计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190600063

一种参数可变的最小存储再生码

王雪冰
天津大学精密仪器与光电子工程学院　天津 300072
中国石油大学胜利学院　山东 东营 257061

摘　要　参数为(n,k,B,d,t)的功能性修复最小存储再生码采用(n,k)删除码策略,依靠d个帮助节点修复t个节点的失效。出于存储空间、修复带宽、可修复节点数等因素的考虑,需要将一个参数为(n_1,k_1,B,d_1,t_1)的功能性修复再生码转换为另一个参数为(n_2,k_2,B,d_2,t_2)的功能性修复再生码,并且希望这个转换过程能够以最小下载数据量的代价进行。针对此问题,采用逻辑节点和物理节点相结合的方法,构造了一种可变参数的功能性修复再生码,证明了该码可以在不同参数组之间相互转换,而且在转换过程中使用最小下载数据量。

关键词:存储码;再生码;修复带宽;可变参数码;逻辑节点;物理节点
中图法分类号　TP393

Minimum Storage Regenerating Code with Variable Parameters

WANG Xue-bing
School of Precision Instruments and Optoelectronics Engineering, Tianjin University, Tianjin 300072, China
Shengli College, China University of Petroleum, Dongying, Shandong 257061, China

Abstract　A functional repair minimum storage regenerating code with the parameters of (n,k,B,d,t) leverages the strategy of (n,k) erasure code to repair a number of t nodes malfunction with the help of d helper nodes. Considering the elements of storage space, repair bandwidth, and the number of repairable nodes, a functional repair regenerating code with parameters of (n_1,k_1,B,d_1,t_1) needs to be transformed into another functional repair regenerating code with parameters of (n_2,k_2,B,d_2,t_2), and hopefully the transforming process can be done with minimum data downloading. To this end, by combining logical nodes with physical nodes, a functional repair regenerating code with variable parameters is constructed. It is proved the code can be transformed between different parameters and the minimum download data is used in the transforming process.

Keywords　Storage code, Regenerating code, Repair bandwidth, Variable parameter code, Logical node, Physical node

1　引言

传统的数据中心把客户数据集中存储在一个房间或一栋建筑内的磁盘阵列中,这种方式的优点是便于管理,但明显的缺点就是随着业务量的增加,数据中心负荷急剧增加,瓶颈效应明显;数据中心的另一个缺点是应对突发事件能力较弱,一旦出现地震、火灾等灾害,数据中心损毁,用户数据将无法恢复,损失巨大。随着互联网技术的飞速发展,数据存储方式逐渐从集中式转变为分布式。与数据中心不同,分布式存储系统(Distributed Storage System,DSS),把客户数据分散存储在空间上相分离且相距较远的若干个磁盘驱动器或存储节点内,这些磁盘可能分布在一个城市的各个角落,或者位于各个城市,甚至是各个国家。DSS最大的特点就是把通信负荷分散到各个存储节点,从而有效解决了数据传输的瓶颈问题。DSS已经取代数据中心成为市场上主流的数据存储技术,商用的DSS有谷歌文件系统、Dropbox、微软OneDrive、亚马逊S3、百度云盘等。

对于一个DSS来说,任何一个存储节点都可能损坏,导致其上存储的数据丢失,因此对DSS的一个基本要求是可修复性,即通过依然存活的存储节点上存储的数据重新生成丢失的数据并重建损坏的节点。修复性可以依靠数据冗余来实现,传统的数据冗余采用"复制"的实现方式,即网络存储节点上存储的数据就是用户原始数据的拷贝,典型例子是谷歌文件系统。这种基于"复制"的分布式存储方式简单、易实现、易管理,但文献[1]证明这种存储方式的存储效率远远低于基于"编码"的存储方式。以(n,k)最大距离可分码(Maximum Distance Separable,MDS)为例,客户的原始数据是k个数据块,经过删除码编码后生成n个码数据块,码数据块和原始数据块的大小相同,把这n个码数据块分别存储在n个节点上,从这n个码数据块中任意选择k个码数据块即可恢复k个原始数据块。可见,(n,k)-MDS码的码率为k/n,且最大能承受$(n-k)$个码数据块的丢失。

DSS为了修复某个损坏的存储节点上丢失的码数据,向依然存活的节点请求的码数据块的个数被称为修复带宽。在上述(n,k)-MDS码存储系统中,假设用户原始文件的大小为$|B|$,则每个数据块的大小为$\alpha=\frac{|B|}{k}$,(n,k)-MDS码把k个原始数据块编码生成n个码数据块并存储在n个存储节点上,因此总的存储数据量为$\frac{|B|}{k}n$。(n,k)-MDS为了修复某个

