中国科学院大学
University of Chinese Academy of Sciences

磁盘和RAID

中国科学院大学计算机学院

2025-12-17

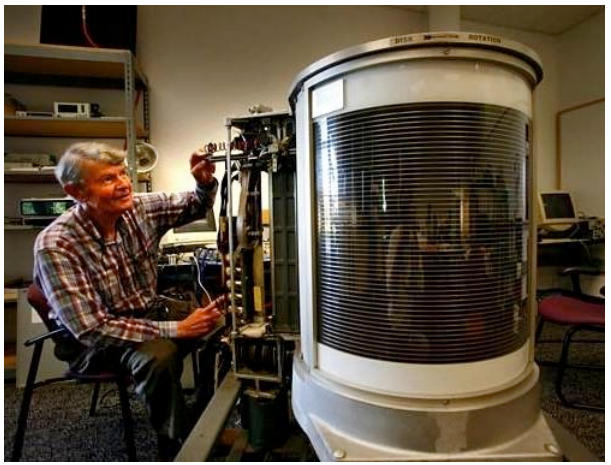
内容提要

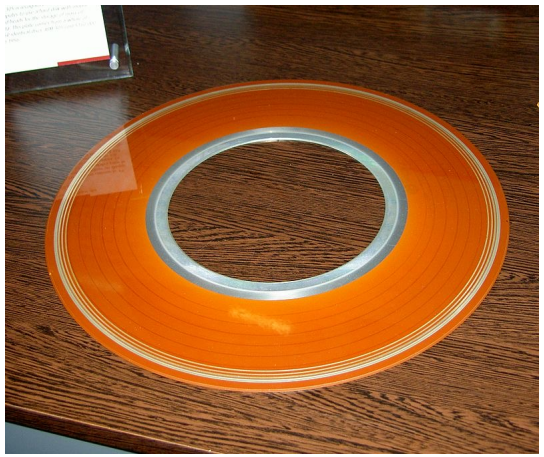
- 磁盘
  - 内部结构
  - 性能特性
  - 磁盘调度
- RAID (磁盘阵列)

第一块磁盘

1956，IBM 305 RAMAC

- 50个盘片（disc）
- 总容量5MB
- 每个盘片 24英寸
- 体积：1.9 m^3
- 重量：910 kg





https://en.wikipedia.org/wiki/IBM_305_RAMAC

磁盘（Hard Disk）

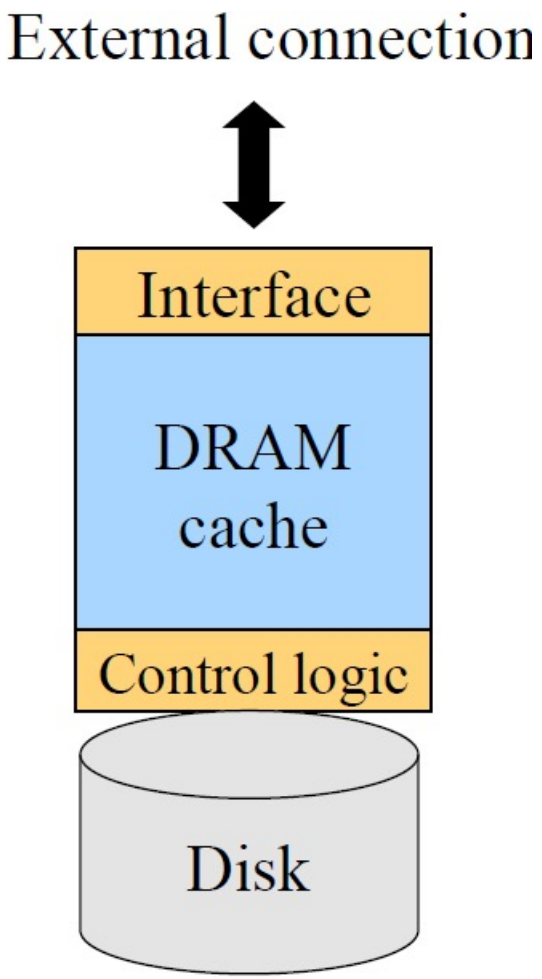
- 持久化、大容量、低成本的存储设备：机械操作，速度慢
- 多种尺寸： 3.5英寸, 2.5英寸
- 多种容量：100GB ~ 14TB
- 多种接口

