计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200300124

分布式存储系统中的预测式纠删码研究

张 航 唐 聃 蔡红亮
成都信息工程大学软件工程学院 成都 610225
(1521717495@qq.com)

摘 要 纠删码消耗的存储空间较少,获得的数据可靠性较高,因此被分布式存储系统广泛采用。但纠删码在修复数据时较高的修复成本限制了其应用。为了降低纠删码的修复成本,研究人员在分组码和再生码上进行了大量的研究。由于分组码和再生码属于被动容错方式,对于一些容易出现失效的节点,采用主动容错的方式能更好地降低修复成本,维护系统的可靠性,因此,提出了一种主动容错的预测式纠删(Proactive basic-Pyramid,PPyramid)码。PPyramid码利用硬盘故障预测方法来调整basic-Pyramid码中冗余块和数据块之间的关联,将预测出的即将出现故障的硬盘划分到同一小组,使得在修复数据时,所有的读取操作在小组内进行,从而减少读取数据块的个数,节省修复成本。在基于Ceph搭建的分布式存储系统中,在修复多个硬盘故障时,将PPyramid码与其他常用的纠删码进行对比。实验结果表明,相比basic-Pyramid码,PPyramid码能降低6.3%~34.9%的修复成本和减少7.6%~63.6%的修复时间,相比LRC码、pLRC码、SHEC码、DLRC码,能降低8.6%~52%的修复成本和减少10.8%~52.4%的修复时间。同时,PPyramid码构造灵活,具有很强的实际应用价值。

关键词:分布式存储系统;硬盘故障;数据修复;纠删码;故障预测
中图法分类号 TP302.8

Study on Predictive Erasure Codes in Distributed Storage System

ZHANG Hang,TANG Dan and CAI Hong-liang
School of Software Engineering,Chengdu University of Information Technology,Chengdu 610225,China

Abstract Erasure coding consumes less storage space and obtains a higher data reliability,thus being widely used by distributed storage systems. However,when erasure codes are used to repair data,their high repair costs limit their application. In order to reduce the repair cost of erasure codes,researchers have researched a lot on block codes and regenerative codes. But block codes and regeneration codes are passive fault tolerance. For some nodes that are prone to failure,using active fault tolerance can better reduce repair costs and maintain the system reliability. Therefore,this paper proposes a proactive basic-Pyramid(PPyramid) code. The PPyramid code uses the hard disk failure prediction method to adjust the association between redundant and data blocks in the Pyramid code,divides hard disks that are predicted to fail into the same group,thus making all read operations to be performed within the team when recovering data,thereby reducing the number of read data blocks and saving repair costs. In a distributed storage system based on Ceph,it is compared with other commonly used erasure codes,when repairing multiple hard drives. Experimental results show that,PPyramid codes can reduce repair costs by 6.3%~34.9% and decrease repair time by 7.6%~63.6% compared with basic-Pyramid. Compared with LRC code,pLRC code,SHEC code and DLRC code,it can reduce repair costs by 8.6%~52% and decrease repair time by 10.8%~52.4%. Meanwhile,PPyramid codes are flexible in construction and have strong practical application value.

Keywords Distributed storage system,Hard disk failure,Data repair,Erasure codes,Failure prediction

1 引言

随着网络技术的发展,以文字为主的互联网信息传播手段正在被视频影音取代,这使得各企业的数据中心存储的数据量越来越大。工信部在“数博会”上表示,到2020年,我国数据总量全球占比有望达到20%,我国将成为数据量最大的

到稿日期:2020-03-23 返修日期:2020-07-20 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:四川省科技计划项目(20ZDYF1156);人工智能重大专项(2018GZDZX0030);四川省科技成果转移转化示范项目(2018CC0093)
This work was supported by the Science and Technology Program of Sichuan Province(20ZDYF1156),Major AI Projects(2018GZDZX0030) and Sichuan Science and Technology Achievements Transfer and Transformation Demonstration Project(2018CC0093).
通信作者:唐聃(tangdan@foxmail.com)

国家之一[1]。爆炸式增长的数据对分布式存储系统的安全性和可靠性提出了巨大的挑战。

为了解决以上问题，容错技术被应用于分布式存储系统。目前，常用的容错技术主要分为两类。

第一类是被动容错方式。被动容错方式是分布式存储系统最常见的容错方式，即当节点中某些硬盘的数据失效后，利用其余健康硬盘上的数据进行修复。被动容错方式主要采用多副本[2]和纠删码技术[3]。多副本技术操作简单，易于实现，但需要消耗大量的存储空间，存储效率低。纠删码技术存储效率高，灵活性好，但修复成本过高。

目前，围绕降低纠删码修复成本的研究主要集中在两个方向：一个是分组码，一个是再生码。分组码主要包含层次分组码和交叉分组码。微软的分布式存储系统采用了增加局部冗余的方式来降低修复成本的 LRC（Locally Repairable Code）层次分组码[4]。同时 Huang 等在 LRC 码的基础上提出了 Pyramid 层次分组码[5]，同样利用了局部冗余的方法。Facebook 在自家的分布式存储系统中采用了 LRCs[6] 和 EX-Pyramid 层次分组码[7]。LRCs 利用局部冗余块能生成全局冗余块的思想，减少了全局冗余块的个数，相比 LRC 码降低了存储成本。EXPyramid 码在 Pyramid 码上增加了纵式局部冗余块，进一步降低了数据修复时的修复成本。Meng 等提出了 DLRC（Dynamic Local Reconstruction Code）层次分组码[8]，DLRC 码通过设定参数动态地调整存储开销和重构开销之间的平衡来降低修复成本。Miyamae 等提出了 SHEC（Shingled Erasure Code）码[9]，该码是一种典型的交叉分组码，将各个分组像屋顶瓦片式的相互重叠，增强数据块与冗余块之间的线性关系，从而减少修复时读取块的个数，以降低修复成本。WEAVER 交叉分组码[10]在数据块和冗余块之间形成互相关联关系，通过搜索的方法来查出最佳的组合放置关系，从而减少修复成本。交叉分组码虽然在一定程度上降低了修复成本，但也相应地增加了存储开销，如 WEAVER 码的存储开销极大，空间利用率不超过 50%，且某些分组码在丢失数据块个数增加的情况下，其数据的修复成本会急剧增加，不能满足分布式存储系统的需求。再生码的研究主要分为两大类：MBR（Minimun Bandwidth Rrgenerating）码[11]和 MSR（Minimun Storage Regenerating）码[12]。MSR 码主要拥有最小的存储开销，MBR 码主要拥有最低的数据修复成本。再生码虽然在一定程度上降低了数据的传输成本，但在修复的过程中相比传统纠删码需要在更多的节点硬盘中读取数据，导致其数据读取量较多。此外，再生码编码复杂度高，灵活度不够，不能很好嵌入到分布式存储系统中。

第二类容错技术是主动容错方式。主动容错方式就是实时收集分布式存储系统中各节点硬盘大量的属性数据信息，并建立相应的硬盘故障预测模型，监测各硬盘的运行情况，对即将出现的硬盘故障进行处理，消除潜在的故障隐患，以提高存储系统的可靠性。

