第 3 章　独立磁盘冗余阵列

3.1　RAID 概述

独立磁盘冗余阵列(redundant arrays of independent disks,RAID)简称盘阵或磁盘阵列(disk array),是一种把多块独立的硬盘(物理硬盘或闪存芯片等)按不同的方式组合起来形成一个虚拟的硬盘(逻辑硬盘),从而提供比单个硬盘更高存储性能和提供数据备份的技术。组成磁盘阵列的不同方式称为 RAID 级别(RAID level)。在用户看起来,组成的磁盘组就像是一个硬盘,用户可以对它进行分区、格式化等操作。总之,对磁盘阵列的操作与单个硬盘一模一样。不同的是,磁盘阵列的存储速度要比单个硬盘快很多,而且引入了冗余数据来提供不同程度的容错保障,以实现可靠性存储。因阵列系统具有部署相对简单、可靠性较高、并发数据访问好等特点,已被广泛应用到大规模存储系统中。

1. RAID 技术的特点

RAID 技术的三大特点如下。

(1) 在容量和管理上,易于灵活地进行容量扩展,“虚拟化”使可管理性极大的增强。

(2) 在性能上,“磁盘分块”技术带来性能的提高。

(3) 在可靠性和可用性上,通过冗余技术和热备份、热切换,提升了可靠性。

热备份盘是一个不参与磁盘阵列但是加电上线的盘,一旦磁盘阵列中的盘出现问题,它就可以自动替换进入磁盘阵列,是一种“自动换盘”技术。

热切换也称热插拔(hot plug),是允许在不关闭系统、不切断电源的情况下取出和更换损坏的硬盘,从而提高系统对灾难的及时恢复能力、扩展性和灵活性。

2. 磁盘阵列的关键概念和技术

磁盘阵列中主要有 3 个关键概念和技术:镜像(mirroring)、数据条带(data stripping)和数据校验(data parity)。

(1) 镜像。镜像是将数据复制到多个磁盘,一方面可以提高可靠性,另一方面可并发从两个或多个副本读取数据来提高读性能。

(2) 数据条带。数据条带是将数据分片保存在多个不同的磁盘,多个数据分片共同组成一个完整数据副本,这与镜像的多个副本是不同的,它通常用于性能考虑。数据条带具有更高的并发粒度,当访问数据时,可以同时对位于不同磁盘上数据进行读写操作,从而获得非常可观的 I/O 性能提升。

(3) 数据校验。数据校验是利用冗余数据进行数据错误检测和修复,冗余数据通常采用海明码、异或操作等算法来计算获得。利用校验功能,可以很大程度上提高磁盘阵列的可靠性、鲁棒性和容错能力。

RAID 主要利用数据条带、镜像和数据校验技术来获取高性能、可靠性、容错能力和扩展性,根据运用或组合运用这 3 种技术的策略和架构,可以把磁盘阵列分为不同的等级,以满足不同数据应用的需求。磁盘阵列技术经过不断的发展,已拥有了从 RAID 0～RAID 7 共 8 种基本的 RAID 级别。另外,还有一些基本 RAID 级别的组合形式,如 RAID 10(RAID 0 与 RAID 1 的组合),RAID 50(RAID 0 与 RAID 5 的组合)等。不同的 RAID 级别代表着不同的存储性能、数据安全性和存储成本。RAID 通常是由在硬盘阵列中的 RAID 控制器或计算机中的 RAID 卡来实现的。

磁盘阵列技术在不断发展的同时也给云存储技术带来了巨大的技术支撑，主要表现高效的 I/O 性能和系统的可靠性这两方面。

3.2 条带化技术

条带化技术是一种自动将 I/O 的负载均衡到多个物理磁盘上的技术，即将一块连续的数据分成很多个小部分并把它们分别存储到不同的磁盘上。这样一来，就能使多个进程同时访问数据的多个不同部分而不会造成磁盘冲突。当需要对这种数据进行顺序访问时，可以获得最大程度上的 I/O 并行能力，从而获得非常好的性能。数据从第一块磁盘传输完毕后，第二块磁盘就能确定下一段数据的传输。相比之下，单个磁盘的读写就慢得多，所以条带化技术在现代的数据库和某些磁盘阵列中得到了广泛的应用。

为什么要进行条带化管理呢？当多个进程同时访问一个磁盘时，大多数磁盘系统都对访问次数（IOPS，每秒进行的 I/O 操作）和数据传输率（TPS，每秒传输的数据量）有限制。当达到这些限制时，后面访问磁盘的进程就需要等待，这就是磁盘冲突。

避免磁盘冲突是优化 I/O 性能的一个重要目标，而 I/O 性能的优化与其他资源（如 CPU 和内存）的优化有着很大的区别，I/O 优化最有效的手段是将 I/O 性能进行最大限度的平衡。

图 3-1 所示为一个未经条带化处理的连续数据分布情况，图 3-2 所示为一个已经被条带化处理的连续数据分布情况，通过比较可以发现，图 3-2 中对连续数据的读写都有最大并发能力。

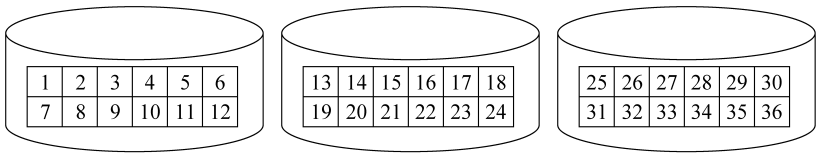


图 3-1 未经过条带化处理的连续数据

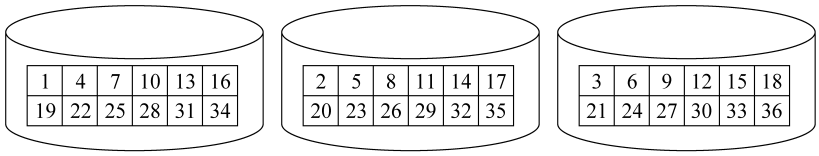


图 3-2 经过条带化处理的连续数据

由于条带化技术在 I/O 性能问题上的优越表现，以至于在应用系统所在的计算环境中的多个层次或平台都涉及了条带化的技术，如操作系统和存储系统这两个层次中都可能使用条带化技术。

当对数据做条带化处理时，数据被切成一个个小数据块，这些小数据块分布存储在不同的硬盘上。可见，影响条带化效果的因素有两个，一是条带大小（stripe size），即数据被切成的小数据块的大小，另一个是条带宽度（stripe width），即数据被存储到多少块硬盘上。

条带宽度是指同时可以并发读或写的条带数量。这个数量等于磁盘阵列中的物理硬盘数量。例如一个经过条带化处理的磁盘阵列中有 4 块物理硬盘，则该磁盘阵列的条带宽度就是 4。通过增加条带宽度，可以增加磁盘阵列的读写性能。很明显，通过增加更多的硬盘，也就增加了可以同时并发读或写的条带数量。在其他条件相同的前提下，一个由 8 块 500GB 硬盘组成的磁盘阵列比一个由 4 块 1TB 硬盘组成的磁盘阵列具有更高的传输性能。

条带大小也被称为 block size、chunk size、stripe length 或者 granularity（粒度）。这个参数指的是写在每块磁盘上的条带数据块的大小。条带大小对性能的影响比条带宽度难以量化得多。如果减小条

带大小，则文件会被分成更多、更小的数据块。这些数据块会被分散到更多的硬盘上存储，因此提高了传输的性能，但是由于要多次寻找不同的数据块，磁盘定位的性能就会下降。如果增加条带大小，则会降低传输性能，提高定位性能。

条带技术相关概念包括数据块(data chunk)、校验块(parity chunk)、编码条带(stripe)、编码(encode)、解码(decode)等。

数据块是阵列系统中用来存取数据的最小逻辑单元，磁盘阵列的数据块大小一般为2～512KB(或者更大)，其数值是2^nB($n=1,2,3,\cdots$)，即2KB、4KB、8KB、16KB。用户存放的数据通常被分为多个数据块，并且通过磁盘阵列的系统控制层根据逻辑地址并发地写入对应的磁盘阵列设备中。在用户读取数据时，也可通过磁盘阵列系统控制层在相应的设备上进行多通道并发读取。

校验块是由同一个条带内来自不同磁盘设备的多个数据块通过异或运算产生的冗余数据。校验块用于保护数据块，以保证在发生数据故障的情况下(在容错能力范围内)，依然能恢复故障数据。

