|  |  |
| --- | --- |
| 学号 | 2017282110232 |

|  |
| --- |
| **大数据应用的云存储系统可靠性调查** |

|  |  |
| --- | --- |
| 院（系）名 称 ： | 计算机学院 |
| 班级 ： | 2017级硕士5班 |
| 学 生 姓 名 ： | 倪钢 |

二〇一七年十二月

摘 要

所谓大数据，就是成熟的，能够处理海量异构数据，并且对其进行快速修改的云存储系统。然而，云存储系统因为是由大型硬件组件组成，会产生许多源于硬件的不可避免的错误。所以提高大数据云存储系统的容错能力应用是一个重大的挑战。主从备份和纠删码是云存储系统中使用的最重要的数据可靠性技术。这两种技术在各种指标（如耐用性，可用性，存储开销，网络带宽和流量，能耗和恢复性能）方面都有自己的侧重点。本调查针对上述指标探讨了在大数据应用的云存储系统中采用这两种技术所涉及的挑战。本文还介绍了一种概念上的混合技术，以进一步提高云计算上大数据应用的可靠性，延迟，带宽使用率和存储效率。

目 录

[1 概要 1](#_Toc500687138)

[2 研究背景 3](#_Toc500687139)

[2.1 云存储和大数据应用程序 3](#_Toc500687140)

[2.2 数据失效 4](#_Toc500687141)

[2.3 数据可靠性 5](#_Toc500687142)

[3 纠删码 8](#_Toc500687143)

[3.1 Non-MDS编码/本地可修复编码 10](#_Toc500687144)

[3.2 重生成编码 11](#_Toc500687145)

[3.2.1 最小存储量和最小带宽重生成编码 12](#_Toc500687146)

[3.2.2 传输修复式重生成编码 13](#_Toc500687147)

[3.2.3 协作修复重生成编码 13](#_Toc500687148)

[4 主从备份 15](#_Toc500687149)

[4.1 静态备份 15](#_Toc500687150)

[4.2 动态备份 15](#_Toc500687151)

[5 主从备份和纠删码之间的比较 17](#_Toc500687152)

[6 大数据应用的云存储可靠性技术的最新进展 20](#_Toc500687153)

[6.1 学术挑战 21](#_Toc500687154)

[6.1.1 储存效率 21](#_Toc500687155)

[6.1.2 带宽效率 21](#_Toc500687156)

[6.1.3 能量效率 21](#_Toc500687157)

[6.1.4 数据访问延迟 22](#_Toc500687158)

[6.2 概念性架构 22](#_Toc500687159)

[6.3 未来的研究方向 24](#_Toc500687160)

[7 总结 25](#_Toc500687161)

[参考文献 26](#_Toc500687162)

# 概要

在当代大数据社会中，数据量增长速度超过存储容量（Gantz and Reinsel，2012）。 Facebook每个星期都需要额外的60TB存储空间才能用于新照片（Beaver，2010）。 YouTube用户每分钟上传400多个视频，每天需要1PB的新存储空间（Baesens，2014）。 根据国际数据公司（IDC）第六次年度研究报告，到2020年，电子数据将每两年翻一番（Gantz and Reinsel，2012）。 云计算提供了一种经济高效的方式来支持大数据和分析应用，从而挖掘数据的商业价值（Groenfeldt，2012）。 到2020年，数字世界中大约40％的数据将被云计算提供商存储或处理（Gantz and Reinsel，2012）。 云存储为大数据的存储提供了合理的可扩展性，并有助于处理数据的种类，数量和速度的稳步增长（Chen and Zhang，2014）。

由于云存储建立在商用服务器和磁盘驱动器上（Ford，2010），因此存在可能危及依赖于它的应用程序运行的故障。 例如，2009年，由于硬盘驱动器故障，Facebook暂时丢失了超过10％的存储照片（Gunawi，2010）。 Amazon Simple Storage Service（S3）遇到由负载平衡器错误引起的数据损坏问题（Wang，2015）。 Amazon Web Services（AWS）由于DynamoDB故障而导致严重中断（AWS，2016）。 在Facebook，在3000个节点的生产集群中，通常有20个或更多的节点故障（Sathiamoorthy，2013）。 由于故障是云存储系统的常态，在数据恢复期间提高数据可靠性同时保持系统性能，是在云计算上部署大数据应用程序时最重要的挑战之一。