通信作者:王雪冰(bbxxww2002@163.com)

节点上存储的数据量为 α 的码数据，需要请求的数据量 $|B|=k\alpha$，这个修复带宽是很低效的。文献[2]提出了再生码(Regenerating Code)的概念，再生码也使用(n,k)删除码把 k 个原始数据编码生成 n 个码数据块，当某个码数据块丢失时，系统对 d 个存活节点中的每一个请求数据量进行修复，因此修复带宽等于 $d\beta$。文献[2]采用信息流图的方法证明了再生码在每个节点存储的数据量和修复单个节点所需的修复带宽之间存在折中曲线，曲线的两端分别是最小存储再生码(Minimum Storage Regenerating code，MSR)和最小带宽再生码(Minimum Bandwidth Regenerating code，MBR)。进一步，文献[3]提出了 3 种修复模型：精确修复、功能性修复和精确修复系统节点。精确修复是指重新生成的码数据精确地等于丢失的码数据；功能性修复是指重生的码数据不等于丢失的码数据，但与存活的码数据依然构成(n,k)删除码关系，即依靠任意的 k 个码数据块都可以恢复 k 个原始数据块；精确修复系统节点采用系统码，即用户原始数据存储在 k 个系统节点中，校验数据存储在剩余节点中，修复时精确重生 k 个系统节点中的用户数据，而校验数据不需要精确修复，只需要保持校验关系即可。

Dimakis 等[2-3]提出的参数为(n,k,d)再生码一次只能修复一个节点失效，对于多个节点的失效问题需要逐个修复。文献[4]和文献[5]提出了参数为$(n,k,d>k,t)$的再生码方案，用于同时修复 t 个节点失效，证明了在失效节点之间互相合作可以减小修复这些失效节点的修复带宽。由文献[4]和文献[5]可知，对于任意一组参数$(n,k,d>k,t)$，都存在参数为(n,k,d,t)的功能性修复 MSR 码，而且这种码存在一种高效的构造方法。参数 k 和 d 的值决定了为了修复 t 个失效节点需要下载的最小数据量，即修复带宽。对于某个固定取值的 k 和 d，取值越大则修复 t 个失效节点的修复带宽就越小。文献[6]指出对于同一组用户数据 B，可以设计出具有不同参数 d 和 t 的功能性修复 MSR 码，即存在多组参数为(n,k,B,d_i,t_i)的功能性修复 MSR 码。实际应用中，DSS 的各个存储节点的容量可能是有差别的、不对等的，文献[7]讨论了在非均匀异构存储系统中进行数据修复时如何选择帮助节点进行修复的问题，提出了树形拓扑修复算法的降低修复成本。文献[8]针对多节点同时失效、新节点之间不再传输数据的情形，利用割型得到了最小割的一般表达式，找到了节点存储和修复带宽的折中关系。文献[9]讨论了 MSR 码在尺度可变网络中的应用，针对单节点或多节点失效问题，提出了并行修复和串行修复两种方法。文献[10]讨论了如何在任何大小的域，尤其是小尺寸的域上设计 MSR 和 MBR 的问题。文献[11]提出了一种中心式的多节点修复策略和把单节点修复转换为多节点修复的框架，并应用乘积矩阵码和干扰对齐对其进行演示。

本文讨论下述问题：对于一组用户原始数据 B，假设存在一个参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码，出于存储空间、修复带宽、可修复节点数目等因素的考虑，需要将其转换为另一个参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码，并且希望这个转换过程能够以最小下载数据量为代价进行，那么该如何设计参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码呢？另外，这个转换过程该如何设计？

类似的问题曾在文献[12-15]中被讨论，但这些工作研究的都是(n_1,k_1)删除码向(n_2,k_2)删除码转换的问题，不能用于多节点失效的场景。具体地，文献[12-13]提出的转换方法不是最优的，这是因为该方法没有得出转换所需的最小数据下载量。文献[14]提出了一种编码策略，实现了对于可行的参数 n_1,k_1,n_2,k_2，可以以最小的下载量把(n_1,k_1)删除码转换为(n_2,k_2)删除码。随后，文献[15]提出了一种编码策略，对于同一个文件 B，以最小的下载量把(n_i,k_i)删除码转换为(n_j,k_j)删除码，$1\leqslant i,j\leqslant m,i\neq j$。

本文对上述工作进行了更为深入的研究和推广，提出了一种可变参数(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码的构造方法，构造过程使用了物理节点和逻辑节点相结合的思想，所构造的(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码可以以最小下载数据量转换为另一个参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。

