-1470|=39), 1750 (|1750-1470|=280) → 选择 1509 移动：39

当前位置：1509

最近：1750 (|1750-1509|=241), 1774 (|1774-1509|=265) → 选择 1750 移动：241

当前位置：1750 移动：|1774 - 1750| = 24

总移动距离 = 13 + 44 + 827 + 35 + 74 + 448 + 39 + 241 + 24 = 1745

c. SCAN

向外移动（143 → 4999）：经过的请求：1470, 1509, 1750, 1774

移动顺序：143 → 1470（移动 1327）1470 → 1509（移动 39）1509 → 1750（移动 241）1750 → 1774（移动 24）1774 → 4999（移动 3225，直到磁盘末端）

向外总移动：1327 + 39 + 241 + 24 + 3225 = 4856

向内移动（4999 → 0）：经过的请求：1022, 948, 913, 130, 86

移动顺序：4999 → 1022（移动 3977）1022 → 948（移动 74）948 → 913（移动 35）913 → 130（移动 783）130 → 86（移动 44）86 → 0（移动 86，直到磁盘起点）

向内总移动：3977 + 74 + 35 + 783 + 44 + 86 = 4999

4856（向外） + 4999（向内） = 9855

d. LOOK

向外移动（143 → 1774）：经过的请求：1470, 1509, 1750, 1774

移动顺序：143 → 1470（移动 1327）1470 → 1509（移动 39）1509 → 1750（移动 241）1750 → 1774（移动 24）向外总移动：1327 + 39 + 241 + 24 = 1631

向内移动（1774 → 86）：经过的请求：1022, 948, 913, 130, 86

移动顺序：1774 → 1022（移动 752）1022 → 948（移动 74）948 → 913（移动 35）913 → 130（移动 783）130 → 86（移动 44)向内总移动：752 + 74 + 35 + 783 + 44 = 1688

总移动距离:1631（向外） + 1688（向内） = 3319

e. CSCAN

向外移动（143 → 4999）：经过的请求：1470, 1509, 1750, 1774

向外总移动：1327 + 39 + 241 + 24 + 3225 = 4856

直接返回 4999 → 0（不处理请求）：移动 4999

重新向外移动（0 → 86 → 130 → 913 → 948 → 1022）：

重新向外总移动：86 + 44 + 783 + 35 + 74 = 1022

C-SCAN 总移动距离

4856（向外） + 4999（返回） + 1022（重新向外） = 10877

f. CLOOK

向外移动（143 → 1774）：经过的请求：1470, 1509, 1750, 1774

向外总移动：1327 + 39 + 241 + 24 = 1631

直接跳到最低请求（1774 → 86）：

移动 1774 - 86 = 1688

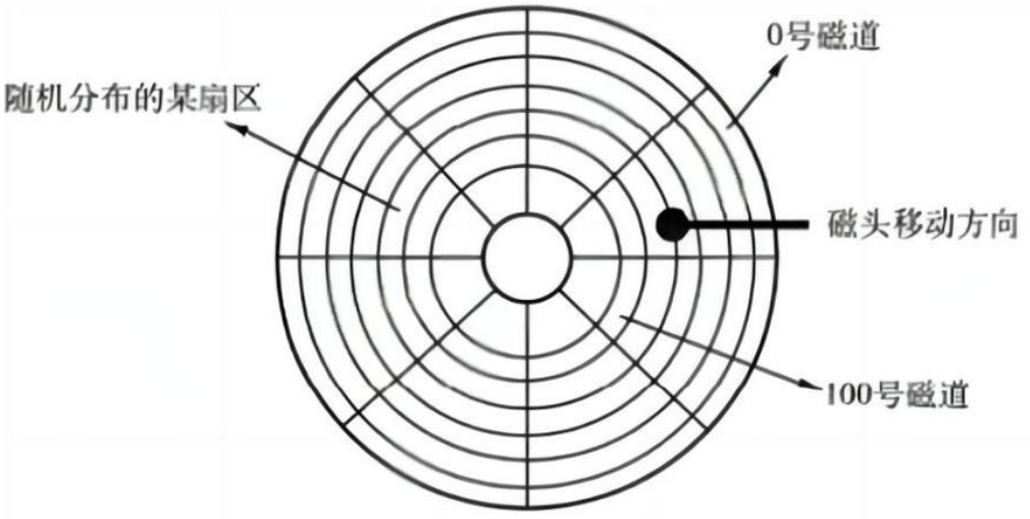
继续向外移动（86 → 130 → 913 → 948 → 1022）：

继续向外总移动：44 + 783 + 35 + 74 = 936

总移动距离:1631（向外） + 1688（跳回） + 936（继续向外） = 4255

Question 3：

1. 如下图所示，假设计算机系统采用C-SCAN(循环扫描)磁盘调度策略，使用 2KB的内存空间记录16384个磁盘块的空闲状态。



1)请说明在上述条件下如何进行磁盘块空闲状态的管理。

2)设某单面磁盘的旋转速度为 6000 转/分，每个磁道有 100 个扇区，相邻磁道间的平均移动时间为 1ms。若在某时刻，磁头位于100号磁道处，并沿着磁道号增大的方向移动(见上图)，磁道号请求队列为 50，90，30，120，对请求队列中的每个磁道需读取1个随机分布的扇区，则读完这4个扇区点共需要多少时间?要求给出计算过程。