随着数据量的增大，分布式存储系统中的节点数量增加，导致硬盘数量越来越多，硬盘故障的频率也随之越来越高。据调查显示，硬盘故障占所有节点故障的 78%[13]，因此，解决好硬盘故障问题就能很好地维护存储系统的稳定性。目前，存储服务器中的硬盘都采用 SMART 技术，可以实时查看硬盘的各种状态。当某些状态超过其硬盘预设的阈值时，存储系统就会发出警报，提示硬盘即将出现故障[14]。

将纠删码技术与主动容错方式相结合能很好地降低纠删码的修复成本，提高存储系统的可靠性。目前，已有相关文献将纠删码与硬盘的故障预测技术相结合，如 Li 等把根据硬盘预测出的即将出现故障的硬盘中的数据复制备份到另一个健康的硬盘中[15]。虽然该方法有效地降低了数据的修复成本，但极大地增加了存储消耗。Hu 等[16]利用硬盘故障预测技术将 LRC 码划分成不均等的小组，从而解决了不均匀的故障修复问题。Zhang 等将 LRC 码与硬盘故障预测模型相结合，把即将出现故障的硬盘划分到组内长度较小的小组内，从而在修复数据时降低读取健康硬盘中的数据量，以降低修复成本[17]。上述运用了硬盘故障预测技术的纠删码大多利用了 LRC 码的分组特性，但 LRC 码中每个小组只能最优修复一个故障中的硬盘数据，并不能同时修复多个硬盘数据。因此，本文考虑将硬盘故障预测技术与分组码中能在小组内修复多个故障的纠删码相结合。在众多分组码中，Pyramid 码构造灵活，可以通过调节参数的形式使得小组内能够修复多个故障，且其本身具有低修复成本的特性，在分布式存储中有较广的应用场景。因此，本文将主动容错技术与 Pyramid 码的 basic-Pyramid 码相结合，提出了一种新的预测式纠删码 PPyramid 码。该码可以针对多个硬盘故障，动态地调整原始数据块和冗余块之间的关联，将预测出即将出现故障的硬盘的节点划分到同一小组中，使得在恢复数据时，所有的修复读取操作在小组内进行，从而进一步降低读取块的个数，节省修复成本，维护系统稳定性，同时在 basic-Pyramid 码的编码结构上不增加额外的存储空间。

本文第 2 节介绍文中所使用技术的相关概念；第 3 节对本文提出的 PPyramid 码的具体编译码方法进行详细阐述，并给出具体实例说明；第 4 节对 PPyramid 码进行量化分析，分析其可靠性和修复能力；第 5 节对 PPyramid 码进行修复成本、修复时间和更新成本的相关实验，并与其他典型的纠删码进行对比；最后总结全文。

2　相关概述与问题定义

2.1　纠删码的相关概述

纠删码常用 (n,k) 表示，是将大小为 M 的数据划分成 k 个原始数据块，k 个原始数据块经过相应的编码算法总共得到 n 个块，这一过程被称为纠删码的编码。当某些块丢失时，若丢失的块个数小于该纠删码的最大容错个数，则可以选取其中剩余的块进行计算并修复出丢失的块数据，这一过程被称为纠删码的解码。

为便于理解，本文给出了一些纠删码常用的基本概念的说明和定义，如下所示。

（1）系统性纠删码：数据块经过计算产生的块中包含原始

的数据块，因其良好的访问性能，成为分布式存储系统的首选。

(2)容错度：纠删码可以容忍的任意块丢失的最大个数。

(3)原始数据块：用户上传的原始数据对象被系统划分后得到的块。

(4)冗余块：数据块经过纠删码算法后产生的所有块。

(5)全局冗余块：条带内所有原始数据块通过相应的编码算法得到的块。

(6)局部冗余块：原始数据块被分成若干个小组，每个小组内的原始数据块通过相应的编码算法得到的块。

(7)条带：由多个数据块和冗余块组成，且满足同一纠删码算法。

(8)数据修复：当分布式存储系统中因某些硬盘故障导致数据丢失时，可以通过从条带内剩余的健康硬盘获取的块数据来恢复出丢失的数据的过程。

2.2 硬盘故障预测技术的相关概念

目前，几乎所有的硬盘都支持 SMART 技术，而主动容错技术一开始就是在以 SMART 技术预测硬盘故障模型的基础上建立的。当模型预测硬盘即将出现故障而发出警报时，备份数据或者良好的容错机制能很好地保证分布式存储系统的稳定性。

硬盘故障预测模型基于大量的硬盘数据进行预测，其收集的每条硬盘数据都包含了很多属性，不同公司生产的硬盘，其收集的数据的属性个数也不尽相同，但收集的属性大多是一样的，如盘片启动时间、信道读取余量、寻到错误率、设备开关次数等。在硬盘故障预测模型中，为了使预测结果准确，降低模型的算法复杂度，常常会删除一些无用的属性，只保留一些有用的基本属性作为系统模型的输入特征。常用的属性如表 1 所列。

表 1 SMART 常用属性表

Table 1 SMART common attributes

序号	属性名称	属性含义
1	Spin Up Time	盘片启动时间
2	Seek Error Rate	寻道错误率
3	Reallocated Sector Count	重定位磁区次数
4	read error rate	底层数据读取错误率
5	Power-On Hours	硬盘加电时间
6	Reported Uncorrectable Errors	报告不可纠正错误
7	High Fly Writes	磁头写入高度
8	Temperature	温度
9	Hardware ECC Recovered	硬件 ECC 恢复次数
10	Current Pending Sector Count	等候重定的扇区次数

一般基于硬盘故障的预测模型都是二分类模型，预测模型返回的结果只能是会发生故障或不会发生故障。硬盘故障预测模型中[18-20]总共有 3 个评价指标，分别是准确率或召回率(False Discovery Rate，FDR)、误报率(False Alarm Rate，FAR)和提前预测时间(Time In Advance，TIA)。FDR 是故障硬盘中可以被准确预测出的比例；FAR 是好盘中被误报为坏盘的比例；TIA 描述的是可以提前多长时间预测出即将到来的故障。这 3 个指标的数值直接描述了一个预测模型的好坏。

2.3 问题定义

为了便于理解，本节给出了分布式存储系统中的纠删码相关问题的定义。

定义 1(可靠性) 对于分布式存储系统，可靠性描述了存储系统保障数据安全、提供数据访问的能力。如果数据丢失，那么将会给用户以及企业都带来不可估量的损失。因此可靠性是分布式存储系统重要的指标之一。

定义 2(修复成本) 修复成本指纠删码在进行数据修复过程中需要读取的数据量。在分布式存储系统中，各节点之间进行着频繁的数据访问操作，如果数据修复时所读取的数据量过大，就会造成存储系统的其余数据访问任务受到影响，从而威胁到系统的稳定性。因此在数据修复中纠删码读取的数据量应尽可能小。

定义 3(修复时间) 纠删码在进行数据修复时所消耗的时间长短。在分布式存储系统中，数据修复时会消耗存储服务器计算、内存等资源。如果修复时间过长，就会长时间地占据这些资源，增大系统运行负荷，从而可能导致其余健康硬盘中的数据失效。因此，对于分布式存储系统中的纠删码来说，修复时间越短越好。