本文第 2 节提出了一种参数可变的编码策略，用于把参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码转换为参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码，分 3 种情况详细讨论了这一转换过程如何进行，并证明了该转换方法具有最小下载量性能；最后总结全文。

2 参数可变的 MSR 编码策略

本文的目的是以最小下载量实现参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码向参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码的转换。参数 B 表示用户原始数据文件，文件大小为$|B|$。对应于两种 MSR 码的总的存储空间分别为 $\frac{|B|}{k_1}n_1$ 和 $\frac{|B|}{k_2}n_2$，比较这两个存储空间的大小存在 3 种可能：

$$\frac{|B|}{k_1}n_1<\frac{|B|}{k_2}n_2;\frac{|B|}{k_1}n_1=\frac{|B|}{k_2}n_2;\frac{|B|}{k_1}n_1>\frac{|B|}{k_2}n_2$$

我们首先给出参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码的设计过程，其次针对上述 3 种可能，分别讨论如何把参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码转换为参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。

参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码的设计过程包括两步。

第 1 步　设计一组参数为$\left(\max(n_1,n_2)\cdot\frac{LCM(k_1,k_2)}{\min(k_1,k_2)},LCM(k_1,k_2),B,D_i,T_i\right)$，$i=1,2,3$

的功能性修复 MSR 码，其中 $LCM(\cdot)$表示最小公倍数。需要说明的是，虽然参数 D_i 和 T_i 可以有多种可能的取值，但与把码(n_1,k_1,B,d_1,t_1)转换为码(n_2,k_2,B,d_2,t_2)相关的参数只有 3 组，分别是：

(1) $D_1=d_1\cdot\frac{LCM(k_1,k_2)}{k_1}$，$T_1=t_1\cdot\frac{LCM(k_1,k_2)}{k_1}$

(2) $D_2=d_2\cdot\frac{LCM(k_1,k_2)}{k_2}$，$T_2=t_2\cdot\frac{LCM(k_1,k_2)}{k_2}$

(3) 如果 $T_3=n_2\cdot\frac{LCM(k_1,k_2)}{k_2}-n_1\cdot\frac{LCM(k_1,k_2)}{k_1}>0$，则 $D_3=n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$；否则 $D_3=0,T_3=0$。

上面设计的 MSR 码需要 $\max(n_1,n_2)\cdot\frac{LCM(k_1,k_2)}{\min(k_1,k_2)}$ 个

存储节点,依靠其中任意的 $LCM(k_1,k_2)$ 个存储节点即可重建原始文件 B。该码可以进行 3 种修复:依靠 D_1 个帮助节点可以修复 T_1 个失效节点,依靠 D_2 个帮助节点可以修复 T_2 个失效节点,依靠 D_3 个帮助节点可以修复 T_3 个失效节点。这样的码是存在的,可以使用文献[6]给出的方法进行构造。

第 2 步　基于第 1 步生成的码构造参数为 (n_1,k_1,B,d_1,t_1) 的功能性修复 MSR 码。首先,把每个物理节点看成由 $\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点构成,那么全部 n_1 个物理节点一共包含了 $n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点。其次,从第 1 步生成的参数为 $(\max(n_1,n_2)\cdot\frac{LCM(k_1,k_2)}{\min(k_1,k_2)},LCM(k_1,k_2),B,D_i,T_i)$ 的功能性修复 MSR 码中任选 $n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$ 个节点,注意因为 $\max(n_1,n_2)\cdot\frac{LCM(k_1,k_2)}{\min(k_1,k_2)}\geqslant n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$,所以这一步一定是可以选出的,然后把选出的这 $n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$ 个节点上存储的码数据一一对应地映射到 $n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点上,这样我们就得到了一组参数为式(1)的功能性修复 *MSR* 码。

$$\left(n_1\cdot\frac{LCM(k_1,k_2)}{k_1},LCM(k_1,k_2),B,D_i,T_i\right)\quad i=1,2,3 \tag{1}$$

其中:(3)$D_1=d_1\cdot\frac{LCM(k_1,k_2)}{k_1}$,$T_1=t_1\cdot\frac{LCM(k_1,k_2)}{k_1}$

(4)如果 $d_2\cdot\frac{LCM(k_1,k_2)}{k_2}\leqslant n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$,则 $D_2=d_2\cdot\frac{LCM(k_1,k_2)}{k_2}$,$T_2=t_2\cdot\frac{LCM(k_1,k_2)}{k_2}$;否则 $D_2=0$,$T_2=0$。

