网络存储系统容错编码技术进展

摘要:专业的大型磁盘存储系统均发展为包含多块磁盘的大型阵列系统。随着系统中的磁盘数目的不断增加,由磁盘失效引起的数据丢失的可能性越来越大。对于由存储系统中部分磁盘失效所引起的数据丢失的问题,目前业界公认的比较好的解决方案是使用冗余容错编码技术来实现磁盘的容错。在工程实践中,目前广泛应用的编码方法大多局限于双容错阵列码。随着系统规模的进一步加大,3容错甚至更多容错的编码方法已引起研究者的重视。今后的5至10年间,对于3容错或多容错的编码方法的研究将会成为新的热点。

关键字:存储系统;容错编码;阵列码

英文摘要:Large professional hard disk storage systems are generally arranged as array systems consisting of many hard disks. However, as the number of hard disks increases, the probability of disk fault and data loss also increases. A redundant fault-tolerant coding technique can be employed that allows for faults among hard disks. Currently, only double fault-tolerant array codes are in widespread use; but with expansion of system size, different fault-tolerant coding streams will need to be investigated. Experts generally agree that triple- fault-tolerant coding will become the dominant technique with the next five to ten years.

英文关键字:storage systems; fault-tolerant coding; array code

基金项目:国家高技术研究发展(“863”)计划(2008AA01Z401);国家自然科学基金(60903028)

1 存储容错编码评价指标

近20年来,随着计算机技术的迅猛发展,大规模存储系统的发展也十分迅速。当前,普通PC机的存储器的容量已经达到了太比特级别,这较之20年前的20 MB存储容量提高了10 000倍。

除了传统的磁盘驱动器之外,新型的固态存储(SSD)存储器也已经走向市场。尽管单个存储器的容量发展迅速,但是却仍然赶不上人们对存储容量需求的增长速度。

随着大型计算机系统由“以计算为中心”向着“以信息处理为中心”的转变,以及信息量的爆炸式增长,人们对海量存储系统的需求日益提高。海量存储系统本质上是将很多的单个存储器件(下面均以磁盘为例),通过系统的接口,连接整合为一个虚拟的容量巨大的单一存储器,即磁盘阵列。

随着阵列中磁盘数目的增多,系统的可靠性也随之下降。工业界一般使用平均数据丢失时间(MTTDL)来衡量阵列的可靠性。

设单个磁盘的平均失效时间为MTTFdisk,则对于包含n块磁盘的无冗余阵列来说,其MTTDL可简单估计为:MTTDL=MTTFdisk/n。可见,当n较大时,整个

系统的可靠性将成比例下降。这对于较大规模的系统来说是不可接受的。利用冗余数据编码来提高系统可靠性是公认的解决这一问题的较好方法。通过巧妙地将 m 块标准大小的磁盘上的数据，增加部分冗余校验信息，编码后存放于 n 块磁盘上，使得系统满足：对于任意 k 块磁盘失效，都可以通过其他 n-k 块未失效盘中的数据解码恢复，则称整个系统是 k 容错的，或者称 k 为系统的容错数。

分析表明[1]，对于 k 容错的系统来说，可以近似估计为：

$$MTTDL = \frac{MTTF_{disk}^{k+1}}{n(n-1)\cdots(n-k)MTTR^{k}} \qquad (1)$$

因而，在大规模系统中，容错数可以说是另一种对系统可靠性的描述方式。市场中一般磁盘的 MTTFdisk 为 105 左右，系统修复时间 MTTR 一般为 10 左右。根据(1)式可以看出，当系统磁盘数为 103～104 时，一般 2 容错或是 3 容错编码就基本上可以满足存储系统的容错要求。

系统用于增加容错能力而添加的冗余越多，系统的额外造价也将越高。因而在具有相同容错数的前提下，人们往往追求更小的冗余度，即(n-m)/n 的值，其中 n 为系统磁盘数、m 为存储用户数据的磁盘数。根据编码理论的 Singleton 界，k 容错系统的最小冗余度为：k/n。达到这一最小值的编码方法称做 MDS 码。目前多数存储编码研究都集中于构造不同参数下的 MDS 码。