定义 4(更新成本) 纠删码在进行编码后，根据硬盘故障预测模型给出的结果，调整纠删码分组时需要读取的数据量。更新成本越小，对系统的影响就越小，更能保证存储系统稳定运行。

3 PPyramid 的设计

由于 PPyramid 码的设计是在 basic-Pyramid 码的基础上进行优化并改进的，本文首先介绍基本的层次分组码 basic-Pyramid 码，然后介绍本文硬盘故障算法的选择，最后介绍 PPyramid 编码算法的设计并举例说明。

3.1 basic-Pyramid 码

根据 basic-Pyramid 码的编码规则，在条带中的全部原始数据块编码产生全局冗余块后，把原始数据块分成几个小组，每个组内编码生成局部冗余块，其中全局冗余块可以由每个小组相应的局部冗余块计算所得。图 1 给出了(11，8)的 RS 码结构。条带中总共含有 $D_1, D_2, \cdots, D_8$，共计 8 个原始数据块。8 个原始数据块经过编码产生 P_1, P_2, P_3 这 3 个全局冗余块。图 2 给出了(12，8)的 basic-Pyramid 码结构，条带中的 8 个原始数据块分成两个小组，其中 D_1, D_2, D_3, D_4 为一个小组，D_5, D_6, D_7, D_8 为一个小组。每个小组仅使用组内的原始数据块编码生成 P_{11}, P_{12} 两个局部冗余块，其中，$P_{11}+P_{12}=P_1$。当组内丢失的块数据个数少于局部冗余块的个数时，读取组内的块数据即可修复，即减少了修复成本。

其生成公式如下，其中，g_i 是与 D_i 在同一有限域中的生成矩阵 $\boldsymbol{G}$ 中的元素。

$$p_1=\sum_{i=1}^{i=8}\boldsymbol{D}_i g_i \tag{1}$$

$$p_{11}=\sum_{i=1}^{i=4}\boldsymbol{D}_i g_i \tag{2}$$

$$p_{12}=\sum_{i=5}^{i=8}\boldsymbol{D}_i g_i \tag{3}$$

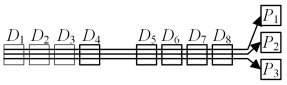

图 1　(11,8)的 RS 码结构图

Fig. 1　(11,8) RS code structure diagram

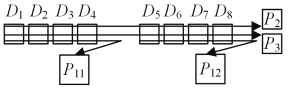

图 2　(12,8)的 basic-Pyramid 码结构图

Fig. 2　(12,8) basic-Pyramid code structure diagram

3.2　硬盘故障预测模型算法选择

SMART 技术通过预先设定阈值的方式来判断硬盘的好坏，该方法技术简单，但不能非常有效地预测出硬盘的故障。实验数据结果表明，SMART 虽然能够达到 0.1%的低故障误报率，但是却只能预测出 3%～10%的硬盘故障，预测准确率非常有限。

为了提高硬盘故障的检测率，目前，大量的研究工作使用统计学或者机器学习的方法来建立硬盘故障预测模型。比如 Murray 等[21]基于 SMART 属性，利用贝叶斯分类方法使得硬盘故障预测的误报率低至 1%，准确率达到了 55%。Hamerly 等[19]利用统计学与秩和检验的方法实现了 0.5%的误报率，准确率达到 60%。Zhu 等采用人工神经网络[18]和决策树[22]的方式，能达到 0.1%的误报率和 95%的准确率。

一个良好的硬盘故障预测模型应该具有良好的硬盘故障预测准确率，同时误报率不应过高。实验使用 Hughes 等开源的 SMART 数据，该数据包含 178 块健康的硬盘和 191 块故障磁盘，并选择了目前硬盘故障技术中最常见的 3 种方法，即人工神经网络、决策树、支持向量机来分别测量其误报率和准确率。其中，人工神经网络按照文献[18]的方式，使用了 3 层 BP 神经网络，3 层的节点数分别为 19，30，1，其中，隐层和输出层都使用 sigmoid 函数作为激活函数。将最大迭代数设置为 400，学习率设置为 0.1。决策树按照文献[22]的方式，使用 ID3 算法生成决策树，其中，minsplit(最小分支节点数)设置为 20，minbucket(树中叶节点包含的最小样本数)设置为 7，complexity parameter(单个节点的复杂度)设置为 0.001。

测试中，对 SMART 数据集上的 10 个特征进行采样，10 个特征如表 1 所列。将采样中的数据集随机拆分成训练数据集、验证数据集和测试数据集，这 3 个数据集分别占总数据集的 70%，20%和 10%，其测试性能如表 2 所列。

表 2　3 种方法的性能比较

Table 2　Performance comparison of three methods

(单位：%)

算法	准确率	误报率
人工神经网络	94	2.64
决策树	93	0.62
向量积	48.6	0.21

由表 2 可知，这 3 种方法中，人工神经网络和决策树能够达到较好的准确率和误报率，但决策树预测硬盘故障模型生成的规则更易于理解，且能够清晰地指出硬盘故障发生的原因，使得模型可以有针对性地解决问题。因此，选择决策树作为实验的基本硬盘故障预测模型。

3.3　PPyramid 算法的设计

为便于理解 PPyramid 码编码过程，本文对频繁使用的符号进行解释，具体如表 3 所列。

表 3　常用符号

Table 3　Common symbols

符号	含义
n	一个条带中块的总个数
k	条带中原始数据块的个数
l	条带中原始数据块划分成小组的个数
r	每个小组生成局部冗余块的个数
m	全局冗余块个数
G	编码矩阵
g_i	编码矩阵中的有限域元素

PPyramid 码的算法设计过程如下：

步骤 1　按照 basic-Pyramid 码的编码结构，将原始数据块划分成不同的小组，生成全局冗余块和局部冗余块，并将它们存储到不同的硬盘上。

步骤 2　根据硬盘故障预测模型得出的结果，将分布式存储系统所有硬盘包含的块数据划分成两个状态，即健康块和差块，健康块为故障预测模型预测暂时不会出现故障的硬盘中的块数据，差块为即将出现故障的硬盘中的块数据。

步骤 3　根据两种情况调换块的位置。

情况 1　选定含差块最多的一个小组，将差块移动到选定的小组内并替换该小组内的健康块，同时将该小组内被替换的健康块移动到原来差块所在的位置。当一个小组内的差块个数等于局部冗余块的个数时，再次选定剩下小组中含差块最多的一个小组，将剩余差块移动到选定的小组中，依次循环，直至用最少个数的小组包含所有的差块。同时对有变化的小组重新生成局部冗余块。

情况 2　选定含差块最多的一个小组，当选定的小组本身含差块的个数已经超过自身的局部冗余块个数时，将超过的差块移动到下一个选定的小组中，直至下一个选定小组内的差块等于组内局部冗余块的个数，再选定下一个小组，依次循环，直至用最少个数的小组包含所有的差块，且每组的差块个数不多于组内局部冗余块的个数。同时对有变化的小组重新生成局部冗余块。