(5)如果 $T_3=n_2\cdot\frac{LCM(k_1,k_2)}{k_2}-n_1\cdot\frac{LCM(k_1,k_2)}{k_1}>0$,则 $D_3=n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$;否则 $D_3=0$,$T_3=0$。

(6)$D_2=0$,$T_2=0$ 意味着 n_1 个物理节点包含的逻辑节点的数目小于 $d_2\cdot\frac{LCM(k_1,k_2)}{k_2}$,因此无法修复 $t_2\cdot\frac{LCM(k_1,k_2)}{k_2}$ 个逻辑失效节点。

我们称式(1)给出的码为码 $\mathscr{C}$,一共包括 $n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点,每个逻辑节点包含的数据量为 $|B|/LCM(k_1,k_2)$,应用任意的 $LCM(k_1,k_2)$ 个逻辑节点即可恢复 $LCM(k_1,k_2)$ 个原始数据。把 $\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点归并成 1 个物理节点,就得到了 n_1 个物理节点,每个物理节点包含 $\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点,应用任意的 k_1 个物理节点即可恢复 k_1 个原始数据。因此,从物理节点的角度看,式(1)给出的码 $\mathscr{C}$ 就对应于 $(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$,$i=1,2,3$ 的功能性修复 MSR 码,其中:

(7)$\boldsymbol{d}_1=\boldsymbol{d}_1$,$\boldsymbol{t}_1=\boldsymbol{t}_1$;

(8)如果 $d_2\cdot\frac{LCM(k_1,k_2)}{k_2}\leqslant n_1\cdot\frac{LCM(k_1,k_2)}{k_1}$,则 $\boldsymbol{d}_2=d_2\cdot\frac{k_1}{k_2}$,$\boldsymbol{t}_2=t_2\cdot\frac{k_1}{k_2}$;否则 $\boldsymbol{d}_2=0$,$\boldsymbol{t}_2=0$。

(9)如果 $n_2\cdot\frac{LCM(k_1,k_2)}{k_2}-n_1\cdot\frac{LCM(k_1,k_2)}{k_1}>0$,则 $\boldsymbol{d}_3=n_1\boldsymbol{t}_3=n_2\cdot\frac{k_1}{k_2}-n_1$;否则 $\boldsymbol{d}_3=0$,$\boldsymbol{t}_3=0$。

注意 $\boldsymbol{d}_2,\boldsymbol{t}_2,\boldsymbol{t}_3$ 可以取非整数,非整数取值的 $\boldsymbol{t}_2,\boldsymbol{t}_3$ 意味着部分节点失效,即某个节点中有一部分数据损坏,非整数取值的 $\boldsymbol{d}_2$ 意味着某些帮助节点中只有部分数据参与数据修复。

对于上面基于码 $\mathscr{C}$ 构造的 $(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$,$i=1,2,3$ 的功能性修复 MSR 码,根据参数 n_1,n_2,k_1,k_2 取值的不同,我们将证明该修复码可以以最小下载数据量转换为参数为 (n_2,k_2,B,d_2,t_2) 的功能性修复 MSR 码。根据存储空间大小的不同,以下分 3 种情况加以讨论。

2.1　$\frac{|\boldsymbol{B}|}{\boldsymbol{k}_1}\boldsymbol{n}_1=\frac{|\boldsymbol{B}|}{\boldsymbol{k}_2}\boldsymbol{n}_2$

定理 1　当 $\frac{|B|}{k_1}n_1=\frac{|B|}{k_2}n_2$ 时,基于码 $\mathscr{C}$ 构造的参数为 $(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$,$i=1,2,3$ 的功能性修复 MSR 码可以以最小下载数据量转换为参数为 (n_2,k_2,B,d_2,t_2) 的功能性修复 MSR 码。

证明:对于 $\frac{|B|}{k_1}n_1=\frac{|B|}{k_2}n_2$,存在 3 种可能的参数取值。

(1)$n_1=n_2$,$k_1=k_2$;

(2)$n_1>n_2$,$k_1>k_2$;

(3)$n_1<n_2$,$k_1<k_2$。

当 $n_1=n_2$,$k_1=k_2$ 时,根据第 2 步中的(7)可得 $\boldsymbol{d}_2=d_2$,$\boldsymbol{t}_2=t_2$,因此所构造的参数为 (n_1,k_1,B,d_1,t_1) 的功能性修复 MSR 码自动成为 (n_2,k_2,B,d_2,t_2) 的功能性修复 MSR 码。

