•信息工程•

DOI:10.15961/j.jsuese.201700066

逆码:一种可容3错的低密度MDS横式阵列码方法

陈　亮[1,2],袁德砦[1,2],滕鹏国[1,2],王晓京[1]

(1.中国科学院 成都计算机应用研究所,四川 成都 610041;2.中国科学院大学,北京 100049)

摘　要:磁盘阵列存储可采用阵列码技术提高系统的容错能力。随着对阵列存储系统的深入认识,阵列码的更新效率也逐步成为一项重要的性能指标。针对当前可容3错横式阵列码更新效率低的问题,提出了一种具有低密度特性的横式阵列码构造方法,称为逆码。不同于传统阵列码是利用特殊几何方法确定编码过程,逆码是从生成矩阵角度出发,通过构造编码分布矩阵确定编码过程。首先,基于域$GF(2^w)$给出了一种具有超正规性质但只有3行元素的矩阵结构,称为逆结构矩阵;然后,利用$w\times w$大小的比特方阵表示域$GF(2^w)$中元素;最后,通过提出的优化算法得到具有低密度性质的编码分布矩阵,进而确定逆码的编码过程。理论分析表明:逆码满足最大距离可分性质,可取得最优的存储效率;与STAR码、RTP码等容3错的阵列码相比,逆码的参数取值范围将不受素数的限制,参数设置更加连续。实验分析表明:相比于同样从生成矩阵确定编码过程的CRS码,逆码的稀疏度、更新效率以及编译码效率均有明显优势;相比于STAR码、RTP码,逆码的更新效率平均可提高20%;为了提高逆码的译码效率,文中也尝试了不同的异或序列技术。

关键词:磁盘阵列;横式阵列码;容3错;低密度;更新效率

中图分类号:TP302　　**文献标志码**:A　　**文章编号**:2096-3246(2017)05-0135-08

Inverse Code: A Low-density MDS Horizontal Array Code Tolerating Triple Faults

CHEN Liang[1,2], *YUAN Dezhai*[1,2], *TENG Pengguo*[1,2], *WANG Xiaojing*[1]

(1.Chengdu Inst. of Computer Applications, Chinese Academy of Sciences, Chengdu 610041, China;
2.Univ. of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Array codes are widely used to improve the fault tolerance ability of redundant arrays inindependent disks(RAID) storage system.With the in-depth understanding of RAID storage system,the update efficiency of array codes has gradually become an important performance metric.In order to deal with the low update efficiency problem of current triple fault tolerant horizontal array codes,a new kind of low-density and triple fault tolerant horizontal array code called inverse code was proposed.Unlike the traditional array codes whose encoding process was determined by using special geometric methods,the encoding process of inverse codes were determined by constructing coding distribution matrix in generator matrix.Firstly,a matrix structure based on $GF(2^w)$ called inverse structure matrix was proposed,which is a super-regular matrix and contains only three rows of elements.Secondly,bit square matrices of size $w\times w$ were used to represent the elements of $GF(2^w)$.Finally,the coding distribution matrix with low density property was produced by a new optimized algorithm and used to determine the encoding process of inverse code.Theoretical analysis showed that the inverse code was maximum distance separable and obtained the optimal storage efficiency.Moreover,the parameters of inverse code were not constrained to be prime numbers and more continuous compared with other triple fault tolerant horizontal codes such as STAR code and RDP code.Experimental results showed that inverse code had the advantages in sparse degree,update efficiency,encoding and decoding efficiency over CRS code whose encoding process was determined by constructing generator matrix.Compared with STAR code and RDP code,the update efficiency of inverse code was improved by 20% on average.Besides,different kinds of 'XOR scheduling' techniques were used to improve the decoding efficiency of inverse code.

Key words: RAID;horizontal array code;triple fault tolerance;low-density;update efficiency

收稿日期:2017 – 01 – 19

基金项目:国家自然科学基金青年科学基金资助项目(61501064);四川省科技厅支撑计划项目资助(2015GZ0088)

作者简介:陈　亮(1990—),男,博士生.研究方向:存储容灾与编码方法.E-mail:chenliangnbanba@163.com

网络出版时间:2017 – 07 – 13 16 : 01 : 36　　**网络出版地址**:http://kns.cnki.net/kcms/detail/51.1773.TB.20170713.1601.001.html

独立磁盘冗余阵列(redundant arrays of independent disks, RAID)通过将数据有序存放在磁盘阵列中,同时生成和存储校验数据,达到提高数据可靠性的效果[1]。早期的RAID系统采用简单的备份或单个奇偶校验位生成校验数据,如RAID1～RAID5,只能容忍单个磁盘丢失或损坏。为了提高存储系统的容错能力,在RAID存储系统中开始采用阵列码技术,使其可以容忍任意2、3个,甚至更多磁盘同时出错。阵列码属于一类纠删码方法,它将数据布局在2维阵列中,通过生成校验数据,提高容错能力。由于RAID存储系统中数据处于不断修改、更新的状态,除了容错能力,阵列码的更新效率也是不可忽视的性能指标。

根据校验数据在2维阵列中分布位置的不同,可将阵列码分为横式阵列码、垂直阵列码和混合结构阵列码。由于各类阵列码之间的数据布局、校验位的运算方法都大不相同,它们的容错能力、存储效率、编译码复杂度、更新复杂度、扩展能力等性能均有偏重[2−3]。横式阵列码通常具有扩展性强,码长易于变换等优点,但不易达到最优更新效率,如EVENODD码[4]、RDP码[5]、STAR码[6]、RTP码[7]等码字,其中STAR码、RTP码分别从EVENODD码、RDP码扩展得到。垂直阵列码通常可取得最优更新效率,但其码长固定,不易变换且扩展能力差,参数限制严苛,如X码[8]、WEAVER码、C码等码字;混合结构阵列码具有较强的容错能力,可容多错,但其存储效率往往很低,如GRID码[9]等。若阵列码具有最大距离可分性质(minimum distance separable, MDS),可满足Singleton边界条件[10],能达到理论最优存储效率。