3.5英寸
12TB

2.5英寸
5TB

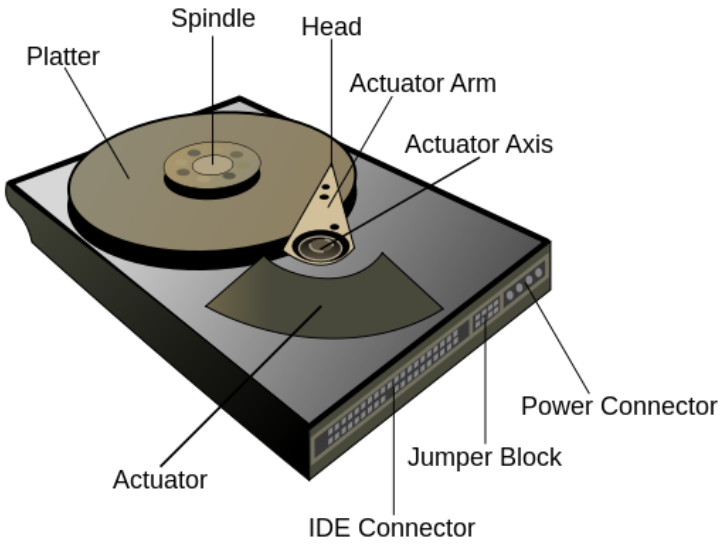
典型的磁盘控制器

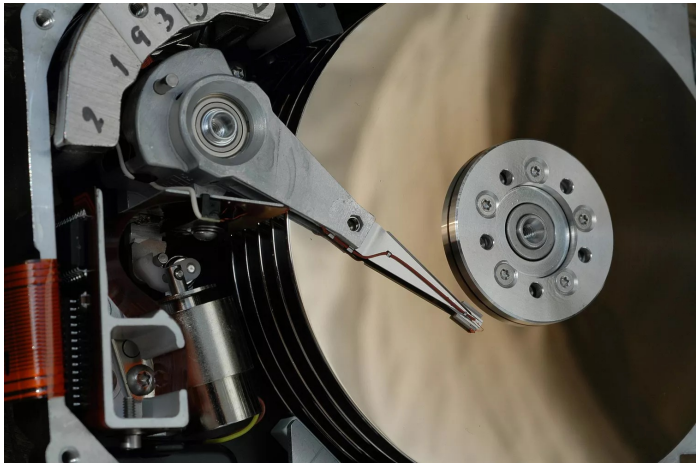
- 与主机的接口
  - SATA (1.0, 2.0, 3.0) , ATA
    - 面向对延迟、吞吐率要求不高，大容量的场景
  - SAS (1, 2, 3)， SCSI / Ultra-SCSI
    - 面向低延迟，高吞吐率，高可靠场景
  - FC: Fiber channel
- 缓存
  - 缓冲数据
- 控制逻辑
  - 读写操作
  - 请求调度
  - 缓存替换
  - 坏块检测和重映射



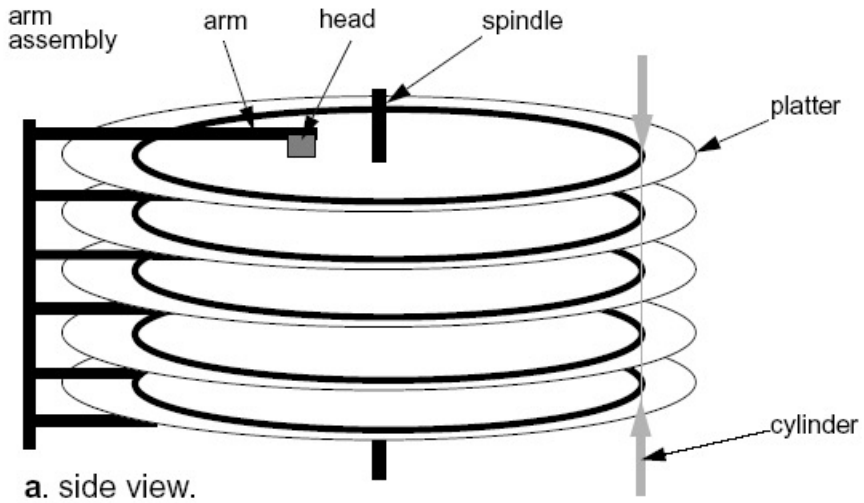
磁盘的结构

- 盘片：一组
  - 按一定速率旋转
- 磁道 (Track)
  - 位于盘片表面的同心圆
  - 用于记录数据的磁介质
  - bit沿着每条磁道顺序排列
- 扇区 (Sector)
  - 磁道划分为固定大小的单元
一般为512字节
- 磁头：一组
  - 用于读写磁道上的数据
- 磁臂：一组
  - 用于移动磁头（多个）

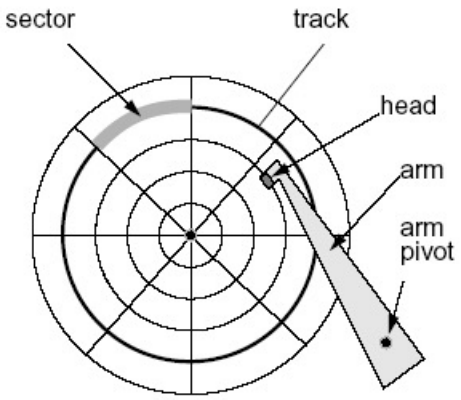




磁盘的结构



a. side view.

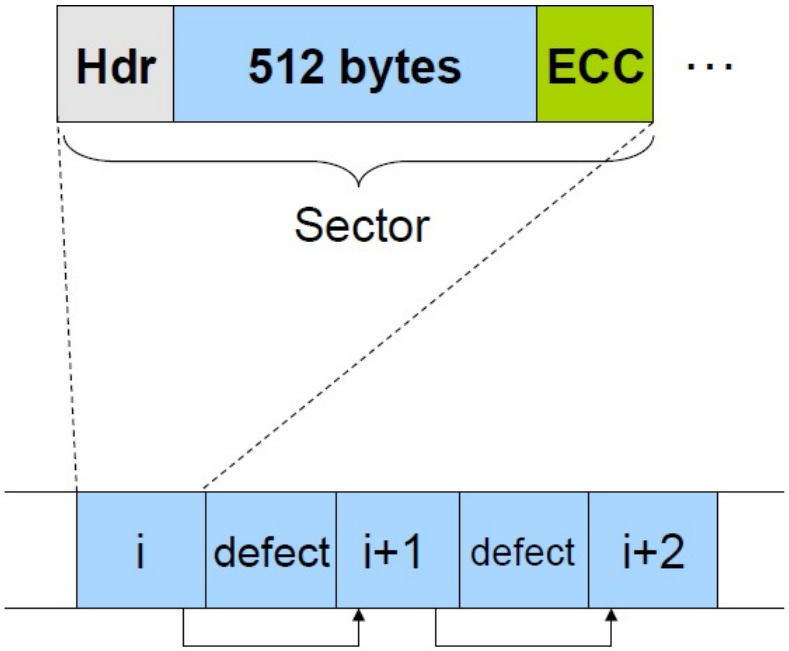


b. top view.