除了上述指标，任何计算机系统的速度与效率永远是需要考量的重要指标。这里我们不讨论如何有效地并行处理多磁盘中的数据读取(那是另外一个较大的课题)，而着重研究由于冗余编码带来的额外计算开销。对于即便是相同的编码方法，由于编/解码算法的不同，可能计算效率的差异较大。由于在计算机系统中，最终的编码运算都会反映为一些二进制运算，因而研究者通常使用编码需要的总的二进制异或运算次数来衡量由于额外冗余编码带来的系统计算开销。对于一个随机存取的存储系统来说，随机小块信息写操作的性能尤为重要。编码运算中每个单元所参与的平均异或次数可以用来衡量这一指标，我们称其为编码的更新复杂度。

综合上面讨论，存储系统容错编码问题可以归结为寻求对如下指标进行优化的编码方法

系统满足需要的容错性能，容错数为 k 的系统。
系统有较小(或最优)的冗余度
系统有较小(或最优)的编码/更新复杂度。

2 线性编码

对于单容错系统来说，简单的奇偶校验即可使得上面的 3 个指标达到最优。经典的系统都是使用的这种方法。然而对于 k 大于 1 的情况，问题的解决就不是那么简单了。从通信编码理论的丰富成果中，两种比较有代表性的编码方法被人们挑选出来，并用于解决存储容错问题，他们是二进制线性码和 RS 码。

2.1 多维阵列码

图 1 所示是二维阵列编码及校验矩阵。二维阵列码是奇偶校验的自然推广，由图 1 很容易看出它是双容错的。二维阵列码保持了单容错时奇偶校验码的最优编码复杂度的特性，但是二维阵列码的冗余度不再是最优的了。

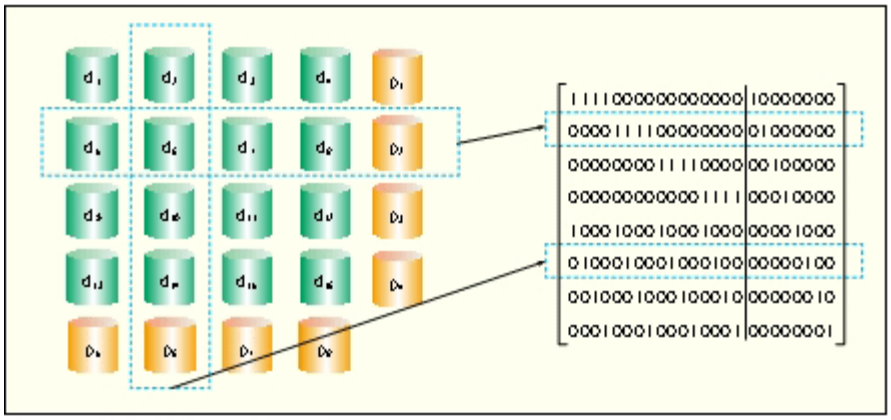


图 1 二维阵列编码及校验矩阵

二维阵列码也很容易推广为 k 维阵列。并且容易得到这样编码的 k 容错特性。但是随着 k 的增大，冗余会越来越大[2-3]。

2.2 Full 码

图 2 所示是 FULL-2 码。FULL-2 码可看做是二维阵列码的推广。

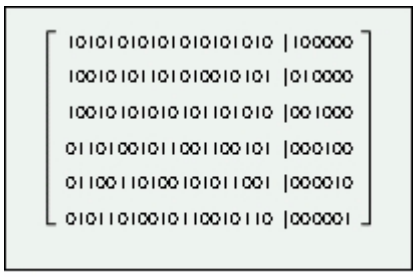
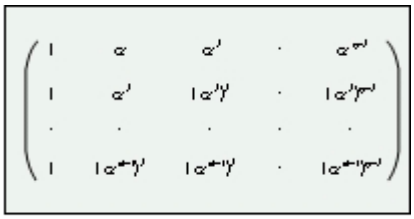