步骤 4　当差块变成丢失块后，读取组内的块数据，修复故障盘丢失的块数据。

步骤 5　当修复丢失的块数据后，将之前移动过的坏块和健康块还原到初始的各小组的位置上。

详细的算法如算法 1 所示。

算法 1　PPyramid 码的设计算法

数组介绍。err_loc：记录条带中差块所在位置的数组；read_loc：需要读取的块在条带集中的位置的数组；read_buff：读数据缓存数组，write_buff：写数据缓存数组。

过程：

```
1. 系统初始执行 Pyramid_encode();
2. err_loc [] = pre_model();
3. for i in range(len(err_loc))
4.     exchange_loc(i);
```

```
5. 更新局部冗余块 group_encode();
6. for i in range(len(err_loc))
7.     if i==FALSE:
8. 记录修复所需的块列表 Read_loc []=min_list_repair();
9.         read_buff:[ ] =Loc_data(Read_loc[]);
10.        write_buff:[]=Loc_decode(read_buff:[ ])
11.        更新 err_loc [];
12.        Re_exchange_loc(i)
13. End for
```

系统一开始执行 1 中的初始编码，将编码后的数据传到各个硬盘。2 执行硬盘故障预测函数，将预测为差块的位置传给 err_loc 数据。3－4 调整差块在各小组内的位置。6－7 判断差块是否变成丢失块。8 获取修复丢失数据需要读取健康块所在的条带位置。9 读取健康块中的数据保存在 read_buff 数组。10 将修复出的数据传入 write_buff 数组。12 将修复后的差块还原到初始小组的位置。

下面从(11,6)的 basic-Pyramid 码出发，构造 PPyramid 码编码过程。其中，白色方框代表健康块，灰色方框代表差块。

如图 3 所示，(19,12)的 Pyramid 码将条带原始数据块划分成每组包含 4 个原始数据块的 3 个小组，每小组各生成 2 个局部冗余块，整个原始数据块编码生成一个全局冗余块。

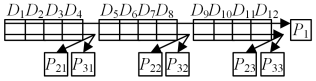

图 3 (19,12) PPyramid 码图

Fig. 3 (19,12) PPyramid diagram

假设如图 4 所示，执行硬盘故障预测模型算法，得出 D_2，D_6 为条带中的差块。

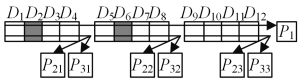

图 4 2 个差块所在位置图

Fig. 4 Location diagram of 2 error blocks

如图 5 所示，首先选定一个包含差块最多的小组，第一个小组和第二个小组含差块个数一样，于是随机选取一个小组。选取第一个小组，将条带内的第二个小组中的 D_6 差块与第一个小组中的 D_1 健康块互换，使得所有的差块在同一个小组。由于 2 个小组的数据块发生变化，2 个小组重新生成局部冗余块 Q_{21}，Q_{31}，Q_{22}，Q_{32}。若第一个小组的差块因为硬盘故障真正变成丢失块，则只需读取第一个小组内的 D_3，D_4 数据块和 Q_{21}，Q_{31} 局部冗余块，总共 4 个块数据即可修复 D_2，D_6 这 2 个丢失块。而 basic-Pyramid 码修复 2 个丢失块需要分别在 2 个小组各读取 4 个块数据，总共 8 个块数据才可以修复丢失块。因此，PPyramid 码相比 basic-Pyramid 码可减少 50%的修复成本。

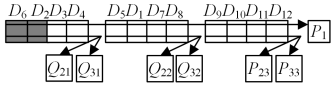

图 5 调整 2 个差块后的结构图

Fig. 5 Structure diagram after adjusting 2 error blocks

4 理论量化分析

本节主要对 PPyramid 码和 basic-Pyramid 码的可靠性和修复能力进行分析和比较。

4.1 可靠性分析

可靠性的好坏反映了一个分布式存储系统提供数据访问服务、保证数据安全的能力。工业界常使用平均无故障时间(Mean Time To Failure，MTTF)来衡量某种产品的可靠性。存储领域通常使用平均丢失时间(Mean Time To Data Loss，MTTDL)来反映一个分布式存储系统的可靠性。根据文献[13]，硬盘的寿命是服从指数分配的，因此本文默认 MTTF 和 MTTDL 也服从指数分配。一般来说，基于某种编码的分布式存储系统的 MTTDL 不仅与编码方法有关，更与单个存储节点的修复率 μ 和单个节点的故障率 λ 有关。修复率 μ 越高，MTTDL 就越大，存储系统的可靠性就越好。λ 越高，MTTDL 就越小，存储系统就更容易丢失数据，其可靠性就越低。以往类似的方法[23-25]使用马尔可夫模型进行硬盘的可靠性建模，并以此来推导出分布式存储系统的 MTTDL。虽然该马尔可夫模型不能完全地反映出两个码的可靠性，但能起到一定的见解作用。

我们分别以(11,6)的 basic-Pyramid 码和 PPyramid 码建立马尔可夫模型。其中，(11,6)basic-Pyramid 码和 PPyramid 码都将条带分成 2 个小组，每个小组包含 3 个原始数据块，每个小组在生成 2 个局部冗余块后再生成 1 个全局冗余块。建立的马尔可夫转换过程如图 6、图 7 所示。

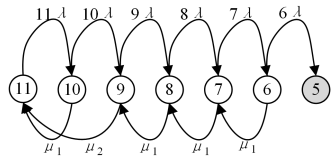

图 6 basic-Pyramid 码的马尔可夫模型

Fig. 6 Markov model of basic-Pyramid codes

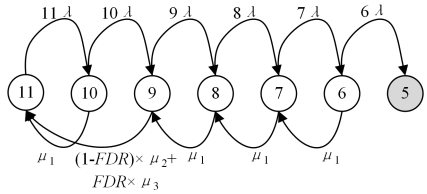

图 7 PPyramid 码的马尔可夫模型

Fig. 7 Markov model of PPyramid codes

由图 6、图 7 可知，共计 11 个硬盘，其状态上的数字代表目前拥有的健康硬盘数量。其中，灰色的状态代表目前有 6 个硬盘出现了故障，已经不能恢复出硬盘丢失的数据。

在马尔可夫正方向的转换过程中，用 λ 表示单硬盘故障概率，因此状态 i 到状态 $i-1$ 的状态转换概率为 $i\lambda$。

在马尔可夫反方向的转换过程中，μ 表示状态 i 到状态 $i+1$ 的修复率。其中，μ_1 表示 basic-Pyramid 码单硬盘故障修复率，μ_2 表示 basic-Pyramid 码双硬盘故障修复率，μ_3 表示 PPyramid 码双硬盘故障修复率。对于 PPyramid 码来说，从

状态 9 到状态 11 的过程中，因为其使用了硬盘预测技术，将所有的即将出现故障的硬盘移动到了所有的小组，所以其转换概率为$(1-FDR)\times\mu_2+FDR\times\mu_3$。

$$MTTDL=\frac{\mu^m+P_1(n)\mu^{m-1}\lambda+\cdots+P_{m-1}(n)\mu\lambda^{m-1}+P_m(n)\lambda^m}{\prod_{i=0}^{m}(n-i)\lambda^{m+1}} \tag{4}$$