针对横式阵列码的更新效率问题,国内外学者从构造低密度的生成矩阵角度出发,寻找可接近最优更新效率的编码方法。Blaum等[11]利用特殊形式的低重量方阵,提出了Blaum-Roth码,它是一种可容2错,且具有低密度特性的MDS码,但是对容多错的情况不具有MDS性质。Plank[12]利用另一种形式的低重量方阵,提出了可容2错且具有MDS性质的Liberation码,同Blaum-Roth码一样,Liberation码无法扩展到容多错的情况。针对容3错阵列码,Zhang等[13]提出了TIP码,其满足MDS性质,且更新复杂度达到理论最优,但由于各数据列均包含校验位,不属于横式阵列码,具有垂直码性质,即码字长度不易变换,参数固定。此外,针对容多错阵列码,Plank等[14]从柯西矩阵出发,将有限域元素利用比特方阵形式表示,即矩阵中只包含“0”或“1”元素,通过优化算法减少矩阵中“1”元素的个数,提出具有低密度特性的CRS码。CRS码的参数摆脱了传统阵列码存在的素数限制,参数更加自由,但针对容3错的情况,矩阵中“1”元素个数多,密度高,稀疏度较低。

作者将在Blaum、Plank等研究[11−12]的基础上,提出一种可容3错的低密度MDS横式阵列码构造方法,称为逆码(inverse code)。

1 横式阵列码与比特方阵

1.1 横式阵列码的生成矩阵形式

对于横式阵列码,其信息位集中布局在2维阵列的左边,校验位集中在右边。图1为有k列信息数据、r列校验数据、每列共有w位的横式阵列码。其中,$d_{i,j}$、$p_{i,j}$分别被称为信息元素、校验元素。

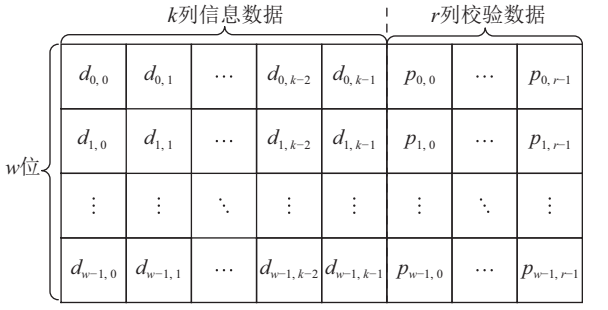

图 1 横式阵列码数据布局示意图

Fig.1 Horizontal array code data layout scheme

图1中,校验元素数据通过信息元素按特定的异或运算得出。当阵列码某列中出现部分数据丢失,则认为该列数据全部丢失。若阵列码任意r列丢失,数据可恢复,称其具有MDS性质。当r=2时,即RAID6编码;当r=3时,即容3错阵列码。

按照文献[12]中方法,将图1中k列信息数据依次从左到右、首尾相连,可得到1维形式的信息向量。同样地,可将r列校验数据转换为1维形式的校验向量;再根据阵列码编码算法,得出各校验元素p与各信息元素d之间异或和运算关系,从而,可推出阵列码的生成矩阵。图2为p=5时,STAR码的生成矩阵形式。其中,生成矩阵$\boldsymbol{G}$灰色部分表示取值为1,白色部分表示取值为0。

图2中,矩阵$\boldsymbol{G}$中上半部分为单位阵,下半部分为编码分布矩阵(coding distribution matrix, CDM)。各校验元素p是根据编码分布矩阵$\boldsymbol{B}$中各行包含元素“1”的分布情况,将对应的信息元素进行异或和运算得到,例如$p_{0,0}=d_{0,0}\oplus d_{0,1}\oplus d_{0,2}\oplus d_{0,3}\oplus d_{0,4}$。阵列码的存储性能主要由编码分布矩阵决定;若矩阵$\boldsymbol{B}$中包含元素“1”的个数越少,意味着编码过程异或次数减少,编译码速率更快,更新效率更高。对于容错性能,一种横式阵列码若为可容r错的MDS码,其编码向量中删除任意r个数据块$\boldsymbol{D}$、$\boldsymbol{P}$,剩余k个可读数据块对应生成矩阵$\boldsymbol{G}$的kw行,构成的$kw\times kw$子矩阵需均为满秩矩

阵，方可恢复丢失数据。

图3为编码分布矩阵的一般形式。

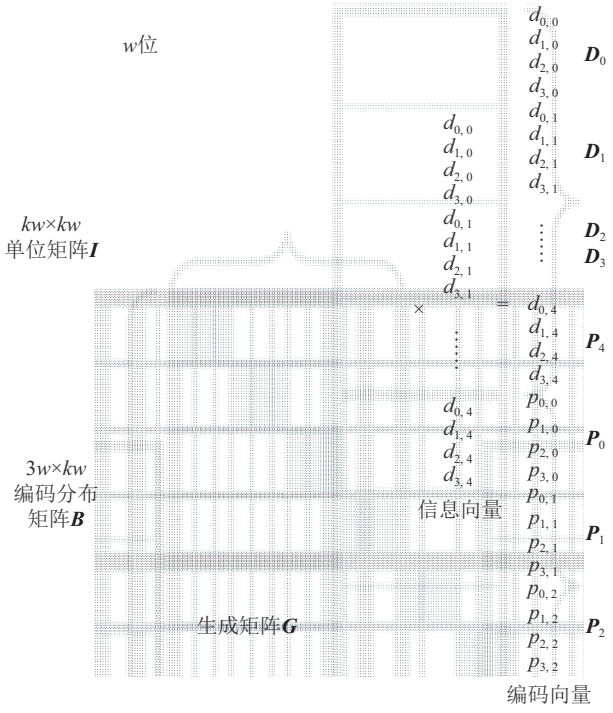

图 2　p=5时，STAR码的生成矩阵形式

Fig.2　Generator matrix of STAR-code (p=5)

