10月12日笔记

地址转换与 cache 查找的顺序:

- **translation before cache**: slow
- **translation after cache**: fast, but has synonym and alias problem

solution for **translation before cache**: Translation Look-aside Buffer. a cache for PT.
把每个进程的 pagetable 放入 TLB，用一个寄存器保存在 pagetable 中的起始位置。
为了保证低 miss rate，TLB 采用 full associative 。因为 TLB 容量不大，采用 full associative + 多路比较器，不会使 hit time 成本较大。
TLB 中的 entry 比 PT 长。在 Page Table 中，virtual page number 作为查找 PTE 的索引，无需储存在 PTE 中；在 TLB 中，需要储存virtual page number 作为对比的 tag。

SPEC: System Performance Evaluation Cooperative

可以通过跑 benchmark 以测试机器的性能。
Whetstone 和 Dhrystone 是两套人造的测试程序。后来改为 自然选择的程序。
SPEC 是一个测试集合，包含了 gcc, spice 等程序。SPECInT92 包含了9 integer programs, SPECfp92 包含了 14 floating point programs. 跑 SEPC 测试集合中的程序，对表现进行加权平均，评测出机器的性能。

Moore定律（cpu 性能与集成度）

每过18个月，单位面积上能放的三极管数翻一倍。（集成度）
即 cpu 的性能 增加一倍。

RAIDS(Redundant Arrays of Inexpensive Disk)

在早期采用单块磁盘进行数据存储和读写时，寻址和读写的时间消耗导致I/O性能非常低，并且存储容量还很小；此外，单块磁盘极其容易出现物理故障，导致数据的丢失。RAID 即独立磁盘冗余阵列，采用多个独立的磁盘组成在一起形成一个大的磁盘系统，从而实现比单块磁盘更好的存储性能和更高的可靠性。

RAID0

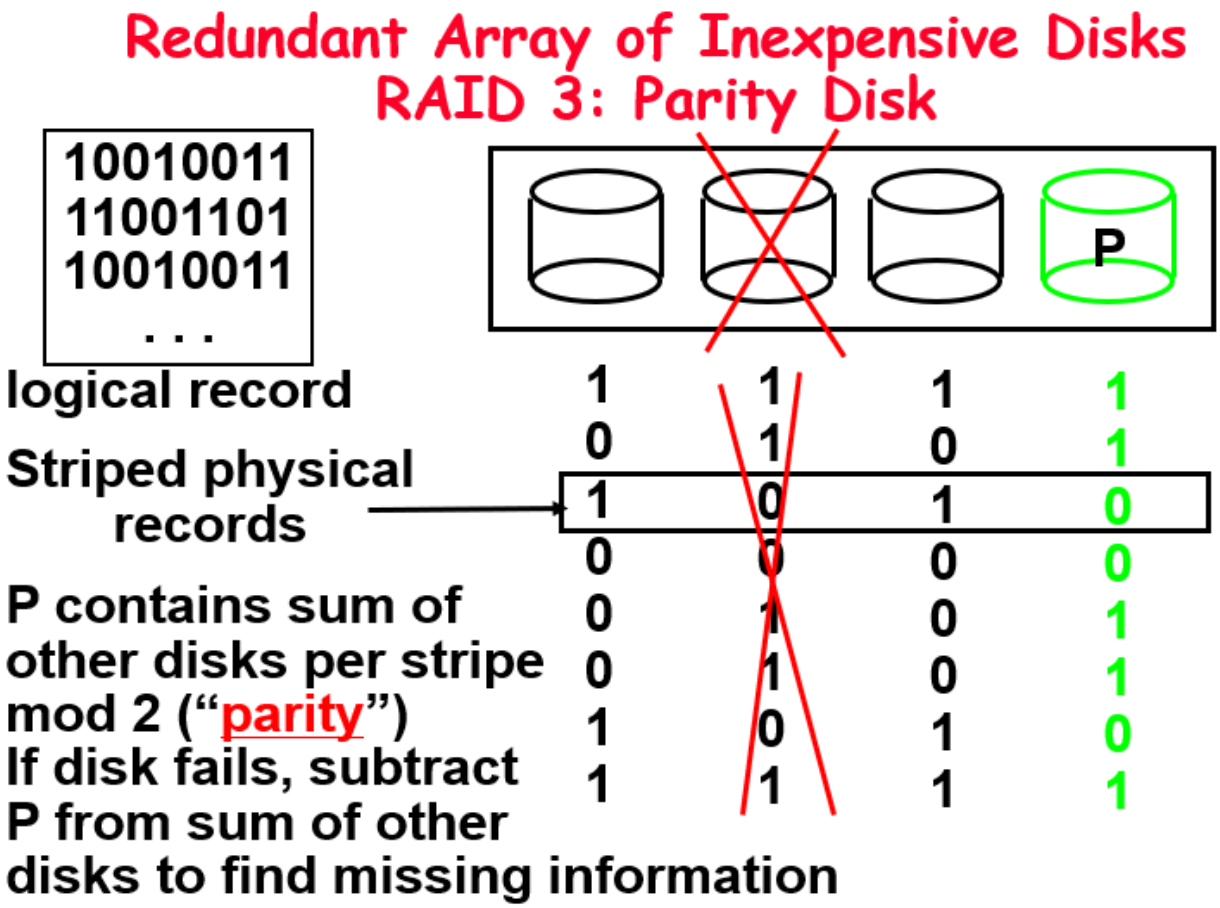
将多块磁盘组合在一起，形成一个大容量存储。写入时会将数据分为N份，以独立的方式实现N块磁盘的读写。这N份数据会同时并发的写到磁盘中，提高执行性能。但寻址时间长，并且无法提供数据校验或冗余备份，因此一旦某块磁盘损坏了，数据将无法恢复。