式(4)为 MTTDL 的计算公式。由于在编码的分布式存储系统中修复率远大于故障率，可以做近似计算，将式(4)简化为：

$$MTTDL\approx\frac{\mu^m}{\prod_{i=0}^{m}(n-i)\lambda^{m+1}} \tag{5}$$

按如下方式进行 MTTDL 的计算：

假设节点硬盘无故障时间 MTTF 为 4 年，则 $1/\lambda=4$，设分布式存储系统的可用带宽为 γ，总共有 N 个节点，每个节点硬盘的存储量为 M。

对于 μ_1 来说，其修复 1 个节点硬盘故障的修复成本 C_1 为：

$$\frac{1}{11}\times6+\frac{10}{11}\times3=\frac{36}{11}$$

因此可以得出修复率 μ_1 为：

$$\mu_1=\gamma\frac{(N-1)\times11}{36M} \tag{6}$$

对于 μ_2 来说，其修复 2 个节点硬盘故障的修复成本 C_2 为：

$$\frac{20}{55}\times3+\frac{25}{55}\times6+\frac{10}{55}\times9=\frac{60}{11}$$

因此可以得出修复率 μ_2 为：

$$\mu_2=\gamma\frac{(N-1)\times11}{60M} \tag{7}$$

对于 μ_3 来说，其修复 2 个节点硬盘故障的修复成本 C_3 为：

$$\left(\frac{20}{55}+\frac{25}{55}\right)\times3+\frac{10}{55}\times9=\frac{45}{11}$$

因此可以得出修复率 μ_3 为：

$$\mu_3=\gamma\frac{(N-1)\times11}{45M} \tag{8}$$

使用 $M=16$ TB，$\gamma=1$ GBPS，$N=400$ 来分别对 PPyramid 码和 basic-Pyramid 码进行 MTTDL 的计算，结果如表 4 所列。

表 4　MTTDL 比较

Table 4　MTTDL comparison

纠删码	MTTDL
basic-Pyramid	2.67×10^{11}
PPyramid	4.68×10^{11}

PPyramid 码使用了硬盘故障预测技术，使得数据修复只在同一个小组内进行，能大幅提高系统的稳定性。由表 4 可知，PPyramid 码相比 basic-Pyramid 码，其 MTTDL 提高了约 75%。

4.2　修复能力

修复能力是衡量一个纠删码性能的重要指标之一。本小节中用于测试纠删码修复能力的方法如下：

假设一个参数为(n,k)的纠删码，在分布式存储系统中，由于硬盘发生故障，其失效的节点个数为 x，根据概率学，其共有$\binom{n}{x}$种失效方式，在这些失效方式中，其能够修复的失效方式和全部失效方式的比值则为该纠删码失效 x 节点的修复能力。

当发生多个硬盘故障时，由于硬盘故障技术的加持，针对 basic-Pyramid 码，如果同一个小组内包含太多失效节点，那么会导致无法修复。而 PPyramid 码因为硬盘故障技术的加持，可以事先把失效节点平均分配在不同的小组中，使得原本无法修复的节点可以重新被修复。如(13,8)的参数下，在修复 4 个节点失效的情况下，假设 4 个失效节点全部集中在一个小组内，则 basic-Pyramid 码无法修复。但是 PPyramid 码可以把其中 2 个失效节点移动到另一个小组中，使得修复依然可以进行。

表 5 列出了(13,8)参数下的 PPyramid 码和 basic-Pyramid 码分别在 1,2,…,5 个硬盘故障节点失效时的修复能力。

表 5　修复能力对照

Table 5　Repair ability comparison

（单位：%）

纠删码	不同节点数失效时的修复率				
	1	2	3	4	5
basic-Pyramid 码	100	100	100	90.2	64.10
PPyramid 码	100	100	100	100	100

由表 5 可知，PPyramid 码在有 4 个节点和 5 个节点失效时，其修复能力明显大于 basic-Pyramid 码，这表明 PPyramid 码拥有较强的多硬盘故障修复能力。

5　实验结果与分析

本节在真实的分布式存储环境下测试 PPyramid 码的修复成本和修复时间，并将这些数据与其余常见的纠删码进行对比。

5.1　实验环境

为了在真实的分布式环境下对 PPyramid 码的各个方面进行测试，本文基于 Ceph 分布式存储系统搭建了纠删码测试平台。

Ceph 是一个可靠、自动重均衡、自动恢复的分布式存储系统。Ceph 的核心组件包括 Ceph OSD，Ceph Monitor 和 Ceph MDS。其中，OSD 节点负责存储数据并恢复数据、平衡数据、与其他 OSD 间进行心跳检查等，并将一些变化情况上报给 Ceph Monitor。当 Ceph 接收客户端上传的数据时，数据首先被传到 Primary OSD 上，然后再由 Primary OSD 将其写到其余的 osd 上。Monitor 负责监视集群的健康状态以及控制集群节点的相关操作。MDS 主要保存文件系统服务的元数据。

基于 Ceph 搭建的纠删码测试平台体系结构如图 8 所示。

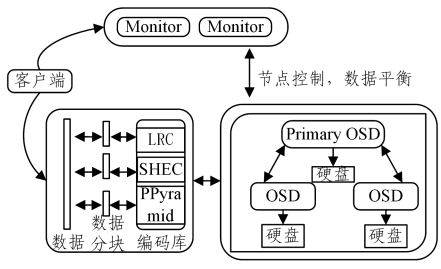

图 8　纠删码测试平台体系结构示意图

Fig. 8　Schematic diagram of erasure code test platform structure

实验使用的纠删码测试平台共包含 25 个节点，除 1 个客户端和 2 个 Monitor 节点以外，其余节点都被作为 OSD 的存储节点。每个节点配置酷睿 i5-5200U 2.2 GHz 处理器、32GB 内存、500g 固态硬盘和 1GBps 以太网卡；每个节点均运行 Centos 7.5 系统，且都安装 Python 3.0 以及 ceph 12 分布式存储系统。

5.2 具体实现

从 Ceph0.8 版本开始，用户可以访问 Ceph 提供的自带纠删码库，实现一些常用的纠删码，如 RS 码、LRC 码等。在 Ceph 中，为了将纠删码逻辑与 OSD 逻辑分开，RADOS 提供了一个纠删码插件接口，允许第三方开发人员提供自己的纠删码，并使用该接口实现与 RADOS 的无缝集成。因为 Ceph 中自带的纠删码主要是一些传统纠删码，所以 Ceph 提供的插件接口不能很好地区分需要修复的丢失冗余块和原始数据块。当 RADOS 需要修复因硬盘故障而丢失的块时，它首先使用自带的 decode 函数对所有 k 个原始数据块进行解码，然后使用 encode 函数生成丢失的块，且不允许使用访问部分块来修复丢失的块。