编码条带是一组来自不同磁盘设备的数据块和校验块构成的集合，该集合的元素在各自的磁盘内通常具有相同的逻辑偏移地址。

编码是指同一条带内所有的数据块通过依照一定规则的运算(如异或运算、有限域运算等)产生特定校验块的过程。

解码是编码操作的逆过程，是指由同一个条带内相关的校验块和部分数据块通过一定规则的运算(如异或运算、有限域运算等)产生特定数据块的过程。

条带大小不同，文件存储的情况差异较大，如图3-3所示。

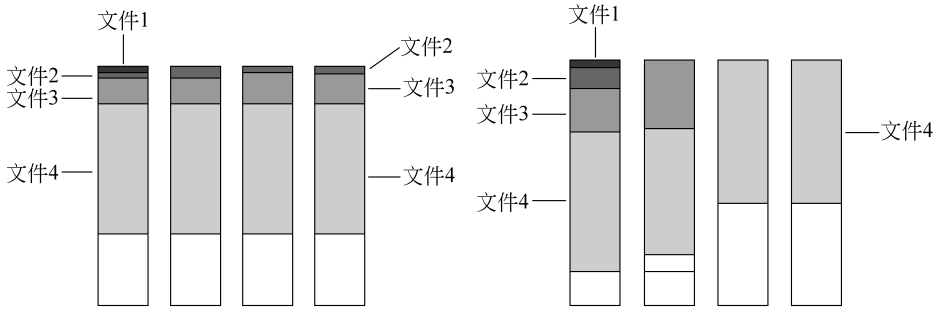


图3-3　不同条带大小时连续数据的存储情况

这是一个由4块硬盘组成的RAID 0阵列，图3-3(a)的条带大小为4KB，图3-3(b)的条带大小为64KB。图3-3(a)中的每一条细格表示4KB大小。

图3-3中文件1大小是4KB，文件2大小20KB，文件3大小为100KB，文件4大小为500KB。

从图3-3中可以看到，不同条带大小对“中型大小”的文件影响很大。不论条带是4KB还64KB，大小为4KB的文件1都分布在一块硬盘的一个数据块上，而对于大小为500KB的文件4，无论条带是4KB还是64KB，都会被分布在4块硬盘上。

但是对于大小为20KB的文件2，如果采用64KB的条带，则会被分布在一块硬盘上，而不是像采用4KB的条带那样分布在4块硬盘上。同理，如果采用64KB的条带，则大小为100KB的文件3会被分布到2块硬盘上，而采用4KB的条带时则文件3会分布到4块硬盘上。可以看到，增加条带大小可以明显增加定位性能。在上边的例子中，条带宽度理所当然是4。

图3-4是使用16KB条带时文件的存储情况，可以对应参考理解。

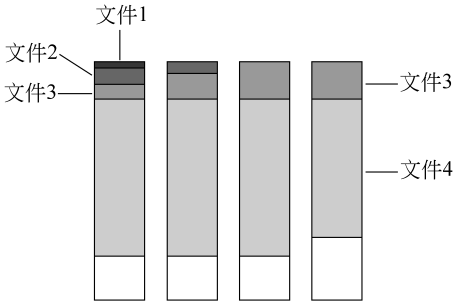


图 3-4　使用 16KB 条带时文件的存储情况

条带大小应该如何设置呢？最好的办法是尝试不同组合，根据应用的不同得到自己的经验规律。另外，不要过高估计不同条带大小的性能差异。它有可能会差距很大，尤其是设置成 4KB 和 256KB 这样的相对极端数值，但对于相差不大的数值，它们的性能差异可能就不明显。对于大多数应用的经验是，当读写大量的小文件时，采用的条带大小大些；当快速访问少量的大文件时，采用的条带大小小些；如果要平衡这两者，最好采用中间值。

条带技术将磁盘阵列的各数据块分散到各成员盘上，输入输出的聚合程度比单个磁盘要高很多。

3.3　磁盘阵列的级别

磁盘阵列技术分为几种不同的级别，分别可以提供不同的速度、容错能力、校验块放置策略等。根据实际情况选择适当的磁盘阵列级别可以满足用户对存储系统可用性、性能和容量的要求。

常用的磁盘阵列级别主要有以下几种：RAID 0、RAID 1、RAID 2、RAID 3、RAID 4、RAID 5、RAID 6，以及新标准的 RAID 7。此外，还有混合 RAID 级别 RAID 01、RAID 10 和 RAID 50 等。

3.3.1　RAID 0

RAID 0 是无容错的条带化磁盘阵列，它代表了所有磁盘阵列级别中最高的存储性能。整个逻辑盘的数据是被分条(stripped)分布在多个物理磁盘上，可以并行读写，提供最快的速度，每个磁盘执行属于它自己的那部分数据请求，要求至少两个磁盘。它将磁盘逻辑条带化，再将数据块(大小可以根据实际情况来调整)分别写到不同的磁盘上，使磁盘的逻辑存储顺序按照图 3-5 中顺序来存储数据，即 D0、D1、D2、D3、D4、D5、D6 等。这样一来，所有的 I/O 访问将会被分担到每个磁盘驱动器上，从而大大提高了 I/O 效率。通过 RAID 0 可以获得更大的单个逻辑盘的容量；通过对多个磁盘的同时读取，可以获得更高的存取速度。

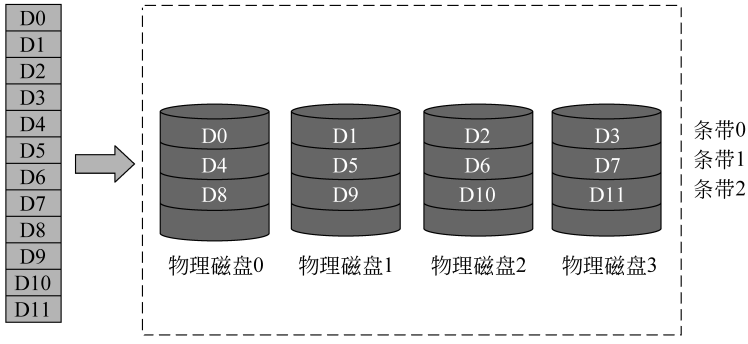


图 3-5　RAID 0

RAID 0 并不是真正的 RAID 结构，不产生冗余数据，数据存储效率 100%。RAID 0 连续地分割数据并对多个磁盘进行并行读写，因此具有很高的数据传输速率。但是，RAID 0 在提高性能的同时并没有提高数据可靠性，如果一个磁盘失效，就会影响整体数据，磁盘阵列的整体可靠性仅为单台设备的 1/N。因此 RAID 0 不可应用于对数据可靠性需求较高的关键应用。

3.3.2　RAID 1

RAID 1 是一种带镜像的双工阵列，即将磁盘两两配对，形成全冗余组合，以确保数据的稳定性和可靠性。在整个镜像过程中，只有一半的磁盘容量是有效的（另一半磁盘容量用来存放同样的数据），要求设备数为偶数个，如图 3-6 所示。与 RAID 0 相比，RAID 1 首先考虑的是安全性，容量减半、速度不变。

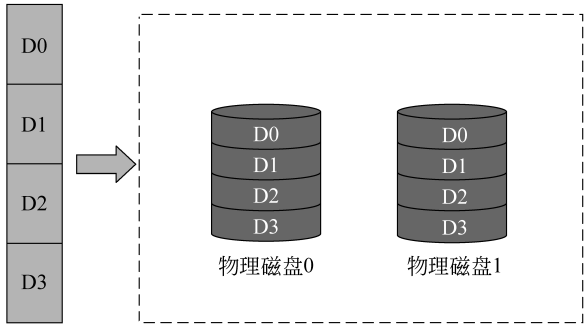


图 3-6　RAID 1

RAID 1 是通过数据镜像实现数据冗余，在两对分离的磁盘上产生互为备份的数据的。RAID 1 可以提高读取性能，当原始数据繁忙时，可直接从镜像副本中读取数据。当一个副本发生错误时，可通过读取另一个副本来恢复丢失的数据。RAID 1 不需要编码和解码，部署相对简单，但是其存储成本是磁盘阵列中最高的。一般适用于要求数据访问性能高且不关注存储成本的应用场景，例如金融、财务以及其他要求高数据可靠性的领域。

3.3.3　RAID 2

RAID 2 是一种带海明码校验的磁盘阵列，海明码是一种具有自纠错功能的校验码。RAID 2 将一个数据字的每一位写在一块数据盘上，每个数据字的校验码写在校验盘上。读取数据时，校验码对正确的数据进行校验，对错误的数据位进行纠错。

由于海明码具有校验及自动纠错功能，因此 RAID 2 方式有了提高数据传输速率的可能。随着数据传输速率的提高，信息部的字长越长，信息部字长与校验部字长的比值就越大，数据传输就越高效；反之，信息部的字长越短，信息部字长与校验部字长的比值就越小，数据传输就越低效。因此，使用这种磁盘阵列方式只有达到特定的传输速率，才能保证不把过多的资源浪费在数据校验上。

RAID 2 是为用于影像处理或 CAD/CAM 等需要连续存取大量数据的计算机设计的，并不适合一般的多用户环境、网络服务器和 PC。由于校验盘数量太多、开销太大、成本昂贵，目前已基本不再使用，如图 3-7 所示。