图 2 FULL-2 码

FULL 码依然保持了最优的编码复杂度，并且冗余度要比阵列码好很多。然而不幸的是，当 k 大于 3 时，FULL-k 码不再是 k 容错的[4]。

2.3 RS 码

图 3 所示是 RS 码的校验矩阵。RS 码从最佳的冗余特性出发。达到 Singleton 界的 RS 码被人们提出并广泛应用。



▲图3 RS码的校验矩阵

校验矩阵通过线性变换可以化为系统矩阵，用存储系统的语言亦可显式地区分出系统中哪些单元用于存储校验单元。可以看出，矩阵中的元素不再是“0”、“1”，而为有限域元素的幂，故编码需要使用有限域运算。在计算机系统中，有限域元素最后还是会被映射为“0”、“1”单元。这时每个有限域元素一般会被映射为多个“0”、“1”单元，而有限域运算也可以被分解为这些“0”、“1”单元的复杂运算。我们仍然以编码所需的异或运算为基本单位，则编码所需的异或运算次数和编码算法的巧妙程度有关。目前较好的域运算算法所需的异或次数大约为O(n3)[5]，计算复杂度相当高。RS码是MDS码，故冗余度是最优的。

3 阵列编码

上述几种编码各有优缺点，那么是否存在对于多指标同时最优的k容错编码方法呢？自文献[5]提出EVENODD码起，一大类只使用异或运算的阵列编码被提出并被人们广泛研究。

多维阵列或FULL码等二进制线性码每块磁盘只取一个逻辑单元进行校验运算。而阵列码则在每块磁盘上取多个逻辑单元，一起交叉进行校验运算。校验计算同2进制线性码一样，只使用二进制异或运算，但冗余度却可以与RS码相同。

3.1 EVENODD码

EVENODD码的想法很简单，每块磁盘中取若干单元，排成方阵，然后将这些单元分成不同的校验组，另外添加两块磁盘用于存储校验单元。所有校验组均使用简单的二进制奇偶校验。

水平校验与对角校验如表1所示。表1中D代表用户数据单元，P代表冗余校验单元。可以看出，Disk1—Disk5存储用户数据单元；Disk6、7存储冗余校验单元。Disk6的各单元为用户数据各行的水平校验和，而Disk7的各单元为用户数据的辅对角线校验和。

▼表1 水平校验与对角校验

Disk1	Disk2	Disk3	Disk4	Disk5	Disk6	Disk7
D1,1	D1,2	D1,3	...	D1,5	P1	P5
D2,5	D2,1	D2,2	D2,3	D2,4	P2	P6
D3,4	D3,5	D3,1	D3,2	D3,3	P3	P7
D4,3	D4,4	D4,5	D4,1	D4,2	P4	P8
0	0	0	0	0	0	P9

设存储用户数据盘的数目为 p(如上例中 p=5)，则系统包含 p+2 块磁盘，前 p+1 块磁盘中的最后一个单元为虚拟 0 元，故每盘实际包含 p-1 个单元，最后一块磁盘包含 p 个单元。可以证明，当 p 为素数时系统是双容错的。

简单计算可知此时的系统的冗余度为(2p-1)/((p+2)(p-1)+1)。由于最后的校验盘多出一个单元，所以冗余度稍稍大于最优的 2/(p+2)。为了达到最优值，文献[5]中使用如下技巧：将多出的单元(即辅对角交验和)叠加到该盘其他单元上，构造 MDS 的 EVENODD 码如表 2 所示。

表 2 构造 MDS 的 EVENODD 码

Disk1	Disk2	Disk3	Disk4	Disk5	Disk6	Disk7
D1,1	D1,2	D1,3	D1,4	D1,5	P1	P1+P5
D2,1	D2,2	D2,3	D2,4	D2,5	P2	P2+P5
D3,1	D3,2	D3,3	D3,4	D3,5	P3	P3+P5
D4,1	D4,2	D4,3	D4,4	D4,5	P4	P4+P5

表 2 也可表示为如表 3 所示。

表 3 构造 MDS 的 EVENODD 码

Disk1	Disk2	Disk3	Disk4	Disk5	Disk6	Disk7
D1,1	D1,2	D1,3	D1,4	D1,5+4	P1	P5
D2,1	D2,2	D2,3	D2,5+4	D2,5	P2	P6
D3,1	D3,2	D3,5+4	D3,5	D3,5	P3	P7
D4,1	D4,5+4	D4,5	D4,5	D4,5	P4	P8