因此，本文利用 Ceph 提供的 librados，在 Ceph 的基础上扩展了纠删码的编解码流程。扩展的编解码流程是对 Ceph 现有纠删码功能的一个扩展，适用于目前绝大多数的纠删码，如 SHEC 码，RDP 码，basic-Pyramid 码等。同时在现有的纠删码库中重新设计了第三方纠删码库，用于存储利用扩展后纠删码的编解码流程实现的纠删码。

扩展流程中主要新增了以下 3 个功能。

(1)-min_list_repair()：根据丢失块的 ID，寻找到修复该块所需块的最少 ID 列表。

(2)-repair()：给定丢失块的 ID，以及由 min_list_repair 返回的列表，重新生成丢失的块。

(3)-plug_erasure_code()：给定纠删码 ID，返回第三方纠删码库中的具体的编译码方法。若在第三方纠删码库中没有找到请求的插件库，则加载原始纠删码库中默认的纠删码。

5.3 实验对比指标和方法

实验总共分为 4 个部分。第一部分为 PPyramid 码与 basic-Pyramid 码的横向对比，分别对比其编码时间、修复成本和修复时间；第二部分为 PPyramid 码与同样使用了硬盘故障预测技术的最新 pLRC 码的横向对比，分别对比其修复成本和修复时间；第三部分为 PPyramid 码与目前分布式存储系统中常用的部分纠删码，如 LRC 码、SHEC 码、DLRC 的横向对比，分别对比其修复成本和修复时间；第四部分为测试 PPyramid 码的更新成本。

以上实验中，测试文件上传数据大小为 1GB，数据块默认分块大小为 8MB。

5.4 PPyramid 码与 basic-Pyramid 码横向对比

5.4.1 平均编码时间

图 9 给出了 PPyramid 码编码所消耗的时间。由图 9 可知，在不调整块的位置时，其时间即为 basic-Pyramid 码编码的时间。在进行硬盘故障预测后，因为无论调整块个数是多少，其都会统一进行组内重新编码以生成新的局部冗余块，所以其调整块位置所消耗的时间并不随着调整块个数的增加而增加，始终保持在一个相对稳定的值。其调整块内位置所消耗的时间相比不调整块的位置所消耗的时间增加了约 48%，但其调整块位置后所消耗的时间并不是集中在一开始数据上传后统一的编码时间段内，而是当分布式系统检测到系统空闲才开始进行的，因此并不会造成系统 CPU、内存等资源的竞争。

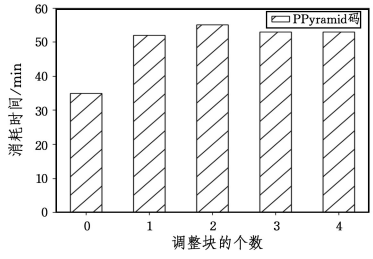

图 9 编码时间图

Fig. 9 Coding time

5.4.2 平均修复成本对比

图 10 给出了(13,8)参数下的 PPyramid 码和 basic-Pyramid 码的修复成本对比。其中，PPyramid 码和 basic-Pyramid 码都产生了 1 个全局冗余块，同时将原始数据块分为 2 个小组，每组产生 2 个局部冗余块。如图 10 所示，横坐标表示硬盘故障的个数，纵坐标表示修复时数据的读取量。由图 10 可知，当 2～3 个节点发生故障时，相比 basic-Pyramid 码，PPyramid码能有效降低数据修复成本约 6.32%～34.7%。当单节点硬盘故障和硬盘故障个数超过局部冗余块个数时，丢失块无法移动，其修复成本和 basic-Pyramid 码的修复成本一致。

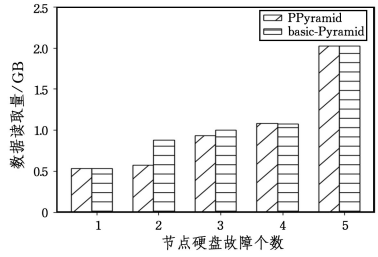

图 10 (13,8)参数下的 PPyramid 码和 basic-Pyramid 码的平均修复成本对比

Fig. 10 Comparison of average repair cost between PPyramid and basic-Pyramid with (13,8) parameter

图 11 给出了(15,8)参数下的 PPyramid 码和 basic-Pyramid 码的修复成本对比。其中，PPyramid 码和 basic-Pyramid 码都产生了 1 个全局冗余块，同时将原始数据块分为 2 个小组，每组产生 3 个局部冗余块。如图 11 所示，横坐标表示硬盘故障个数，纵坐标表示修复时数据的读取量。由图 11 可知，相比(13,8)参数，当发生多硬盘故障时，PPyramid 码的局部冗余块越多，其能更加有效地将丢失块移动到小组内去修复，从而有效地减少修复成本。如图 11 所示，当 2～4 个节点发生故障时，相比 basic-Pyramid 码，PPyramid 码能有效降低数据修复成本约 8.5%～26.7%。

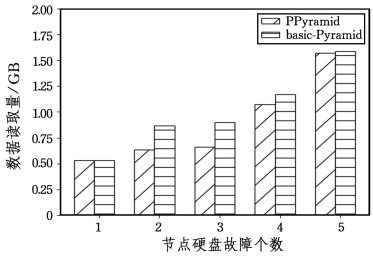

图 11　(15,8)参数下的 PPyramid 码和 basic-Pyramid 码的平均修复成本对比

Fig. 11　Comparison of average repair cost between PPyramid and basic-Pyramid with (15,8) parameter

图 12 给出了(19,12)参数下的 PPyramid 码和basic-Pyramid 码的修复成本对比。其中，PPyramid 码和 basic-Pyramid 码都产生 1 个全局冗余块，同时将原始数据块分为 3 个小组，每组产生 2 个局部冗余块。如图 12 所示，横坐标表示硬盘故障个数，纵坐标表示修复时数据的读取量。相比(13,8)参数，当其多硬盘发生故障时，PPyramid 码的分组越多，其越能有效地减少修复成本。当 2～5 个节点发生故障时，相比 basic-Pyramid 码，PPyramid 码能有效降低数据修复成本约 11.5%～34.9%。

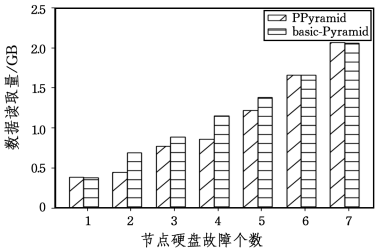

图 12　(19,12)参数下的 PPyramid，basic-Pyramid 平均修复成本对比图

Fig. 12　Comparison of average repair cost between PPyramid and basic-Pyramid with (19,12) parameter

5.4.3　平均修复时间对比

图 13 为(13,8)参数下 PPyramid 码和 basic-Pyramid 码的多节点硬盘故障失效的平均修复时间对比。图中的横坐标表示硬盘故障个数，纵坐标表示修复所花费的时间。由于 PPyramid 码的大部分修复都能在组内进行，其数据量读取小，修复时间与 basic-Pyramid 相比消耗时长较小。从图中可以看出，在多节点硬盘故障个数为 2～3 时，PPyramid 码比 basic-Pyramid 的修复时间约减少 7.6%～63.6%。