当 $n_1>n_2$,$k_1>k_2$ 时,为了把参数为 $(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$ 的 MSR 码转换为参数为 (n_2,k_2,B,d_2,t_2) 的 MSR 码,把 n_2 个物理节点中的每一个看成由 $\frac{LCM(k_1,k_2)}{k_2}$ 个逻辑节点构成,因为:

$$\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}>0 \tag{2}$$

$$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_2=\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2) \tag{3}$$

所以只需要把 (n_1,k_1,B,d_i,t_i) MSR 码中的任意一组 n_1-n_2 个物理节点上的数据转移到另外的 n_2 个物理节点上。转移方法如下:在 (n_1,k_1,B,d_i,t_i) MSR 码中,每个物理节点对应 $\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点,n_1-n_2 个物理节点一共包括 $\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)$ 个逻辑节点,把这些逻辑节点上存储的数据平均分配给剩余的 n_2 个物理节点,因此每个目标节点将分得 $\frac{LCM(k_1,k_2)}{k_1}\cdot\frac{(n_1-n_2)}{n_2}$ 个逻辑节点,加上原有的 $\frac{LCM(k_1,k_2)}{k_1}$ 个逻辑节点,每个物理节点一共包含 $\frac{LCM(k_1,k_2)}{k_2}$ 个逻辑节点。由此,我们就获得了一组参数为式(4)的功能性修复 MSR 码,每个物理节点对应 $\frac{LCM(k_1,k_2)}{k_2}$

个逻辑节点，每个逻辑节点包含的数据量仍为$\frac{|B|}{LCM(k_1,k_2)}$。从物理节点的角度看，该码就是参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。上述转换过程把 n_1-n_2 个物理节点上的数据转移到另外的 n_2 个物理节点上，因此使用了最小的数据下载量。

$$\left(\frac{n_2\cdot LCM(k_1,k_2)}{k_2}-LCM(k_1,k_2),B,D_i,T_i\right),i=1,2,3 \tag{4}$$

当 $n_1<n_2,k_1<k_2$ 时，为了把参数为$(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$的 MSR 码转换为参数为$(n_2,k_2,B,d_2,t_2)$的 MSR 码，仍然把 n_2 个物理节点中的每一个物理节点看成由$\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点构成。但此时式(5)、式(6)成立。

$$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}>0 \tag{5}$$

$$\left(\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}\right)\cdot n_1=\frac{LCM(k_1,k_2)}{k_2}\cdot(n_2-n_1) \tag{6}$$

为了实现码的转换，首先新建 n_2-n_1 个物理节点，其次从 n_1 个物理节点中任选$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点，一共选出$\left(\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}\right)\cdot n_1$个逻辑节点，把这些逻辑节点平均分配到 n_2-n_1 个物理节点上，使得每个物理节点分得$\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点。由此我们获得了参数为式(7)的功能性修复 MSR 码。

$$\left(\frac{n_2\cdot LCM(k_1,k_2)}{k_2},LCM(k_1,k_2),B,D_i,T_i\right),i=1,2,3 \tag{7}$$

从物理节点的角度看，该码包含了参数为$(n_2,k_2,B,\boldsymbol{d}_2,\boldsymbol{t}_2)$的功能性修复 MSR 码的属性，而且具有最小下载量。

当$\frac{|B|}{k_1}n_1=\frac{|B|}{k_2}n_2$时，比较两种码转换前后的参数。首先，码的可修复节点数和修复带宽分别由参数 t 和$\frac{|B|}{k}d$表征，转换前后视参数的不同而不同。其次，如前所述，(n,k)-MDS 码把 k 个原始数据块编码生成 n 个码数据块存在 n 个存储节点上，因此总的存储数据量为$\frac{|B|}{k}n$，因此当$\frac{|B|}{k_1}n_1=\frac{|B|}{k_2}n_2$时码的转换前后占用的存储空间不变。最后，考察转换过程中下载的数据量，对于情况 1)，下载量为 0；对于情况 2)，需要转移$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)$个逻辑节点上的数据，每个逻辑节点包含的数据量为$\frac{|B|}{LCM(k_1,k_2)}$，因此总的下载量为$\frac{|B|}{k_1}(n_1-n_2)$；同理，对于情况 3)，总的下载量为 $n_1|B|\frac{k_2-k_1}{k_1k_2}$。

2.2 $\frac{|\boldsymbol{B}|}{\boldsymbol{k}_1}\boldsymbol{n}_1>\frac{|\boldsymbol{B}|}{\boldsymbol{k}_2}\boldsymbol{n}_2$