- 柱面 (Cylinder)
  - 由所有盘片上半径相同的磁道组成
- Zone
  - 不同磁道的扇区个数不同：外道多，内道少
  - 所有柱面划分为zone，同一zone中每条磁道的扇区数相同
  - 1000-5000个柱面/zone，其中几个为备用柱面（spare cylinder）

磁盘扇区（Sector）

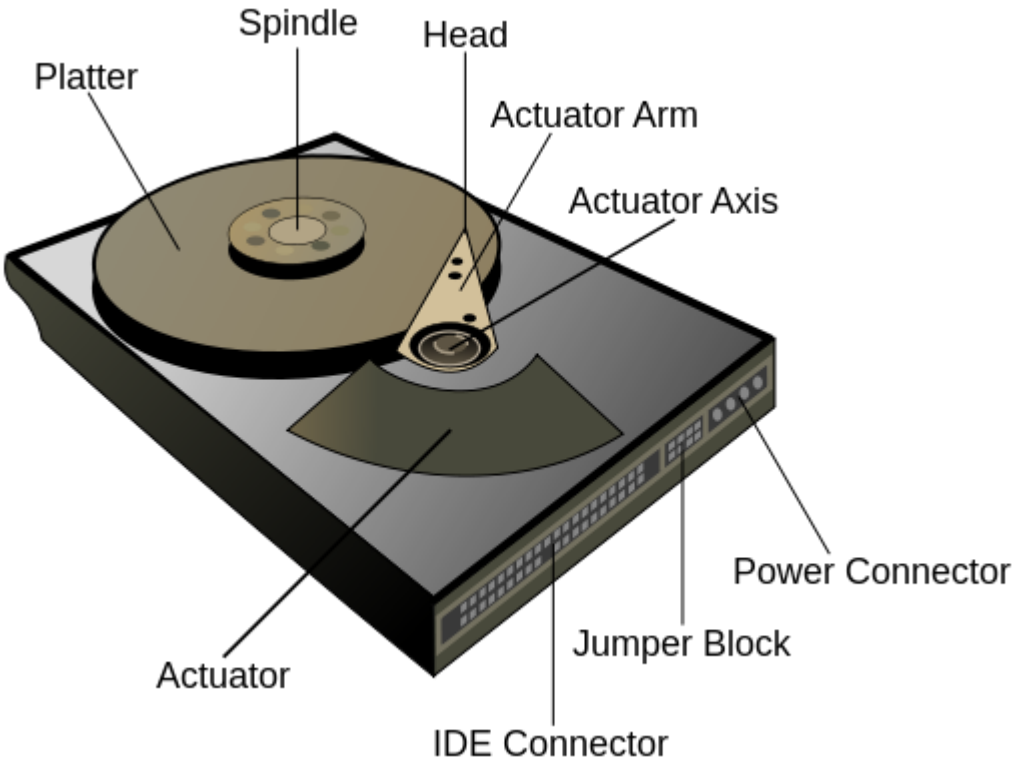
- 扇区的创建
  - 磁盘格式化
  - 逻辑块地址映射到物理块地址
- 扇区的格式
  - 头部：ID，损坏标志位，…
  - 数据区：实际用于存储数据的区域，典型大小为512B
  - 尾部：ECC校验码
- 坏扇区
  - 发现坏扇区 → 先用ECC纠错
  - 如果不能纠错，用备用扇区替代
  - 坏扇区不再使用
- 磁盘容量
  - 格式化损耗20%左右：每个扇区的头部和尾部 + 坏扇区



读写操作

读写某个柱面的某个扇区

- 定位柱面，移动磁臂使磁头对准柱面 → 寻道 seek
- 等待扇区旋转到磁头下方 → 旋转 rotation
- 进行数据读写 → 数据传输 transfer



60年的变化（来源Mark Kryder @SNW 2006）

	IBM RAMAC (1956)	Seagate ST4000NM0035 (2019)	变化
容量	5MB	4TB	800,000 ↑
价格	$1,000/MB	￥0.25/GB	↓
转速	1,200 RPM	7,200 RPM	6 ↑
寻道时间	600 ms	4.16 ms	144 ↓
传输速率	10KB/s	226MB/s	22,600 ↑
功耗	5000 W	6.9W	725 ↓
重量	910 KG	680 g	↓

磁盘性能

- 有效带宽 = 数据量 /耗时
- 耗时
  - 寻道时间（seek time）
    - 把磁头移动到目标柱面的时间
    - 典型：3.5 ~ 9.5ms
  - 旋转延迟（rotation delay)
    - 等待目标扇区旋转到磁头下方的时间
    - 典型：7,200 ~ 15,000 RPM
  - 数据传输时间（data transfer time）
    - 典型传输带宽：70~250 MB/sec
- 例子：
  - 假设BW=100MB/s, seek=5ms, rotation=4ms
  - 访问1KB数据的总时间= 5ms + 4ms + (1KB/100MB/s) = 9.01ms
  - 寻道时间和旋转延迟占99.9%
  - 有效带宽 = ?
  - 访问1MB数据的有效带宽呢？