也就是说当第一辅对角校验和为 1 时，其他各对角校验为奇校验；当第一辅对角校验和为 0 时，其他各对角校验为偶校验。这就是它被命名为 EVENODD 码的原因。

3.2 RDP 码

从表 2 可以看出，为了得到冗余最优，EVENODD 码的辅对角线上的单元的更新复杂度很高。每次更新这些单元的数据时都要同时更新其他 p 个校验单元。对于双容错编码来说，最优值为 2。文献[6]中构造的 RDP 编码将这些单元的更新复杂度均衡到每个单元，从而有效地消除了写操作中更新性能的不均衡。一个包含水平校验的对角线校验如表 4 所示。

表 4 包含水平校验的对角线校验

Disk1	Disk2	Disk3	Disk4	Disk5	Disk6
D4,1	D1,2	D1,3	D1,4	P1	P6
D4,2	D2,4	D2,5	D2,6	P2	P7
D4,3	D3,6	D3,1	D3,7	P3	P8
D4,4	D4,7	D4,2	D4,5	P4	P9
0	0	0	0	0	P5

与 EVENODD 不同处在于，做对角校验时也包含了水平校验单元的一列(因此，数据单元也比 EVENODD 少了一列)。

同样的，RDP 的最后一个校验盘多出一个单元，使得整个系统不为 MDS 码。但 RDP 码的优势在于，简单地将多出的单元删去，系统仍然为双容错的。即得到如表 5 所示阵列。

表 5 RDP 的 MDS 码构造

Disk$_1$	Disk$_2$	Disk$_3$	Disk$_4$	Disk$_5$	Disk$_6$
$D_{1,1}$	$D_{1,2}$	$D_{1,3}$	$D_{1,4}$	$P_{1,1}$	$P_{1,2}$
$D_{2,4}$	$D_{2,5}$	$D_{2,6}$	$D_{2,1}$	$P_{2,1}$	$P_{2,2}$
$D_{3,3}$	$D_{3,4}$	$D_{3,5}$	$D_{3,6}$	$P_{3,1}$	$P_{3,2}$
$D_{4,6}$	$D_{4,1}$	$D_{4,2}$	$D_{4,3}$	$P_{4,1}$	$P_{4,2}$

从表 5 可以看出，所有数据单元的更新负载为 2 或 3，分布比 EVENODD 码要均匀，不会产生由编码方式带来的额外“瓶颈”，但系统的平均更新复杂度是相同的。

3.3 Liberation 码

从前面几种编码可以看出，所使用的方法都是水平校验加其他一种校验共同构成双容错。不同之处就在于“另一种校验”的不同选择。如将另一校验盘上的校验元看作一个“0”、“1”向量，每块数据盘上的单元对这些校验元的影响可用一个“0”、“1”矩阵来表示。如表 5 中的第 1 列的 4 个数据单元对 Disk7 中的各校验元的影响可表示为如图 4 所示矩阵。

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

图 4 列校验的矩阵

在这种表示下，前面所说的更新复杂度就对应着矩阵中 1 的个数。于是构造一个双容错阵列码的问题就转变为：寻找若干个这样的矩阵，使得其中 1 的个数尽量少，并且任意 2 个之和为满秩。

在 p 为素数时，文献[7]中构造的 Liberation 码使得 p×p 阶矩阵 1 的数目不超过 p+1，其构造的 p 个矩阵可简单地描述为：各对角线加一个额外单元。第 k 个矩阵的额外的 1 单元的位置可描述为(k(p-1)/2 Mod p,1+k(p=1)/2 Mod p)。得到的编码如表 6 所示。

表 6 Liberation 码

3.4 PDHLatin 码

前面这些编码为 MDS 码的充要条件均为：码长与素数相关(RDP 为 p+1，其他为 p+2)。它们的双容错解码方法均为根据一个已知单元，然后通过校验关系与失效单元形成的链式关系依次恢复所有单元。这使人们理解到其容错能力的本质是任意两列都可以形成这样的关联结构。文献[8]中利用拉丁方构造了 PDHLatin 码，使得码长不再必须关联一个素数。

所谓拉丁方是指 n×n 的方阵中填入 n 个不同符号，使得每行每列的符号都不重复。显然拉丁方的每两列构成一个 n 元置换。所谓汉密尔顿拉丁方是指拉丁方的任何两列构成的置换为单环的。图 5 为一个 9 阶汉密尔顿拉丁方。