定理 2 当$\frac{|B|}{k_1}n_1>\frac{|B|}{k_2}n_2$时，基于码 $\mathscr{C}$ 构造的参数为$(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$，$i=1,2,3$ 的功能性修复 MSR 码可以以最小下载数据量转换为参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。

证明：对于$\frac{|B|}{k_1}n_1>\frac{|B|}{k_2}n_2$，转换之后存储空间变小，存在 5 种可能的参数取值。

(1)当 $n_1=n_2,k_1<k_2$ 时，基于码 $\mathscr{C}$ 构造的参数为$(n_1,k_1,B,\boldsymbol{d}_i,\boldsymbol{t}_i)$，$i=1,2,3$ 的功能性修复 MSR 码中共有 n_1 个物理节点，每个物理节点包含$\frac{LCM(k_1,k_2)}{k_1}$个逻辑节点，从中删除$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点就得到了参数为$(n_2,k_2,B,d_2,t_2)$的功能性修复 MSR 码，删除的数据量为 $n_1|B|\frac{k_2-k_1}{k_1k_2}$。

(2)当 $n_1>n_2,k_1=k_2$ 时，从 n_1 个物理节点任意选择并删除 n_1-n_2 个物理节点就得到了(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码，删除的数据量为$|B|\frac{n_2-n_1}{k_1k_2}$。

(3)当 $n_1>n_2,k_1>k_2$ 时，在这种情况下满足：

$$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)>\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_2 \tag{8}$$

任选 n_1-n_2 个物理节点，一共包括$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)$个逻辑节点，从中任选$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_2$个逻辑节点转移到剩余的 n_2 个物理节点上，让每个目标节点获得$\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}$个逻辑节点上的数据，由此可得参数为$(n_2,k_2,B,d_2,t_2)$的功能性修复 MSR 码，对应的下载量为 $n_2|B|\frac{k_2-k_1}{k_1k_2}$。

(4)当 $n_1>n_2,k_1<k_2$ 时，此$\left(\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}\right)\cdot>0$。首先从 n_1 个物理节点中选择并删除 n_1-n_2 个物理节点；其次对于剩余的 n_2 个物理节点中的每一个物理节点，删除其中的$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点，由此可得参数为$(n_2,k_2,B,d_2,t_2)$的功能性修复 MSR 码，删除的数据量为$|B|\frac{n_1-n_2}{k_1}+n_2|B|\frac{k_2-k_1}{k_1k_2}$。

(5)当 $n_1<n_2,k_1<k_2$ 时，在这种情况下满足：

$$\left(\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}\right)\cdot n_2>\frac{LCM(k_1,k_2)}{k_2}\cdot(n_2-n_1) \tag{9}$$

首先从每个物理节点中选择$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点，一共选择出$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_1$个逻辑节点，把这些逻辑节点中的数据转移到新建的 n_2-n_1 个物理节点中，使得每个目标节点唯一地获得$\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点，根据式(9)，这种转移是可行

的。由此可得参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码,对应的下载量为$|B|\frac{n_2-n_1}{k}$。

上述 5 种情况在码的转换过程中都使用了最小的数据下载量。

2.3 $\frac{|\boldsymbol{B}|}{\boldsymbol{k}_1}\boldsymbol{n}_1<\frac{|\boldsymbol{B}|}{\boldsymbol{k}_2}\boldsymbol{n}_2$

定理 3 当$\frac{|B|}{k_1}n_1<\frac{|B|}{k_2}n_2$时,基于码$\mathscr{C}$构造的参数为$(n_1,k_1,B,d_i,t_i)$,$i=1,2,3$的功能性修复 MSR 码可以以最小下载数据量转换为参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。

证明:对于$\frac{|B|}{k_1}n_1<\frac{|B|}{k_2}n_2$,转换之后存储空间变大,存在 5 种可能的参数取值。

(1)当$n_1=n_2,k_1>k_2$时,需要为每个物理节点补充$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)$个逻辑节点,这相当于一个修复问题:应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个帮助节点修复产生$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_1$个逻辑节点。回忆我们应用码$\mathscr{C}$构造的参数为$(n_1,k_1,B,d_i,t_i)$,$i=1,2,3$的功能性修复 MSR 码能够应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个帮助节点以最小下载数据量修复$\frac{LCM(k_1,k_2)}{k_2}\cdot n_2-\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点,因此可以完成该修复问题。由此我们就得到了参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。该过程使用的下载数据量为$\frac{n_1|B|}{k_1}$。

(2)当$n_1>n_2,k_1>k_2$时,$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)>0$,且有:

$$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_2>\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)\quad(10)$$