磁盘性能

- 一次传输多少数据才能达到磁盘带宽的90%？
  - 假设磁盘BW=100MB/s, seek=5ms , rotaion=4ms
  - BW×90% = size / (size/BW + rotation + seek)
  - size = BW× (rotation + seek) × 0.9/(1-0.9)
    = 100MB×0.009*9 = 8.1MB

Block Size (Kbytes)	% of Disk Transfer Bandwidth
9Kbytes	1%
100Kbytes	10%
0.9Mbytes	50%
8.1Mbytes	90%

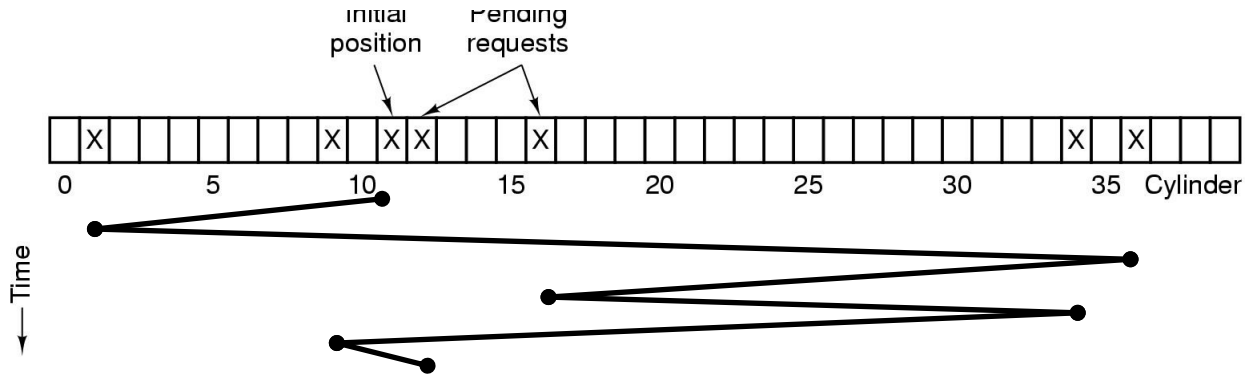
- 对于小粒度的访问，时间主要花在寻道时间和旋转时间上
  - 磁盘的传输带宽被浪费
  - 缓存：每次读写邻近的多个扇区，而不是一个扇区
  - 调度算法：减少寻道开销

磁盘缓存

- 方法
  - 磁盘内用少量DRAM来缓存最近访问的块
    - 典型大小为 64~256MB
  - 由控制器管理，OS无法控制
  - 块替换策略：LRU
- 优点
  - 如果访问具有局部性，读性能受益
- 缺点
  - 需要额外的机制来保障写的可靠性

磁盘寻道算法FIFO (FCFS)

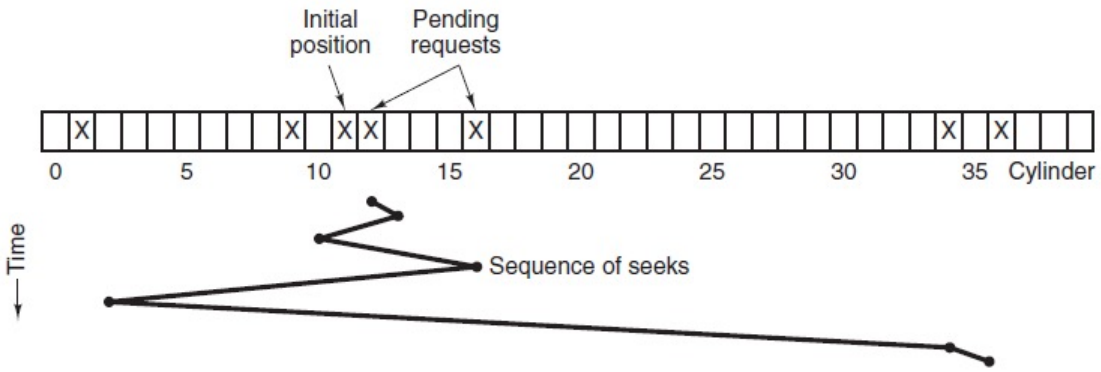
- 例子



  - 请求到达顺序：11→1→36→16→34→9→12（柱面编号）
  - FIFO服务顺序：11→1→36→16→34→9→12
  - FIFO总寻道距离：10+35+20+18+25+3 = 111
- 好处
  - 公平性
  - 磁盘请求的服务顺序是可以预期的
- 不足
  - 请求到来的随机性，可能导致长距离寻道
  - 可能发生极端情况：比如横扫整个磁盘

磁盘调度SSF (Shortest Seek First)

