随机二元扩展码:一种适用于分布式存储系统的编码

陈 亮[1),2)] 张景中[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 成都 610041)

[2)](中国科学院大学 北京 100049)

摘 要 随着分布式存储系统的存储容量快速增长,备份容灾存储效率低的缺陷日益明显,基于纠删码的容灾方法越来越受到重视.然而,应用于存储系统的纠删码研究起步较晚,可供选用的码类少,并且大多数属于通信领域的编码方法,不能很好满足存储领域的特殊需求.该文将提出一种新颖的存储编码方法,称为随机二元扩展码(Random Binary Extensive Code,RBEC),为数据容灾存储系统提供一种新的选择.RBEC 是一种基于异或运算的系统码,编码矩阵由一个单位阵和一个随机阵构成,采取自底向上的设计模式,通过控制随机矩阵中各个元素生成,达到码字整体上高性能.相比其他传统码类,RBEC 参数具有动态调整能力,其编码矩阵的行列可以自由伸缩.进而,存储系统可根据应用需求的变化,动态调整码率和纠删能力.对于(k,δ,t)参数 RBEC 码,该文给出了容任意 t 删除错的成功译码概率下界及其证明,并指出通过增加 δ 值可使译码概率下界无限趋近 1(100%).为了提高译码效率,该文进一步给出了一种简化译码矩阵规模的方法.最后介绍了 RBEC 在分布式存储系统的应用.

关键词 分布式存储系统;容灾;纠删码方法;动态调整

中图法分类号 TP302 **DOI 号** 10.11897/SP.J.1016.2017.01980

Random Binary Extensive Code(RBEC): An Efficient Code for Distributed Storage System

CHEN Liang[1),2)] ZHANG Jing-Zhong[1),2)] TENG Peng-Guo[1),2)] WANG Xiao-Jing[1)]

[1)](*Chengdu Institute of Computer Applications, Chinese Academy of Sciences, Chengdu* 610041)

[2)](*University of Chinese Academy of Sciences, Beijing* 100049)

Abstract With the rapid growth of the capacity of distributed storage system, the defect of low storage efficiency for replication fault tolerance has become more and more obvious. In this context, erasure code for fault tolerance has attracted much attention in recent years. However, the research of erasure code applied in storage system starts relatively late and the existing types are too less. Furthermore, most of them are used in the communication field but can't satisfy the special requirement of storage field. In this paper, we present a novel code called Random Binary Extensive Code (RBEC), which provides a new option for fault tolerance storage system. The RBEC is a kind of XOR-based systematic code, in which the generator matrix consists of an identity matrix and a random matrix. The bottom-up design model has been utilized in the design of the RBEC. Overall high performance can be achieved by controlling the generation of each element in the random matrix. In contrast to traditional codes, the RBEC has the ability of dynamically adjusting parameters, and the rows and columns of generator matrix can be scaled freely. Therefore, storage system can dynamically adjust the rate and the tolerance ability when the application requirements changed. For the (k,δ,t) RBEC, our work demonstrates and then

收稿日期:2015-12-30;在线出版日期:2016-09-29.本课题得到国家自然科学基金(61501064)和四川省科技厅支撑计划项目(2015GZ0088)资助.**陈 亮**,男,1990 年生,博士研究生,主要研究方向为存储系统可靠性、RAID 编码.E-mail:chenliangnbanba@163.com.**张景中**,男,1986 年生,博士研究生,主要研究方向为秘密分享、存储系统可靠性.**滕鹏国**,男,1986 年生,博士研究生,主要研究方向为存储系统可靠性、RAID 编码.**王晓京**,男,1953 年生,研究员,博士生导师,主要研究领域为信息安全、秘密分享和网络存储等.

gives the probability lower bound of successfully decoding in tolerating any t erasure errors, whose value can tend to 1(100%) infinitely by increasing the value of δ. In order to further improve decoding efficiency, we also propose a method to minimize the size of decoding matrix. Finally, we introduce the application of the RBEC in distributed storage system.

Keywords　distributed storage system; fault tolerance; erasure code method; dynamic adjustment

1　引　言

存储系统的安危是保证信息数据可靠性的根基.作为大数据时代的各级数据存储中心及网络云存储系统,基于冗余机制的数据容灾方法是其必备的保护措施[1-2].当前,基于复制备份的容灾技术与基于纠删码的容灾技术是两种主要的数据容灾方法.然而,应用复制备份方法的存储系统中数据通常维持 $N\geq 3$ 副本,有效存储空间为 $1/N$,存储效率低、系统成本高,已逐步难以满足现代存储系统需求[3-4].相比之下,纠删码方法通过牺牲部分计算效率,能够获得更高存储效率和容灾能力,越来越受到学者和企业的关注.

目前在纠删编码技术的数据冗余措施方面,基于 RS(Reed-Solomon)码方法的分布式存储容灾系统研究最为广泛,最有代表性.RS 码[5]是一种基于有限域运算,参数选择自由,并达到了理论上最优存储效率的编码方法.在工程应用领域,如谷歌 GFS (Google File System)[6]、微软 Azure[7]、淘宝 TFS (Taobao File System)等企业级存储系统已对 RS 码进行了研究.谷歌在 2009 年开发的第二代 GFS (Colossus)采用了 RS 码①;针对 RS 码在重构过程高带宽消耗问题,Huang 等人[8]对 RS 码做了改进,推出 LRC 码(Local Reconstruction Codes),通过减少部分容灾能力,换取更少的重建代价,并将 LRC 码应用到微软的 Azure 系统;Facebook 也在 2013 年将 LRC 码应用在 HDFS 系统(Hadoop 分布式文件系统)[9].在学术研究领域,对于 RS 码在有限域上乘法运算复杂度高的问题,Blomer 等人[10]提出将有限域中元素采取'0、1'矩阵形式表示,进而将乘法转化为异或运算;Plank[11]在此基础上提出了 CRS 码(Cauchy Reed-Solomon Code).此外,Luo 等人[12]从算法角度对有限域乘法运算进行了优化和改进;CPU 硬件方面处理能力的提高也对有限域计算效率起到了积极作用[13].

但是,RS 码的构造及其编译码运算始终无法摆脱有限域的束缚:随着存储系统规模的增长,RS 码的构造规模亦需增长;当 RS 码的参数不断增大时,有限域也将不断扩大,运算效率问题将再次成为大数据存储系统发展的桎梏.进而,学者开始在计算效率与存储效率两者间做出权衡,尝试其他只需异或运算的编码方法.

国际上研究用于数据冗余存储技术的其它候选码类还有 LDPC 码(Low Density Parity Check Code)[14]、喷泉码[15-18]以及网络编码(Network Coding)[19]等.与 RS 码相比,它们运算过程可只存在异或操作,摆脱了有限域的运算,并且采用概率型恢复模式,获得到更高编译码性能.虽然这些码类存储效率达不到理论最优,但仍远高于复制备份冗余方法.

在 2004 年,Plank 等人[20]对 LDPC 在存储系统中的应用做了实际测试分析.在 2010 年,Harihara 等人[21]利用 LDPC 码构建了存储系统 Spread-Store.由 LDPC 编码发展的喷泉码(Fountain Code)也在存储系统中得到应用,例如,OceanStore[22]存储系统采用 Tornado 码[15-16],RobuStore[23]存储系统采用无码率的 LT 码(Luby Transform Code)[17-18]等.此外,Dimakis 等人[24]尝试将网络编码应用于存储系统.Acedanski 等人[25]分析了存储系统采用完全随机码进行数据保护时的容灾性能.

然而,LDPC 编码的构造相对较难,参数选取限制严苛,扩展能力差;Tornado、LT 等码字受到译码方法影响,编译码时数据块参与数量庞大,存取流程复杂;而网络编码技术目前发展尚不成熟,并且要求存储网络中各节点均有存储和计算能力,工程量大,距离存储容灾的实用还有很大距离;Acedanski 利用的随机码过于简单,只观察了容灾性能,未综合考虑其他性能(如运算效率),有很大局限性.基于异或运算的编码方法若要在分布式存储系统中得到实际应用还需要更深入的研究.

大数据时代下数据量日益增长,存储系统对容

① Storage Architecture and Challenges. http://bit.ly/nUylRW, 2010

灾技术的需求也在逐步提高.当前主流的纠删码技术(如前所列举)大都继承通信技术上的编码方法,可选择的码类有限且不能完全满足存储系统的需求.因此发展大规模数据容灾存储编码技术提上了迫切日程.为此,本文在喷泉码、LDPC 码等编码方案基础上,提出一种用于数据冗余容灾的纠删编码方法:RBEC(Random Binary Extensive Code),为存储系统提供一种新的选择.RBEC 编译码过程只需用到异或运算,码字构造简单,并且具有突出的扩展能力和容灾能力,以及数据灾难高概率(概率可控)恢复等优点.

在文中第 2 节,将进一步介绍已有编码技术和数据块间运算机制;第 3 节,主要介绍 RBEC 编码结构以及编码矩阵的数学基础;第 4 节为 RBEC 的实验分析;第 5 节,介绍它在存储系统方面应用;第 6 节将做总结和后续研究展望.

2 相关工作

RS 码类、低密度奇偶校验码 LDPC 和喷泉码为当前主要纠删码技术,并均可采用编码矩阵 $\boldsymbol{G}$ 或译码矩阵 $\boldsymbol{H}$ 进行描述.根据编码矩阵形式可分为系统码和非系统码;相对非系统码,系统码生成的码字信息数据与校验数据相互独立,应用更加方便.

2.1 应用在存储系统中常见的编码方法

RS 码为 MDS(Maximum Distance Separable)码,(n,k)RS 码对 k 份信息块进行编码,生成 n 份数据块,n 份数据块中任意 k 份均能将原信息恢复,达到了参数(n,k)下的最大容删能力.然而,RS 码是在有限域上运算,乘法运算复杂度高.为了保证存储系统计算效率,通常限制域 $GF(2^m)$大小,导致应用过程中参数(n,k)取值受限.文献[10-11]中将域中元素以 $m\times m$ 的'0、1'方阵形式表示,使得乘法运算转化为异或运算(如图 1).不过这只是在 RS 码表现形式上的变化,对系统计算效率提升有限.与此同时,变换后的编码矩阵规模将以 m^2 级增大,数据编码存储也将更加复杂,容删能力依然未变.

Tornado 码是一种分层结构的编码,具有线性编译码时间,属于系统码.Tornado 码纠删性能由中间各层不规则稀疏图的构造决定,主要采用随机方式或 LDPC 码.Tornado 码构造时需要提前确定码率,再设计各层稀疏图,保证各层均能译码成功.由于编码设计复杂,且扩展性差,在实际应用有限.

LT 码为一种数字喷泉码,无码率特性可使冗余位无限扩展,译码过程采用迭代译码方式,其度分布设计尤为重要,直接影响着编译码性能.LT 码长普遍较长,需要数以千计的数据块参与编译码操作才能达到最优性能.当前 LT 码主要应用在通信领域,在存储系统并没有得到推广.

