HW11

姓名: 陈锐林, 学号:21307130148

2023年11月29日

Chapter37-HDD

Question1:

(1) 以最复杂的 7,30,8 为例: 首先需要转 15° 开始传输 7, 共用时 15+30; 之后寻道进入最内圈, 花了 40*2=80, 需再转 300-80=220, 之后传输 30, 共花 330; 之后要再寻道出来, 花 80, 这时和传输'8' 的起始位置差 50°, 所以得转 310, 共花 420。所以在这三个阶段各花 45,330,420。(2) 而 10,11,12,13 就是分成两批次, 不寻道 + 转 + 连续传输和寻道 + 转 + 传输。对于题目中的所有情况,结果如下:

| 命令 | 用时 | 过程 |
|----------------|-----|-----------------------------------|
| -a 0 | 195 | 0+165+30 |
| -a 6 | 375 | 0+345+30 |
| -a 30 | 375 | 80+265+30 |
| -a 7,30,8 | 795 | (0+15+30)+(80+220+30)+(80+310+30) |
| -a 10,11,12,13 | 585 | (0+105+30+30)+(40+320+30+30) |

Question4:

- (1)FIFO 中,很明显,像 Question1 讨论的,需要两次寻道,可能是增加了开销的。
- (2) 如果采用 SSTF, 过程应该如下: 转 15, 连续传输"7","8", 用时 60, 总共 75; 再寻道进入内圈, 用时 80, 旋转 190, 传输 30, 用时 300。总用时为 375。所以总结如下: 对"7"为 0+15+30; 对"8"为 0+0+30; 对"30"为 80+190+30。

Question 10:

如果采用序列-a 20,11,25,那么这 4 种策略都会给出 585 的结果,因为他们总是把 25 放在最后解决。但如果用 20-25-11 的顺序就能用时更短。具体过程如下:"20":40+5+30;"25":40+80+30;"11":80+190+30。共用时 525。这说明贪心的策略并不总是最优的。

Chapter38-RAID

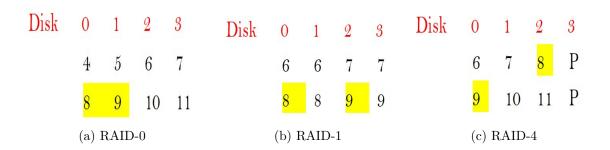
Question1:

- (1)/(3) 我使用了类似"python3 ./raid.py -n 5 -L 0 -R 20 -c -s 0" 的指令来理解; 其中-L 取 0,1,4,5,表示 RAID level; -s 取不同值表示不同随机种子。
- (2) 对于 RAID-5 的左对称和左不对称问题,我使用这两条指令"python3./raid.py-n5-L5-R20-c-5 LS"和"python3./raid.py-n5-L5-R20-c-5 LA",这个命令分别要求访问地址 8,9,10,15,16。最后输出的结果和我认知的左对称和左不对称格局 (如下)是一样的:排列时,左对称每行起始地址在上一行的检验和下,之后再补全;左不对称则直接按从左到右的顺序排。

| Disk | 0 | 1 | 2 | 3 | | Disk | 0 | 1 | 2 | 3 |
|---------|----|----|----|----|----------|------|----|----|----|----|
| | 0 | 1 | 2 | P | | | 0 | 1 | 2 | P |
| LS | 4 | 5 | P | 3 | | LA | 3 | 4 | P | 5 |
| | 8 | P | 6 | 7 | | | 6 | P | 7 | 8 |
| | P | 9 | 10 | 11 | | | P | 9 | 10 | 11 |
| | 12 | 13 | 14 | P | | | 12 | 13 | 14 | P |
| | 16 | 17 | P | 15 | | | 15 | 16 | P | 17 |
| (a) 左对称 | | | | | (b) 左不对称 | | | | | |

Question4:

(1) 随着申请块的大小的增加,在读取时就要跨越 Disks。对于 RAID-0,如读地址 8,就要用到 D0(off-2), D1(off-2)。对于 RAID-1,可能就是读 D0(off-4),D2(off-4)。 RAID-4 读取为 D2(off-2),D0(off-0)。而对于 RAID-5,如上面给出的,左对称时就是 D0(off-2),D1(off-3); 左不对称时是 D3(off-2),D1(off-3).



(2) 对于顺序读取的情况,设置大小为 8K,就会读 0,2,4,6,8,...; 设置大小为 12K,就会读取 0,3,6,9,12,...; 为了让 I/O 更加高效,在每一次的申请读数据时要 让大小更大些,比如这里可以采用 12K 和 16K; 让更多的磁盘同时动起来。