首先,从n_1个物理节点中任选n_1-n_2个物理节点,其中包括$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)$个逻辑节点,把这些逻辑节点转移到另外的$n_2$个物理节点上,基于式(10),每个目标物理节点获得不多于$\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}$个逻辑节点。删除所选的$n_1-n_2$个物理节点,系统中只剩余$n_2$个物理节点。其次,我们还需要总数为$\frac{LCM(k_1,k_2)}{k_2}\cdot n_2-\frac{LCM(k_1,k_2)}{k_1}n_1$个逻辑节点,可以应用修复过程获得这些数据,即应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点修复$\frac{LCM(k_1,k_2)}{k_2}\cdot n_2-\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点,由此可得参数为$(n_2,k_2,B,d_2,t_2)$的功能性修复 MSR 码。该过程使用的下载数据量为$\frac{n_1|B|}{k_1}+n_2|B|\frac{k_1-k_2}{k_1k_2}$。

(3)当$n_1<n_2,k_1=k_2$时,需要新建n_2-n_1个物理节点,共包含$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_1-n_2)$个逻辑节点,可以应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点通过修复过程产生。因为码$\mathscr{C}$,即$(n_1\cdot\frac{LCM(k_1,k_2)}{k_1},LCM(k_1,k_2),B,D_i,T_i)$,$i=1,2,3$,可以以最小下载数据量应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点修复$\frac{LCM(k_1,k_2)}{k_1}\cdot(n_2-n_1)$个逻辑节点,所以我们就以最小下载量把$(n_1,k_1,B,d_i,t_i)$,$i=1,2,3$的功能性修复 MSR 码转换为参数为$(n_2,k_2,B,d_2,t_2)$的功能性修复 MSR 码。该过程使用的下载数据量为$\frac{n_1|B|}{k_1}$。

(4)当$n_1<n_2,k_1>k_2$时,一方面需要为已有的n_1个物理节点中的每个物理节点补充$\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}$个逻辑节点;另一方面需要新建$n_2-n_1$个物理节点,并为其中每一个物理节点生成$\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点,因此一共需要$\frac{LCM(k_1,k_2)}{k_2}\cdot n_2-\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点。如前所述,这些逻辑节点可以应用$\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点修复得到。该过程使用的下载数据量为$\frac{n_1|B|}{k_1}$。

(5)当$n_1<n_2,k_1<k_2$时,首先从每个物理节点中任选$\frac{LCM(k_1,k_2)}{k_1}-\frac{LCM(k_1,k_2)}{k_2}$个逻辑节点,一共选出$\left(\frac{LCM(k_1,k_2)}{k_2}-\frac{LCM(k_1,k_2)}{k_1}\right)\cdot n_1$个逻辑节点,把这些逻辑节点平均分配给$n_2-n_1$个新建的物理节点。此外,仍需要$\frac{LCM(k_1,k_2)}{k_2}\cdot n_2-\frac{LCM(k_1,k_2)}{k_1}\cdot n_1$个逻辑节点,可以应用修复过程得到。该过程使用的下载数据量为$\frac{n_1|B|}{k_1}+n_1|B|\frac{k_2-k_1}{k_1k_2}$。

结束语 本文提出了一种可变参数的功能性修复最小存储再生码的构造方法,该方法的关键点在于把逻辑节点和物理节点相结合。应用该方法可以构造一个参数为(n_1,k_1,B,d_1,t_1)的功能性修复 MSR 码,该码可以转换为另一个参数为(n_2,k_2,B,d_2,t_2)的功能性修复 MSR 码。根据转换前后两个码占用的存储空间的比较,分 3 种情况详细讨论了如何进行码的转换,并证明了该转换过程只需要使用最小下载数据量。

参 考 文 献

[1] LIU Y T,LIU H. A Cloud Storage System Based on Network Coding [J]. Computer Science,2018(45):293-298.

[2] DIMAKIS A G,GODFREY P B,WU Y,et al. Network Coding for Distributed Storage Systems [J]. IEEE Transactions on Information Theory,2010,56(9):4539-4551.

[3] DIMAKIS A G,RAMCHANDRAN K,WU Y,et al. A Survey on Network Codes for Distributed Storage [J]. Proceedings of the IEEE,2011,99:476-489.

(下转第 317 页)

[16] LOPATKA J,PEDZISZ M. Automatic modulation classification using statistical moments and a fuzzy classifier[C]// WCC 2000-ICSP 2000. 2000 5th International Conference on Signal Processing Proceedings. 16th World Computer Congress 2000. IEEE,2000:1500-1506.