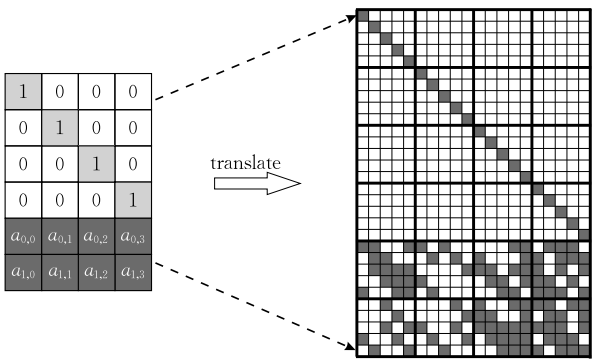


图 1 编码矩阵转化为'0、1'矩阵示意图

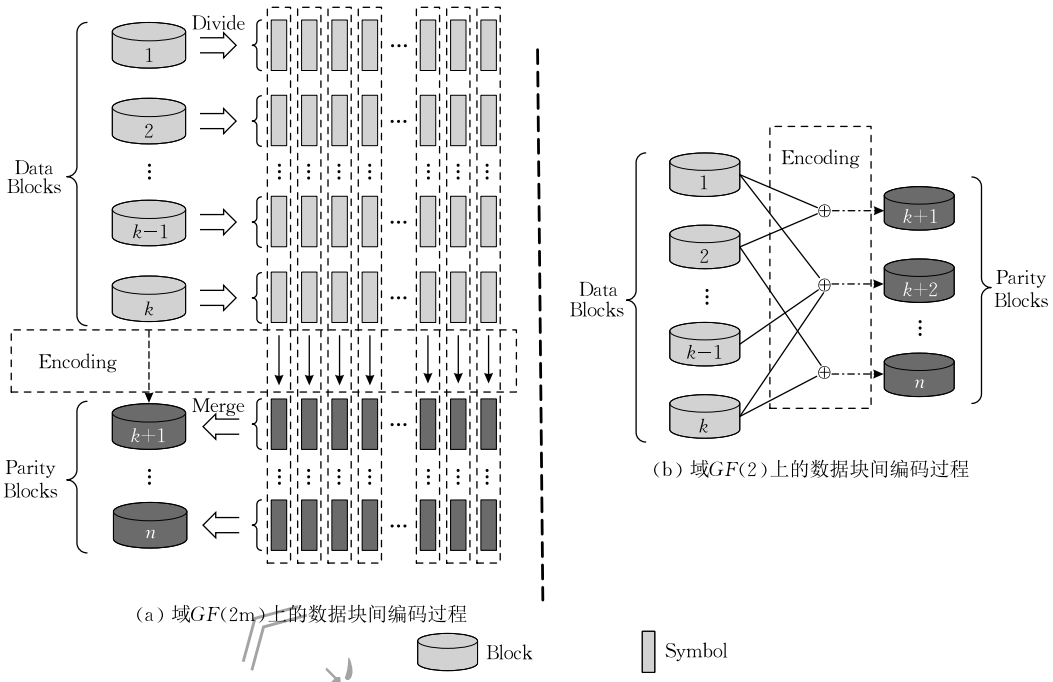
LDPC 码是一类由校验矩阵确定的线性分组码,可通过 Tanner 图进行构建.LDPC 的校验矩阵中大部分元素为'0',具有良好的低密度特性;编译码过程简单,速度快.对于长码字的 LDPC 码,目前主流研究集中在理论方面,确切给出码字构造方法并达到理论界限的成果很少.应用于存储系统的 LDPC 码还需不断探索与研究.

2.2 数据块间运算机制

不同构造方法的纠删码,其编译码运算基于的有限域规模各有不同.基于纠删码技术的容灾存储系统存取数据时,需对数据块进行预处理,将运算操作规约到限定的有限域范围,保证编译码过程顺利进行.当前存储系统采用的编码技术可分为基于 $GF(2^m)$运算和基于 $GF(2)$运算两大类.

传统 RS 码类的编码矩阵中各元素均属于域 $GF(2^m)$($k<n\leq 2^m$),一般取 $GF(2^8)$或 $GF(2^{16})$.然而待编码数据块(block)大小普遍以 MBytes、GBytes 为单位,此时数据块需要划分为若干个 mbits 大小模块(symbols),再对各模块采取编码技术,使得运算封闭在域 $GF(2^m)$,如图 2(a).

域 $GF(2)$上(n,k)纠删码的编码矩阵中各元素只存在'0、1',可以作为任意域 $GF(2^m)$的子域,此时 m 大小已无任何限制.给定 k 份、大小任意的待编码数据块 block,总存在域 $GF(2^X)$,X 为数据块 block 比特大小,使得各编码数据块都属于 $GF(2^X)$的元素;编码矩阵中的'0、1'元素可视为 $GF(2^X)$中的单位元和零元;则将运算封闭在域 $GF(2^X)$且只存在异或操作,如图 2(b).



(a) 域$GF(2m)$上的数据块间编码过程　　(b) 域$GF(2)$上的数据块间编码过程

图 2　不同域下编码过程示意图

3　随机二元扩展码

本节将给出一种新型编码方法:随机二元扩展码(Random Binary Extensive Code,RBEC),一种基于$GF(2)$运算的近似 MDS 码.它具有高容错能力、高概率恢复、突出的扩展能力等特性.下面首先介绍 RBEC 的编码结构,再给出一种适用于 RBEC 的译码方法,以及译码概率分析和理论证明,最后叙述 RBEC 参数动态调控能力.

3.1　随机二元扩展码 RBEC 的结构和编码过程

RBEC 的参数主要包括n,k,δ,t,其中n为码字的长度且满足$n=k+\delta+t$,k为信息部分的长度,δ为保证高概率满秩所需的冗余位数,t为 RBEC 可容最多丢失的个数,即最大容删除错能力.RBEC 的编码矩阵包含$k+\delta+t$行、k列,结构采取单位阵与随机矩阵结合方式.编码矩阵$\boldsymbol{G}$前k行为单位阵$\boldsymbol{I}_{k\times k}$,后$\delta+t$行为随机矩阵$\boldsymbol{R}_{(\delta+t)\times k}$且矩阵中各元素$r_{ij}$均按概率 0.5 独立选取'0'或'1',如图 3 所示.

$$\boldsymbol{G}=\begin{bmatrix}\boldsymbol{I}_{k\times k}\\ \boldsymbol{R}_{(\delta+t)\times k}\end{bmatrix}$$

$$\boldsymbol{I}_{k\times k}=\begin{bmatrix}1&0&\cdots&0\\0&1&\cdots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&1\end{bmatrix}\quad \boldsymbol{R}_{(\delta+t)\times k}=\begin{bmatrix}r_{1,1}&r_{1,2}&\cdots&r_{1,k}\\r_{2,1}&r_{2,2}&\cdots&r_{2,k}\\\vdots&\vdots&\ddots&\vdots\\r_{\delta+t,1}&r_{\delta+t,2}&\cdots&r_{\delta+t,k}\end{bmatrix}$$

图 3　RBEC 的编码矩阵结构示意图

由文中 2.2 节可知,基于域$GF(2)$构建的编码矩阵在编译码过程中,需保证各信息块$\boldsymbol{D}_i(1\leq i\leq k)$大小一致,同属于一个域,使得运算封闭;编码后得到的各校验块$\boldsymbol{P}_i(1\leq i\leq\delta+t)$也将与信息块大小保持一致.如图 4 所示,$\boldsymbol{\alpha}$为各信息块构成的信息向量,$\boldsymbol{\beta}$为编码矩阵$\boldsymbol{G}$与信息向量$\boldsymbol{\alpha}$运算后得到的编码向量.由于 RBEC 编码阵采取单位阵与随机阵结合方式,属于系统码;编码向量$\boldsymbol{\beta}$中信息部分与校验部分相互独立,可直接提取原始信息.

$$\begin{bmatrix}\boldsymbol{I}_{k\times k}\\ \boldsymbol{R}_{(\delta+t)\times k}\end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{D}_1\\\boldsymbol{D}_2\\\vdots\\\boldsymbol{D}_{k-1}\\\boldsymbol{D}_k\end{bmatrix}=\begin{bmatrix}\boldsymbol{D}_1\\\boldsymbol{D}_2\\\vdots\\\boldsymbol{D}_k\\\boldsymbol{P}_1\\\vdots\\\boldsymbol{P}_{\delta+t}\end{bmatrix}$$

$$\boldsymbol{G}_{(k+\delta+t)\times k}\cdot\boldsymbol{\alpha}_{k\times 1}=\boldsymbol{\beta}_{(k+\delta+t)\times 1}$$

图 4　RBEC 的编码过程

若编码向量$(\boldsymbol{D}_1,\boldsymbol{D}_2,\cdots,\boldsymbol{D}_k\mid\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{\delta+t})$中出现任意$t$个元素丢失,RBEC 能以高于$1-1/2^{\delta}$概率完全恢复原始信息数据$(\boldsymbol{D}_1,\boldsymbol{D}_2,\cdots,\boldsymbol{D}_k)$,文中 3.3 节给出了完整的数学证明.RBEC 的编码矩阵构造过程中,参数k、t取值没有严苛的限制.但是,δ值大小至关重要,它直接影响着 RBEC 的译码恢复概率;δ越大,恢复概率越高;例如当$\delta=20$时,恢复任意t位删除错的概率可以超过 0.999 999.

图 5 为 $k=20,\delta=10,t=10$ 时，RBEC 编码示例：图中编码矩阵 $\boldsymbol{G}$ 规模为 40×20，空白处值为 0，阴影部分值为 1；矩阵 $\boldsymbol{G}$ 上半部分为单位阵，下半部分矩阵随机生成，其中各元素按概率 0.5 独立选取 0 或 1.

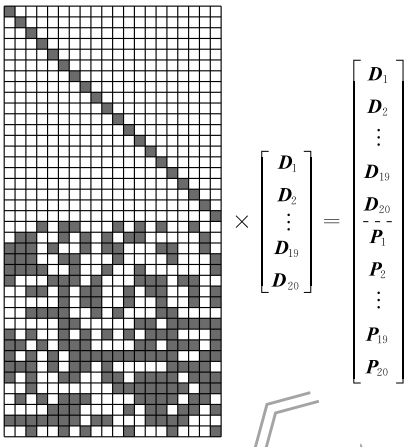


图 5 (40,20,10,10)RBEC 的编码过程

图 5 中编码向量校验部分($\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{20}$)是由信息部分($\boldsymbol{D}_1,\cdots,\boldsymbol{D}_{20}$)根据随机阵各行中元素值，进行异或操作得到. 例如，$\boldsymbol{P}_1$ 对应随机阵第 1 行，$\boldsymbol{P}_1$ 值为 $\boldsymbol{P}_1=\boldsymbol{D}_4\oplus\boldsymbol{D}_6\oplus\boldsymbol{D}_7\oplus\boldsymbol{D}_{11}\oplus\boldsymbol{D}_{13}\oplus\boldsymbol{D}_{15}\oplus\boldsymbol{D}_{18}$；$\boldsymbol{P}_{20}$ 对应第 20 行，$\boldsymbol{P}_{20}=\boldsymbol{D}_3\oplus\boldsymbol{D}_6\oplus\boldsymbol{D}_8\oplus\boldsymbol{D}_{12}\oplus\boldsymbol{D}_{16}\oplus\boldsymbol{D}_{20}$；其它行同理. (40,20,10,10)RBEC 能保证当编码向量($\boldsymbol{D}_1,\cdots,\boldsymbol{D}_{20}|\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{20}$)中出现任意 10 位损毁或丢失时，以高于 $1-1/2^{10}\approx0.999$ 概率将信息部分($\boldsymbol{D}_1,\cdots,\boldsymbol{D}_{20}$)完全恢复.

3.2 RBEC 译码方法

在进行译码操作时，(n,k,δ,t)RBEC 码的编码向量若出现任意 t 位删除错，可利用剩余 $k+\delta$ 位数据以及对应于编码矩阵 $\boldsymbol{G}_{(k+\delta+t)\times k}$ 中 $k+\delta$ 行，构造 $(k+\delta)\times k$ 大小的译码方程组，通过求解方程组将原始 k 位信息数据恢复. 若采用高斯消去法，运算复杂度为 $o(k^3)$；亦或者，利用译码矩阵 $\boldsymbol{H}_{(k+\delta+t)\times(\delta+t)}$ 与编码向量 $\boldsymbol{\beta}_{(k+\delta+t)\times1}$ 之间满足关系 $\boldsymbol{H}\cdot\boldsymbol{\beta}=0$，构造 $(\delta+t)\times t$ 大小的译码方程组，运算复杂度为 $o(t^3)$.