RAID1

为了提高存储可靠性，将工作磁盘中的内容原封不动地写入镜像磁盘，给数据一个冗余备份。任何一块磁盘损坏时，都可以基于另外一块磁盘去恢复数据。但实际空间使用率只有50%，是一种比较昂贵的方案。

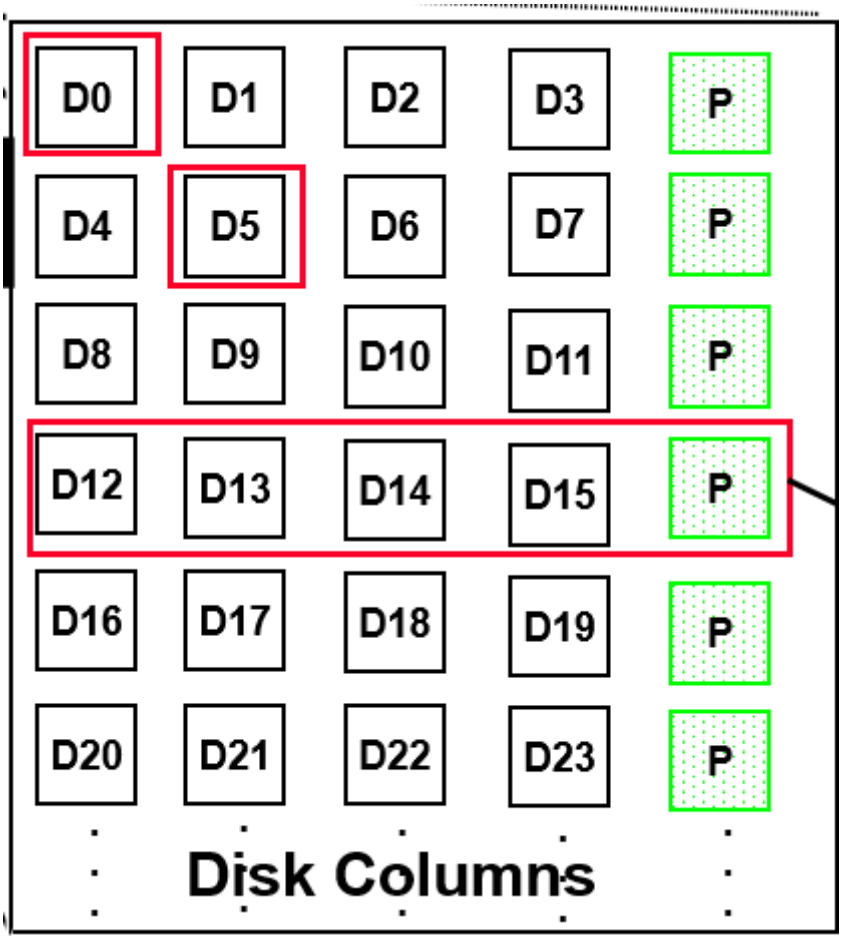
RAID3

采用奇偶校验。有 n 个普通磁盘和一个校验磁盘。在数据写入时，也要更改校验磁盘中的数据，保证同一 bit 的 n+1 个磁盘数据之和为偶数。一旦某一块磁盘故障，可以利用其它的 n 块磁盘去恢复数据。校验磁盘的数据写入十分频繁，因此也最容易出现故障。



在进行读和写时，都会涉及对 n+1 块磁盘的操作。读数据时，也需要把所有磁盘该位置的数据全部读出并进行奇偶校验，以保证数据的可靠性。这使得不同磁盘的读和写都不能同时进行。下图中的 D0 和