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 4 & 8 & 9 & 3 & 5 & 1 & 7 & 6 \\ 3 & 1 & 9 & 2 & 8 & 7 & 5 & 6 & 4 \\ 4 & 5 & 2 & 3 & 1 & 8 & 6 & 9 & 7 \\ 5 & 7 & 4 & 1 & 6 & 9 & 8 & 3 & 2 \\ 6 & 9 & 5 & 8 & 7 & 4 & 2 & 1 & 3 \\ 7 & 8 & 6 & 5 & 9 & 2 & 3 & 4 & 1 \\ 8 & 6 & 1 & 7 & 4 & 3 & 9 & 2 & 5 \\ 9 & 3 & 7 & 6 & 2 & 1 & 4 & 5 & 8 \end{bmatrix}$$

图 5 汉密尔顿拉丁方

从一个给定的汉密尔顿拉丁方，我们可以用与 EVENODD 码类似的方法构造编码，只不过各单元对于第二校验盘的校验关系不再依单元所在对角线位置决定，而是根据拉丁方相应位置的符号决定。根据图 5，得到表 7 所示的 PDHLatin 码。

表 7 PDHLatin 码

$Disk_1$	$Disk_2$	$Disk_3$	$Disk_4$	$Disk_5$	$Disk_6$	$Disk_7$	$Disk_8$	$Disk_9$	$Disk_{10}$	$Disk_{11}$
$D_{0,1}$	$D_{0,2}$	$D_{0,3}$	$D_{0,4}$	$D_{0,5}$	$D_{0,6}$	$D_{0,7}$	$D_{0,8}$	$D_{0,9}$	P_0	P_1
$D_{1,0}$	$D_{1,2}$	$D_{1,3}$	$D_{1,4}$	$D_{1,5}$	$D_{1,6}$	$D_{1,7}$	$D_{1,8}$	$D_{1,9}$	P_1	P_2
$D_{2,0}$	$D_{2,1}$	$D_{2,3}$	$D_{2,4}$	$D_{2,5}$	$D_{2,6}$	$D_{2,7}$	$D_{2,8}$	$D_{2,9}$	P_2	P_3
$D_{3,0}$	$D_{3,1}$	$D_{3,2}$	$D_{3,4}$	$D_{3,5}$	$D_{3,6}$	$D_{3,7}$	$D_{3,8}$	$D_{3,9}$	P_3	P_4
$D_{4,0}$	$D_{4,1}$	$D_{4,2}$	$D_{4,3}$	$D_{4,5}$	$D_{4,6}$	$D_{4,7}$	$D_{4,8}$	$D_{4,9}$	P_4	P_5
$D_{5,0}$	$D_{5,1}$	$D_{5,2}$	$D_{5,3}$	$D_{5,4}$	$D_{5,6}$	$D_{5,7}$	$D_{5,8}$	$D_{5,9}$	P_5	P_6
$D_{6,0}$	$D_{6,1}$	$D_{6,2}$	$D_{6,3}$	$D_{6,4}$	$D_{6,5}$	$D_{6,7}$	$D_{6,8}$	$D_{6,9}$	P_6	P_7
$D_{7,0}$	$D_{7,1}$	$D_{7,2}$	$D_{7,3}$	$D_{7,4}$	$D_{7,5}$	$D_{7,6}$	$D_{7,8}$	$D_{7,9}$	P_7	P_8
$D_{8,0}$	$D_{8,1}$	$D_{8,2}$	$D_{8,3}$	$D_{8,4}$	$D_{8,5}$	$D_{8,6}$	$D_{8,7}$	$D_{8,9}$	P_8	P_9
0	0	0	0	0	0	0	0	0	0	P_9

3.5 X 码

上面介绍的几种编码方法虽然都达到了冗余的最优，但在更新复杂度方面均稍高于最优值，那么是否可以达到两者同时最优呢？文献[9]提出的 X 码是一种这样的双容错编码。

X 码的想法也很简单，仍然是在阵列中采用主对角线和辅对角线两种校验，但是通过巧妙地将校验单元分布到各个磁盘中(而不是像其他方法中，校验单元被分离出来，独立存放于校验盘)，使得系统同时达到了两方面指标同时最优。