RBEC 为系统码，其编码向量中校验部分只起到冗余保护作用，且可独立重新构造. 本部分利用其只需保证恢复信息部分中缺失数据的特点，依然从译码矩阵出发，介绍一种最小化译码方程组的方法，使其运算复杂度只与剩余 $k+\delta$ 位数据中包含的校验位数目 p 有关，具体简化过程如图 6 所示.

(1) 假设剩余 $k+\delta$ 位中包含 p 个校验位、$k+\delta-p$ 个信息位元素；由于信息位至多为 k，所以 $p\geq\delta$. 恢复过程主要是将缺失的 $k-(k+\delta-p)=p-\delta$ 信息位元素恢复；

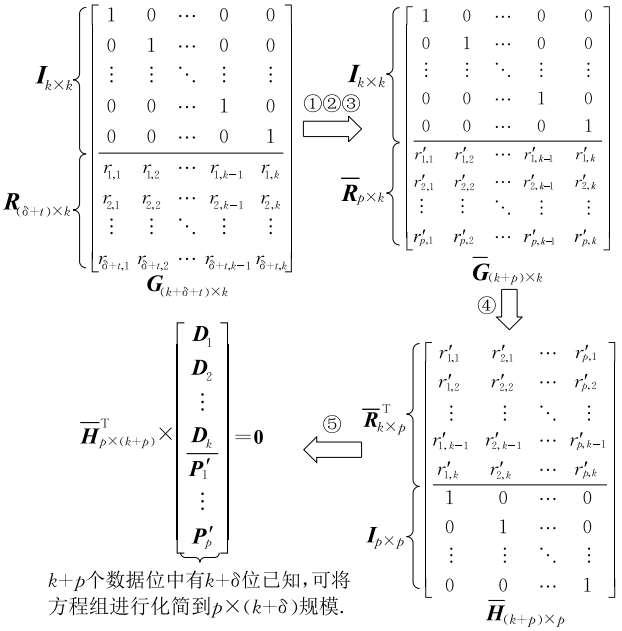


图 6 译码方程组简化过程

(2) 将 p 个校验位对应于编码矩阵 $\boldsymbol{G}$ 的 p 个行取出，可构造 $p\times k$ 矩阵 $\overline{\boldsymbol{R}}$；矩阵 $\overline{\boldsymbol{R}}$ 为随机阵 $\boldsymbol{R}$ 的子矩阵，依然保持随机阵的性质；

(3) 构造编码矩阵 $\overline{\boldsymbol{G}}_{(k+p)\times k}$，上半部分为单位阵 $\boldsymbol{I}_{k\times k}$，下半部分为随机阵 $\overline{\boldsymbol{R}}_{p\times k}$；

(4) 通过矩阵 $\overline{\boldsymbol{G}}_{(k+p)\times k}$，构造译码矩阵 $\overline{\boldsymbol{H}}_{(k+p)\times p}$，上半部分为矩阵 $\overline{\boldsymbol{R}}$ 的转置 $\overline{\boldsymbol{R}}^{\mathrm{T}}$，下半部分为单位阵 $\boldsymbol{I}_{p\times p}$；

(5) 利用译码矩阵 $\overline{\boldsymbol{H}}_{(k+p)\times p}$ 与剩余已知 $k+\delta$ 位信息进行数据恢复，可构造出规模为 $p\times(p-\delta)$ 的译码方程组，求解方程组将缺失的 $p-\delta$ 位信息数据恢复.

步骤(4)中，矩阵 $\overline{\boldsymbol{G}}_{(k+p)\times k}$ 相对矩阵 $\boldsymbol{G}_{(k+\delta+t)\times k}$ 缺少了随机阵 $\boldsymbol{R}$ 中的 $\delta+t-p$ 行；它们只起到冗余校验作用，将其删除并不会影响信息位的恢复过程，同时又简化了译码方程组.

上述过程使得最后译码方程组规模只与 $k+\delta$ 个已知数据中包含的校验位数目 p 有关，为 $p\times(p-\delta)$. 不论 k 与 t 之间大小关系，p 取值范围定满足 $\delta\leq p\leq\min\{\delta+t,\delta+k\}$，进而上述过程译码复杂度满足 $o((p-\delta)^3)\leq\min\{o(t^3),o(k^3)\}$.

假设 RBEC 码字上各元素被选取或丢失概率相等，任取 $k+\delta$ 位，其中包含校验位数目 p 的大小可近似为 $(k+\delta)\times\dfrac{\delta+t}{k+\delta+t}$. 此时，优化后的方程组译码复杂度 $o\left(\left(\dfrac{kt}{k+\delta+t}\right)^3\right)$ 与 $o(k^3)$ 相比，只有后者

的$\left(\frac{t}{k+\delta+t}\right)^3$;$\frac{t}{k+\delta+t}$值越小,优化效果越明显.

例如,在图 5 的(40,20,10,10)RBEC 例子中,假设编码向量($\boldsymbol{D}_1,\cdots,\boldsymbol{D}_{20}\,|\,\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{20}$)可读元素为$\boldsymbol{D}_6\sim\boldsymbol{D}_{20}$、$\boldsymbol{P}_1\sim\boldsymbol{P}_{15}$共 30 位,其中$\boldsymbol{D}_1\sim\boldsymbol{D}_5$信息位丢失.根据步骤(1)~(3),将图 5 中矩阵$\boldsymbol{G}_{40\times20}$中校验位$\boldsymbol{P}_1\sim\boldsymbol{P}_{15}$对应第 21 至第 35 行取出,构造图 7 中矩阵$\overline{\boldsymbol{G}}_{35\times20}$;再按步骤(4)构造译码矩阵$\overline{\boldsymbol{H}}_{35\times15}$;最后根据矩阵$\overline{\boldsymbol{H}}_{35\times15}$与向量$\boldsymbol{\alpha}_{35\times1}$满足关系$\overline{\boldsymbol{H}}^{\mathrm{T}}_{35\times15}\times\boldsymbol{\alpha}_{35\times1}=0$($\boldsymbol{\alpha}_{35\times1}=(\boldsymbol{X}_1,\cdots,\boldsymbol{X}_5,\boldsymbol{D}_6,\cdots,\boldsymbol{D}_{20}\,|\,\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{15})$,前 5 位未知,为待求解信息位),将$\boldsymbol{\alpha}_{35\times1}$中已知项移到等式右边,可将方程组化为 15×5 规模,如图 7 所示.

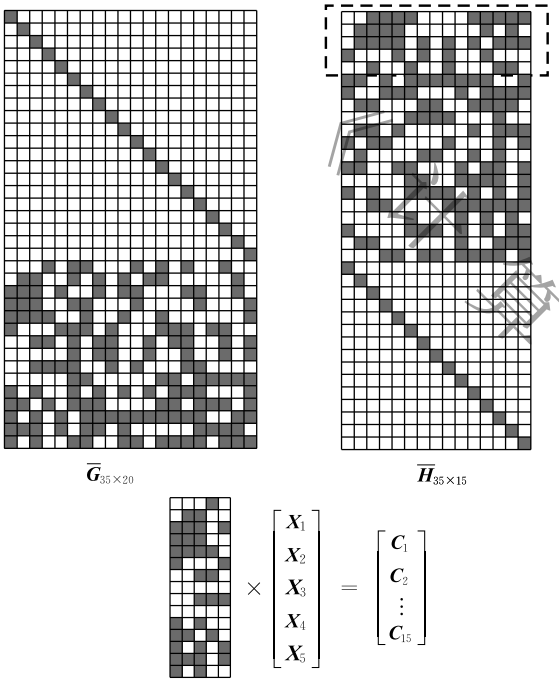


图 7　译码过程示例

3.3　RBEC 的译码恢复概率

由 3.2 节可知,(n,k,δ,t)RBEC 码能否成功恢复任意 t 删除错的关键是:编码矩阵 $\boldsymbol{G}$ 删除任意 t 行,余下 $k+\delta$ 行构成的矩阵是否满足列满秩性质.本节将给出 RBEC 编码矩阵中任取 $k+\delta$ 行满足列满秩性质概率的下界,并指出随 δ 值增加,列满秩概率下界将不断趋近 1(100%).由于随机矩阵是编码矩阵重要组成部分,随机阵满秩概率是本文方法的重要理论依据之一.本节将先引出随机矩阵满秩概率,然后通过定理 2 和推论 3 给出 RBEC 编码矩阵任意 $k+\delta$ 行的列满秩概率下界和理论证明.

定义 1.　随机矩阵$\boldsymbol{R}_{n\times k}$($n\geq k>0$)中各元素 r_{ij} ($0\leq i<n,0\leq j<k$)取值相互独立且满足概率分布:

$$P\{r_{ij}=m\}=\begin{cases}T, & \text{for } m=1\\ 1-T, & \text{for } m=0\end{cases},\ T\in(0,1).$$

引理 1.　$n\geq k>0$,$n\times k$ 列满秩矩阵的总个数C_{Total}满足以下关系:

$$C_{\mathrm{Total}}=\sum_{i=0}^{nk}C_i=\prod_{i=1}^{k}(2^n-2^{i-1})$$

C_i:包含 i 个 1 且满足列满秩的 $n\times k$ 矩阵个数.

证明.　由于"矩阵列满秩"等价于"矩阵中各列线性无关",按照规则"第 i 列不为前 $i-1$ 列线性组合,i 从 1 到 k"逐步迭代,可得到 $n\times k$ 列满秩矩阵的总个数$C_{\mathrm{Total}}=\prod_{i=1}^{k}(2^n-2^{i-1})$.

再根据"矩阵中包含 1 的个数"对C_{Total}个列满秩矩阵进行分类,将"包含 i 个 1 的列满秩矩阵"归为一类,包含的元素个数记为 C_i,易得

$$C_{\mathrm{Total}}=\sum_{i=0}^{nk}C_i=\prod_{i=1}^{k}(2^n-2^{i-1}).\qquad\text{证毕.}$$

定理 1.　$n\geq k>0$,随机矩阵$\boldsymbol{R}_{n\times k}$列满秩概率为

$$P_{rank(\boldsymbol{R})=k}=\sum_{i=0}^{nk}Q_i\cdot C_i \tag{1}$$

$$(Q_i=T^i(1-T)^{nk-i},\ T\in(0,1);$$

$$\sum_{i=0}^{nk}C_i=\prod_{i=1}^{k}(2^n-2^{i-1}))$$

C_i:包含 i 个 1 且满足列满秩的 $n\times k$ 矩阵个数;Q_i:随机构造矩阵$\boldsymbol{R}_{n\times k}$,并满足在指定 i 个位置为 1 其余全为 0 的概率.

证明.　给定矩阵$\boldsymbol{M}_{n\times k}$,其矩阵中元素只有 i 个 1,其余为 0;可推出,随机矩阵$\boldsymbol{R}_{n\times k}$等于$\boldsymbol{M}_{n\times k}$的概率为$Q_i=T^i(1-T)^{nk-i}$.此外,"随机矩阵$\boldsymbol{R}_{n\times k}$列满秩概率"可等价于"$\boldsymbol{R}_{n\times k}$等于各个列满秩矩阵的概率总和".由引理 1 可知,$n\times k$ 列满秩矩阵的总个数为$\prod_{i=1}^{k}(2^n-2^{i-1})$,将包含 i 个 1 的满秩矩阵归为一类,总个数记为 C_i,i 从 0 取到 nk;进而,随机矩阵$\boldsymbol{R}_{n\times k}$列满秩概率为

$$P_{rank(\boldsymbol{R})=k}=\sum_{i=0}^{nk}Q_i\cdot C_i(Q_i=T^i(1-T)^{nk-i}).$$

证毕.