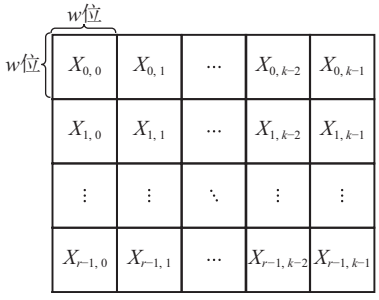

图 3　横式阵列码编码分布矩阵形式

Fig.3　Coding distribution matrix form of horizontal array code

图3中，每个元素$\boldsymbol{X}_{i,j}$对应一个$w\times w$的比特矩阵，矩阵中元素只为“0”或“1”，w值等于图1中阵列码的行数，k、r分别对应阵列码中数据列的个数、校验列的个数。针对容错性能，文献[11–12]从编码分布矩阵角度出发指出若阵列码为一种可容任意r错的MDS码，则其编码分布矩阵需要满足超正规（super-regular）[15]性质，即由矩阵中任意m行m列（$1\le m\le r$）上元素构造的$wm\times wm$子矩阵必须为满秩矩阵。

1.2　域$GF(2^w)$元素的方阵表示方法

传统Galois域$GF(2^w)$中元素是采用多项式形式表示，元素间加法运算为简单的异或操作，运算效率高；但是，元素间的乘法运算通常需要预先建立离散对数表或乘法表，再利用查表方式才能处理，复杂度高、效率低。当有限域规模较大时，建表与查表的耗时以和内存开销将不容忽视。为了避开有限域中乘法运算复杂度高的问题，文献[16]将域$GF(2^w)$中元素利用$w\times w$的比特矩阵表示，并进行编译码操作，使得乘法操作转换为异或运算。文献[17]基于$GF(2)$上的w次本原多项式$f(x)=a_0+a_1x+\cdots+a_{w-1}x^{w-1}+x^w$，给出域$GF(2^w)$矩阵形式的生成元，称其为多项式的友阵（companion matrix）：

$$\boldsymbol{A}=\begin{bmatrix}0&0&\cdots&0&a_0\\1&0&\cdots&0&a_1\\0&1&\cdots&0&a_2\\\vdots&\vdots&&\vdots&\vdots\\0&0&\cdots&1&a_{w-1}\end{bmatrix}。$$

通过矩阵$\boldsymbol{A}$不断自乘，可得出域$GF(2^w)$中所有非零元素。

图4给出了域$GF(2^3)$中所有元素的矩阵表示形式，以及元素间加法、乘法运算示例。

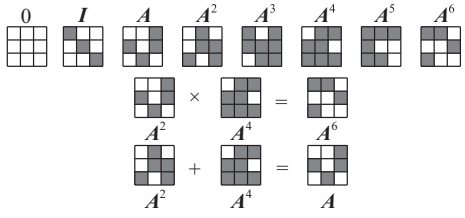

图 4　域$GF(2^3)$中元素的矩阵形式表示

Fig.4　Matrix representation of the elements of $GF(2^3)$

图4中，生成元矩阵$\boldsymbol{A}$对应的本原多项式为$f(x)=1+x+x^3$，有限域中乘法操作转换为比特矩阵间的点乘操作，只存在异或运算，提高了乘法计算效率，并且元素间加法操作的性能仍保持不变。

2　逆　码

2.1　逆结构矩阵

逆结构矩阵（inverse structure matrix，ISM）中，各元素均属于有限域$GF(2^w)$（$w\ge1$），其行数为3、列数可达到2^w-1，且矩阵中同一列上的第2、3行元素互为逆元素。它是一类具有超正规性质的矩阵结构，下面的定理给出其详细证明，具体矩阵结构如下：

$$\begin{bmatrix}1&1&1&\cdots&1&1\\\alpha_0&\alpha_1&\alpha_2&\cdots&\alpha_{k-2}&\alpha_{k-1}\\\alpha_0^{-1}&\alpha_1^{-1}&\alpha_2^{-1}&\cdots&\alpha_{k-2}^{-1}&\alpha_{k-1}^{-1}\end{bmatrix}。$$

其中，$\alpha_i\in GF(2^w)$，$\alpha_i\ne0$，$0\le i\le k-1$，$k\le2^w-1$。

定　理　逆结构矩阵中任意r行r列（$1\le r\le3$）上元素构造的子矩阵均满秩。

证明：1）当r=1时，由于逆结构矩阵中全为非零元素，显然任意1×1阶矩阵为满秩矩阵。

2）当r=2时，任意2×2阶矩阵可概括为如下3种形式：

$$\begin{bmatrix} 1 & 1 \\ \alpha_i & \alpha_j \end{bmatrix}、\begin{bmatrix} 1 & 1 \\ \alpha_i^{-1} & \alpha_j^{-1} \end{bmatrix}、\begin{bmatrix} \alpha_i & \alpha_j \\ \alpha_i^{-1} & \alpha_j^{-1} \end{bmatrix},$$

其中，$0\le i<j\le k-1$。

由于元素α_i、α_j各不相同，对上述3类矩阵进行简单初等列变换，可知：前两个矩阵为满秩矩阵；第3个矩阵的秩可化简为$\alpha_i\alpha_j\left(\alpha_j^{-1}+\alpha_i^{-1}\right)\left(\alpha_j^{-1}-\alpha_i^{-1}\right)$，在域$GF(2^w)$上，$\alpha_j^{-1}+\alpha_i^{-1}$与$\alpha_j^{-1}-\alpha_i^{-1}$均不为零，则第3个矩阵也为满秩矩阵。

3）当r=3时，任意3×3阶矩阵可概括为如下形式：

$$\boldsymbol{G}=\begin{bmatrix} 1 & 1 & 1 \\ \alpha_i & \alpha_j & \alpha_u \\ \alpha_i^{-1} & \alpha_j^{-1} & \alpha_u^{-1} \end{bmatrix},0\le i<j<u\le k-1。$$