为了满足双容错的要求，X 码也要求阵列中包含的列数(或说码长)为素数。码长为素数 p 的 X 码中，每一列包含 p-2 个用户数据单元，2 个冗余校验单元。

3.6 B 码

是否还存在与 X 码相同特性的其他编码方案呢？显然将两个 X 码阵列重叠，系统仍然保持最优冗余与最优更新复杂度。

这样得到的新编码，在磁盘数目不变的情况下，每块盘需要关联的单元数目加倍。而在实际中，为了简化实现，我们实际上需要每块盘关联的单元数目尽量少。对于 n 块磁盘，在保持最优冗余与最优更新复杂度的条件下，每块盘最少需要多少个单元来关联校验呢？文献[10]提出的 B 码在双容错的情况下很好地解决了这一问题。

通过将编码构造等同于图论中的完全图完美-因子分解问题。并根据图论已有的结论，给出一种各方面性能均达到最优的编码。依据一个完全图的一种完美 1 因子分解方案，我们可以构造如表 8 所示的双容错编码——B 码。

表 8 B 码

$Disk_1$	$Disk_2$	$Disk_3$	$Disk_4$	$Disk_5$
$D_{0,0}$	$D_{0,1}$	$D_{0,2}$	$D_{0,3}$	$D_{0,4}$
P_0	P_1	P_2	P_3	$D_{1,4}$

这种编码，每块磁盘包含至多 1 个校验单元，并且只有一块磁盘不包含校验单元。它将 n 个符号的所有 2 元组分划为多列，并且满足双容错要求，因而在保

持了最优冗余度与更新复杂度的前提下，码长达到最长。因而这种编码也被称做最长最低密度阵列码。

3.7 T 码

对于 3 容错的最长最低密度阵列码的构造较之双容错要复杂很多。文献[11]最先给出了一种这样的构造，并利用计算机辅助证明了某些参数下，3、4 容错最长最低密度阵列码的 MDS 性。在文献[12]中独立构造了同样的编码并利用组合结构近乎可分的不完全区组设计(NRB)给出了这种编码的组合解释，同时也给出了简明的代数证明。

T 码从形式上与 B 码相同，每块磁盘包含至多 1 个校验单元，并且只有一块磁盘不包含校验单元。文献[12]证明了对于任意容错的最长最低密度阵列码均满足这种性质。

对于普遍参数的 T 码，或任意容错的最长最低密度阵列码的构造，仍是困难问题。

3.8 Weaver 码

前面的编码都将优化冗余率最优设为第一目标，同时兼顾编码/更新复杂度。但在一些系统中，如果冗余率的适当损失可换来更好的性能或更易于部署，则也是可选择的。文献[13]从优先考虑系统编码/更新复杂度的角度，提出了易于构造的 Weaver 码。

由 B 码、T 码的构造也可以看出，在保持更新复杂度最优的前提下，校验单元分布在各磁盘中的编码比较容易构造。为了简化问题，文献[13]选择具有循环对称性的阵列进行研究。也就是说要求编码满足：(1)所有数据单元参与的校验组数为常数；(2)所有校验组包含的单元数目为常数；(3)如果磁盘 i 上的数据单元 j 参与磁盘 k 上的校验单元 p 所代表的校验组，则必有对于任何 $0 \leqslant x < n$ 满足第 i+x mod n 块盘上的数据单元 j 参与磁盘 k+x mod n 的校验单元 p 所代表的校验组。

为了更容易地得到 k 容错编码，文献[13]放宽了冗余的要求，只研究每块磁盘中，冗余校验单元不少于用户数据单元的情况。这样，Weaver 码的最好冗余率只有 50%。

4 结束语

阵列码尽管有着很多性能优势，但在目前的存储系统中，还是 RS 码及层叠 RAID(如 RAID1+0 等)使用得比较多。笔者认为其原因主要为以下几个方面：

首先是实现上的简单性因素：RS 码已经是工业界流行的技术，无论软硬件都有成熟的实现方案，而层叠 RAID 原理十分简单，所以这两种编码实施最简单易行。与之相对，阵列码多种多样、原理复杂，实施需要一定的投入。目前海量存储系统正处于发展阶段，什么是“最好的”编码尚不能形成定论，因而就目前阶段来讲，最简单的就是最好的。

其次，受到目前大部分应用的存储需求影响：尽管将多个单个部件合成一个统一的虚拟部件会有好处，但也会有相应的问题。如对 10 000 块磁盘是合成 1 个系统好呢？还是组成 10 每个包含 1 000 块磁盘的小系统好呢？这要根据需求来判断。一般来说小一些的系统会更容易管理和维护。目前只有极少的应用需要对超过 1 000 块盘容量的数据并行的处理，因而将系统分为多个较小系统是有益的。