推论 1.　当 $T=0.5$,$n=k+\delta$,$k>0$,$\delta\geq0$ 时,随机矩阵$\boldsymbol{R}_{n\times k}$列满秩概率为

$$P_{rank(\boldsymbol{R})=k}=\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right) \tag{2}$$

证明.　由定理 1 可知,当 $T\in(0,1)$时,随机矩阵$\boldsymbol{R}_{n\times k}$列满秩概率:$P_{rank(\boldsymbol{R})=k}=\sum_{i=0}^{nk}Q_i\cdot C_i$.当 $T=$

0.5 时，$Q_i=T^i(1-T)^{nk-i}=\left(\frac{1}{2}\right)^i\left(1-\frac{1}{2}\right)^{nk-i}=\left(\frac{1}{2}\right)^{nk}$. 同时，$\sum_{i=0}^{nk}C_i=\prod_{i=1}^{k}(2^n-2^{i-1})$ 且 $n=k+\delta$，由式(1)可知

$$P_{rank(\boldsymbol{R})=k}=\sum_{i=0}^{nk}Q_i\cdot C_i=\left(\frac{1}{2}\right)^{nk}\sum_{i=0}^{nk}C_i$$
$$=\left(\frac{1}{2}\right)^{nk}\prod_{i=1}^{k}(2^n-2^{i-1})$$
$$=\prod_{i=1}^{k}\left(1-\frac{1}{2^{n-(i-1)}}\right)=\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right).$$

证毕.

推论 2. $P_{rank(\boldsymbol{R})=k}=\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)>1-\frac{1}{2^{\delta}}$ (3)

证明. 欲证 $\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}$，只需证 $\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+k}}$.

① 当 $k=1$，$\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)=1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+k}}=1-\frac{1}{2^{\delta+1}}$；两者相等，不等式成立；

② 假设当 $k=N-1$，不等式 $\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+k}}$ 成立；

③ 当 $k=N$ 时：

$$\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)=\left[\prod_{i=1}^{N-1}\left(1-\frac{1}{2^{\delta+i}}\right)\right]\left(1-\frac{1}{2^{\delta+N}}\right)$$
$$1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+k}}=1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+N}}=\frac{2^{\delta+N}-2^N+1}{2^{\delta+N}}$$
$$=\left(\frac{2^{\delta+N}-2^N+2}{2^{\delta+N}}\right)\left(\frac{2^{\delta+N}-2^N+1}{2^{\delta+N}-2^N+2}\right)$$
$$=\left(1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+N-1}}\right)\left(1-\frac{1}{2^{\delta+N}-2^N+2}\right)$$

由于 $N>1$，可得 $1-\frac{1}{2^{\delta+N}}>1-\frac{1}{2^{\delta+N}-2^N+2}$；

同时根据假设 $k=N-1$ 时，不等式 $\prod_{i=1}^{N-1}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+N-1}}$ 成立，可以推导出 $\prod_{i=1}^{N}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+N}}$；当 $k=N$ 时，不等式成立. 由①②③知，对于 k 取任何正整数，不等式 $\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)\geq 1-\frac{1}{2^{\delta}}+\frac{1}{2^{\delta+k}}$ 均成立；进而 $\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)>1-\frac{1}{2^{\delta}}$ 成立. 证毕.

定理 2. $T=0.5$，构造 $(k+\delta+t)\times k$ 矩阵 $\boldsymbol{G}_{(k+\delta+t)\times k}$（$k>0$，$\delta>0$，$t>0$），且满足形式 $\left[\frac{\boldsymbol{I}_{k\times k}}{\boldsymbol{R}_{(\delta+t)\times k}}\right]$，$\boldsymbol{I}_{k\times k}$ 为单位阵，$\boldsymbol{R}_{(\delta+t)\times k}$ 为随机矩阵. 从 $\boldsymbol{G}_{(k+\delta+t)\times k}$ 中任取 $k+\delta$ 行构造矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$，$\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率满足 $P_{rank(\boldsymbol{M})=k}>\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$.

证明. 矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 列满秩概率可按“$k+\delta$ 行中包含单位阵中的行数”进行分类讨论；设单位阵各行为 $\boldsymbol{e}_i$（$0\leq i<k$）：

① 若 $k+\delta$ 行取至底部矩阵 $\boldsymbol{R}_{(\delta+t)\times k}$，此时 $\boldsymbol{M}_{(k+\delta)\times k}$ 等价于随机构造，列满秩概率：

$$P^0_{rank(\boldsymbol{M})=k}=\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$$

② 若 $k+\delta$ 行有 1 行取至单位阵 $\boldsymbol{I}_{k\times k}$，其余 $k+\delta-1$ 行取至矩阵 $\boldsymbol{R}_{(\delta+t)\times k}$. 设矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 取得的单位阵行为 $\boldsymbol{e}_i$（$0\leq i<k$），行 $\boldsymbol{e}_i$ 只有第 i 个元素值为 1，其余全为 0；进而，矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 第 i 列定不能由其它列线性表示，只需保证余下 $k-1$ 列线性无关；并且余下 $k-1$ 列在 $\boldsymbol{e}_i$ 所处行中各元素均为 0，也无需考虑. 最后，$\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率等价于余下矩阵 $\overline{\boldsymbol{M}}_{(k+\delta-1)\times(k-1)}$ 满秩概率，同时矩阵 $\overline{\boldsymbol{M}}$ 为随机矩阵 $\boldsymbol{R}$ 的子矩阵，可得

$$P^1_{rank(\boldsymbol{M})=k}=\prod_{i=1}^{k-1}\left(1-\frac{1}{2^{\delta+i}}\right)$$

如图 8 所示，从矩阵 $\boldsymbol{G}$ 中提取 $k+\delta$ 行构造矩阵 $\boldsymbol{M}$，矩阵 $\boldsymbol{M}$ 第 1 行为 $\boldsymbol{G}$ 中单位阵的第 2 行. 矩阵 $\boldsymbol{M}$ 第 2 列元素无法由其它 $k-1$ 列线性表示，为了使矩阵 $\boldsymbol{M}$ 满秩，只需保证剩余 $k-1$ 列线性无关；除去第

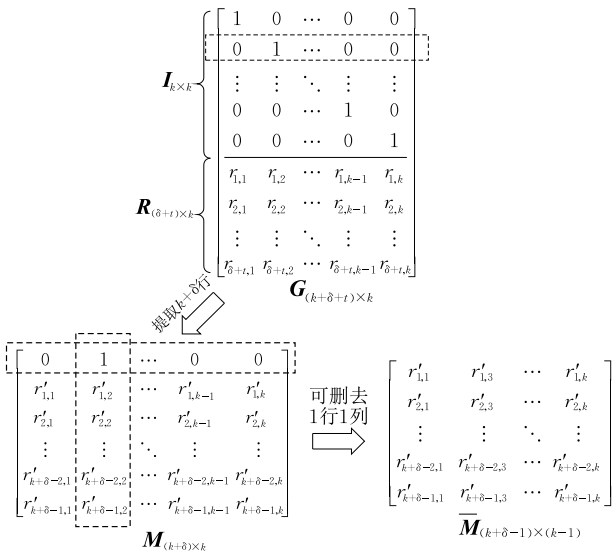


图 8 矩阵 $\boldsymbol{M}$ 提取 $\boldsymbol{G}$ 中单位阵的第 2 行

2 列,矩阵 $\boldsymbol{M}$ 在第 1 行元素全为 0. 最后,只需考虑 $(k+\delta-1)(k-1)$ 矩阵 $\overline{\boldsymbol{M}}$ 的满秩概率.

③若 $k+\delta$ 行有 2 行取至单位阵 $\boldsymbol{I}_{k\times k}$,同理②;矩阵 $\boldsymbol{M}$ 将有 2 行与 2 列不需要讨论;最后,只需求 $(k+\delta-2)\times(k-2)$ 随机矩阵满秩概率:

$$P^{2}_{rank(\boldsymbol{M})=k}=\prod_{i=1}^{k-2}\left(1-\frac{1}{2^{\delta+i}}\right)$$

以此类推.

④若 $k+\delta$ 行有 $k-1$ 行取至单位阵 $\boldsymbol{I}_{k\times k}$:

$$P^{k-1}_{rank(\boldsymbol{M})=k}=\prod_{i=1}^{1}\left(1-\frac{1}{2^{\delta+i}}\right)$$

⑤若 $k+\delta$ 行有 k 行取至单位阵 $\boldsymbol{I}_{k\times k}$:

$$P^{k}_{rank(\boldsymbol{M})=k}=1$$

随着矩阵 $\boldsymbol{M}$ 包含的单位阵中行数目逐渐增多,公式 $P^{i}_{rank(\boldsymbol{M})=k}(0\leq i\leq k)$ 中项数逐渐减少,并且缺失项的通式为 $1-\frac{1}{2^{\delta+i}}(i\geq 1)$,均小于 1;容易推出:

$$P^{k}_{rank(\boldsymbol{M})=k}>P^{k-1}_{rank(\boldsymbol{M})=k}>\cdots>P^{1}_{rank(\boldsymbol{M})=k}>P^{0}_{rank(\boldsymbol{M})=k}$$

上述各分类下的矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率均大于或等于随机构造 $(k+\delta)\times k$ 矩阵的满秩概率,即 $P^{i}_{rank(\boldsymbol{M})=k}\geq P^{0}_{rank(\boldsymbol{M})=k}(0\leq i\leq k)$. 进而,矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率 $P_{rank(\boldsymbol{M})=k}>P^{0}_{rank(\boldsymbol{M})=k}=\prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$.

证毕.

推论 3.　$T=0.5$,构造 $(k+\delta+t)\times k$ 矩阵 $\boldsymbol{G}_{(k+\delta+t)\times k}(k>0,\delta>0,t>0)$,且满足形式 $\left[\frac{\boldsymbol{I}_{k\times k}}{\boldsymbol{R}_{(\delta+t)\times k}}\right]$,$\boldsymbol{I}_{k\times k}$ 为单位阵,$\boldsymbol{R}_{(\delta+t)\times k}$ 为随机矩阵. 从 $\boldsymbol{G}_{(k+\delta+t)\times k}$ 中任取 $k+\delta$ 行构造矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$,$\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率满足 $P_{rank(\boldsymbol{M})=k}>1-\frac{1}{2^{\delta}}$.

证明.　由推论 2 与定理 2 容易推导.　证毕.

从推论 3 可以得出,若 $T=0.5$,从形式为 $\left[\frac{\boldsymbol{I}_{k\times k}}{\boldsymbol{R}_{(\delta+t)\times k}}\right]$ 的矩阵 $\boldsymbol{G}$ 中任意选取 $k+\delta$ 行或删去 t 行,构造出的矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率,存在一个由 δ 决定,与 k 无关的概率下界;随着 δ 取值不断增大,矩阵 $\boldsymbol{M}_{(k+\delta)\times k}$ 满秩概率将不断趋近 1(100%). RBEC 的编码矩阵结构正采用单位阵与随机阵结合方式,其译码恢复概率符合定理 2 与推论 3 的结论. 我们大量的实验也表明,当 $\delta\geq 20$,矩阵列不满秩概率已经降到百万分之一以下,非常接近理论公式 $1-1/2^{\delta}$ 所给出的估值.

3.4　RBEC 参数动态调整能力

当前主流纠删码技术主要从宏观整体出发,构造出性能优异的码字,其编码矩阵或译码矩阵内部各元素联系紧密,耦合度强,如 RS 码、LDPC 等. 当编码参数发生变化,编码方法无法直接在原有的编码矩阵或译码矩阵上通过行列调整进行相应变换;通常只有摒弃原有码字,再根据新的参数构造新的码字,过程繁琐复杂,代价高.

与传统编码方法不同,RBEC 编码矩阵结构为单位阵与随机阵相结合,采用自底向上的设计模式,通过控制随机阵 $\boldsymbol{R}$ 中各元素独立生成,获得整体上码字高性能. RBEC 编码矩阵特殊的设计模式,使得其参数具有动态调整能力,主要体现为:可动态对矩阵 $\boldsymbol{G}$ 的行列进行增减,可随时对参数 k,t,δ 进行调整,并且调整过程不需要重新设计码字,直接在原编码矩阵基础上完成. 下面将详细描述 k,δ,t 的调整过程.