将矩阵$\boldsymbol{G}$进行初等列变换可得：

$$\begin{bmatrix} 1 & 0 & 0 \\ \alpha_i & \alpha_j-\alpha_i & \alpha_u-\alpha_i \\ \alpha_i^{-1} & \alpha_j^{-1}-\alpha_i^{-1} & \alpha_u^{-1}-\alpha_i^{-1} \end{bmatrix}。$$

矩阵$\boldsymbol{G}$的满秩情况将由子矩阵$\boldsymbol{G}'=\begin{vmatrix} \alpha_j-\alpha_i & \alpha_u-\alpha_i \\ \alpha_j^{-1}-\alpha_i^{-1} & \alpha_u^{-1}-\alpha_i^{-1} \end{vmatrix}$的满秩情况决定。矩阵$\boldsymbol{G}'$秩的计算式如下：

$$\begin{aligned}|\boldsymbol{G}'|=&(\alpha_j-\alpha_i)(\alpha_u^{-1}-\alpha_i^{-1})-(\alpha_u-\alpha_i)(\alpha_j^{-1}-\alpha_i^{-1})=\\&\alpha_j\alpha_u^{-1}-\alpha_j\alpha_i^{-1}-\alpha_i\alpha_u^{-1}-\alpha_u\alpha_j^{-1}+\alpha_u\alpha_i^{-1}+\alpha_i\alpha_j^{-1}=\\&\alpha_j\alpha_u^{-1}(1-\alpha_u\alpha_i^{-1})-\alpha_i\alpha_u^{-1}(1-\alpha_u\alpha_j^{-1})-\\&(1-\alpha_u\alpha_i^{-1})+(1-\alpha_u\alpha_j^{-1})=\\&(\alpha_j\alpha_u^{-1}-1)(1-\alpha_u\alpha_i^{-1})-(\alpha_i\alpha_u^{-1}-1)(1-\alpha_u\alpha_j^{-1})=\\&(\alpha_j\alpha_u^{-1}-1)(1-\alpha_u\alpha_i^{-1})(1-\alpha_i\alpha_j^{-1})。\end{aligned}$$

由于α_i、α_j、α_u均属于域$GF(2^w)$中非零元素且各不相同，故$\alpha_j\alpha_u^{-1}-1$、$1-\alpha_u\alpha_i^{-1}$、$1-\alpha_i\alpha_j^{-1}$也必然均不为零元素，所以矩阵$\boldsymbol{G}'$为满秩矩阵。综上所述，逆结构矩阵中任意3×3阶矩阵都满秩。证毕。

将逆结构矩阵中各元素利用比特方阵表示，即得到逆码的编码分布矩阵。由上述定理可知，该编码分布矩阵满足超正规性质，根据第1.1节所述，不难推出逆码是一种可容3错的MDS横式阵列码。为了使编码分布矩阵具有低密度特性，下面给出一种逆码的优化构造算法。

2.2 逆码的优化构造算法

由于各横式阵列码的存储性能主要由编码分布矩阵决定，进而从构造低密度的编码分布矩阵出发，提出逆码（inverse code）的优化构造算法。如图1所示，阵列码参数主要包括k、w、r，由于逆码是一种针对容3错的MDS横式阵列码，可得r=3。当确定k、w值（$3\le k\le 2^w-1$）时，具有低密度性质的逆码的具体构造步骤如下：

1）输入参数k、w，并判断k、w取值是否合理。若k值属于[3, 2^w–1]，进行第2）步；反之，退出。

2）根据w值，确定$GF(2)$上一个系数最少且最高次数为w的本原多项式$f(x)$。关于本原多项式的选取可参照文献[18]。

3）构造$f(x)$对应的友阵$\boldsymbol{A}$，并将矩阵$\boldsymbol{A}$不断自乘，得到$GF(2^w)$上所有非零矩阵元素，记录各个矩阵包含“1”元素的个数。

4）遍历$GF(2^w)$中所有非零矩阵$\boldsymbol{A}^i$，并与对应逆矩阵$\boldsymbol{A}^{-i}$组合成一个矩阵对，计算两个矩阵包含“1”元素的个数之和，记为C_i（$0\le i\le 2^w-2$）。

5）根据C_i值从小到大的顺序，对上述矩阵对（$\boldsymbol{A}^i$，$\boldsymbol{A}^{-i}$）进行排序。

6）选取前k个矩阵对，即重量最小的k个矩阵对，放置在逆结构矩阵中。矩阵对（$\boldsymbol{A}^i$，$\boldsymbol{A}^{-i}$）中，原矩阵$\boldsymbol{A}^i$放在第2行，逆矩阵$\boldsymbol{A}^{-i}$放在对应的第3行。

7）满足参数w、k且能容3错的逆码的编码分布矩阵构造完成，码字构造完成。

图5为w=4，k=5时，采用$f(x)=1+x+x^4$本原多项式，按照上述算法步骤得到逆码的编码分布矩阵，其中共有74个元素值为“1”。而图2中具有相同容错能力和阵列规模的STAR码的校验矩阵中有84个元素为“1”，逆码的编码分布矩阵密度低了12%。同样，对比具有相同阵列规模的CRS码（n=5，m=2，w=4）[19]，CRS码有80个元素为“1”，逆码的矩阵密度降低了7.5%。

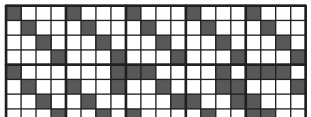

图 5 w=4，k=5时，逆码的编码分布矩阵

Fig.5 Inverse structure matrix of inverse code (w=4, k=5)

2.3 逆码的译码方法

逆码为一种可容3错的阵列码，将其应用于RAID系统时，若系统中出现任意不多于3个磁盘同时出错，均保证数据可恢复。逆码利用有限域元素比特矩阵形式，构造低重量的编码分布矩阵确定编码过程，其编码分布矩阵中“1”元素的个数虽然稀疏，但其分布未有明确规律。目前，可采用求解方程组方式恢复数据。