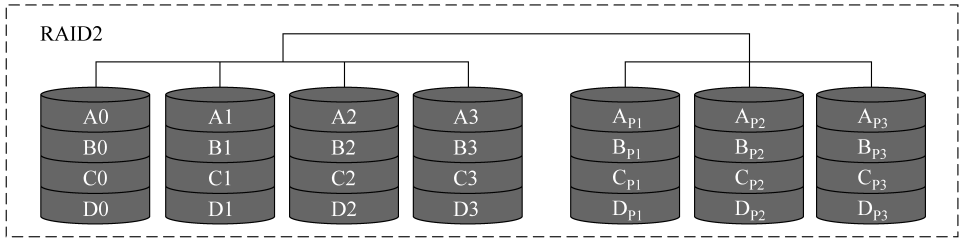


图 3-7　RAID 2

3.3.4 RAID 3

RAID 3 是一种带奇偶校验的并行传输阵列，它采用了相对简单的校验码来保证数据的完整性。它将数据块分割成条带（按位交叉）后分别存储在各个数据盘上，然后将计算出的校验码写入专用的校验磁盘，供每次读操作时进行校验。由于在一个硬盘阵列中，多个硬盘同时出现故障的概率很小（更换一个新硬盘后，系统可以重新恢复完整的校验容错信息），所以一般情况下，使用 RAID 3 的安全性可以得到保障。RAID 3 至少需要 3 块磁盘。与 RAID 0 相比，RAID 3 的读写速度较慢。由于使用的容错算法和分块大小决定了 RAID 使用的应用场合，所以在通常情况下，RAID 3 比较适合视频编辑、硬盘播出机、大型数据库等文件大且安全要求较高的应用，如图 3-8 所示。

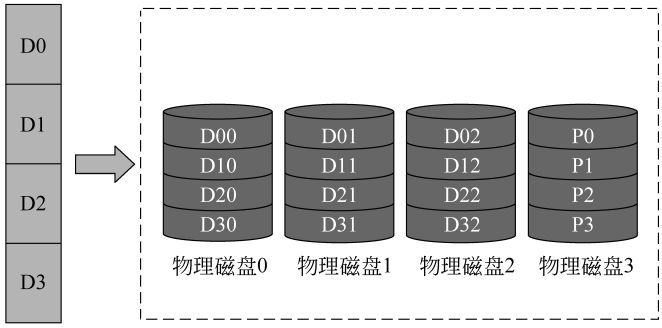


图 3-8 RAID 3

由于 RAID 3 对系统资源的消耗较大，因此基本不用软件方式实现。此外，由于 RAID 3 按条带存储，无逻辑单元，因此不论 I/O 操作的数据量大小，都必须对整个条带进行校验位的计算并将结果写入校验盘。从而每次读写都要牵动整个组，每次只能完成一次 I/O 操作，这是 RAID 3 的最大缺点。

3.3.5 RAID 4

RAID 4 是一种带独立的数据磁盘与共享的校验磁盘（independent data disks with shared parity disk）的磁盘阵列，其技术和 RAID 3 基本一样，只是条带单位不同，它是以块为单位。为了设计方便，块的大小一般以一个文件为单位，即一个文件写在一个磁盘里面，这样做的好处是相关内容聚合在一个磁盘上，不像 RAID 3 那样分布在各个磁盘上，缺点是会牺牲并发读写特性，如图 3-9 所示。

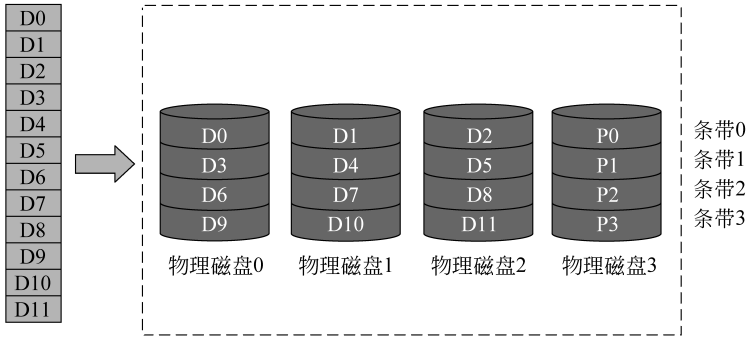


图 3-9 RAID 4

在 RAID 4 中有一块专用的校验磁盘，用于存储校验块数据，因此至少使用 3 个磁盘。它将数据分

为更大的块，然后并行传输到各个成员磁盘，同时将计算出的 XOR 校验数据存放到专用的校验磁盘上。RAID 4 具有较高的读取性能和较低的写入性能，存储效率为$(N-1)/N$。当单个磁盘发生故障时，可通过其他磁盘及校验磁盘进行修复。

由于所有的检验数据集中放置在同一个磁盘内，所以数据的更新等操作会造成校验磁盘成为磁盘阵列系统 I/O 性能的瓶颈，易发生数据磁盘与校验磁盘争写的问题。由于校验的粒度太粗，所以发生磁盘失效之后的数据重建比较复杂。除此之外，还存在控制器结构复杂的问题。由于使用同一个共享的冗余磁盘存放块校验信息，因此读取块的速度不高，仅与单磁盘相当。基于以上原因，RAID 4 的使用并不广泛。

3.3.6　RAID 5

RAID 5 是一种带分布式奇偶校验的磁盘阵列，是 RAID 4 的升级版，是一种兼顾存储性能、数据安全和存储成本的存储解决方案，也是目前最常用的磁盘阵列方式。RAID 5 不对存储的数据进行备份，而是把数据和相应的奇偶校验信息存储到组成 RAID 5 的各个磁盘上。当磁盘阵列中的某个磁盘发生损坏后，可利用剩下的数据和相应的奇偶校验信息修复被损坏的数据，如图 3-10 所示。

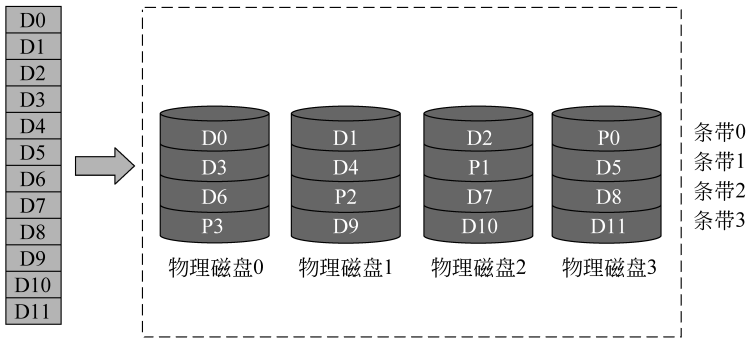


图 3-10　RAID 5

如图 3-10 所示，P0 为校验块，由数据块 D0、D1 以及 D2 经过异或操作计算产生。当上述任意一个数据块丢失时，例如 D0 丢失，可通过读取其他剩余数据块 D1、D2 和校验块 P0，再对这些数据进行异或运算重新计算恢复得到 D0。

可以将 RAID 5 理解为 RAID 0 和 RAID 1 的折中方案，虽然都可以为磁盘阵列系统提供数据安全保障，但保障程度比镜像低，磁盘空间利用率比镜像高。RAID 5 具有和 RAID 0 相似的数据读取速度，只是多了一个奇偶校验信息，写入速度也比单个磁盘的写入速度稍慢。由于多个数据对应一个奇偶校验信息，所以 RAID 5 的磁盘空间利用率比 RAID 1 高，存储成本相对较低。

RAID 5 没有单独指定的奇偶校验磁盘，而是交叉地存取数据并将奇偶校验信息存放到所有磁盘上。在 RAID 5 上，读写指针可同时对磁盘阵列中的存储设备进行操作，提供了更高的数据流量，所以 RAID 5 更适合随机读写的小数据块。RAID 3 每进行一次数据传输都涉及磁盘阵列中所有的存储设备，而 RAID 5 中大部分数据传输只对一块磁盘进行，且可以并行操作。在 RAID 5 中有“写损失”，即每进行一次写操作都会产生 4 个实际的读写操作，即读出旧的数据、读出奇偶信息、写入新的数据、写入奇偶信息。

但 RAID 5 对控制器的要求很高，既要求控制器具备快速传输能力，又要求有很强的计算能力。由于 RAID 5 只能保证发生单盘故障时不影响整个存储系统，所以一旦发生两块磁盘同时发生故障，便无

法恢复数据。虽然大部分 RAID 级别都只能保证单盘故障的容错性，但 RAID 5 具有较高的读出性能，中等的写入性能，是应用比较广泛的一种磁盘阵列。

3.3.7 RAID 6

RAID 6 是一种带有独立数据磁盘的磁盘阵列，它所带的数据磁盘带有两种独立的分布式校验方案。它是在 RAID 5 的基础上发展而成的，因此它的工作模式与 RAID 5 有异曲同工之妙。RAID 5 将校验码写入一个磁盘，而 RAID 6 将校验码写入两个磁盘，因此后者增强了磁盘的容错能力。RAID 6 允许两个磁盘同时发生故障，磁盘阵列中的磁盘数量最少要 4 个。若磁盘数量为 $N+2$ 个，则存储效率为 $(N-2)/N$。