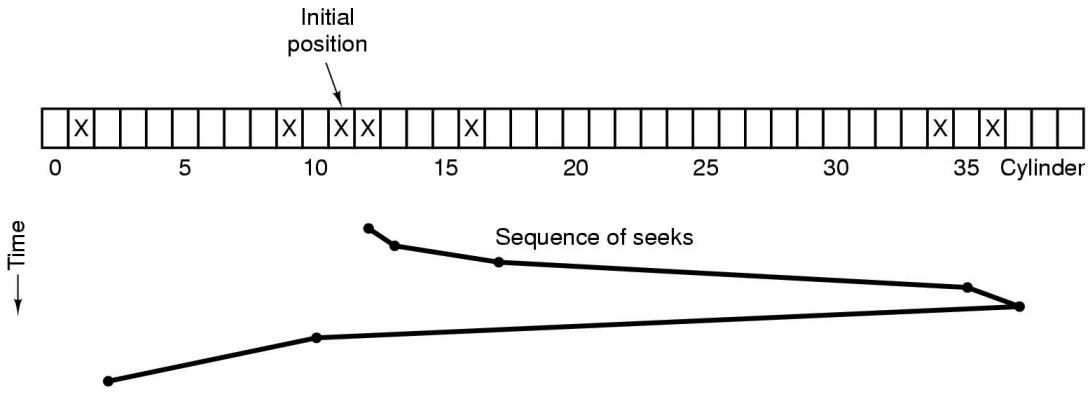
- 方法
  - 选择磁头移动距离最短的请求（需要缓冲一部分请求）
  - 计入旋转时间
  - 请求到达顺序：11→1→36→16→34→9→12（柱面编号）
  - SSF服务顺序：11→12→9→16→1→34→36
  - SSF总寻道距离：1+3+7+15+33+2 = 61
- 好处
  - 试图减少寻道距离
- 不足
  - 可能产生饥饿



电梯调度 (SCAN/LOOK)

- 方法
  - 磁头按一个方向到另一端，再折回，按反方向回到这端，不断往返
  - 只服务当前移动方向上寻道距离最近的请求
  - LOOK：如果磁盘移动方向上没有请求，就折回
  - 请求到达顺序：11→1→36→16→34→9→12（柱面编号）
  - SCAN服务顺序：11→12→16→34→36→9→1
  - SCAN总寻道距离：1+4+18+2+27+8 = 60

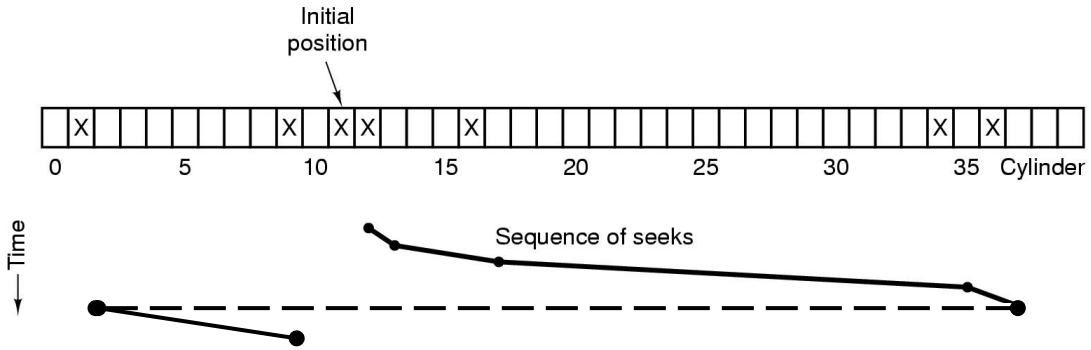
- 好处
  - 消除饥饿：请求的服务时间有上限
  - 减少磁头随机移动，提升性能
- 不足
  - 反方向的请求需等待更长时间



C-SCAN/C-LOOK (Circular SCAN)

- 方法
  - 将SCAN算法改为折回时不服务请求，立即回到磁盘最外层重新向内扫描
- 好处
  - 避免反方向请求等待过长时间，

  服务时间更加均匀
- 不足
  - 请求分布不均时，磁头频繁返回起点并不必要，反而降低效率



磁盘调度算法

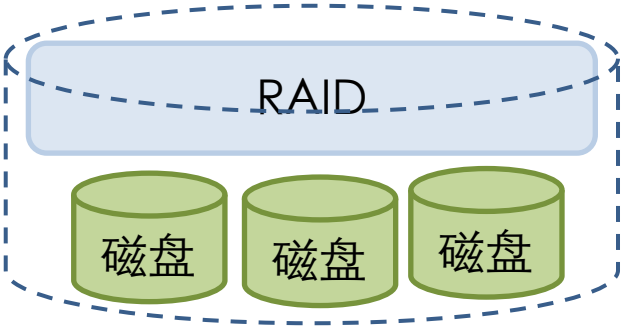
- 调度算法
  - FIFO：实现简单，但寻道时间长
  - SSF：贪心算法，可能造成饥饿现象（距离初始磁头位置较远的请求长期得不到服务）
  - SCAN/LOOK：减少饥饿和随机移动
  - C-SCAN/C-LOOK：避免SCAN算法返回时，反方向请求等待时间过长
- 磁盘I/O请求缓冲
  - 把请求缓冲在控制器缓冲区，便于进行调度
  - 缓冲区大，调度机会增加
- 优化方向
  - 既寻道最短，又旋转延迟最短

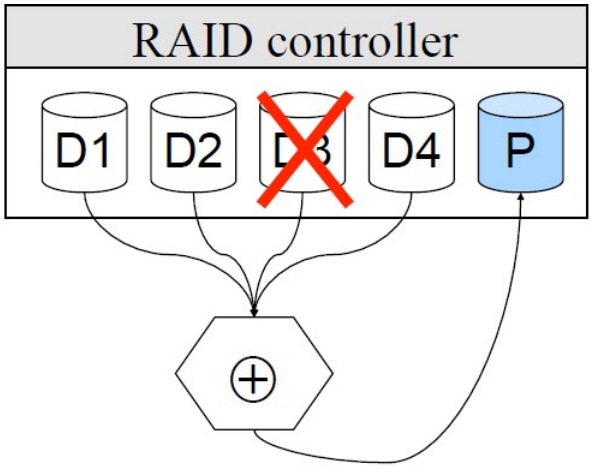
内容提要

- 磁盘
- RAID (磁盘阵列)