横式阵列码出现任意r（$1\le r\le 3$）列数据丢失时，对应图2编码向量将有r个数据块丢失，其中可包含信息块$\boldsymbol{D}$或校验块$\boldsymbol{P}$。同时，从剩余数据块中取出k个可读数据，并从生成矩阵中取出对应的kw行，可构造出$kw\times kw$的满秩矩阵$\boldsymbol{G}'$；矩阵$\boldsymbol{G}'$与原始信息向量相乘

即得到对应k个可读数据；为了恢复r个丢失的数据，需先求出矩阵$\boldsymbol{G}'$的逆矩阵$\boldsymbol{G}'^{-1}$，将逆矩阵$\boldsymbol{G}'^{-1}$与k个可读数据相乘，得出全部原始信息数据，再重新进行编码，可得到所有丢失的校验数据。

以图5中k=5、w=4、r=3逆码为例，当数据$\boldsymbol{D}_0$、$\boldsymbol{D}_1$、$\boldsymbol{D}_2$丢失，k个可读数据为$\boldsymbol{D}_3$、$\boldsymbol{D}_4$、$\boldsymbol{P}_0$、$\boldsymbol{P}_1$、$\boldsymbol{P}_2$；可构造$kw\times kw$满秩矩阵$\boldsymbol{G}'$，进而求出相应逆矩阵$\boldsymbol{G}'^{-1}$，将$\boldsymbol{G}'^{-1}$中对应丢失数据$\boldsymbol{D}_0$、$\boldsymbol{D}_1$、$\boldsymbol{D}_2$的行取出构造译码矩阵，与可读数据$\boldsymbol{D}_3$、$\boldsymbol{D}_4$、$\boldsymbol{P}_0$、$\boldsymbol{P}_1$、$\boldsymbol{P}_2$组成的向量进行运算，可将丢失数据$\boldsymbol{D}_0$、$\boldsymbol{D}_1$、$\boldsymbol{D}_2$恢复。运算过程，如图6所示。

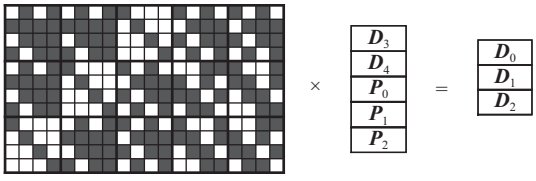

图 6　逆码译码过程示例

Fig.6　Instance of inverse code decoding

图6的译码运算过程中，若直接采用译码矩阵与向量相乘的方法，将会进行大量的异或运算，译码效率不高。但此类译码矩阵中“1”元素的个数较多，每行的运算会存在大量重复操作。可利用配套异或序列[12,20–21]技术对上述问题进行优化。

3　性能分析

实验在配置为Intel Core i3 CPU 3.07 GHz/3.06 GHz和4 GB RAM的PC机上进行。为了保证横式阵列码性能的可比性，选取了STAR码、RTP码、CRS码3类可纠3错的MDS横式阵列码与本文中提出的逆码在参数自由度、稀疏度、编码效率、更新效率、译码效率等性能进行对比。为了上述各性能分析的实验结果具有可比性，实验中设定STAR码、RTP码、CRS码、逆码的编码分布矩阵规模相同，即4类阵列码均在k、w、r值相同情况下进行讨论，此时编码分布矩阵的规模均为$3w\times kw$。

3.1　参数自由度

横式阵列码的参数主要包括k、r、w，其中，k表示数据列的个数，r表示校验列的个数，w表示2维阵列的行数（如图1所示）。目前，大多数横式阵列码的各校验列数据可通过不同斜率上的数据块运算得到，参数k、w值受到素数p限制，取值离散。如：STAR码的w值固定为p–1，k值也最多取为p；RTP码的w值也固定为p–1，k值最多取为p–1。虽然上述阵列码可将部分数据列视为全“0”元素的列，对k值进行调整，但始终无法超过上限素数p。

逆码和CRS码是通过编码分布矩阵和有限域元素的方阵形式构造，其k、w的取值摆脱了素数限制，w值可取区间[2，+∞]内任意整数，k值范围也可达到2^w–1，可取参数范围更广泛。例如：计算机采用二进制运算，而w=8时，当前很多阵列码无法构造此参数的码字，而逆码与CRS码可以做到；w=11时，逆码与CRS码的k值范围为3～1 023；而STAR码、RTP码的k值最多为11，码长可取范围远小于逆码。当固定码长时，逆码与CRS码也可利用更小的w值构造码字，w值越小意味着编译码过程中总体的异或次数越少，涉及的数据块个数也越少。

3.2　稀疏度

稀疏度指阵列码转换为生成矩阵形式，生成矩阵$\boldsymbol{G}$中编码分布矩阵（CDM）包含“1”元素的个数；编码分布矩阵的稀疏度直接影响阵列码的编码复杂度、更新复杂度。

图7为w=18这一固定值时，各阵列码的编码分布矩阵中包含“1”元素的个数随k值变化的趋势。由于STAR码、RTP码受到编码方法限制，k取值最多分别为19、18，故图7中横坐标没有继续延伸。此外，变换k值的过程中，STAR码是从右向左将数据列设为全零列，而RTP码是从左向右将数据列设为全零列，以取得最优稀疏度（在对其他性能进行比较时，均按此方法变换k值）。当w=18时，CRS码、逆码的有限域构造采用$f(x)=1+x^7+x^{18}$本原多项式。

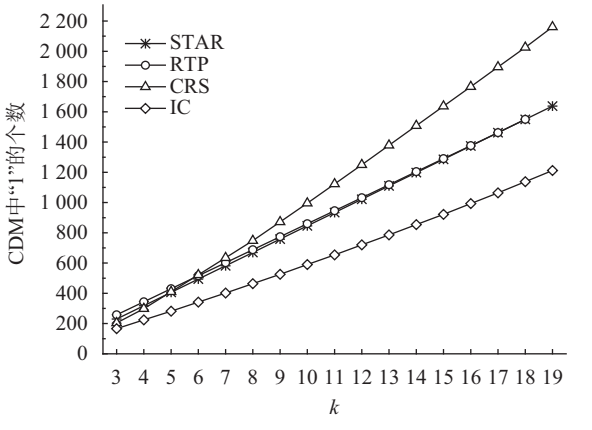

图 7　w=18，各阵列码稀疏度随k值的变化

Fig.7　Different array codes’ sparse degree change with k (w=18)