3)若将磁盘替换为随机访问的 Flash 半导体存储器(如U盘、固态硬盘等)，是否有比C-SCAN 更高效的磁盘调度策略?若有，给出磁盘调度策略的名称并说明理由;若无，说明理由。

Answer 3：

1) 位图：使用一个位图（Bitmap）来表示磁盘块的使用情况。每个位对应一个磁盘块，1 表示空闲，0 表示已使用。由于需要使用 2KB 的内存来记录 16384 个磁盘块的空闲状态，每个磁盘块用 1 位表示，总共需要 16384 位，即 2KB。

2)

① 从 100 到 120：

移动磁道数：120 - 100 = 20。

移动时间：20 * 1ms = 20ms。

读取一个随机扇区：

平均旋转延迟：0.5 * 10ms = 5ms。

传输时间：0.1ms。

总时间：20ms + 5ms + 0.1ms = 25.1ms。

② 从 120 返回到 0：

假设最大磁道号为 120，返回到 0 的移动时间：120 * 1ms = 120ms。

从 0 到 50：

移动磁道数：50 - 0 = 50。

移动时间：50 * 1ms = 50ms。

读取一个随机扇区：

平均旋转延迟：5ms。

传输时间：0.1ms。

总时间：50ms + 5ms + 0.1ms = 55.1ms。

③ 从 50 到 90：

移动磁道数：90 - 50 = 40。

移动时间：40 * 1ms = 40ms。

读取一个随机扇区：

平均旋转延迟：5ms。

传输时间：0.1ms。

总时间：40ms + 5ms + 0.1ms = 45.1ms。

④ 从 90 到 30：

移动磁道数：90 - 30 = 60。

移动时间：60 * 1ms = 60ms。

读取一个随机扇区：

平均旋转延迟：5ms。

传输时间：0.1ms。

总时间：60ms + 5ms + 0.1ms = 65.1ms。

⑤ 总时间：25.1ms + 120ms + 55.1ms + 45.1ms + 65.1ms= 310.4ms。

3) 对于 Flash 存储器（如 U 盘、固态硬盘等）,由于其随机访问特性，不需要像磁盘那样进行寻道和旋转延迟的计算。因此，可以采用更高效的调度策略：随机访问调度策略：由于 Flash 存储器支持随机访问，可以直接访问任意位置的数据，无需等待磁头移动或磁盘旋转。因此对于 Flash 存储器，随机访问的先来先服务（FCFS）的调度策略比 C-SCAN 更高效

Question 4：某计算机系统中的磁盘有 300 个柱面，每个柱面有 10 个磁道，每个磁道有 200 个扇 区，扇区大小为 512B。文件系统的每簇包含 2 个扇区。请回答下列问题:

1) 磁盘的容量是多少?

2)设磁头在 85 号柱面上，此时有 4 个磁盘访问请求，簇号分别为 100260，60005，101660 和 110560。采用最短寻道时间优先 SSTF 调度算法，系统访问簇的先后次序是什么?

3)簇号 100530 在磁盘上的物理地址是什么?将簇号转换成磁盘物理地址的过程由 I/O 系统的什么程序完成?

Answer 4：

1) 每个柱面有 10 个磁道，每个磁道有 200 个扇区，所以每个柱面有 10×200=2000 个扇区。

磁盘有 300 个柱面，所以总扇区数为 300×2000=600000 个扇区。

每个扇区的大小为 512 字节，所以磁盘的总容量为：

600000×512 B=307200000 B=300 MB

2)

计算每个柱面的扇区数：

每柱面扇区数=每柱面磁道数×每磁道扇区数=10×200=2000 扇区

计算每个柱面的簇数：

每柱面簇数=每柱面扇区数÷每簇扇区数=1000 簇

柱面号=簇号÷每柱面簇数（向下取整）

100260 → 100

60005 → 60

101660 → 101

110560 → 110

SSTF 选择顺序：100（距离 15）然后是 101（距离 1）接着是 110（距离 9）最后是 60

SSTF 访问顺序：100, 101, 110, 60 对应的簇号顺序：100260, 101660, 110560, 600053)计算柱面号：柱面号=100

计算簇在柱面内的偏移量：偏移簇号=530

扇区号=簇号×每簇扇区数=530×2=1060

计算磁道号和扇区偏移：每磁道扇区数：200

磁道号=1060/200=5

扇区偏移=1060mod200=60

物理地址

柱面号（Cylinder）：100

磁头号（Head）：5

扇区号（Sector）：60

转换程序:由 I/O 系统的磁盘驱动程序（Disk Driver）完成。具体来说，是磁盘调度程序（Disk Scheduler）或块设备驱动（Block Device Driver）负责将逻辑簇号映射为物理地址（CHS 或 LBA 格式）。