RAID 6 的每个磁盘都存有两个校验值，而 RAID 5 只能为每个磁盘提供一个校验值，由于校验值的使用可以达到恢复数据的目的，因此多增加一位校验位，数据的恢复能力就会得到增强，如图 3-11 所示。在增加了一位校验位后，会带来新的问题，例如需要更加复杂的控制器进行控制，降低磁盘的写入能力，以及需要占用一定的磁盘空间。RAID 6 有很高的数据有效性和可靠性，特别适用于可靠性要求很高的领域。随着技术的不断完善，RAID 6 将得到更加广泛的应用。

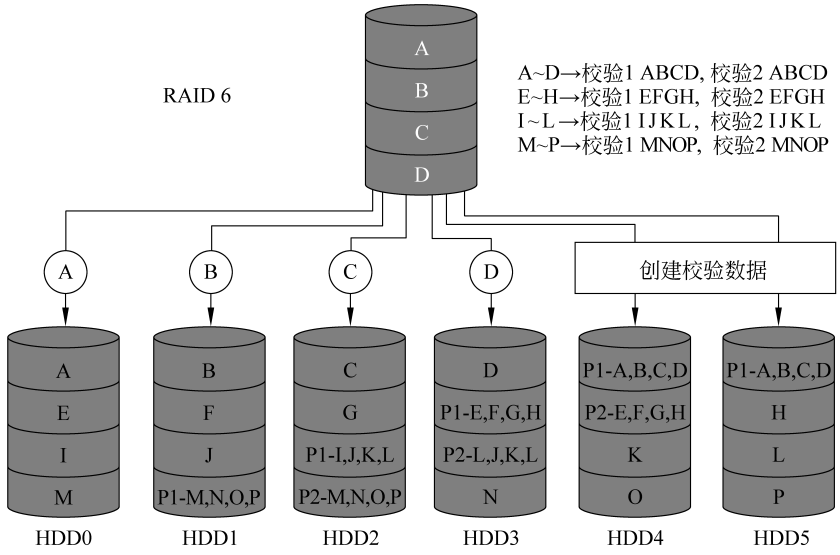


图 3-11 顺次写入 A、B、C、D 时的 RAID 6 情形

3.3.8 RAID 7

RAID 7 是一种最优化的异步高速输入输出和数据传输(optimized asynchrony for high I/O and data transfer rates)磁盘阵列，是一种新的 RAID 标准。它与以前的 RAID 级别有明显的不同，可以理解成一个独立的“存储计算机”，RAID 7 自身带有智能化实时操作系统和用于存储管理的软件工具，可以完全独立于主机运行，不占用主机的 CPU 资源。

RAID 7 具有最优化的高速数据传送磁盘结构，所有的输入输出操作均是同步进行、分别控制的，这就提高了系统的并行性和系统存取数据的速度。RAID 7 的每个磁盘都带有高速缓冲存储器，实时操作系统可以使用任何实时操作芯片实现不同实时系统的需要，如图 3-12 所示。RAID 7 允许使用简单网络管理协议(simple network management protocol，SNMP)进行管理和监视，可以对校验区指定独立的

传送信道以提高效率。RAID 7 可以连接多台主机，当多用户访问系统时，访问时间几乎接近于 0。如果发生系统断电，会把高速缓冲存储器内的数据全部丢失，因此 RAID 7 需要和 UPS 一起工作。此外，RAID 7 系统的成本很高。

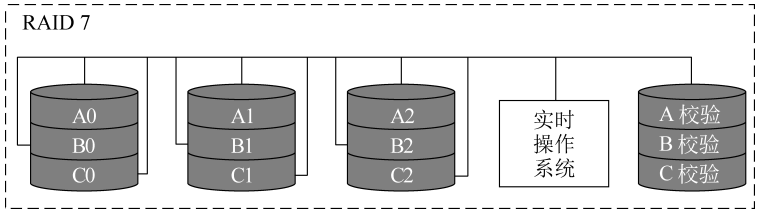


图 3-12　RAID 7

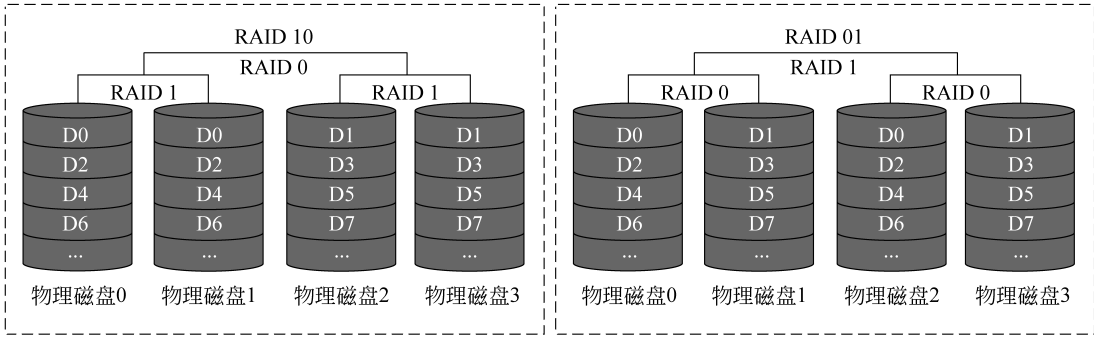
3.3.9　组合 RAID

不同级别的 RAID 可以在性能、冗余、价格等方面做不同程度的折中，组合出不同级别的磁盘阵列，目的是扬长避短，产生具有优势特性的混合磁盘阵列级别。下面重点介绍 RAID 10、RAID 01 和 RAID 50。

1. RAID 10 和 RAID 01

RAID 10 结合 RAID 0 和 RAID 1 磁盘阵列技术，为两层磁盘阵列结构，要求存储设备的总数是不少于 4 的偶数，如图 3-13(a)所示。顶层结构为 RAID 0，负责条带化数据，对外提供访问；底层结构为 RAID 1，负责将顶层条带化后的数据进行备份以及数据保护。其存储效率与 RAID 1 相同，适用场景与 RAID 1 类似，因具有条带化的 RAID 0，能进一步提升原有底层 RAID 1 的访问性能。

RAID 01 至少需要 4 块磁盘来构成。它实际上是将两个 RAID 0 系统进行镜像，综合了 RAID 0 和 RAID 1 的技术，如图 3-13(b)所示。其容灾性能等同于 RAID 5，冗余度等同于 RAID 1。由于具有 RAID 0 的性能，它具有很高的输入输出速度。



(a) RAID 10　　(b) RAID 01

图 3-13　RAID 10 和 RAID 01

RAID 10 是先做镜像，然后再做条带；RAID 01 则是先做条带，然后再做镜像。

若磁盘阵列中都有 6 块磁盘，则 RAID 10 是先将盘分成 3 组镜像，然后再对这 3 个 RAID 1 做条带；RAID 01 则是先利用 3 块盘做 RAID 0，然后将另外 3 块盘作为 RAID 0 的镜像。

下面，以图 3-13 所示的由 4 块磁盘组成的磁盘阵列为例介绍 RAID 10 和 RAID 01 在安全性方面的差异。

(1) RAID 10。这种情况下,假设当 D0 盘损坏,在剩下的 3 块盘中,只有当 D1 盘发生故障时,才会导致整个磁盘阵列失效,可计算出故障率为 1/3。

(2) RAID 01。这种情况下,仍然假设 D0 盘损坏,这时左边的条带将无法读取。在剩下的 3 块盘中,D2、D3 盘中的任何一个损坏,都会导致整个磁盘阵列失效,可简单计算故障率为 2/3。

由此可见,RAID 10 比 RAID 01 在安全性方面强。

在正常的情况下,RAID 01 和 RAID 10 数据存储的逻辑位置是完全一样的,而且每一个读写操作所产生的输入输出量也是一样的,所以两者在读写性能上没什么区别,但是当磁盘出现故障时(例如前面假设的 D0 损坏时)可以发现,这两种情况下的读取性能是不同的,RAID 10 的读取性能将优于 RAID 01。

2. RAID 50

RAID 50 具有 RAID 5 和 RAID 0 的共同特性。它先是各用 3 个磁盘组成两组 RAID 5,再把两组 RAID 5 组建成 RAID 0,如图 3-14 所示。利用 RAID 5 的校验来纠错和重建资料,利用 RAID 0 的高速来提升系统性能。RAID 50 需要至少 6 个磁盘,如果每组 RAID 5 中各损坏一个磁盘,数据可以顺利恢复。