从图7中不难发现，随着k值的不断增加，各阵列码的编码分布矩阵规模也逐渐变大，包含“1”元素的个数均呈现上升趋势；逆码的编码分布矩阵中“1”元素的个数始终保持少于其他阵列码，并随着k值的不断增加，优势越明显。

图8为设定阵列码的数据列k与w值相同时，w取值为3～18，各阵列码的稀疏度随w值变化的趋势。其中，各w值下的本原多项式可参照文献[18]。由于STAR

码、RTP码构造过程要求w+1为素数，它们在图8中部分横坐标才有数据（即图8中竖虚线，另图10、12中竖虚线原因同上）。

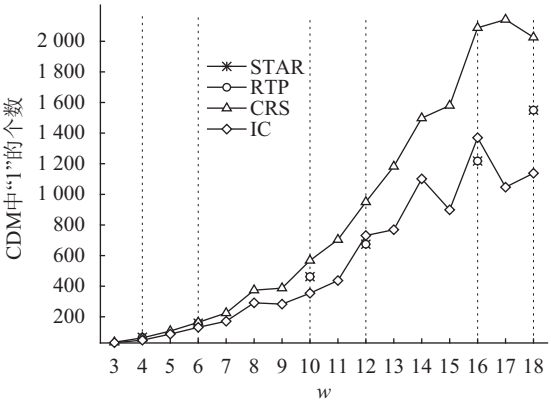

图 8　k=w, 各阵列码稀疏度随w值的变化

Fig.8　Different array codes' sparse degree change with w (k=w)

从图8中可看出：在不同w值下，逆码的稀疏度始终优于CRS码；STAR码与RTP码的稀疏度保持相同；在大部分情况下，逆码的稀疏度也优于STAR码和RTP码。总体上，逆码的编码分布矩阵稀疏度是优于其余阵列码，稀疏度可得到保证。

对于w=12、16时，逆码的编码分布矩阵中"1"元素的个数略高于STAR码和RTP码，其原因是采用的本原多项式中系数个数较多，分别为：$f(x)=1+x+x^4+x^6+x^{12}$、$f(x)=1+x+x^3+x^{12}+x^{16}$；两个本原多项式除去最高位，均有4个不为零的系数，对应友阵$\boldsymbol{A}$自乘生成的有限域中稀疏的矩阵元素较少；当数据列较多时，逆码的编码分布矩阵会存在较多的密度高的矩阵元素，使得整体密度下降。而当w=10、18时，本原多项式为$f(x)=1+x^3+x^{10}$、$f(x)=1+x^7+x^{18}$，除最高位，只有2个不为零的系数，此类有限域中稀疏的矩阵元素更多，生成的逆码稀疏度更高。

3.3　编码效率

各阵列码的编码运算只存在简单的异或操作，其中，编码开销指生成单个校验元素时数据元素间平均需要运行的异或次数，记为C_{E}。对于有k个数据列的阵列码，生成单个校验元素最少只需k–1次异或运算[2]。实验以各阵列码的编码开销除以k–1得到的比例度量各阵列码的编码效率，理论最优的编码效率对应值为1。实验中，STAR码、RTP码编码过程生成单个校验元所需异或次数均按照各自对应的编码算法计算得到；而对于CRS码、逆码，由于没有特殊的编码运算方法，以编码分布矩阵中各行平均包含"1"元素的个数减1的值为生成单个校验元所需异或次数。

图9为w=18时，各阵列码编码效率随k值的变化趋势。

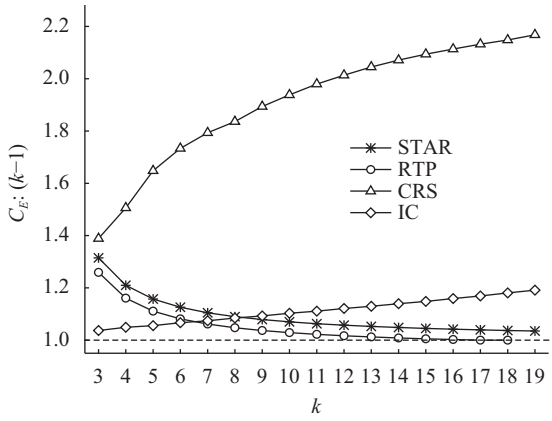

图 9　w=18, 各阵列码编码效率随k值的变化

Fig.9　Different array codes' encoding efficiency change with k (w=18)

从图9中可知：当k值较小时，逆码的编码效率优于其他阵列码；随着k值不断增大，逆码、CRS码的编码效率不断降低；相反地，STAR码与RTP码的编码效率不断提高，随k值增大，优于逆码；逆码的编码效率始终优于CRS码。造成上述现象主要原因是，STAR码与RTP码运算过程中均先计算了公共因子，如STAR码的调节因子、RTP码的第1列校验元素，此类运算减少了大量异或操作；而逆码、CRS码的编码运算过程，各校验元素相互独立，未查找公共因子进行优化；所以即便逆码的稀疏度优于STAR码、RTP码，而编码效率可能会比它们差。

图10为设定阵列码的数据列k与w值相同时，w取值为3～18，各阵列码编码效率随w值变化的趋势。从图10中可观察到，逆码的编码效率低于STAR码、RTP码，但始终优于CRS码。对于逆码的编码效率问题，可以应用异或序列技术进行优化。