RAID (Redundant Array of Independent Disks)

- 主要思想
  - 由多个磁盘构成一个存储设备
- 好处
  - 提高性能：多个磁盘并行工作
  - 增加容量：聚合多个磁盘的空间
  - 提高可靠性：数据冗余，有磁盘损坏时，数据不损坏
- 开销
  - 控制器变得复杂
- 牵涉的问题
  - 多块盘做块映射：逻辑块LBN → <磁盘#, 块#>
  - 如何做冗余机制保护数据可靠性



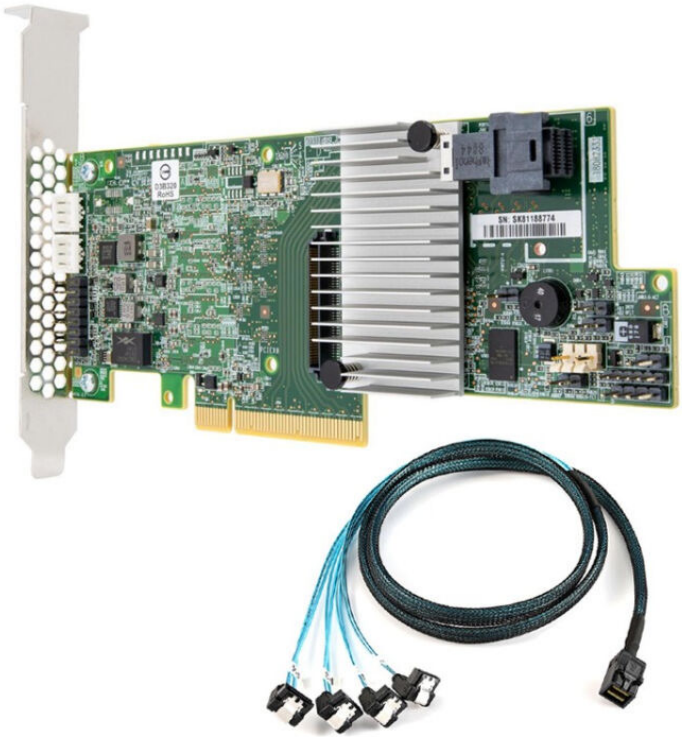


$P = D1 \oplus D2 \oplus D3 \oplus D4$

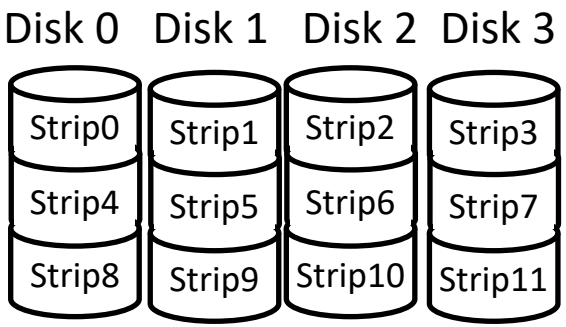
$D3 = D1 \oplus D2 \oplus P \oplus D4$

RAID (Redundant Array of Independent Disks)

- 示例



RAID-0



Disk 0	Disk 1	Disk 2	Disk 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

RAID Level 0

- 以条带（stripe）为粒度映射到N块磁盘（轮转方式），条带宽度为N，即有N个条（strip）组成
- 1个strip = K个块，即1个条由K个块组成
- 无冗余

容量

- N × 单个磁盘容量

可靠性

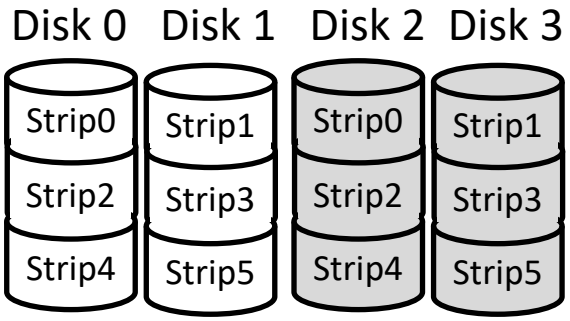
- （单个磁盘可靠性）^{N}

性能

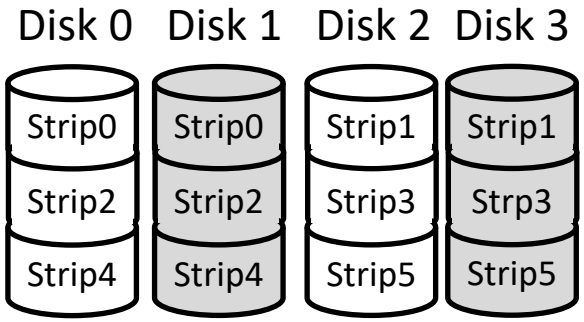
- 带宽 = N × 单个磁盘带宽
- 延迟 = 单个磁盘的延迟

RAID-1

RAID-01 / RAID-0+1



RAID-10 / RAID-1+0



Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

RAID Level 1

- 镜像
- 镜像级别R：数据存R份
- 通常与RAID-0结合使用
  RAID-01或RAID-10

容量

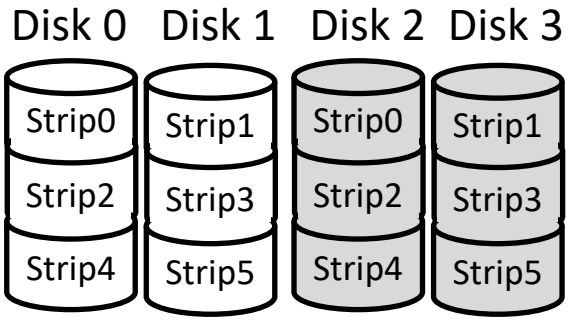
- （N ×单个磁盘容量）/ R

可靠性（R=2)