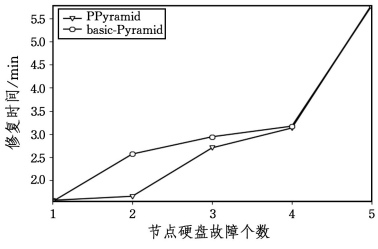

图 13　PPyramid，basic-Pyramid 平均修复时间对比图

Fig. 13　Comparison of PPyramid and basic-Pyramid average repair time diagram

5.5　PPyramid 码与 pLRC 码的横向对比

5.5.1　平均修复成本对比

图 14 给出了(13,8)参数下的 PPyramid 码和 pLRC 码的修复成本对比。其中，PPyramid 码产生 1 个全局冗余块，同时将原始数据块分为 2 个小组，每组产生 2 个局部冗余块。pLRC 码产生 3 个全局冗余块，将原始数据块分成 2 个小组，每组产生 2 个局部冗余块。如图所示，横坐标表示硬盘故障个数，纵坐标表示修复时数据的读取量。当发生单硬盘故障时，pLRC 码将失效块所在小组变成(2,1)的编码形式，使得修复成本较小，但随着硬盘故障个数的增加，此编码方式不再适用于多节点，因此其修复成本大幅度上升。相反，PPyramid 码在发生多节点硬盘故障时能有效地降低修复成本。当 2～4 个节点发生故障时，相比 pLRC 码，PPyramid 码能有效降低数据修复成本约 37.9%～52%。

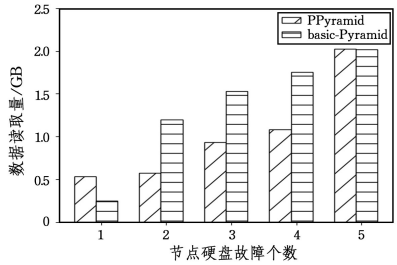

图 14　(13,8)参数下的 PPyramid 码和 pLRC 码平均修复成本对比图

Fig. 14　Comparison of average repair cost between PPyramid and pLRC with (13,8) parameter

5.5.2　平均修复时间

图 15 给出了(13,8)参数下的 PPyramid 码和 pLRC 码在多节点硬盘故障失效时的平均修复时间对比。如图 15 所示，横坐标表示硬盘故障个数，纵坐标表示修复所花费的时间。由图可知，在发生多节点硬盘故障时，PPyramid 码比 pLRC 码的修复时间减少约 38.3%～52.4%。

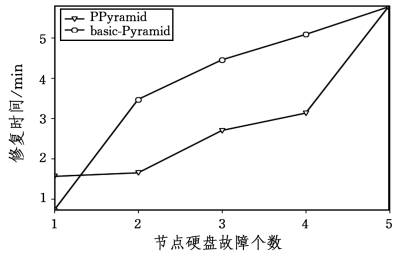

图 15　PPyramid 和 pLRC 平均修复时间对比图

Fig. 15　Comparison of average repair time between PPyramid and pLRC

5.6　PPyramid 码与 LRC 码、SHEC 码、DLRC 码的横向对比

5.6.1　平均修复成本对比

图 16 给出了(15,10)参数下的 PPyramid 码、(15,10)参数下的 LRC 码、(16,10)参数下的 SHEC 码、(16,10)参数下的 DLRC 码的修复成本对比。其中，(15,10)参数下的 PPyramid 码产生 1 个全局冗余块，同时将原始数据块分为 2 个小组，每组产生 2 个局部冗余块。(15,10)参数下的 LRC 码

产生 3 个全局冗余块，将原始数据块分成 2 个小组，每组产生 2 个局部冗余块。(16,10)参数下的 SHEC 码将原始数据块分成 6 个小组，每个小组产生 1 个局部冗余块。(16,10)参数下的 DLRC 码产生 2 个全局冗余块，将原始数据块分为 4 个小组，每个小组产生 1 个局部冗余块。如图所示，横坐标表示硬盘故障个数，纵坐标表示修复时数据的读取量。当发生单硬盘故障时，PPyramid 码的修复成本没有优势，但随着硬盘故障个数的增加，相比对比码，其修复成本大幅降低。当 2～3 个节点发生故障时，PPyramid 码相比 LRC 码，其修复成本降低约 23.8%～46.4%，相比 SHEC 码，其修复成本降低约 8.6%～12.4%，相比 DLRC 码，其修复成本降低约 19.7%～21.3%。

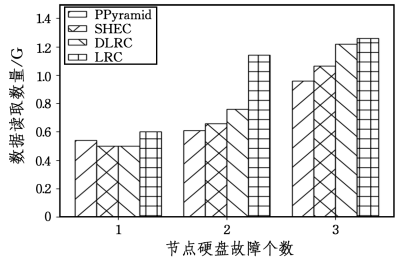

图 16 PPyramid 码与 SHEC 码、DLRC 码、LRC 码的平均修复成本对比图

Fig. 16 Comparison of average repair cost between PPyramid and SHEC, DLRC, LRC

5.6.2 平均修复时间对比

图 17 给出了(15,10)参数下的 PPyramid 码、(15,10)参数下的 LRC 码、(16,10)参数下的 SHEC 码、(16,10)参数下的 DLRC 码的修复时间对比。如图所示，横坐标表示硬盘故障个数，纵坐标表示修复所花费的时间。当 2～3 个节点发生故障时，PPyramid 码相比 LRC 码，其修复时间降低约25.6%～47.5%，相比 SHEC 码，其修复时间降低约 10.8%～12.7%，相比 DLRC 码，其修复时间降低约 21.7%～25.2%。

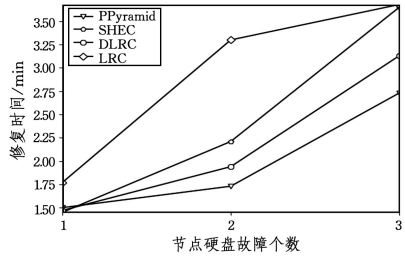

图 17 PPyramid 码与 SHEC 码、DLRC 码、LRC 码平均修复时间对比

Fig. 17 Comparison of average repair time between PPyramid and SHEC, DLRC, LRC

5.7 平均更新成本

图 18 给出了 PPyramid 码的更新成本。由图可知，其更新成本与需要移动的差块数成正比，其移动的差块越多，更新成本就越大。虽然在调整差块位置时会带来一定的更新成本，但相比其降低的数据修复成本和修复时间，其更新成本在尚可接受范围内。

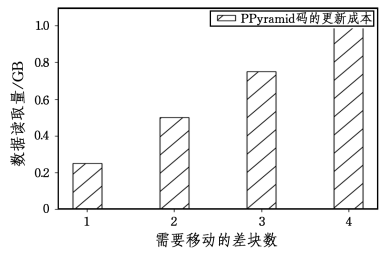

图 18 更新成本

Fig. 18 Update cost

结束语 在分布式存储系统中，由节点硬盘故障导致的数据丢失会对企业造成极大的损失。虽然目前各分布式存储系统都配置了自己的容错系统，以保障自身数据的安全，但其使用的方法依然是被动容错方式，因此不能做到提前预防，合理地变换容错方法。