第三，硬盘的造价较低且发展迅速：这使得人们可以比较“奢侈”地使用存储空间，因而大型存储系统的建造目前还处于“粗旷经营”阶段。相对于易实施性、易维护性、易扩展性，当前阶段冗余率还并不是主要决定因素。

但是，随着单磁盘容量的日趋饱和，系统对性能、容错、节能等需求的不断变化，海量存储系统构造相应的也会不断发展。明天的存储系统将会需要具备什么特性的编码形式，还需我们不断探索。

5 参考文献

[1] HARTLINE J R, RAO K K. Notes on Reliability Models for Non-MDS Erasure Codes [R]. Research Report. RJ10391(A0610-035). San Jose, CA,USA: IBM. 2006.

[2] 王新梅, 肖国镇. 纠错码——原理与方法 [M]. 西安：西安电子科技大学出版社, 2001.

[3] 林胜. 存储系统容错及阵列编码 [D]. 天津：南开大学, 2010.

[4] HELLERSTEIN L, GIBSON G A, KARP R M, et al. Coding Techniques for Handling Failures in Large Disk Arrays [J]. Algorithmic, 1994,12(3/4):182-208.

[5] BLAUM M, BRADY J, BRUCK J, et al. EVENODD: An Optimal Scheme for Tolerating Double Disk Failures in RAID Architectures [J]. ACM SIGARCH Computer Architecture News, 1994,22(1):245-254.

[6] CORBETT P, ENGLISH B, GOEL A. Row-diagonal Parity for Double Disk Failure Correction [C]//Proceedings of the 3rd USENIX Conference on File and Storage Technologies(FAST’04), Mar 31-Apr 2, 2004, San Francisco, CA,USA. Berkeley, CA, USA: USENIX Association, 2004: 14p.

[7] PLANK J S. The RAID-6 Liberation Codes [C]//Proceedings of the 6th USENIX Conference on File and Storage Technologies(FAST’08), Feb 26-29, 2008, San Jose, CA, USA. Berkeley, CA, USA: USENIX Association, 2008:97-110.

[8] WANG Gang, LIN Sheng, LIU Xiaoguang, et al. Combinatorial Constructions of Multi-erasure-correcting Codes with Independent Parity Symbols for Storage Systems [C]//Proceedings of the 13th IEEE Pacific Rim International Symposium on Dependable Computing(PRDC’07), Dec 17-19, 2007, Melbourne, Australia. Piscataway, NJ, USA: IEEE, 2007:61-68.

[9] XU Lihao, BRUCK J. X-code: MDS Array Codes with Optimal Encoding [J]. IEEE Transactions on Information Theory, 1999,45(1):272-276.

[10] XU Lihao, BOHOSSIAN V, BRUCK J, et al. Low Density MDS Codes and Factors of Complete Graphs [J]. IEEE Transactions on Information Theory, 1999,45(6): 1817-1826.
[11] LOUIDOR E, ROTH R M. Lowest-density MDS Codes over Extension Alphabets [C]//Proceedings of the IEEE International Symposium on Information Theory (ISIT’03), Jun 29-Jul 4,2003, Yokohama, Japan. Piscataway, NJ, USA: IEEE, 2003:58.
[12] LIN Sheng, WANG Gang, STONES D S, et al. T-code: 3-erasure Longest Lowest-density MDS Codes [J]. IEEE Journal on Selected Areas in Communications, 2010, 28(2):289-296.
[13] HAFNER J L. WEAVER Codes: Highly Fault Tolerant Erasure Codes for Storage Systems [C]//Proceedings of the 4th USENIX Conference on File and Storage Technologies(FAST’05), Dec 13-16,2005, San Francisco, CA,USA. Berkeley, CA, USA: USENIX Association, 2005.

林胜，南开大学计算机专业博士毕业，天津理工大学副教授，研究方向为存储编码、组合算法。

刘晓光，南开大学计算机专业博士毕业，南开大学信息技术科学学院副教授，研究方向为高性能计算、海量信息存储技术。

王刚，南开大学计算机专业博士毕业，南开大学信息技术科学学院教授，研究方向为海量信息存储技术、并行计算。