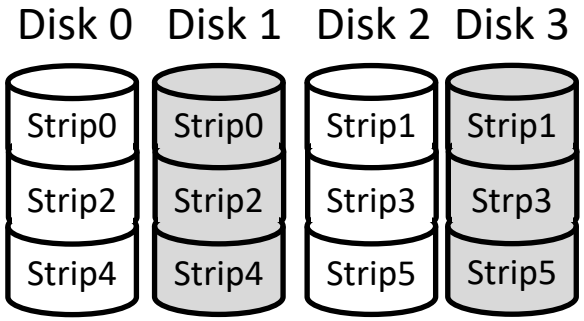
- 容忍任何一个磁盘坏
- 特殊情况下可容忍N/2个磁盘坏

RAID-1

RAID-01 / RAID-0+1



RAID-10 / RAID-1+0



Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

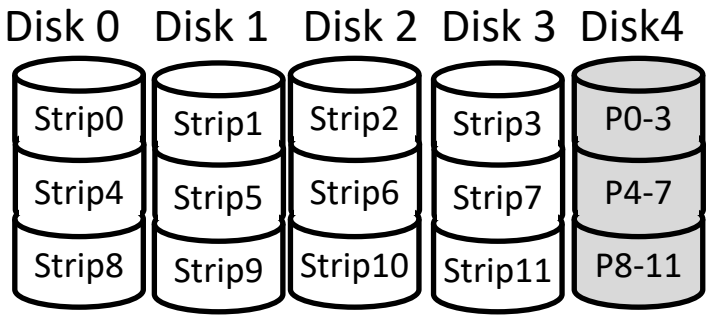
带宽

- 写带宽
  - (N × 单个磁盘写带宽) / R
- 读带宽
  - N × 单个磁盘读带宽

延迟

- ~单个磁盘的延迟

RAID-4



Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

校验块计算使用XOR

Block0	Block1	Block2	Block3	Parity
00	10	11	10	11
10	01	00	01	10

RAID Level 4

- 条带化 + 1个校验块
- 所有校验块在同一块磁盘上(校验盘)
- 缺点：校验盘为写性能瓶颈，易坏

每次写都更新校验块

方法一：读所有数据盘

1、并行读所有磁盘的对应块

2、计算新校验块

3、并行写新块和新校验块

方法二：读一个数据盘和校验盘

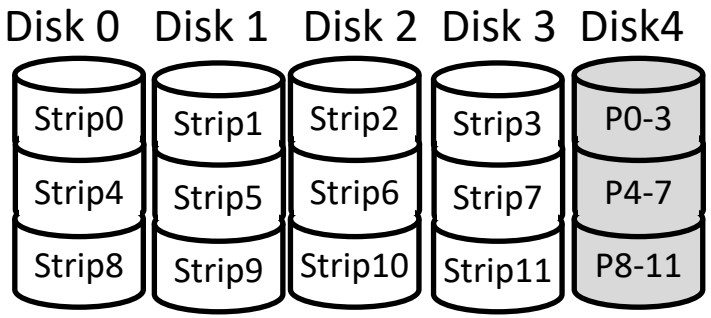
1、并行读旧块和旧校验块

2、计算新校验块

$Pnew = (Bold \oplus Bnew) \oplus Pold$

3、并行写新块和新校验块

RAID-4



Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
*4	5	6	7	+P1
8	9	10	11	P2
12	*13	14	15	+P3

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

容量

- (N-1) ×单个磁盘容量

可靠性

- 容忍任何一块磁盘坏
- 用XOR重构坏盘数据

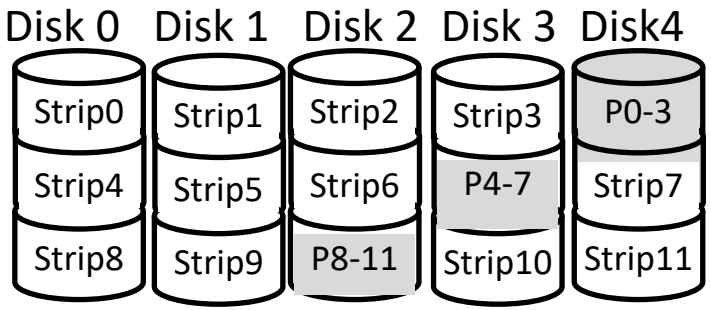
延迟

- 读延迟等于单个磁盘的延迟
- 写延迟约等于2倍单个磁盘延迟

带宽

- 读带宽=(N-1) × 单个磁盘带宽
- 校验盘为写瓶颈，所有校验块串行写

RAID-5



Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
5	6	7	P1	4
10	11	P2	8	9
15	P3	12	13	14
P4	16	17	18	19