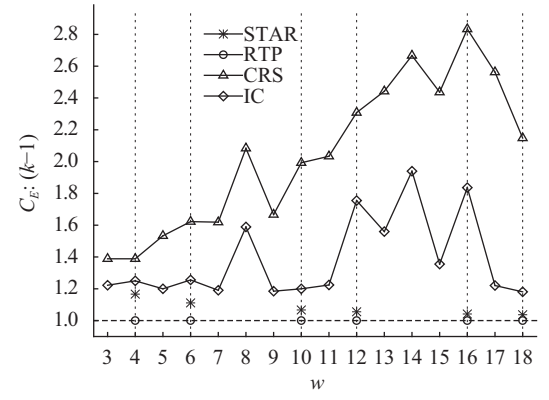

图 10　k=w, 各阵列码编码效率随w值的变化

Fig.10　Different array codes' encoding efficiency change with w (k=w)

3.4　更新效率

RAID系统中，编译码操作并不十分常见，更为常见的是连续不断的I/O操作和小写操作。由于RAID

系统采用了阵列码技术，系统中将出现校验数据，为了保证编码数据一致性，更新单个信息元素，同时需更新若干校验元素。这里，更新效率是指修改单个信息元素时平均需要更新的校验元素个数，记为C_U，则更新的校验个数越少，更新效率越高。对于可容3错的MDS阵列码，最优的更新效率为只更新3个校验元素。实验中，各阵列码的更新效率以各对应的编码分布矩阵包含“1”元素的个数除以矩阵中列数的结果进行度量，计为C_U。

图11为w=18时，各阵列码更新效率随k值的变化趋势。从图11中可知：w=18时，逆码的更新效率始终优于其他阵列码；当k值相同时，相对于其他阵列码，逆码平均可以少更新1个以上的校验元素，更新效率至少提高了20%。将逆码应用到RAID系统中，将节省系统的更新开销。

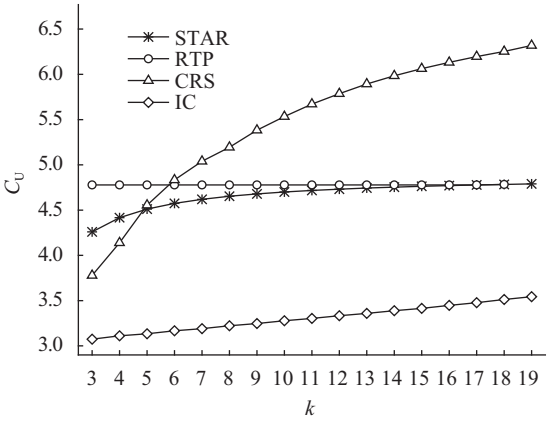

图 11　w=18, 各阵列码更新效率随k值的变化

Fig.11　Different array codes' update efficiency change with k (w=18)

图12为k=w情况下，各阵列码的更新效率随w值变化的趋势。由于更新效率是由编码分布矩阵中“1”元素的个数直接决定，故各阵列码的更新效率优劣情况与图7类似，原因也一致，此处不再赘述。

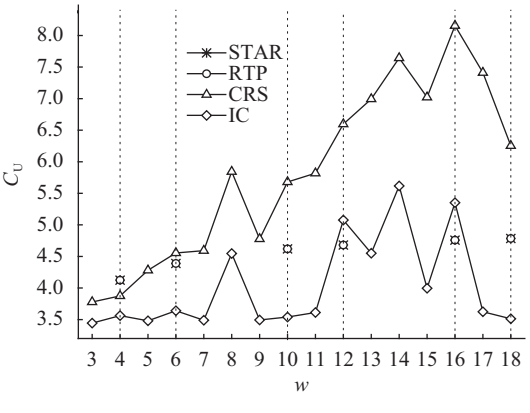

图 12　k=w, 各阵列码更新效率随w值的变化

Fig.12　Different array codes' update efficiency change with w (k=w)

3.5　译码效率

对于可容3错的逆码，若校验列丢失，可通过编码方式重新运算构造；若出现单个数据列丢失，可按RAID5方式进行恢复；若同时出现2、3个数据列丢失，均可按照第2.3节中译码方法进行恢复。由于当前逆码的译码过程是求解方程组，选择复杂度最高的3数据列出错情况，测试译码效率。实验中，针对每组k值，均遍历所有$\binom{k}{3}$种出错情况，得到恢复单个信息元素平均需要的异或次数，记为C_D；并使用其与理论上最少异或次数k–1的比例度量译码效率，则比例越低，效率越高。图13为w=18时，各阵列码译码效率随k值的变化趋势。

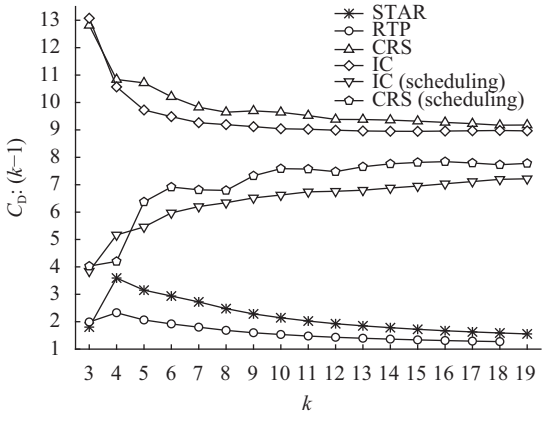

图 13　w=18, 各阵列码译码效率随k值的变化

Fig.13　Different array codes' decode efficiency change with k (w=18)

图13中，STAR码、RTP码均按各自独有译码算法计算异或次数，CRS码、逆码直接以译码矩阵每行平均包含“1”元素的个数再减1的值为译码所需异或次数；CRS(scheduling)、IC(scheduling)是对译码矩阵采用了文献[12]中异或序列进行优化后，计算译码过程所需异或次数。

从图13中可知，逆码的译码效率优于CRS码；但由于逆码和CRS码采用求解方程组形式进行译码，它们的译码效率低于STAR码、RTP码；CRS(scheduling)、IC(scheduling)的译码效率相比CRS、IC平均只可取得20%～30%性能提高，译码复杂度仍然很高。此外，还尝试了文献[20]的优化算法，可将CRS码与IC码的C_D与k–1的比例从9～10降低到2.3以下，但是算法的耗时更长。