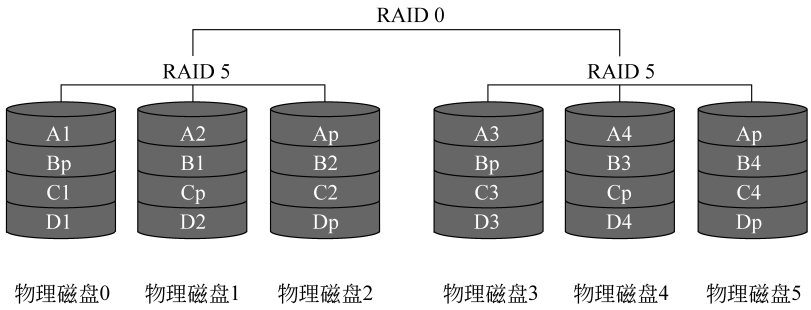


图 3-14 RAID 50

RAID 等级的不同主要是对数据存储要素(速度、容量、容错性)的不同取舍,各有各的应用场景。

(1) 速度。RAID 通过在多个磁盘上同时存储和读取数据来大幅提高存储系统的数据吞吐量。在 RAID 中,可以让很多磁盘驱动器同时传输数据,而这些磁盘驱动器在逻辑上又是一个磁盘驱动器,所以使用 RAID 可以达到单个磁盘驱动器几倍、几十倍甚至上百倍的速率。

(2) 容量。可以将多个磁盘连接起来,对比传统的单个磁盘存储,RAID 将存储的量级拔高了一个台阶。但依旧有其局限性,因为 RAID 始终是放在单台计算机上,计算机上的磁盘卡槽不可能无限增加,磁盘数量也不可能一直增多。

(3) 容错性。不同等级的 RAID 使用不同的数据冗余策略,保证数据的容错性。比如最简单的 RAID 1 就是数据在写入磁盘时,将一份数据同时写入两块磁盘,这样任何一块磁盘损坏都不会导致数据丢失,而插入一块新磁盘就可以通过复制数据的方式自动修复,具有极高的可靠性。

在 RAID 阵列中,磁盘与阵列存在“部分”与“整体”的关系。

整体和部分的辩证关系如下。

(1) 整体和部分是相互区别的。整体居于主导地位,整体统率着部分,具有部分所不具备的功能;部分处于被支配的地位,部分服务于整体。在磁盘阵列中,阵列处于主导地位,磁盘则处于被支配地位,并受阵列管理。

(2) 整体和部分又是相互联系、密不可分的。整体是由部分构成的,离开了部分,整体就不复存在。

部分的功能及其变化会影响整体的功能，关键部分的功能及其变化甚至对整体的功能起决定作用。阵列是由磁盘组成的，离开了磁盘，阵列不复存在，而磁盘的功能及其变化会影响整个阵列的功能。

(3) 磁盘与阵列的部分与整体的辩证关系，启发人们要树立全局观念，立足于整体，统筹全局。必须重视部分的作用，搞好局部，推动整体的发展。在抗击新冠疫情时，全社会就像一台“磁盘阵列”，各地区各部门各行业的工作都息息相关，各环节务必紧紧咬合，才能高效运转。既独立作战，又不各自为战，要协调配合、相互支持、彼此补位，才能最终战胜疫情。

3.4　RAID 卡

3.4.1　RAID 卡概述

RAID 卡就是用来实现磁盘阵列功能的板卡，通常由 I/O 处理器、硬盘控制器、硬盘连接器和缓存等一系列组件构成，如图 3-15(a)所示。RAID 卡主要解决了两个功能：一是通过不同的 RAID 级别实现容错功能，二是可以让很多磁盘驱动器同时传输数据，而这些磁盘驱动器在逻辑上又是一个磁盘，从而实现单个的磁盘驱动器几倍、几十倍甚至上百倍的速率。

磁盘阵列有硬件和软件两种实现方式。通过硬件实现磁盘阵列的设备称为硬 RAID 卡，它是一种独立的硬件设备。主板上集成的 RAID 芯片也属于硬 RAID 卡，但是有人称其为半硬 RAID 卡。通过软件进行模拟的磁盘阵列称为软 RAID 卡，这种方式占用的 CPU 资源较多，因此绝大部分服务器设备都使用硬件方式实现。

按照磁盘接口的不同，磁盘阵列分为 SCSI RAID、IDE RAID 和 SATA RAID 3 种。其中，SCSI RAID 主要用于要求高性能和高可靠性的服务器和工作站，而台式机中主要采用 IDE RAID 和 SATA RAID。

RAID 技术问世时是基于 SCSI 接口(称为 SCSI RAID)的，主要面向服务器等高端应用，价格较高。在市场的推动下，由于 PC 中 IDE 设备的价格大幅降低、性能大幅提高，因而 RAID 技术被移植到 IDE 接口上，推出了基于 IDE 接口的 RAID 应用(称为 IDE RAID)。SATA RAID 是诞生不久的 RAID 方式，它与 IDE RAID 类似，最大的优点是低成本，其他方面和 IDE RAID 接近，以逐步取代 IDE RAID。

目前，通过主板上集成的 SATA RAID 独立 I/O 芯片提供 RAID 功能已越来越普遍，因其价格低廉而广泛应用于塔式或机架式 RAID 系统中，在操作系统安装前就能提供 RAID 支持。它不占用主板的板卡插槽资源，只需 SATA 接口即可，无须另外安装 RAID 卡，且支持硬盘热插拔。图 3-15(b)是一款 SATA RAID 芯片。


(a) Adaptec 71605Q　　(b) SATA RAID 独立芯片 PromisePCD20375

图 3-15　两款 RAID 卡

顾名思义，硬盘热插拔就是带电插拔(hot swap)，即在不停机的情况下更换时插拔硬盘。当有一个硬盘出现故障损坏的情况下，阵列存储器可以不用关机，直接带电拔出故障硬盘并换上新硬盘。磁盘阵列在硬盘出现故障时，一般情况下故障盘的相应指示小灯会显现异常并伴随自动鸣叫警示，提示硬盘发生故障，需要及时更换。

3.4.2 SCSI RAID 卡的工作原理

RAID 卡有自己的 CPU、缓存，通过集成或借用主板上的 SCSI 控制器来管理硬盘，属于智能化设备。RAID 卡的分类一般根据集成的 SCSI 控制器的不同进行划分。如果没有集成 SCSI 控制器，而是借用主板上的 SCSI 控制器来管理硬盘，则为零通道 RAID 卡。根据 RAID 卡集成的 SCSI 控制器的通道数量不同，可以分为单通道、双通道、三通道的 RAID 卡。还可以按照 SCSI 控制器的标准来划分 RAID 卡的种类，如 Ultra Wide、Ultra2 Wide、Ultra160 Wide。

RAID 卡处理器是一个 PCI 从设备，用于接收并执行来自系统的命令。同时占用 PCI 中断，代表 SCSI 磁盘子系统向系统提出中断请求，请求占用 PCI 总线，返回对系统命令的响应，如输送 SCSI 硬盘上的数据，其结构如图 3-16 所示。

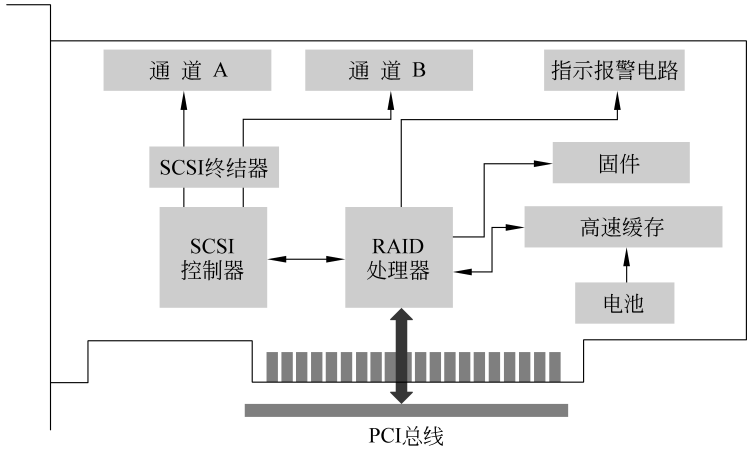


图 3-16 RAID 卡处理器的结构

RAID 处理器通过执行闪存中的固件(firm ware，即写入 EPROM(可擦写可编程只读存储器)或 EEPROM(电可擦可编程只读存储器))中的程序控制 SCSI 控制器、高速缓存以及指示报警电路来实现 RAID 卡的功能，其运作流程如下。

(1) 初始化 RAID 卡寄存器。

(2) 读取 NVRAM 的上次 RAID 参数，与硬盘实际信息进行比较，显示结果。

(3) 发送配置提示、响应 HOST 命令进入配置界面。

(4) 提供配置菜单、将用户提供的 RAID 卡参数、RAID 参数存入 NVRAM。

(5) 根据 RAID 参数，通过 SCSI 控制器对硬盘进行初始化写操作。

(6) 完成配置。

(7) 等待主机发出读写操作命令。

1. 高速缓存

高速缓存(cache)是 RAID 卡与外部总线交换数据的场所，RAID 卡先将数据传送到缓存，再由高速缓存和外边数据总线交换数据。它是 RAID 卡电路板上的一块存储芯片，与硬盘盘片相比，具有极快的存取速度，实际上就是相对低速的硬盘盘片与相对高速的外部设备(例如内存)之间的缓冲器。高速缓存的大小与速度是直接关系到 RAID 卡的实际传输速度的重要因素，大的高速缓存能够大幅度地提高数据命中率从而提高 RAID 卡整体性能。RAID 卡提高磁盘读写性能的另一方法是利用磁盘的高速缓