RAID Level 5

- 条带粒度映射 + 1个校验块
- 校验块分散在不同磁盘上
- Rebuild：复杂 & 速度慢

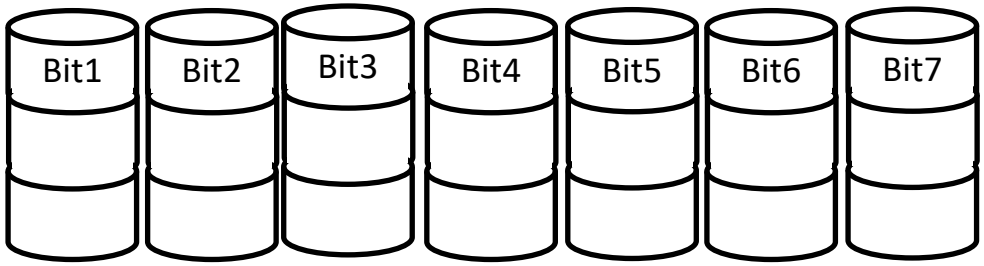
写带宽

- 写并行：校验块并行写
- 写带宽=(N × 单个磁盘带宽) /4

读带宽

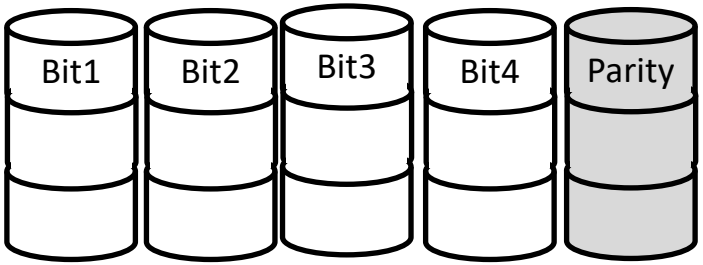
- 正常状态：只读数据块
- 读带宽：N * 单个磁盘带宽

其它RAID级别



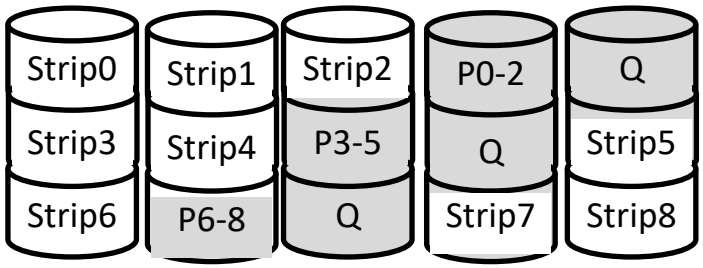
RAID Level 2

- 按位为粒度映射 + ECC
- 每4位 + 3位海明码
- 所有磁盘同步读写：寻道+旋转



RAID Level 3:

- 按位为粒度映射 + Parity位
- 已知坏磁盘时，可纠错

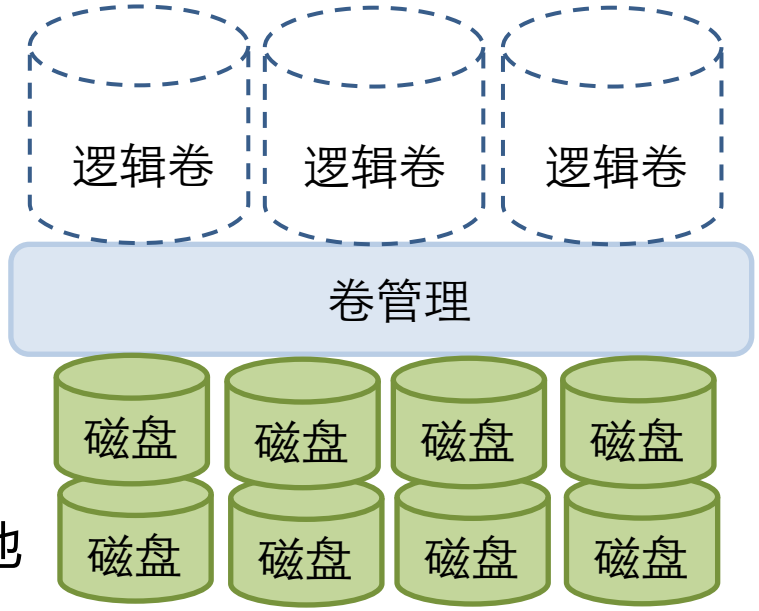


RAID Level 6

- 能容忍两块磁盘同时坏
- 条带化 + 2个校验块

卷管理（Volume manager）

- 虚拟块设备
  - 在多个磁盘上创建一个或多个逻辑卷
  - 逻辑卷：一个虚拟块设备
  - 采用RAID技术将逻辑卷的块地址映射到物理设备
- 好处
  - 提供虚拟的容量和性能
  - 容错
- 实现
  - Logical Volume Manager(LVM)
    - Linux内核的逻辑卷管理
  - 物理卷：LVM最基本的单元
    - 对应一个磁盘分区或整个磁盘
    - 例子：pvcreate /dev/sdb1
  - 卷组：一个或多个物理卷组成的逻辑存储池
    - 例子：vgcreate myvg /dev/sdb1 /dev/sdc1
  - 逻辑卷：卷组中创建的逻辑分区，可以动态调整大小
    - 例子：lvcreate –L 10G –n mylv myvg



总结

- 磁盘
  - 机械设备，内部很复杂，读写性能受限
  - 顺序大粒度读写对磁盘很友好，可以获得高带宽
  - 磁盘调度算法用于减少寻道开销
- RAID提高可靠性和I/O带宽
  - RAID 0
  - RAID 10，RAID 01
  - RAID 5
- 卷管理提供虚拟块设备
  - 通常基于RAID管理底层物理块设备