D5 虽在不同的磁盘中，但也无法同时进行读操作。

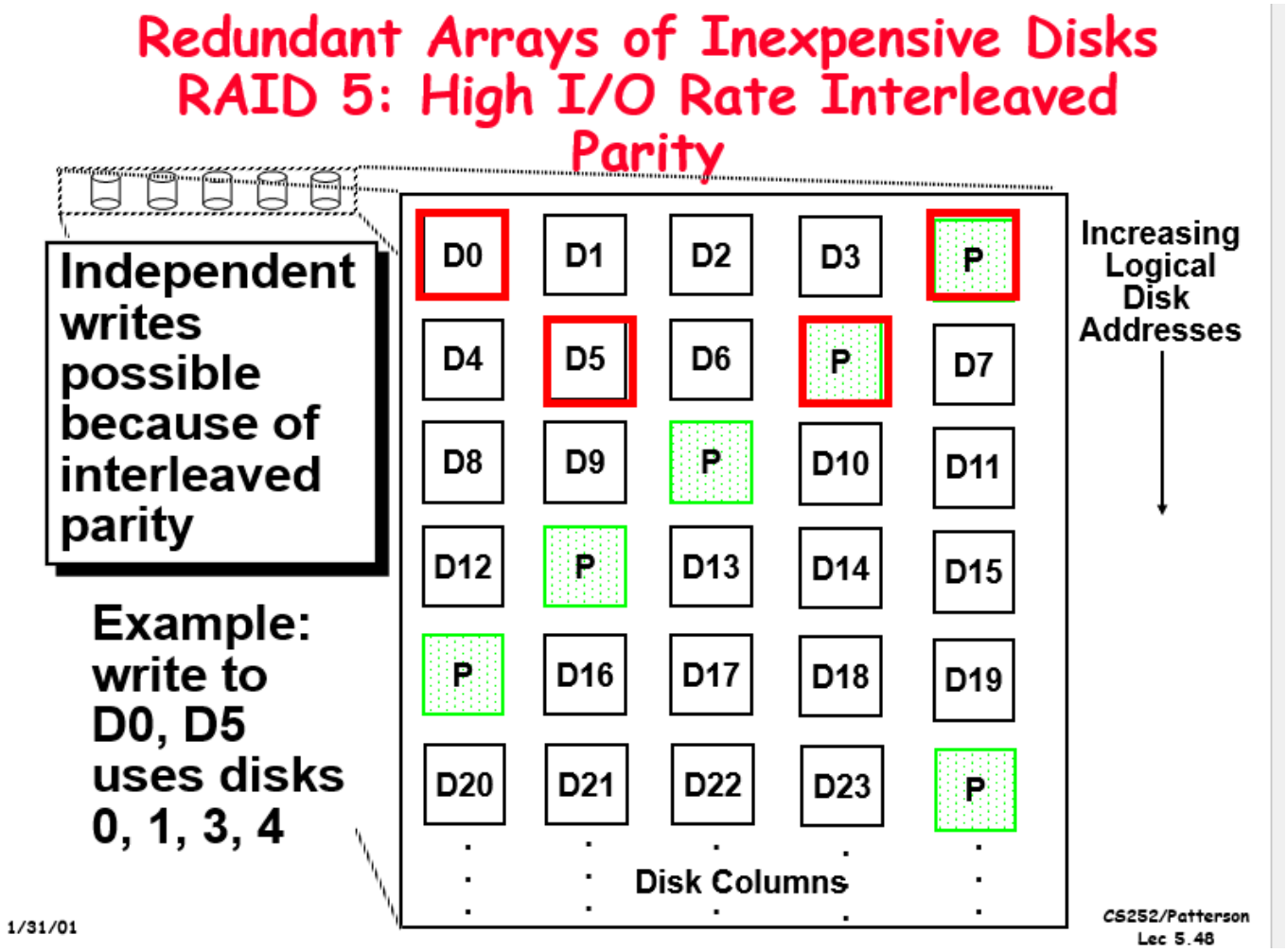


RAID4

在磁盘的 track 上有许多不连续的 sector. 在每个 sector 上设置一个 detection field. 在 read 时，通过 sector 上的 detection field 进行校验，无需涉及到其他的磁盘。此时不同磁盘上的 D0 与 D5 可以被同时独立地读。 但写数据仍然需要涉及到所有的磁盘，无法同时进行。“small read, large write"

RAID5

将校验磁盘的扇区交叉插入磁盘列。（strip)



当 D0 和 D5 同时发生改变时：

$D_0 \rightarrow D_0'$: $P_1' = D_0' + D_1 + D_2 + D_3$ = $P_1 + D_0' - D_0$ 对磁盘1,5进行操作

$D_5 \rightarrow D_5'$: $P_2' = D_5' + D_4 + D_6 + D_7$ = $P_2 + D_5' - D_5$ 对磁盘2,4进行操作

不存在冲突！因此可以同时进行。