(1)若将 t 增大为 $t+c(c>0)$,需向编码矩阵 $\boldsymbol{G}$ 中随机阵 $\boldsymbol{R}$ 尾部增加 c 行,并且增添的 c 行中各元素均按概率 0.5 独立选取'0'或'1',保持矩阵 $\boldsymbol{R}$ 随机一致性,操作完成;如图 9.

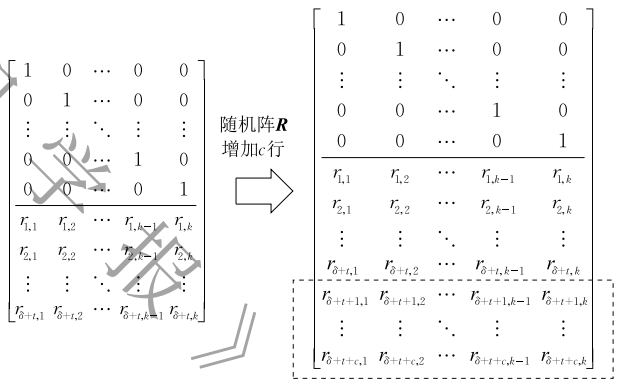


图 9　RBEC 参数 t 扩展 c 位示意图

若将 t 减小为 $t-c(0<c<t)$,只需从随机阵 $\boldsymbol{R}$ 部分选择 c 行删除,操作完成.

(2)参数 δ 调整过程同参数 t.

(3)若将 k 增大为 $k+c(c>0)$,需要对单位阵 $\boldsymbol{I}$ 和随机阵 $\boldsymbol{R}$ 进行调整. 单位阵 $\boldsymbol{I}$ 底部和右端增加 c 行和 c 列,将单位阵规模从 $k\times k$ 扩展为 $(k+c)\times(k+c)$;随机阵 $\boldsymbol{R}$ 右端增加 c 列,并且增添的 c 列中各元素均按概率 0.5 独立选取'0'或'1',保持矩阵 $\boldsymbol{R}$ 随机的一致性,操作完成;如图 10.

若将 k 减小为 $k-c(0<c<k)$,需从编码矩阵 $\boldsymbol{G}$ 中选择 c 列删除,然后消去单位阵中全零行,操作完成.

按照上述过程,不论增添或者删减矩阵行列,调整后的编码矩阵依然符合定理 2 中矩阵模式,保持

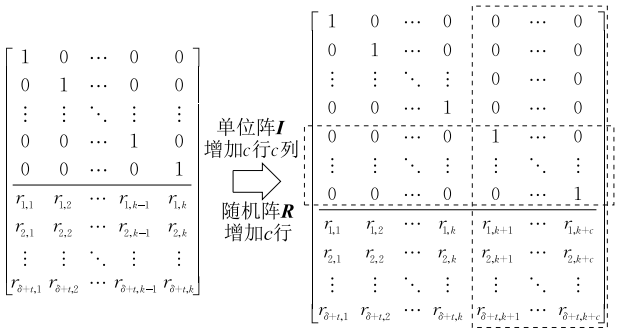


图 10 RBEC 参数 k 扩展 c 位示意图

高概率恢复的特性. RBEC 的特殊编码矩阵结构,使得 RBEC 在参数 k、δ、t 动态调整方面具有天然的优势,并且调整后的编码矩阵依然保持系统码结构. 在编码设计初期,RBEC 的纠删能力与码率的设定范围更广,可在后期根据实际应用环境再做准确的调整.

由 RBEC 的参数动态调整能力不难得到,给定任意大小的 k 或 t、码率 $k/(k+\delta+t)$ 或纠错率 $t/(k+\delta+t)$,RBEC 均能给出确定编码结构. 即便 k 与 t 取值趋于无穷,仍能构造出码率与纠错率保持常数的码字,拥有近似香农好码性质.

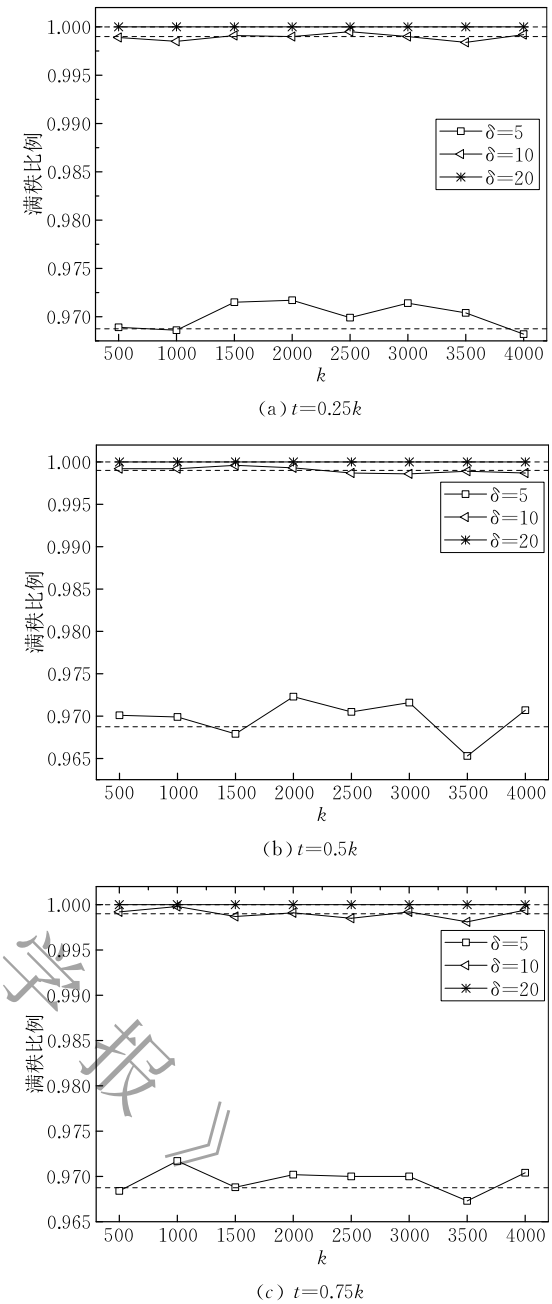
4 性能分析

本节实验主要在配置为 Intel Core i3 CPU 3.07 GHz/3.06 GHz 和 4 GB RAM 的 PC 机上进行. 首先为了验证推论 3,RBEC 编码矩阵任意 $k+\delta$ 行的列满秩概率下界只与 δ 有关,实验分析了参数 k,δ,t 对满秩概率的影响;然后针对 RBEC 的存储效率、纠错率、编译码效率和重构带宽进行了分析,并与 RS 码、CRS 码等编码进行比较.

4.1 RBEC 的满秩概率实验分析

虽然在 3.3 节中对 RBEC 容任意 t 删除错的译码成功概率下界给出了理论上的证明,为了更加直观说明 RBEC 编码矩阵中任意 $k+\delta$ 行构造的矩阵满秩概率下界只与 δ 有关,做了如下实验:我们对 k,δ,t 选取了不同组合,针对每种组合,分别构造 RBEC 编码矩阵 $\boldsymbol{G}$,并从中随机选取 $k+\delta$ 行 10 000 次,记录满秩的次数所占比例. 实验过程中,t 取值 $0.25k$,$0.5k$,$0.75k$;δ 分别取 5,10,20;针对每组 t,δ,k 值以 500 为间隔,从 500 取到 4000;实验结果如图 11 所示.

图 11 中(a)、(b)、(c)分别为 $t=0.25k$,$t=0.5k$,$t=0.75k$,k,δ 取不同值时,实验结果. 从图 11



(a) $t=0.25k$

(b) $t=0.5k$

(c) $t=0.75k$

图 11 k,δ,t 对满秩概率的影响

(a)、(b)或(c)中均可观察到,当 δ,t 确定,参数 k 的变化并未对满秩比例有显著的影响,即无论 k 如何变化,满秩的比例始终处于一个稳定的范围;对比图 11(a)、(b)和(c)可以得出,当 k,δ 确定,不同的 t 值下满秩比例分布依然保持一致,即参数 t 的变化也没有明显的效果;但是当确定 k,t 时,不同的 δ 值,满秩比例变化明显,随着 δ 值不断增大,满秩比例不断增大,并且与推论 3 中给出的 $1-1/2^{\delta}$ 理论下界相符合,如表 1 所示.

通过上述实验进一步可以确定 RBEC 容任意 t 删除错的译码成功概率下界由 δ 决定,δ 值可根据

表 1　δ 取若干个特殊值时,满秩概率下界

δ	$1-1/2^{\delta}$
1	0.5
5	0.96875
10	0.9990
20	0.9999990
30	0.9999999990

具体的应用环境动态设定,k,t 值变化造成的影响可以忽略.我们的理论推导及实验都表明 δ 通常达到 20～30 时,就足以保障"高概率".因此 δ 相对于参数 k 取值较大时(数千、数万),几乎是微不足道的,亦即非常接近 MDS 性质(码率达到最高),这是其他现有二元域上的存储编码所没有的特性.

4.2　RBEC 的存储效率和纠错率

在存储系统中,纠删码的码率也可称为存储效率,即编码向量中信息位个数 k 占整体码长 n 的比例 k/n;纠错率指纠删码的容删除错能力与整体码长的比例,若编码向量中任意 t 位同时丢失时,纠删码均可恢复,其纠错率为 t/n.

对于信息位个数为 k,并能保证容任意 t 位删除错的编码方案,最佳的存储效率为 $k/(k+t)$,称该编码具有 MDS 性质,例如 RS 码.相比之下,(n,k,δ,t)-RBEC 编码存储效率 $k/(k+\delta+t)$ 并未达到理论最优,但是 RS 码的 MDS 性质是以在有限域 $GF(2^m)$ 上运算为代价,乘法复杂度高,并随着 n 变大,有限域也将增大;而 RBEC 一直保持在 $GF(2)$ 上进行异或运算;如图 12 所示.

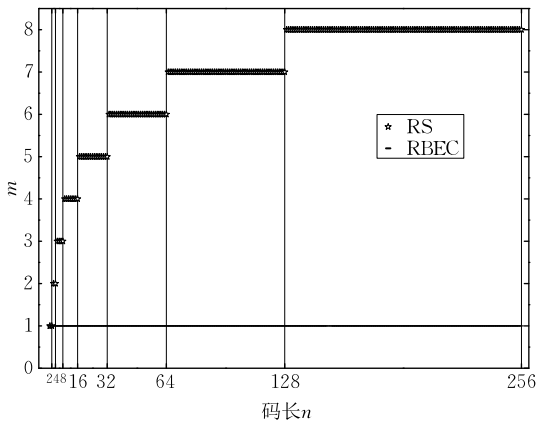


图 12　RS 码与 RBEC 码运算域随码长 n 的变化趋势

RS 在实际系统应用中,如存储系统,为了保证运行效率,通常有限域 $GF(2^m)$ 取值为 $GF(2^8)$ 或 $GF(2^{16})$,造成码字长度 n 受到限制($n\leq 2^m$).然而,RBEC 的参数 δ 通常可设定为很小的常数量,如 $\delta=20$;同时 k,t 取值自由,码长 n 设置并无限制.随着 k,t 不断增大,RBEC 存储效率 $k/(k+\delta+t)$ 与纠错率 $t/(k+\delta+t)$,将十分接近 $k/(k+t)$ 与 $t/(k+t)$,具有近似 MDS 性质.

4.3　RBEC 的编译码效率

编译码效率是衡量纠删码性能的重要指标.对此,本节将从编码过程中生成单个冗余位需要的异或次数度量编码效率,以及恢复单个信息位需要的异或次数度量译码效率.由于 RS 码的编译码过程在高阶有限域上进行,为了使得运算域等价,将 RBEC 编码与 CRS 码对比.CRS 码是一类利用柯西矩阵生成的 RS 码,也为系统码,并且其有限域元素可利用'0、1'方阵表示,使得编译码过程只存在异或操作(如图 1 的转化示例).