存，如图 3-17 所示。对于磁盘 I/O 来说，如果没有高速缓存，就直接从硬盘读写；如果有高速缓存，则首先从高速缓存读写。

RAID 卡高速缓存的作用是回写和预读。回写是通过暂时将数据存放在高速缓存，从而推迟将数据写到慢设备（如硬盘，磁带机）的一种工作方式。

2. 预读

预读可提高计算机系统中读取硬盘的性能，尤其是在读取含有大量文件碎块的文件时。具有良好预读功能的 RAID 卡能在很随机的读取中，识别出读取磁盘的规律，通过这个规律将系统要读取的数据放在高速缓存中；主要有 Read ahead、Pre-Fetch 两种方式。

（1）Read ahead 方式。由于硬盘数据经常是以一簇连续的硬盘扇区组织起来的，所以有时如把系统请求的扇区随后的一个扇区里的数据同时读进来是有价值的。

（2）Pre-Fetch 方式。当 RAID 卡发现系统要读的是先前已经读过的数据时，便将这一个数据块的数据写到高速缓存里。

RAID 总线结构如图 3-18 所示。

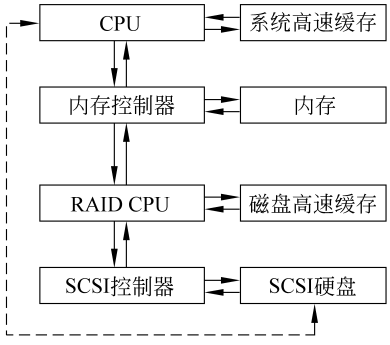


图 3-17　RAID 功能运作流程

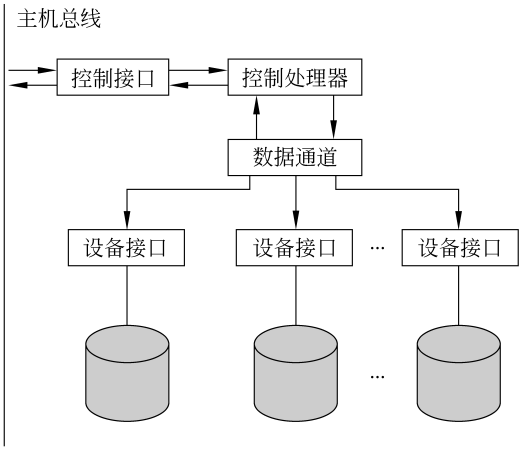


图 3-18　RAID 总线结构

关于 RAID 卡、半硬 RAID 体验，可参见习题 3 实验题的第 1 题。

3.5　基于软件的 RAID

基于软件的 RAID，就是 RAID 的所有功能都是由操作系统与 CPU 来完成，没有第三方的控制、处理/与 I/O 芯片。这样，有关 RAID 的所有任务的处理都由 CPU 来完成，显然这是效率最低的一种 RAID。由于全软 RAID 是在操作系统下实现 RAID，不能保护系统盘，即系统分区不能参与实现 RAID。有些操作系统，RAID 的配置信息存在系统信息中，而不是存在磁盘上，当系统崩溃，需重新安装时，RAID 的信息也会丢失。尤其是全软 RAID 5 是 CPU 的增强方式，会导致 30%～40%的 I/O 功能降低，所以在服务器中不建议使用全软 RAID。目前常用的操作系统，包括 Windows、Linux 等，都支持软件 RAID。

3.5.1　Linux 下基于软件的 RAID

mdadm（multiple devices admin，简称 MD）是 Linux 下的一款标准的软件 RAID 管理工具，能完成

所有的软 RAID 管理功能。mdadm 的特点如下。

(1) 能够诊断、监控和收集详细的阵列信息。

(2) 是一个单独集成化的程序而不是一些分散程序的集合,因此对不同 RAID 管理命令有共通的语法。

(3) 支持的 RAID 级别有 RAID 0、RAID 1、RAID 4、RAID 5 以及 RAID 6。

(4) 可以基于多块硬盘、分区以及逻辑卷来创建 RAID。对于硬件实现 RAID 来说,就只能是基于多块硬盘。

(5) 已创建的软件 RAID 对应于/dev/md*n*,其中 *n* 表示的是第 *n* 个 RAID,如第 1 个创建的 RAID 对应/dev/md0,第 2 个创建的 RAID 就对应/dev/md1,该名字根据需要可以自行更改。

(6) RAID 的信息保存在/proc/mdstat 文件中,或者通过 mdadm 命令来查看。

(7) 能够执行几乎所有的功能而不需要配置文件(也没有默认的配置文件)。如果需要一个配置文件,mdadm 将帮助管理它的内容。

mdadm 不采用/etc/mdadm.conf 作为主要配置文件,它完全可以不依赖该文件也不会影响阵列的正常工作。该配置文件的主要作用是方便跟踪软 RAID 的配置。对该配置文件进行配置是有好处的,但不是必需的,推荐对该文件进行配置。建立方法如下:

```
mdadm -D -s > /etc/mdadm.conf
```

或

```
mdadm --detail --scan > /etc/mdadm.conf
```

(8) mdadm 是命令行工具,要求超级用户(root)的权限。

mdadm 命令基本语法:

```
mdadm [mode] [options]
```

其中,参数[mode] 有下列 7 种。

Assemble: 将以前定义的某个阵列加入当前在用阵列。

Build: Build a legacy array,每个 device 没有 superblocks(超级块)。

Create: 创建一个新的阵列,每个 device 具有 superblocks。

Manage: 管理阵列,比如 add 或 remove。

Misc: 允许单独对阵列中的某个 device 做操作,比如抹去 superblocks 或终止在用的阵列。

Follow or Monitor: 监控 RAID 1、4、5、6 和 multipath 的状态。

Grow: 改变 RAID 容量或阵列中的 device 数目。

而 [options]选项的参数较多,主要介绍如下几个。

-f: fail,将一个磁盘设置为故障状态。

-l: LEVEL,设置磁盘阵列的级别。

-r: 移除故障设备。

-a: 添加新设备进入磁盘阵列。

-S: 停止一个磁盘阵列。

-v--verbose: 显示细节。

-D--detail：打印一个或多个 md device 的详细信息。

-x--spare-devices：指定一个备份磁盘，也就是指定初始阵列的冗余 device 数目即 spare device 数目。

-n：指定磁盘的个数。

-A--assemble：加入一个以前定义的阵列。

-B --build：创建一个没有超级块的阵列(build a legacy array without superblocks)。

-C --create：创建一个新的阵列。

-F --follow,--monitor：选择监控(monitor)模式。

-G --grow：改变激活阵列的大小或形态。

-I --incremental：添加一个单独的设备到合适的阵列，并可能启动阵列。

--auto-detect：请求内核启动任何自动检测到的阵列。

-h --help：帮助信息，用在以上选项后，则显示该选项信息。

--help-options：显示更详细的帮助。

-V --version：打印 mdadm 的版本信息。

-b --brief：较少的细节。用于--detail 和--examine 选项。

-Q --query：查看一个 device，判断它为一个 md device 或是一个 md 阵列的一部分。

-E --examine：打印 device 上的 md superblock 的内容。

-c --config：指定配置文件，默认为/etc/mdadm.conf。

-s --scan：扫描配置文件或/proc/mdstat 以搜寻丢失的信息。配置文件是/etc/mdadm.conf。

一般 Linux 没有安装 mdadm，因而需要下载安装。在 Ubuntn 下的安装命令如下：

```
sudo apt-get install mdadm
```

此后就可以进行创建阵列等操作。例如：

创建阵列：

```
sudo mdadm --create --verbose /dev/md0 --level= 5 --raid-devices=4 /dev/sd[bcde]
```

查看阵列：

```
sudo mdadm -D /dev/md0
cat /proc/mdstat
```

删除阵列：

```
sudo mdadm --stop /dev/md0
```

对于创建好 RAID，需要将 RAID 的信息保存到 /etc/mdadm.conf 文件中，这样在操作系统重启时，系统就会自动加载此文件来启用 RAID。命令如下：

```
# mdadm -D --scan > /etc/mdadm.conf
# cat /etc/mdadm.conf
```

相关实验见习题 3 实验题的第 2 题。

3.5.2 Windows 下基于软件的 RAID

在 Windows 2000 中，引入了基本磁盘和动态磁盘这两个概念，并将它们用于 Windows 系统的管理员工具。大多数 PC 都将硬盘配置为基本磁盘，只有注重提高性能和可靠性的高级用户才会使用动态磁盘。