4　结　论

RAID存储系统应用阵列码技术时，不仅需要考虑容错能力和存储效率，其更新效率也是一项重要的性能指标。本文提出了一种针对容3错的低密度横式阵列码方法，被称为逆码，它具备MDS性质，能达

到最优的存储效率。首先，在域$GF(2^w)$上给出了一种具有超正规性质的矩阵结构，称之为逆结构矩阵；再将有限域利用方阵形式表示，填充逆结构矩阵，构造出对应的编码分布矩阵。为了取得低密度特性，提出了构造逆码的编码分布矩阵的优化算法。相比于传统容3错的MDS阵列编码，逆码的参数选取突破了素数限制，选择更宽泛；在绝大多数参数情况下，逆码的编码分布矩阵稀疏度、更新效率均优于STAR码、RTP码等码字；与CRS码相比，逆码的稀疏度、更新效率、译码效率更是全面占优。因此，逆码适用于I/O操作、小写操作出现频繁的存储系统。

今后，逆码的进一步研究方向为：1）除了文中提到的友阵，将尝试更多的有限域矩阵生成元构造方法，进一步优化逆码稀疏度，实现在任何参数下，更新效率均高于STAR码等码类；2）对逆码的编译码过程，将研究如何摆脱求解方程组形式的复杂译码过程，开发针对逆码的译码算法或异或序列技术，提高效率；3）将逆结构矩阵推向一般化，使其在环上，也能满足矩阵的超正规性质；4）将逆结构矩阵规模扩大，尝试推广至可容更多错。

参考文献：

[1] Patterson D A,Gibson G,Katz R H.A case for redundant arrays of inexpensive disks (RAID) [C]//Proceedings of the 1988 ACM SIGMOD international conference on Management of data.New York:ACM,1988:109–116.

[2] Luo Xianghong,Shu Jiwu.Summary of research for erasure code in storage system[J].Journal of Computer Research and Development,2012,49(1):1–11.[罗象宏,舒继武.存储系统中的纠删码研究综述[J].计算机研究与发展,2012,49(1):1–11.]

[3] Subedi P,He Xubin.A comprehensive analysis of XOR-based erasure codes tolerating 3 or more concurrent failures[C]//Proceedings of the 27th IEEE International Symposium on Parallel & Distributed Processing,Workshops and Phd Forum.New York:IEEE,2013:1528–1537.

[4] Blaum M,Brady J,Bruck J,et al.EVENODD:An efficient scheme for tolerating double disk failures in RAID architectures[J].IEEE Transactions on Computers,1995,44(2):192–202.

[5] Corbett P,English B,Goel A,et al.Row-diagonal parity for double disk failure correction[C]//Proceedings of the 3rd USENIX Conference on File and Storage Technologies.New York:ACM,2004:1–14.

[6] Huang Cheng,Xu Lihao.STAR:An efficient coding scheme for correcting triple storage node failures[J].IEEE Transactions on Computers,2008,57(7):889–901.

[7] Goel A,Corbett P.RAID triple parity[J].ACM SIGOPS Operating Systems Review,2012,46(3):41–49.

[8] Xu Lihao,Bruck J.X-code:MDS array codes with optimal encoding[J].IEEE Transactions on Information Theory,1999,45(1):272–276.

[9] Li Mingqiang,Shu Jiwu,Zheng Weimin.GRID codes:Strip-based erasure codes with high fault tolerance for storage systems[J].ACM Transactions on Storage,2009,4(4):1–22.

[10] MacWilliams F J,Sloane N J A.The Theory of Error-Correcting Codes[M].New York:North Holland,1977.

[11] Blaum M,Roth R M.On lowest density MDS codes[J].IEEE Transactions on Information Theory,1999,45(1):46–59.

[12] Plank J S.The RAID-6 Liberation Codes [C]//Proceeding of the 6th Usenix Conference on File and Storage Technologies.New York:ACM,2008:97–110.

[13] Zhang Yongzhe,Wu Chentao,Li Jie,et al.TIP-Code:A three independent parity code to tolerate triple disk failures with optimal update complextiy[C]//Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks.Rio de Janeiro:IEEE,2015:136–147.

[14] Plank J S,Xu Lihao.Optimizing Cauchy Reed Solomon codes for fault-tolerant network storage applications [C]//Proceeding of the Fifth IEEE International Symposium on Network Computing and Applications.New York:ACM,2006:173–180.

[15] Roth R M,Lempel A.On MDS codes via Cauchy matrices[J].IEEE Transactions on Information Theory,1989,35(6):1314–1319.

[16] Blomer J,Kalfane M,Karp R,et al.An XOR-based erasure-resilient coding scheme[R].Berkeley,California:International Computer Science Institute,1995.

[17] Lidl R,Niederreiter H.Finite fields[M].2nd ed.London:Cambridge University Press,1997.

[18] Lin Shu,Costello J.Error Control Coding fundamentals and aplications[M].2nd ed.Upper Saddle River:Prentice Hall,Inc.,2004.

[19] Plank J S.Enumeration of optimal and good Cauchy matrices for Reed-Solomon coding[R].Knoxville:University of Tennessee,2005.

[20] Cheng Huang,Li Jin,Chen Minghua.On optimizing XOR-based codes for fault-tolerant storage applications[C]//Proceedings of the 2007 IEEE Information Theory Workshop.New York:IEEE,2007:218–223.

[21] Plank J S,Schuman C D,Robsion B D.Heuristics for optimizing matrix-based erasure codes for fault-tolerant storage systems[C]//Proceedings of the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks.New York:IEEE,2012:1–12.

（编辑　赵　婧）

引用格式:Chen Liang,Yuan Dezhai,Teng Pengguo,et al.Inverse Code: A low-density MDS horizontal array code tolerating triple faults[J].Advanced Engineering Sciences,2017,49(5):135–142.[陈亮，袁德砦，滕鹏国，等. 逆码：一种可容3错的低密度MDS横式阵列码方法[J].工程科学与技术,2017,49(5):135–142.]