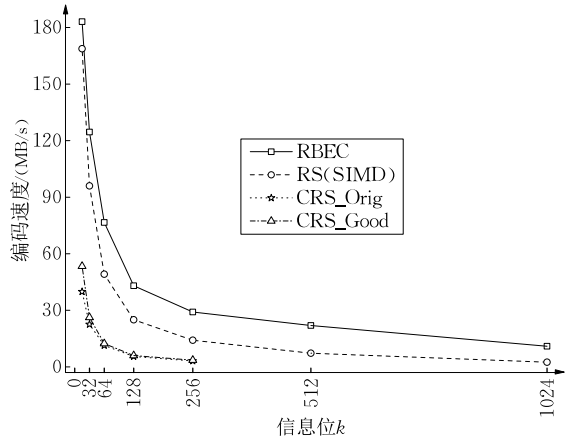
将 RBEC 码和 CRS 码信息位 k 与容错能力 t 设置同等情况下,RBEC 码的编码矩阵 $\boldsymbol{G}$ 规模为 $(k+\delta+t)\times k$,但是 CRS 码的编码矩阵在'0、1'表示下规模将为 $(k+t)m\times km$,$k+t\leq 2^m$.由于 RBEC 码的编码矩阵中随机阵部分,各元素按概率 0.5 独立选取 0 或 1,随机阵部分每行中 1 的个数大致占整行的 1/2,生成每个冗余位大致需要 $(1/2)k-1$ 次异或运算;相对而言,CRS 码的编码矩阵校验部分'1'的个数普遍多于'0'的个数,即便按照文献[11,26]中优化方法得到的 CRS 码编码矩阵,其生成单个冗余位的异或运算次数优化效果也十分有限,仍处于 RBEC 码的 m^2 量级.与编码效率类似,在恢复相同数量删除错时,CRS 码译码效果依然低于 RBEC 码.

图 13 为存取 1GB 文件时,RBEC 码与 RS 码、CRS 码关于编译码速度对比的实验结果.其中,RS 码类在域 $GF(2^{16})$ 上运算;RBEC 的参数 δ 设定为 20,其编译码过程是基于 Plank 发布的 Erasure Coding 算法开源库 jerasure2.0[①] 中提供的接口上实现的.图中 RS(SIMD)指应用 SIMD(Single Instruction Multiple Data)指令集加速了有限域运算的 RS 码[13];CRS_Orig 为原始柯西矩阵,CRS_Good 为经过优化的柯西矩阵[26].

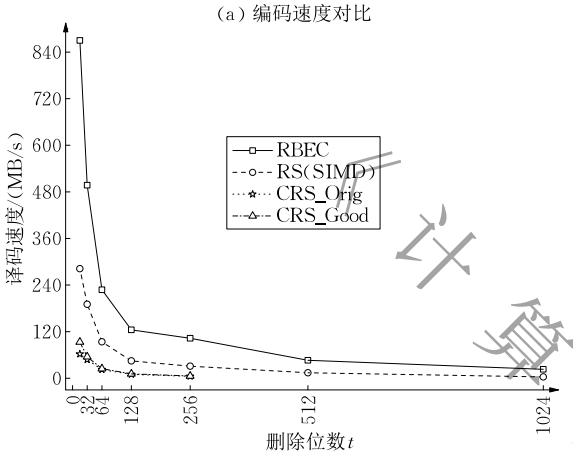
实验中各编码方法编译码过程均未采用任何优化算法,其中译码方法均使用高斯消去法求解丢失数据;并设定码率 0.5,通过不断增加 k 值与 t 值 $(k,t=16,32,64,128,256,512,1024)$,记录随着矩阵规模不断扩大,各编码方法的速度变化.

从图 13(a)中不难观测出,当各编码方法的 k

① http://www.kaymgee.com/Kevin_Greenan/Software.html
http://web.eecs.utk.edu/～plank/plank/www/software.html



(a) 编码速度对比



(b) 译码速度对比

图 13 各编码方法编译码速度对比结果

值相同时,不论 k 值大小,RBEC 的编码速度始终高于 CRS 码;即便对于应用 SIMD 指令集加速了有限域计算的 RS 码,其编码速度也不及 RBEC 码. 图 13(b)是在文件存储后,随机删除 t 份数据,再恢复原始文件时,记录的译码速度;与编码速度类似,当删除位数相同时,RBEC 的译码速度优于其它编码方法.

4.4 RBEC 的重构带宽

重构带宽是纠删码技术在存储系统应用时需考虑的一项重要指标,本节将利用重构带宽比进行度量:即恢复缺失数据所需的数据量与已丢失数据量的比例. 重构带宽与存储效率、纠错率等性能有密切的联系.

当前存储系统冗余容灾技术,重构带宽最优方法当属复制备份策略,它对于任何损毁数据只需利用等量的备份数据进行还原即可,重构带宽比例为 1. 但是复制备份容灾能力十分受限,若系统要求可容任意 t 节点错误,需再将原数据复制 t 份,存储效率低.

虽然纠删码技术可取得低冗余高可靠效果,但是也牺牲了部分重构带宽性能,是纠删码应用过程中需要面对的共性问题. 以文中提出 (n,k,δ,t) RBEC 码和 (n,k) RS 码为例 $(t\leq k)$,当存储系统出现 $d(0<d\leq t)$ 个节点丢失,利用高斯消去方式译码恢复时,RBEC 码与 RS 码分别需要 $k+\delta$ 个节点和 k 个节点. 此时,重构带宽比分别为 $(k+\delta)/d$ 和 k/d,若 d 值较小,如 $d=1$ 时的单点错误修复,RBEC 码、RS 码的重构带宽与复制相比将造成数倍消耗. 但是,当 d 值较大时,如 $d=t$ 时,RBEC 码与 RS 码的重构带宽比趋近 1,将与复制方式相差不大,甚至相等. 进而,随着出错节点数 d 的变大,纠删码重构带宽性能越高.

图 14 展示了备份方法、(180,80,20,80)RBEC 码以及(160,80)RS 码的重构带宽比随出错个数 d 变化的趋势. 图中设定参数 RBEC 码与 RS 码最高均能容忍任意 80 位的删除错. 随着 d 增大,RBEC 码与 RS 码的重构带宽比逐步减小,并趋近于 1. 当 $d=80$ 时,RBEC 码与 RS 码的重构带宽比将分别为 1.25 和 1.

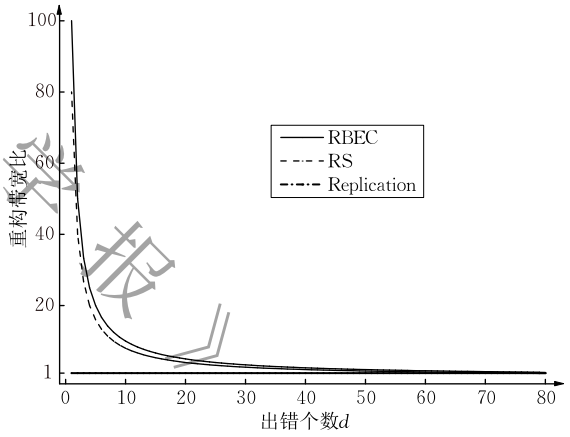


图 14 重构带宽比随出错个数 d 变化趋势图

RBEC 码与 RS 码相比,由于重构过程中会多取 δ 份数据,当恢复损毁节点个数 d 相同时,RBEC 码的重构开销始终高于 RS 码,$(k+\delta)/d>k/d$. 但是 RBEC 码的 δ 值为较小常数量,最大容删除错 t 取值并无限制. t 取值较大时,δ 额外开销影响将可忽略不计,即 $\delta/t(d=t)$ 近似 0;此时,RBEC 码重构开销与 RS 码近似.

对于纠删码技术在存储系统应用过程中碰到恢复单个节点或少数节点时需面对的高重构带宽问题,可利用 XOR Scheduling[27,28] 等专门配套技术途径取得一定程度优化.

4.5 RBEC 与其它概率型编码的比较

应用于存储系统的概率型编码方法主要包

括 LT码、LDPC 码、随机线性码(Random Linear Code,RLC). RLC 码为一类编码矩阵完全随机构造的纠删码方法,文献[25]给出了 RLC 码应用于存储系统的性能分析. RLC 码与 RBEC 码主要区别是 RBEC 的编码矩阵为单位阵和随机矩阵结合形式,RLC 码的编码矩阵为完全随机矩阵. 编码矩阵结构上的不同,使得 RBEC 码为系统码,且编码矩阵能以系统码结构自由扩展,而 RLC 码不具备以上优异的特性. LT 码与 LDPC 码受到译码方法的影响,若要拥有稳定的性能,如果保证译码成功概率,它们的码字长度普遍较长. 同时,LT、LDPC 等码类的设计参数理论极为复杂,扩展能力差,推广开来仍有不少障碍.

由于概率型编码方案不具备 MDS 性质,在译码恢复过程中,需要 fk 份数据才能以接近 100% 概率恢复原始 k 份数据,其中 $f>1$,称为开销因子(Overhead Factor). (n,k,δ,t)RBEC 在恢复过程中需提取 $k+\delta$ 份数据以超过概率 $1-1/2^{\delta}$ 恢复 k 份数据;此时 $f=1+\delta/k$,其中 δ 取值独立且只需为数十大小的常量,k 值为数百量级时,f 已十分接近 1. 相比其它概率型编码,文献[18]中指出的 LT 需提取 $1.05k$ 数据量才能接近 100%恢复 k 份数据,并且要求 k 取值为数万以上量级,短码需要的冗余更多,限制苛刻;文献[20]中指出 LDPC 恢复过程中,所需的开销因子 f 不仅与码率(信息位 k 占码字长度比例)相关,而且 $k\approx10\,000$ 时,f 取值也在 1.05 左右,当 k 值较小时,f 取值将超过 1.10.

表 2 中详细列举了 RBEC 编码与其它常用纠删码的性能比较,其中扩展能力指编码矩阵行列是否可以随意增加或减少;开销因子是在 $k\approx10\,000$ 时得到的结果;码字类型中短码与长码主要由 k 取值范围决定,短码 k 值在数十、数百数量级,长码 k 值则将达到数千、数万数量级.

表 2　RBEC 与其它编码性能对比

	运算域	系统码	扩展能力	恢复类型	开销因子	码字类型
RBEC	$GF(2)$	是	行列伸缩	概率型	近似 1	短码长码
LT	$GF(2)$	否	行扩展	概率型	1.05左右	长码
LDPC	$GF(2)$	否	无	概率型	1.05左右	短码长码
RLC	$GF(2)$	否	无	概率型	近似 1	短码长码
CRS	$GF(2^m)$	是	受有限域限制	确定型	1	短码

5　RBEC 应用于分布式存储系统

分布式存储系统通常由成千甚至上万台存储设备(节点)构成,以实现海量数据存储,并保证数据的高可靠性. 然而,现有可利用的成熟纠删码类几乎都来自通信领域,直接应用于大规模数据存储领域仍然存在不少弊端和未解问题. 本文提出的数据存储编码方案 RBEC 为分布式存储系统的纠删码容灾方法技术途径提供了一种新的选择.

当前,将编码技术应用于分布式存储系统主要流程包括:先将待存储文件划分为数据块,再编码生成冗余块,最后将数据块和冗余块分别存储在系统中各节点. 但是当系统若要调整参数,如码率、纠错率,需将各数据块回收并恢复原文件,再根据新参数下的编码再做一次存储流程,过程异常复杂. 本文将从存储节点层面上应用 RBEC 建立编码关系,进而使存储系统具有高容灾,节点数目可动态调整等特性.