(1) 基本磁盘上的分区包括主分区、扩展分区和逻辑驱动器(逻辑驱动器即逻辑分区，又称为简单卷)，其中主分区最多有 4 个。扩展分区是一种特殊的主分区，可容纳多个逻辑驱动器(最多 128 个)。一个基本磁盘只能有一个扩展分区，因此一块基本磁盘上最多可以有 4 个主分区或者 3 个主分区加一个扩展分区。基本磁盘上的分区不能与其他分区共享或拆分数据，每个分区都可看作该硬盘上的一个独立单元。

(2) 动态磁盘只有卷(动态卷)或卷集，没有分区的概念，卷与基本磁盘上的主分区类似，但是在一块动态磁盘可以容纳大量(大约 2000 个)的卷。在 Windows 中，动态磁盘不但可以方便地改变卷的大小，而且可以将多个独立动态磁盘上的卷视为一个卷(跨区卷)使用，如图 3-19 所示。

(a) 基本磁盘　　(b) 动态磁盘

图 3-19　基本磁盘与动态磁盘

磁盘阵列就是动态磁盘的一种体现。它可以将数据拆分到多个磁盘(RAID 0) 以提高性能，也可以在多个磁盘之间镜像数据(RAID 1) 以提高可靠性。

在默认情况下，用户使用的都是基本磁盘。将新磁盘连入系统或在全新磁盘上初次安装系统时，会被初始化为基本磁盘(组 RAID 时除外)。如果有需要，在系统安装完成后可通过磁盘管理工具将其无损地转为动态磁盘。从基本磁盘转换成动态磁盘后，除非重新创建卷或者使用一些磁盘工具(例如分区助手)，否则不能将它无损转变回去。由于是有损地转换，所以转回基本磁盘前必须备份数据。

若要将两块 500GB 的硬盘划分为一个 800GB 的分区和一个 200GB 的分区，就只能使用动态磁盘实现，这是因为基本磁盘不能跨硬盘进行分区。动态磁盘实现跨越物理磁盘分区管理的功能与 RAID 有点类似。

与基本磁盘相比，动态磁盘有以下不同。

1. 可以任意更改磁盘容量

动态磁盘在不重新启动计算机的情况下可更改磁盘容量大小，而且不会丢失数据，而基本磁盘若不使用 PQMagic 等特殊磁盘工具软件改变分区容量就会丢失全部数据。

2. 磁盘空间的限制

动态磁盘可被扩展到磁盘中不连续的磁盘空间，还可以创建跨磁盘的卷集，将几个磁盘合为一个大卷集，而基本磁盘的分区必须是同一磁盘上的连续空间，分区的最大容量是磁盘的容量。

3. 卷集或分区个数

在一个动态磁盘上可创建的卷集个数没有限制，而在一个基本磁盘上最多只能分 4 个区。

动态磁盘只能在 Windows 系统中使用，其他的操作系统无法识别。

如果要将基本磁盘升级为动态磁盘，最少需要 1MB 没有被分配的磁盘空间。升级方法是右击“磁盘管理”界面右侧的磁盘盘符，在弹出的快捷菜单中选中“动态磁盘”选项。在升级过程中，会重新启动计算机，重启次数＝磁盘分区数量－1(例如磁盘分了 4 个区，那就需要重启 3 次)。升级过程会自动完成，在此期间，磁盘中的数据不会丢失。

如果要将动态磁盘转回基本磁盘，必须先备份从动态磁盘转换为基本磁盘的所有卷，然后在命令提示符窗口通过自带系统维护工具(diskpart 命令)实施转换。磁盘分区实用程序(diskpart 命令)是一个用于管理 Windows 操作系统中硬盘的命令行工具，是 MS-DOS 操作系统中 FDisk 的替代工具。

因为大部分用户的磁盘都是基本磁盘，为了实现基于软件的 RAID，必须将其转换为动态磁盘，具体方法是，在“控制面板”窗口中选中“管理工具”，在弹出的“计算机管理”窗口中选中“磁盘管理”，通过“查看”菜单将其中的一个窗口切换为磁盘列表。这时就可以通过右键菜单将磁盘转换为动态磁盘。

在动态磁盘中，分卷称为卷。这个卷又分简单卷、跨区卷、带区卷、镜像卷和 RAID 5 卷，如图 3-20 所示。它的盘符一般是 DISK1、DISK2 等类似排序，不受字母排序限制，此外，也可以自己定义卷名称。用户可以管理很多个卷，也可以将不同硬盘分到一个卷，实现共同管理数据。动态磁盘具有很高的数据管理容错能力，例如在 RAID 5 模式下，当某个硬盘出现故障时，操作系统会自动读取另一个硬盘的数据，以保证数据的安全；而基本磁盘就没有如此高的数据保护能力，当硬盘损坏时，系统会立刻崩溃，无法挽救系统。

(a) 简单卷　(b) 条带卷　(c) 跨区卷　(d) 镜像卷　(e) RAID 5卷

图 3-20　Windows 的卷类型

(1) 简单卷：和分区功能一样，简单卷包含单一磁盘上的磁盘空间。当系统中有两个或两个以上的动态磁盘并且两个磁盘上都有未分配的空间时，能够选择跨区卷或带区卷这两种分卷方式。

(2) 跨区卷：跨区卷用于将来自多个磁盘的未分配空间合并到一个逻辑卷中。

(3) 带区卷：带区卷用于组合多个(2～32 个)磁盘上的未分配空间到一个卷。当上述系统中的两个动态磁盘容量一致时，会看到另一个分区方式。

(4) 镜像卷：镜像卷是单一卷的两份相同副本，每份副本各在一个硬盘上，即 RAID 1。

(5) RAID 5 卷：当拥有 3 个或 3 个以上的动态磁盘时，就可以使用更加复杂的 RAID 方式——RAID 5，此时在分卷界面中会出现新的分卷形式。

RAID 5 卷相当于带奇偶校验的带区卷，即 RAID 5 方式。

可以把 RAID 5 理解成一种使用磁盘驱动器的方法，即将一组磁盘驱动器用某种逻辑方式联系起来，在逻辑上作为一个磁盘驱动器(即磁盘阵列)使用，一般应用于 SCSI 接口。

使用 RAID 5 后，除了数据被分散写入各个硬盘外，也会建立一份奇偶校验数据信息并保存在不同硬盘上(带校验的带区卷)。

RAID 5 方式的特点如下。

(1) 磁盘性能高(低于带区卷)。

(2) 利用率没有带区高(总共要有一块物理磁盘做校验)，假如有 N 块磁盘，则它的利用率为 $(N-1)/N\times100\%$。

(3) 由于需要计算校验位，所以写入性能有所下降，读取性能有所提高。

(4) 当一块磁盘损坏时，另一块磁盘中的数据会自动恢复，因此安全性高。

如图 3-21 所示，在跨区卷进行数据存储时，只有在第一块物理分区被占用完后才会使用第二块物

理分区的剩余空间；在带区卷进行数据存储时，会将所要存储的数据平均分散到具有相同类型的动态磁盘的相同分区；在镜像卷进行数据存储时，只能使用两块物理磁盘，但是可同时向两块物理磁盘存入相同的数据。

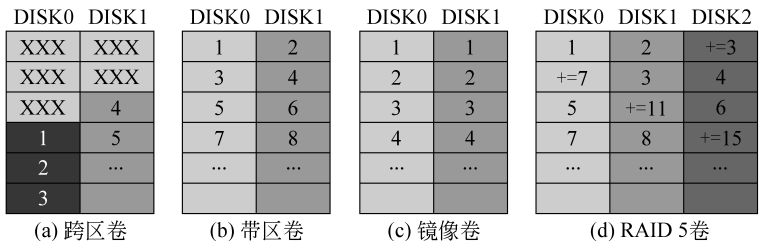


图 3-21 卷数据存储情况

RAID 5 卷至少需要 3 块硬盘，其中的一块用于存储校验信息，其他磁盘上的数据都是平均分配。

相关实验见习题 3 实验题的第 3 题。

3.6 基于硬件和基于软件的 RAID 技术比较

目前，基于硬件的 RAID 解决方案比基于软件的 RAID 技术在使用性能和服务性能上略胜一筹，在可检测和修复多位错误、RAID 保护的可引导阵列、错误磁盘自动检测、剩余空间取代和阵列重建、共有或指定的剩余空间、彩色编码报警等许多方面优于后者。此外，它还提供从单一控制实施的对多 RAID 安装、多操作系统远程检测和管理的能力。