[17] KIM K,POLYDOROS A. Digital modulation classification: the BPSK versus QPSK case[C]// MILCOM 88,21st Century Military Communications—What's Possible?'. Conference record. Military Communications Conference. IEEE,1988:431-436.

[18] GULDEMIR H,SENGUR A. Comparison of clustering algorithms for analog modulation classification[J]. Expert Systems with Applications,2006,30(4):642-649.

[19] WONG M L D,NANDI A K. Automatic digital modulation recognition using artificial neural network and genetic algorithm [J]. Signal Processing,2004,84(2):351-365.

[20] ZHANG X,GE T. Automatic Modulation Recognition of Communication Signals Based on Instantaneous Statistical Characteristics and SVM Classifier[C]// 2018 IEEE Asia-Pacific Conference on Antennas and Propagation (APCAP). IEEE,2018:344-346.

[21] ASLAM M W,ZHU Z,NANDI A K. Automatic modulation classification using combination of genetic programming and KNN[J]. IEEE Transactions on Wireless Communications, 2012,11(8):2742-2750.

[22] O'SHEA T J,CORGAN J,CLANCY T C. Convolutional radio modulation recognition networks[C]// International Conference on Engineering Applications of Neural Networks. Springer, Cham,2016:213-226.

[23] O'SHEA T J,HITEFIELD S,CORGAN J. End-to-end radio traffic sequence recognition with recurrent neural networks [C]// 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP). IEEE,2016:277-281.

[24] TANG B,TU Y,ZHANG Z,et al. Digital signal modulation classification with data augmentation using generative adversarial nets in cognitive radio networks[J]. IEEE Access,2018,6: 15713-15722.

[25] HARTIGAN J A,WONG M A. Algorithm AS 136: A k-means clustering algorithm[J]. Journal of the Royal Statistical Society. Series C (Applied Statistics),1979,28(1):100-108.

[26] HOCHREITER S,SCHMIDHUBER J. Long short-term memory[J]. Neural Computation,1997,9(8):1735-1780.

[27] O'SHEA T,HOYDIS J. An introduction to deep learning for the physical layer[J]. IEEE Transactions on Cognitive Communications and Networking,2017,3(4):563-575.

CHEN Jin-yin, Ph.D, associate professor. Her main research interests include artificial intelligence security, graph data mining and evolutionary computing.

(上接第 309 页)

[4] HU Y,XU Y,WANG X,et al. Cooperative Recovery of Distributed Storage Systems from Multiple Losses with Network Coding [J]. IEEE Journal on Selected Areas in Communications, 2010,28:268-276.

[5] SHUM K W. Cooperative Regenerating Codes for Distributed Storage System [C]// Proceeding of IEEE International Conference on Communications (ICC). 2011.

[6] KERMARREC A,STRAUB G,SCOUARNEC L. Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes [C]// Proceeding of International Symposium on Network Coding (NetCod),2011.

[7] ZHONG F Y,WANG Y,LI N S. Node Selection Scheme for Data Repair in Heterogeneous Distributed Storage Systems [J]. Computer Science,2019(8):35-41.

[8] WANG L S,TANG X H. Regenerating Codes for New Multi-node Repair Model [J]. Application Research of Computers, 2018(2):527-531.

[9] ZHANG H Y,LI H,ZHU B,et al. Minimum Storage Regenerating Codes for Scalable Distributed Storage [J]. IEEE Acess, 2017,5:7149-7155.

[10] NETANEL R. Asymptotically Optimal Regenerating Codes over Any Field [J]. IEEE Transactions on Information Theory, 2018,64(11):7178-7187.

[11] MARWEN Z,ZHIYING W. Centralized Multi-node Repair Regenerating Codes [J]. IEEE Transactions on Information Theory,2019,65(7):4180-4206.

[12] ZHANG G,ZHENG W,SHU J. ALV: A New Data Redistribution Approach to RAID-5 Scaling [J]. IEEE Transactions on Computers,2010,59:345-357.

[13] ZHANG G,LI K,WANG J,et al. Accelerate RDP RAID-6 Scaling by Reducing Disk I/Os and XOR Operations [J]. IEEE Transactions on Computers,2015,64:32-44.

[14] RAI B K,DHOORJATI V,SAINI L,et al. On Adaptive Distributed Storage Systems [C]// IEEE International Symposium on Information Theory (ISIT). 2015.

[15] RAI B K. Adaptive Erasure Code Based Distributed Storage Systems [C]// IEEE 14th Canadian Workshop on Information Theory. 2015.

WANG Xue-bing, born in 1974, associate professor. His main research interests include network communication, information security, intelligent material association and artificial intelligence.