本文将硬盘的故障预测技术与常用的分组码 basic-Pyramid 码相结合，提出新的 PPyramid 码。PPyramid 码能根据系统提前预测出节点硬盘即将出现故障的个数，动态地调整各小组中数据块的位置，从而使得数据修复工作集中在小组内进行，且将执行修复的小组个数降到最少。虽然这会带来编码时间的增加和一定的更新成本，但相比其降低的数据修复成本和修复时间，其所增加的编码时间和更新成本尚在可接受范围内。实验结果表明，PPyramid 码相比 basic-Pyramid 码能降低 6.3%～34.9%的修复成本和 7.6%～63.6%的修复时间，相比 LRC 码能降低 23.8%～46.4%的修复成本和 25.6%～47.5%的修复时间，相比 pLRC 码能降低 37.9%～52%的修复成本和 38.3%～52.4%的修复时间，相比 SHEC 码能降低 8.6%～12.4%的修复成本和 10.8%～12.7%的修复时间，相比 DLRC 码能降低 19.7%～21.3%的修复成本和 21.7%～25.5%的修复时间。同时 PPyramid 码拥有较强的修复能力，能提升分布式存储系统的可靠性和稳定性。

参 考 文 献

[1] Daily economic news, China's total data will account for 20% of global data in 2020. Information infrastructure protection is the key to big data security[EB/OL]. https://baijiahao.baidu.com/s?id=1601722855211864246&wfr=spider&for=pc.

[2] WANG Y J, SUN W, ZHOU S, et al. Key technologies of distributed storage in cloud computing environment [J]. Journal of Software, 2012, (4): 232-256.

[3] WANG Y J, LI S. Research and performance evaluation of data replication technology in distributed storage systems[J]. Computers & Mathematics with Applications, 2006, 51(11): 1625-1632.

[4] HUANG C, SIMITCI H, XU Y, et al. Erasure coding in windows azure storage[C]// Proceedings of the 2012 USENIX Conference on Annual Technical Conference. USA: USENIX Association, 2012: 2-2.

[5] HUANG C, CHEN M, LI J. Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems [C]// Sixth IEEE International Symposium on Network Compu-

ting and Applications (NCA 2007). Piscataway: IEEE, 2007: 79-86.

[6] SATHIAMOORTHY M, ASTERIS M, PAPAILIOPOULOS D, et al. XORing Elephants: Novel Erasure Codes for Big Data [C]// VLDB2013: Proceedings of the 39th International Conference on Very Large Data Bases. Trento: VLDB Endowment, 2013: 325-336.

[7] ZHOU S, WANG Y J. EXPyramid: An Array-Based Flexible Coding Scheme with High Fault-Tolerance and Low Recovery-Overhead[J]. Journal of Computer Research & Development, 2011, 48(s1): 30-36.

[8] MENG Y, ZHANG L, XU D, et al. A Dynamic Erasure Code Based on Block Code[C]// Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks. USA: Junction Publishing, 2019: 379-383.

[9] MIYAMAE T, NAKAO T, SHIOZAWA K. Erasure code with shingled local parity groups for efficient recovery from multiple disk failures[C]// HotDep 2014: Proceedings of the 10th Workshop on Hot Topics in System Dependability. USA: USENIX Association, 2014: 5-5.

[10] HAFNER, JAMES L. WEAVER Codes: Highly Fault Tolerant Erasure Codes for Storage Systems [C] // Proceedings of the FAST '05 Conference on File and Storage Technologies. USA: USENIX Association, 2005: 16-16.

[11] RASHMI K V, SHAH N B, KUMAR P V, et al. Explicit construction of optimal exact regenerating codes for distributed storage[C]// 2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton). Piscataway: IEEE, 2009: 1243-1249.

[12] WU Y, DIMAKIS A G. Reducing repair traffic for erasure coding-based storage via interference alignment [C] // ISIL2009: Proceedings of the 2009 IEEE international conference on Symposium on Information Theory. Piscataway: IEEE, 2009: 2276-2280.

[13] SCHROEDER B, GIBEON G A. Disk failures in the real world: What does an MTTF of 1 000 000 hours mean to you? [J]. ACM Transactions on Storag, 2007, 7(1): 1-16.

[14] HUGHES G F, MURRAY J F, KREUTZDELGADO K, et al. Improved disk-drive failure warnings[J]. IEEE Transactions on Reliability, 2002, 51(3): 350-357.

[15] LI P, LI J, STONES R J, et al. ProCode: A Proactive Erasure Coding Scheme for Cloud Storage Systems[C]// Proceedings of the 2016 IEEE 35th Symposium on Reliable Distributed Systems. Piscataway: IEEE, 2016: 219-228.

[16] HU Y, LIU Y, LI W, et al. Unequal Failure Protection Coding Technique for Distributed Cloud Storage Systems [J]. IEEE Transactions on Cloud Computing, 2017(99): 1-1.

[17] ZHANG X Y, XU J, HU Y. Predictive Local Repair Codes in Cloud Storage Systems [J]. Journal of Computer Research and Development, 2019, 56(9): 1988-2000.

[18] ZHU B, WANG G, LIU X, et al. Proactive drive failure prediction for large scale storage systems [C] // Proceedings of the 2013 IEEE 29th Symposium on Mass Storage Systems and Technologies. Piscataway: IEEE, 2013: 1-5.

[19] HAMERLY G, ELKAN C. Bayesian approaches to failure prediction for disk drives[C]// Proceedings of the Eighteenth International Conference on Machine Learning. USA: Morgan Kaufmann Publishers Inc, 2001, 1: 202-209.

[20] HUGHES G F, MURRAY J F, KREUTZ-DELGADO K, et al. Improved disk-drive failure warnings[J]. IEEE transactions on reliability, 2002, 51(3): 350-357.

[21] MURRAY J F, HUGHES G F, KREUTZ-DELGADO K. Machine Learning Methods for Predicting Failures in Hard Drives: A Multiple-Instance Application[J]. Journal of Machine Learning Research, 2005, 6(1): 783-816.

[22] LI J, JI X, JIA Y, et al. Hard Drive Failure Prediction Using Classification and Regression Trees [C] // Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. Piscataway: IEEE, 2014: 383-394.

[23] CIDON A, ESCRIVA R, KATTI S, et al. Tiered replication: a cost-effective alternative to full cluster geo-replication[C]// Proceedings of the 2015 USENIX Conference on Usenix Annual Technical Conference. USA: USENIX Association, 2015: 31-43.

[24] FORD D, LABELLE F, POPOVICI F I, et al. Availability in Globally Distributed Storage Systems. [C]// Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation. USA: USENIX Association, 2010: 61-74.

[25] SILBERSTEIN M, GANESH L, WANG Y, et al. Lazy means smart: Reducing repair bandwidth costs in erasure-coded distributed storage[C]// Proceedings of International Conference on Systems and Storage. New York: Association for Computing Machinery, 2014: 1-7.

ZHANG Hang, born in 1995, postgraduate. His main research interests include coding theory and distributed storage systems.

TANG Dan, born in 1982, Ph.D, professor, is a member of China Computer Federation. His research main research interests include coding theory and distributed storage systems.