从安装过程来看，两种 RAID 解决方案的安装过程都比较容易，安装耗时也相差无几；从 CPU 占用率看，基于硬件的 RAID 显然能够减少 CPU 的中断次数，同时降低主 PCI 总线的数据流量，使系统的性能产生一个提升；从 I/O 资源占用角度看，两种解决方案的差别并不算很大。基于硬件的 RAID 方案仅在减少 RAID 5 阵列在降级模式的运行时间和平行引导阵列的能力两方面有一定优势。另外，在硬件解决方案中，可以用 RAID 01 取代 RAID 1 来提高性能。尽管基于硬件的 RAID 方案具有优势，但是在产品的价格上仍然无法与基于软件的 RAID 抗衡，这是因为后者完全免费的。尽管如此，硬件解决方案的价格也不是不可接受的，一般情况下，只需增加少许投资即可获得一套基于硬件的入门级 RAID 解决方案。此外，在计算总拥有成本时，还必须考虑基于软件的 RAID 解决方案的隐性成本，例如用户生产效率、管理成本和重新配置的投资等。这些成本综合起来，往往会超过基于硬件的 RAID 解决方案所需的投资。

现在，任务密集型数据已应用于各种商业活动。为了使数据获得更好的保护，许多企业已经开始利用 RAID 技术。优秀的 RAID 解决方案一般都具有较高的可行性、友好的用户界面和简单的热键，可使用户在第一次使用时就能非常方便地运行系统。此外，还要为高级用户进行优化配置提供方便。

在使用系统功能或者 RAID 软件来实现 RAID 时，由于没有独立的硬件和接口，所以需要占用一定的系统资源（CPU、硬盘接口速度），并且受到操作系统稳定性的影响。在软件 RAID 中不能提供硬盘热插拔、硬盘热备份、远程阵列管理、可引导阵列支持等功能。不过，新一代 SATA 接口的软 RAID 可支持热插拔、热备份配置。

通过独立或主板集成的 RAID 硬件在使用时不需要占用其他硬件资源，稳定性和速度都比基于软件的 RAID 强，所以对服务器来说，最好是使用 RAID 硬件来提高计算机的性能。

3.7　性能检测与管理工具

3.7.1　文件系统的检测工具

IOzone 是一个文件系统检测工具，可从其官网下载。它可在不同的操作系统和文件系统中测试 write、re-write、read、re-read、random read、random write、random mix、backwards read、record rewrite、strided read、fwrite、frewrite、fread、mmap 等模式下硬盘的性能。在 Linux 平台上测试时，测试文件的大小一定要大于内存（一般为内存的两倍），否则会把读写的内容存入缓存，使结果失真。

1. IOzone 各项测试的定义

（1）write：测试写入新文件的性能。当写入一个新文件时，不仅需要存储文件中的数据，还包括定位存储介质中数据具体存储位置的额外信息。这些额外信息被称为元数据，其中包括目录信息、所分配的空间和一些与该文件有关但又并非该文件所含数据的其他数据。受这些额外信息影响，write 的性能通常会比 re-write 的性能低。

（2）re-write：测试写入已存在的文件的性能。当一个已存在的文件被写入时，所需工作量会较少，因为此时元数据已经存在，所以 re-write 的性能通常比 write 的性能高。

（3）read：测试读取已存在的文件的性能。

（4）re-read：测试读取最近读过的文件的性能。re-read 性能会高些，这是因为操作系统通常会缓存最近读过的文件数据。这个缓存可以被用于读以提高性能。

（5）random read：测试读取文件中的随机偏移量的性能。许多因素都可能影响这种情况下的系统性能，例如操作系统缓存的大小、磁盘数量、寻道延迟等。

（6）random write：测试写入文件中随机偏移量的性能。同样，有许多因素可能影响这种情况下的系统性能，例如操作系统缓存的大小、磁盘数量、寻道延迟等。

（7）random mix：测试读写文件中随机偏移量的性能。许多因素可能影响这种情况下的系统性能运作，例如操作系统缓存的大小、磁盘数量、寻道延迟等。这个测试只有在吞吐量测试模式下才能进行。每个线程或进程运行读或写测试。这种分布式读写测试是基于 round robin 模式的，因此最好使用多于一个线程或进程执行此测试。

（8）backwards read：测试使用倒序读取文件的性能。这种读文件的方法可能看起来有点奇怪，但是有些应用适合这种测试。例如，MSC Nastran 是一个使用倒序读文件的应用程序的一个例子。它所读的文件都十分大（大小从吉字节级别到太字节级别）。尽管许多操作系统使用一些特殊实现来优化顺序读文件的速度，很少有操作系统会注意并增强倒序读文件的性能。

（9）record rewrite：测试写与覆盖写文件中特定块的性能。如果这个块足够小（比 CPU 数据缓存小），则测出来的性能将会非常高；如果这个块比 CPU 数据缓存大而比转换检测缓冲区（translation lookaside buffer，TLB）小，则测出来的是另一个阶段的性能；如果比此二者都大，但比操作系统缓存小，则得到的性能又是一个阶段；若大到超过操作系统缓存，则又是另一番结果。

（10）strided read：测试跳跃读取文件的性能。例如，在 0 偏移量处读 4KB，然后在间隔 200KB 处读 4KB，再在间隔 200KB 处读取 4KB，如此反复。此时的模式是读取 4KB，间隔 200KB 并重复这个模式。这是一个典型的应用行为，当文件中使用了某个特定的数据结构并且需要访问这个数据结构特定区域的应用程序常常这样做。

许多操作系统并没注意到这种行为或者针对这种类型的访问做一些优化。同样，这种访问行为也可能导致一些性能异常。如在一个数据碎片化的文件系统里，应用程序的跳跃会导致某个特定的磁盘

成为性能的瓶颈。

(11) fwrite：测试调用库函数 fwrite()写入文件的性能。这是一个执行缓存与阻塞写操作的库例程。缓存位于用户空间内。如果一个应用程序想要写很小的传输块，fwrite()函数中的缓存与阻塞 I/O 功能能通过减少实际的操作系统调用并在调用时增加传输块的大小来增强应用程序的性能。这个测试是写一个新文件，所以需要写入元数据。

(12) frewrite：测试调用库函数 frewrite()写入文件的性能。这也是一个执行缓存与阻塞写操作的库例程。缓存位于用户空间内。如果一个应用程序想要写很小的传输块，frewrite()函数中的缓存与阻塞 I/O 功能可以通过减少实际的操作系统调用并在调用时增加传输块的大小来增强应用程序的性能。由于这个测试是写入已存在的文件，无需元数据操作，所以测试的性能会高些。

(13) fread：测试调用库函数 fread()读取文件的性能。这是一个执行缓存与阻塞读操作的库例程。缓存在用户空间之内。如果一个应用程序想要读很小的传输块，fread()函数中的缓存与阻塞 I/O 功能能通过减少实际的操作系统调用并在调用时增加传输块的大小从而增强应用程序的性能。

(14) mmap：许多操作系统支持 mmap()的使用来映射一个文件到用户地址空间。映射之后对内存的读写将与文件同步。这使应用程序能十分方便地将文件当作内存块使用。一个例子是内存中的一块将同时作为一个文件保存在于文件系统中。

2. IOzone 的用法

IOzone 语法如下。

```
iozone [-s filesize_Kb] [-r record_size_Kb ] [-f[path]filename]
       [-itest] [-E] [-p] [-a] [-A] [-z] [-Z] [-m] [-M] [-tchildren] [-h] [-o]
       [-l min_number_procs] [-u max_number_procs] [-v] [-R] [-x]
       [-d microseconds] [-F path1 path2...] [-V pattern] [-jstride]
       [-T] [-C] [-B] [-D] [-G] [-I] [-H depth] [-k depth] [-Umount_point]
       [-S cache_size] [-O] [-K] [-L line_size] [-gmax_filesize_Kb]
       [-n min_filesize_Kb] [-N] [-Q] [-P start_cpu] [-c] [-e] [-bfilename]
       [-J milliseconds] [-X filename] [-Y filename] [-w] [-W]
       [-y min_recordsize_Kb] [-q max_recordsize_Kb] [-+mfilename]
       [-+u ] [ -+d ] [-+p percent_read] [-+r] [-+t ] [-+A #]
```

由于参数过多，下面仅列出了部分参数的用法。

(1) -R：产生 Execl 格式的输出日志。

(2) -a：全面自动模式，使用的块大小为 4KB～16MB，当文件大于 32MB 时会自动停止使用低于 64KB 的块大小测试，这将节省测试时间。

(3) -A：由于测试时间过长，该参数告诉 IOzone 不介意等待，即使在文件非常大时也希望进行小块的测试。

(4) -b：输出结果时将创建一个兼容 Excel 的二进制文件。

(5) -B：使用 mmap()文件。这将使用 mmap()接口来创建并访问所有测试用的临时文件。一些应用程序倾向于将文件当作一个内存块使用。这些应用程序对文件执行 mmap()调用，然后就可以用读写内存的方式访问该块来完成文件 I/O。

(6) -c：计算时间时将运行 close()所用的时间包括进来。

(7) -C：显示吞吐量测试中每个客户传输的字节数。

(8) -d #：穿过“壁垒”时允许存在微秒级的延迟。在测试吞吐量时所有线程或进程在执行测试前