RBEC 为 $GF(2)$ 上的系统码,应用在分布式存储系统时,信息节点和校验节点相互独立,并且编译码过程只存在存储节点之间的异或运算,因而执行效率高. 当信息节点未损坏时,能直接从信息节点中读取-传输数据而无需解码,如图 15 所示. RBEC 的参数 n,k,δ,t 在存储系统中分别对应着特定的实际含义:n 为存储系统的总节点数,包括信息节点(Data Node)和校验节点(Parity Node);k 为存储系统中信息节点的个数;冗余指数 δ 控制着恢复所有丢失节点数据的成功概率(δ 越大成功概率越高);t 为存储系统最大容删能力,即系统至多容 t 个节点(不论其所处的位置地点)同时损毁或数据丢失;$\delta+t$ 为存储系统中校验节点的个数. 其中存储效率 k/n 和纠错率 t/n 为存储系统的两个重要指标.

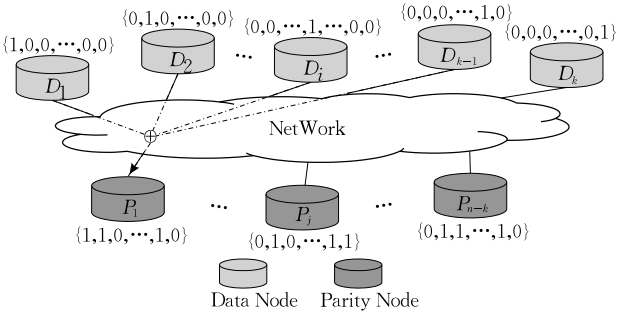


图 15　RBEC 应用于分布存储系统的示意图

5.1　构造基于 RBEC 的存储系统

根据存储系统中节点数目和实际应用的需求,确定 RBEC 的各项参数 n,k,δ,t,依照确定的参数值按如下步骤建立系统.

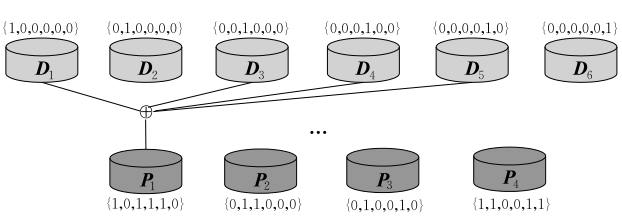
首先,系统中各存储节点均独自生成并保留一

个 $1\times k$ 的节点向量$\{a_1,a_2,\cdots,a_k\}$. 其中,信息节点根据编号 D_i 得到一个单位向量,向量中各元素取值为 $a_{m=i}=1,a_{m\neq i}=0,1\leq m\leq k$;校验节点的向量随机生成,且各元素 a_m 按概率 0.5 取值 0 或 1.

然后,校验节点根据自身节点向量与相应信息节点建立编码联系. 若生成的节点向量中值为 1 的元素分别在第$\{m_1,m_2,\cdots,m_{k'}\}$位,校验节点数据为对应编号的各信息节点异或和.

下面将以参数 $n=10,k=6,\delta=1,t=3$ 的 RBEC 为例介绍本文提出的编码方案在分布式存储系统中应用过程.

图 16(a)为以参数(10,6,1,3)RBEC 搭建的存储系统示意图,共包含 10 个存储节点;其中有 6 个信息节点,4 个校验节点,并能保证以高于 $1-1/2^1$ 概率恢复至多 3 个节点同时出错. 系统中各存储节点均拥有 1×6 的节点向量,$D_1\sim D_6$ 信息节点的向量是根据编号得到的单位向量,$P_1\sim P_4$ 校验节点的向量随机生成. 如图中 P_1 的节点向量为{1,0,1,1,1,0},P_1 数据为信息节点 D_1,D_3,D_4,D_5 异或和,即 $P_1=D_1\oplus D_3\oplus D_4\oplus D_5$. (10,6,1,3)RBEC 编码矩阵 $\boldsymbol{G}_{10\times6}$ 由各存储节点的向量组合构成,如图 16(b)所示.



(a)(10,6,1,3)RBEC下存储系统节点排布示意图

$$\begin{array}{c} \\ D_1\\D_2\\D_3\\D_4\\D_5\\D_6\\ \hdashline P_1\\P_2\\P_3\\P_4 \end{array}\begin{bmatrix} 1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&0&1&0&0\\0&0&0&0&1&0\\0&0&0&0&0&1\\ \hdashline 1&0&1&1&1&0\\0&1&1&0&0&0\\0&1&0&0&1&0\\1&1&0&0&1&1 \end{bmatrix}$$

(b)(10,6,1,3)RBEC的编码矩阵$\boldsymbol{G}_{10\times6}$

图 16 (10,6,1,3)RBEC 下分布式存储系统

若存储系统同时出现 3 个节点损毁或丢失,可利用余下存活的 7 个节点以高于 $1-1/2^1$ 概率将丢失节点恢复. 例如,当信息节点 D_1,D_2,D_3 同时损毁,按照文中 3.2 节的译码方法可得 $D_1=D_6\oplus P_3\oplus P_4,D_2=D_5\oplus P_3,D_3=D_5\oplus P_2\oplus P_3$. 但是,当信息节点 D_1,D_4,D_6 同时损毁,则节点无法恢复.

为了避免系统 t 个节点同时损坏但无法恢复情况出现,参数 δ 大小至关重要. 由 3.3 节中译码恢复概率分析可知,当 $\delta\geq20$ 时,基于 RBEC 的存储系统出现任意 t 个节点同时损坏,余下 $k+\delta$ 个节点的向量组合成的矩阵满秩概率将超过 0.999 999,恢复失败概率将小于百万分之一;并随着 δ 值越大,概率越高,恢复概率趋近 1.

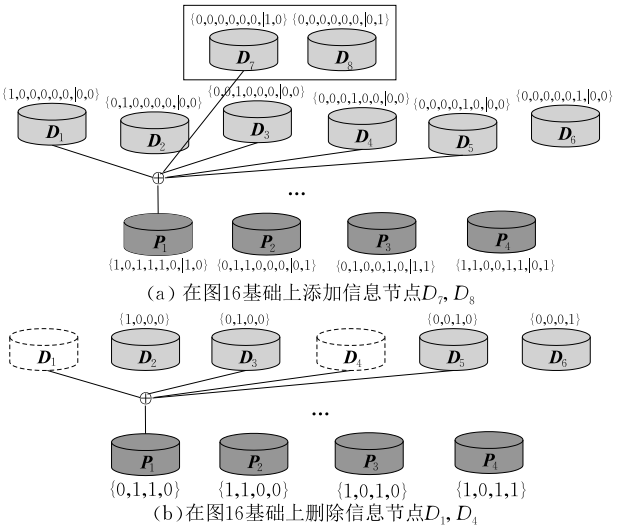
当 δ 值确定后,k 与 t 值在编码方案实现上并无限制,对于任意 k 与 t 值均能依照上述步骤架构系统. 同时,RBEC 参数动态调整能力也为存储系统实现实时调整存储效率和纠错率提供了编码基础. 存储系统可通过对信息节点或校验节点动态的增减,实现运行过程中出现的增大或缩小存储容量、调整容灾能力等需求变化,操作也十分简便.

5.2 动态调整信息节点

将 RBEC 编码方案应用于存储系统,系统可动态控制信息节点数目. 通过增加或减少信息节点个数,实现对有效存储容量的调整,改变系统存储效率 $k/(k+\delta+t)$. 相对传统编码方案,系统只需对部分校验节点数据进行更新,而不存在重编码再整合各节点数据过程,具体如下描述.

系统若需增加 $l(l>0)$个新的信息节点时,包含如下步骤:(1)各存储节点更新自身的节点向量,向量长度由 $1\times k$ 更改为 $1\times(k+l)$. 其中,信息节点的向量依然为单位向量,只是长度发生变化;校验节点保留原有节点向量信息,并在尾部添加 l 个新元素,且各元素仍然按照概率 0.5 取值 0 或 1;(2)各校验节点根据各自向量添加的元素值与对应增添信息节点进行异或操作建立编码联系. 如图 17(a)是在图 16 的存储系统基础上添加了信息节点 D_7,D_8,k 值由 6 变为 8,其存储效率也由 0.60 增大为 0.67;校验节点 P_1 的向量由{1,0,1,1,1,0}更新为{1,0,1,1,1,0,|1,0},节点 P_1 数据需再与节点 D_7 进行异或,此时 $P_1=D_1\oplus D_3\oplus D_4\oplus D_5\oplus D_7$;扩充后(12,8,1,3)RBEC 的编码矩阵 $\boldsymbol{G}_{12\times8}$ 如图 18(a)所示.

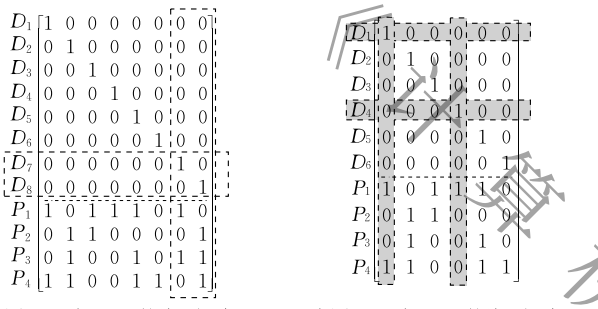
将 $l(0<l<k)$个信息节点从系统中删除时,包含如下步骤:(1)各存储节点更新自身的节点向量,向量长度由 $1\times k$ 缩短为 $1\times(k-l)$. 其中需确定待删除信息节点编号$\{D_{i1},\cdots,D_{il}\}$;然后,各节点只需将向量的第$\{i1,\cdots,il\}$位元素删除;(2)校验节点若与删除信息节点存在编码联系,需要与删除信息节点进行异或操作完成数据更新. 例如,图 17(b)是在图 16 的存储系统基础上删去信息节点 D_1,D_4,k 值由 6 减少为 4,其存储效率也由 0.6 减少为 0.5;各存储节点的向量将删除第 1 位和第 4 位元素. 如校验



(a) 在图16基础上添加信息节点D_7, D_8

(b) 在图16基础上删除信息节点D_1, D_4

图 17　分布存储系统增加或减少信息节点示意图



(a) 在图17(a)中RBEC的编码矩阵$\boldsymbol{G}_{12\times8}$　(b) 在图17(b)中RBEC的编码矩阵$\boldsymbol{G}_{8\times4}$

图 18　调整信息节点数目后 RBEC 的编码矩阵

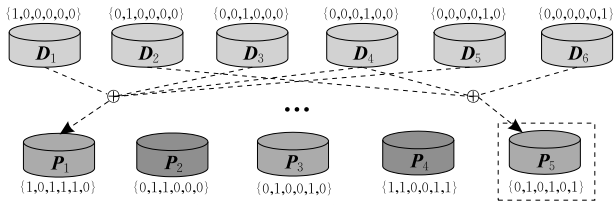
节点 P_1 的节点向量由{1,0,1,1,1,0}更新为{0,1,1,0},由于节点 P_1 先前与 D_1,D_4 存在编码关系,需再进行异或运算完成数据更新,此时 $P_1=D_3\oplus D_5$. 系统删除信息节点后,参数为(8,4,1,3)RBEC 的编码矩阵 $\boldsymbol{G}_{8\times4}$ 如图 18(b)所示.

从图 18 易观察出,不论是增加或减少信息节点数目,应用于系统中的 RBEC 编码矩阵均是在原矩阵基础上进行调整,并未从全局重新设计码字,各存储节点的数据更新也相互独立;同时,变换后的 RBEC 编码矩阵依然保持着系统码结构,相比传统编码方案,省去了单位化编码矩阵的过程.

5.3　动态调整校验节点

由于系统中校验节点数据是起到冗余容灾作用,在未出现信息节点损毁或丢失情况下,校验节点数目多少并不会影响到信息节点. 因此,相比调整信息节点,系统动态调整校验节点更为简单.

系统若需增加 $l(l>0)$ 个新的校验节点时,包含如下步骤:(1) 新添的校验节点根据参数 k 随机生成一个 $1\times k$ 的节点向量,各元素按照概率 0.5 取值 0 或 1;(2) 新添校验节点根据生成的节点向量中值为 1 的元素位置与对应编号下的信息节点进行异或操作,建立编码联系. 如图 19(a)是在图 16 的存储系统基础上添加了校验节点 P_5;根据 P_5 生成的节点向量{0,1,0,1,0,1},P_5 与节点 D_2,D_4,D_6 存在编码联系,此时 $P_5=D_2\oplus D_4\oplus D_6$;系统增添节点 P_5 后,参数为(11,6,2,3)RBEC 的编码矩阵 $\boldsymbol{G}_{11\times6}$ 如图 19(b)所示.



(a) 在图16基础上添加校验节点P_5

$$\begin{array}{c} \\ D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \end{array}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

(b) 添加校验节点后的RBEC编码矩阵

图 19　分布存储系统增加校验节点示意图

系统若要删除 $l(l>0)$ 个校验节点,直接将待删除 l 个校验节点从系统中移除即可.

通过增加或减少系统中的校验节点个数,系统可实时调整最大容删能力 t,同时也可调整恢复任意 t 个损毁或丢失节点数据的最低成功概率 $1-1/2^{\delta}$. 并且整个调整过程呈现在编码矩阵上,只是矩阵行的增加或减少,操作方便.

5.4　应用效果

将 (n,k,δ,t)RBEC 应用于分布式存储系统,存储节点之间只存在异或运算,信息节点和校验节点相互分离,易于读取数据. 从系统中任取 $k+\delta$ 个节点均能以高于 $1-1/2^{\delta}$ 概率将 k 个信息节点数据恢复,其中 δ 取值范围只在数十以内,而 k 与 t 值并无限制,可取数十、数百、甚至数万;当 $\delta\ll k$,RBEC 将拥有近似 MDS 性质. 利用文中 3.2 节方法可优化最终译码方程组规模,同时由于译码矩阵完全由‘0、1’构成,使用 XOR Scheduling 等方法可进一步对恢复带宽或译码时间做出优化.

基于 RBEC 的存储系统中各存储节点信息相互独立,只保留了自身的节点向量,并没有系统宏观编码矩阵的数据;但从系统层面看,这些独立节点正是通过 RBEC 进行整合,使得存储系统不但可保证

高概率容错能力,而且拥有动态调节的能力.

此外,RBEC 应用可不限于分布式存储,若将节点改换成硬盘,即可在 RAID 系统得到应用,显然它只有磁盘间数据块的异或运算,并且纠删粒度可达扇区,具有发展潜力;若将节点改换为传感器,即可在存储空间极其宝贵而容灾要求非常高的传感网络领域得到应用,等等.RBEC 的容灾存储应用值得更加广泛的探讨与深入的研究.

6 总结与展望

在大数据时代,数据是最宝贵的资源,存储系统如何保证数据高可靠抗风险已然成为每个用户、企业关心的要点.当前分布式存储系统的数据可靠性保障方式正逐步从单一的复制备份方式向着与纠删码容灾方式协同方向发展.基于纠删码的容灾技术策略逐步被技术界及产业界认识和采纳.本文提出了一种新的数据存储编码方法:随机二元扩展码(RBEC).它为分布式存储系统的纠删码容灾技术途径提供了一种新的选择.

RBEC 是一种只需异或运算的随机码类,编码结构采用自底向上的设计模式,达到码字整体上性能优势.与 RS 码相比,RBEC 完全摆脱了有限域运算的限制,参数选择更加自由,运算效率更高;与 LT、LDPC 等编码方法相比,RBEC 的编码结构更加简洁,构造过程更加方便易行,易于工程开发与应用.同时,与现有主流编码方法相比,RBEC 对信息位和校验位均有动态调整的能力,编码矩阵具有良好的伸展收缩特性.从而,给定任意参数 k,δ,t 的组合,RBEC 都能给出明确的构造方法.当 RBEC 出现任意 t 位丢失或删除时,我们给出了 RBEC 成功译码概率下界,以及数学证明和实验分析;表明当冗余指数 $\delta \geq 20$,恢复失败概率将小于百万分之一,并随 δ 的增大将不断变小.对于 RBEC 编码矩阵特殊结构,文中给出一种最小化译码方程组规模的方法,可以有效的节省译码时间.将 RBEC 应用在存储系统时,系统中节点只有异或操作,并且可根据实际需求的变化,实时调整信息节点或校验节点数目,操作简单.后续工作中,我们将研究编码矩阵中随机矩阵部分以小于 0.5 概率生成各个元素时,RBEC 的容错能力,进一步稀疏化编码矩阵;以及开发针对 RBEC 结构的高效译码算法,并且尝试 RBEC 的更多应用环境.

参 考 文 献

[1] Schroeder B, Gibson G A. Disk failures in the real world: What does an MTTF of 1 000 000 hours mean to you?//Proceedings of the 5th USENIX conference on File and Storage Technologies. San Jose, USA, 2007: 1-16

[2] Pinheiro E, Weber W D, Barroso L A. Failure trends in a large disk drive population//Proceedings of the 5th USENIX conference on File and Storage Technologies. San Jose, USA, 2007: 17-28

[3] Weatherspoon H, Kubiatowicz J D. Erasure coding vs. replication: A quantitative comparison//Proceedings of the 1st International Workshop on Peer to Peer Systems. Cambridge, USA, 2002: 328-337

[4] Zhang Z, Deshpande A, Thereska E, et al. Does erasure coding have a role to play in my data center? Microsoft Corporation, USA: Technical Report MSR-TR-2010-52, 2010

[5] Reed I S, Solomon G. Polynomial codes over certain finite fields. Journal of Society for Industrial and Applied Mathematics, 1960, 8(2): 300-304

[6] Ghemawat S, Gobioff H, Leung S T. The google file system //Proceedings of the 19th ACM Symposium on Operating Systems Principles. New York, USA, 2003: 29-43

[7] Calder B, Wang J, Ogus A, et al. Windows azure storage: A highly available cloud storage service with strong consistency //Proceedings of the 23rd ACM Symposium on Operating Systems Principles. Cascais, Portugal, 2011: 143-157

[8] Huang C, Simitci H, Xu Y K, et al. Erasure coding in windows azure storage//Proceedings of the 2012 USENIX Conference on Annual Technical Conference. Boston, USA, 2012: 15-26

[9] Sathiamoorthy M, Asteris M, Papailiopoulos D, et al. XORing elephants: Novel erasure codes for big data//Proceedings of the 39th International Conference on Very Large Data Bases. Riva del Garda, Italy, 2013: 325-336

[10] Blomer J, Kalfane M, Karp R, Karpinski M, Luby M, et al. An XOR-Based erasure-resilient coding scheme. USA: International Computer Science Institute, UC Berkeley, Technical Report: ICSI TR-95-0 48, 1995

[11] Plank J S. Optimizing cauchy Reed-Solomon codes for fault-tolerant//Proceedings of the 5th IEEE International Symposium on Network Computing and Applications. Washington, USA, 2006: 173-180

[12] Luo J, Bowers K D, Oprea A, Xu L. Efficient software implementations of large finite fields $GF(2^n)$ for secure storage applications. ACM Transactions on Storage, 2012, 8(2): 1-27

[13] Plank J S, Greenan K M, Miller E L. Screaming fast galois field arithmetic using intel SIMD instructions//Proceedings of the 11th Usenix Conference on File and Storage Technologies. San Jose, USA, 2013: 299-306

[14] Gallager R G. Low-density parity-check codes. IRE Transactions

on Information Theory, 1962, 8(1): 21-28

[15] Luby M G, Mitzenmacher M, Shokrollahi M A, Spielman D A, Stemann V. Practical loss-resilient codes//Proceedings of the 29th Annual ACM Symposium on Theory of Computing. El Paso, USA, 1997: 150-159

[16] Luby M, Mitzenmacher M, Shokrollahi M A, Spielman D A. Efficient erasure correcting codes. IEEE Transactions on Information Theory, 2001, 47(2): 569-584

[17] Luby M. LT codes//Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science. Vancouver, Canada, 2002: 271-280

[18] MacKay D J C. Fountain codes. IEE Proceedings Communications, 2005, 152(6): 1062-1068

[19] Li S-Y R, Sun Q T, Shao Z. Linear network coding: Theory and algorithms. Proceedings of the IEEE, 2011, 99(3): 372-387

[20] Plank J S, Thomason M G. A practical analysis of low-density parity-check erasure codes for wide-area storage applications //Proceedings of the 2004 International Conference on Dependable Systems and Networks. Florence, Italy, 2004: 115-124

[21] Harihara S G, Janakiram B, Chandra M G, Aravind K G, et al. SpreadStore: A LDPC erasure code scheme for distributed storage system//Proceedings of the 2010 International Conference on Data Storage and Data Engineering. Bangalore, India, 2010: 154-158

[22] Kubiatowicz J, Bindel D, Chen Y, et al. Oceanstore: An architecture for global-scale persistent storage. ACM SIGPLAN Notices, 2000, 35(11): 190-201

[23] Xia Huaxia, Chien A. RobuSTore: A distributed storage architecture with robust and high performance//Proceedings of the 2007 ACM/IEEE Conference on Supercomputing. Reno, USA, 2007: 1-11

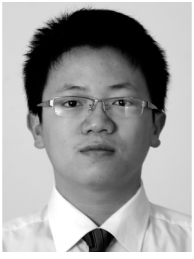
[24] Dimakis A G, Godfrey P B, Wu Y, et al. Network coding for distributed storage systems. IEEE Transactions on Information Theory, 2010, 56(9): 4539-4551

[25] Acedanski S, Deb S, Medard M, Koetter R. How good is random linear coding based distributed networked storage// Proceedings of the 1st WorkShop Network Coding, Theory and Applications. Riva del Garda, Italy, 2005: 1-6

[26] Plank J S, Simmerman S, Schuman C D. Jerasure: A Library in C/C++ Facilitating Erasure Coding for Storage Applications. Department of Electrical Engineering and Computer Science, University of Tennessee, USA: Technical Report: CS-08-627, 2008

[27] Huang C, Li J, Chen M. On optimizing XOR-based codes for fault-tolerant storage applications//Proceedings of the 2007 IEEE Information Theory Workshop. Lake Tahoe, USA, 2007: 218-223

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



CHEN Liang, born in 1990, Ph. D. candidate. His current research interests include storage system reliability and RAID Code.

ZHANG Jing-Zhong, born in 1986, Ph. D. candidate. His current research interests include secret sharing and storage system reliability.

TENG Peng-Guo, born in 1986, Ph. D. candidate. His current research interests include storage system reliability and RAID Code.

WANG Xiao-Jing, born in 1953, professor, Ph. D. supervisor. His main research interests include information security, secret sharing and network storage, etc.

Background

With the arrival of big data era, the capacity of distributed storage system has been growing rapidly and erasure code gradually replaces replication for fault tolerance. Although erasure code based fault tolerance has been researched for a while, most of them are based on RS code and the arithmetic in finite fields, which typically have higher multiplication complexity and lower operation efficiency. Some XOR-based codes have also been analyzed in storage system, such as Tornado, LT, LDPC, etc. However, these codes are designed for communication. The use of them not only makes distributed storage system become more complicated but also limits the selection of tolerance parameters.

Considering the types of erasure code used in storage system are too less, we herein provide a novel coding method called Random Binary Extensive Code (RBEC). The RBEC is a XOR-only systematic code and has a simple construction that can be easily implemented in distributed storage system. As the RBEC can dynamically modulate information part and parity part, it has few limitations on parameters and shows excellent dynamic extension ability. For the special construction of RBEC, we also provide a method to minimize the size of decoding matrix, which can largely reduce the decoding time.

This research is supported by the National Natural Science Foundation of China under Grant No. 61501064 and the Sichuan Science & Technology Support Program under Grant No. 2015GZ0088.