云存储中的数据故障由各种数据冗余技术处理。 最常见的冗余技术是主从备份和纠删码。 主从备份是一个简单的数据冗余机制。 相同的数据被复制并存储在存储系统上的多个位置。 如果请求的数据在一个磁盘中不可用，则数据由下一个可用磁盘提供（Plank，2013）。 纠删码是一个更复杂的数据冗余机制。 奇偶校验数据与原始数据一起被创建和存储，当所请求的数据出错或者丢失时，可以由奇偶校验数据重构请求的数据。 校验码的存储开销远远小于主从备份，因此降低了数据存储的硬件需求，并在数据中心节约了大量的成本和能源（Huang，2012）。 然而，请求数据失败后的数据重构涉及高重建开销和网络流量。

上述就是云服务提供商为了提高可靠性和降低系统运营成本而采用纠删码技术的主要原因。 Facebook使用纠删码，将存储效率从2.1倍提高到3.6倍（Muralidhar，2014）。 Microsoft Azure使用纠删码将存储开销从3倍降低到了1.33倍，从而节省了超过50％的成本（Huang，2012）。

对Facebook的数据仓库集群的研究（Rashmi，2013）显示，每天有超过50个机器故障事件触发。 由于这些故障事件而导致的数据重建增加了网络流量。 Facebook仅对8％的数据将应用了Reed-Solomon纠删码技术。 因为与主从备份相比，纠删码修复每比特数据的网络开销要大上10倍，根据计算，如果有50％的数据由Reed-Solomon纠错机制重构，修复网络传输可能会使集群的网络链路饱和（Sathiamoorthy，2013）。

使用纠错技术的另一个问题是修复传输会增加延迟。 存储系统中高达40％的数据中心总能量的消耗（Harnik，2009）产生于读/写延迟造成的能源利用率效率下降（Kumar，2014）。 因此，减少数据恢复中涉及的等待时间可以节省大量的能源。 如前所述，纠删码机制提供更好的存储效率，可靠性和可用性，但另一方面，重构丢失的数据会增加网络传输量和网络传输延迟延迟。

本文介绍了使用主从备份和纠删码在云计算中提高大数据应用程序的数据可靠性方面正在进行的研究和挑战。 由于这两种技术各有利弊，本文关注于研究人员为了实现两种技术优势互补而所做的尝试。 我们还提出了利用这两种技术的优势的混合技术。 所提出的混合技术旨在优化云存储系统的重要参数，提高可靠性和性能，同时减少存储开销。

本文的内容安排如下。 在第2节中，我们讨论了用于云储存中，解决大数据应用的数据恢复问题的各类存储系统和文件系统。在第3节和第4节中，我们分别讨论纠删码和主从备份中涉及的技术状态和挑战。 在第5节中，我们介绍了主从备份和纠删码之间的综合比较。 在第6节中，我们介绍了大数据应用的云存储的可靠性技术的现状、挑战并提出混合技术概念的架构。 最后，在第7节中，我们总结并提供了这个研究领域未来发展的基础。

# 研究背景

在本节中，我们将简要讨论在大数据应用的云存储系统中使用的存储系统和文件系统的类型。 随后介绍了数据失效和数据可靠性的分析。

## 云存储和大数据应用程序

云存储系统由许多通过网络连接的存储设备组成。 它通常由网络附加存储（NAS）或存储区域网络（SAN）的分布式存储组成，具有存储虚拟化的特点（Zeng，2009）。 存储虚拟化是一种通过应用程序抽象物理存储并将逻辑存储映射到物理存储的技术。 存储设备网络可以被视为一个单一的存储设备，不论信息处于哪个物理位置，采用何种存储模式，用户都可以访问该信息。

根据客户如何访问数据和调用数据接口，云存储系统可以分为文件存储，块存储和对象存储（Mesnier，2003）

文件存储：在文件存储中，文件按层次结构组织。 有关文件的信息作为元数据存储在存储系统中。 文件可以通过指定单个文件的路径来访问。 它为应用程序提供了更高级别的存储抽象，并支持在不同平台之间传输安全的数据。 在文件数量和元数据有限的情况下，文件储存系统在局域网（LAN）展现了良好的性能。 文件服务器维护元数据并授权I / O在多个客户端之间共享文件。 但是，文件服务器争用会影响数据检索性能。

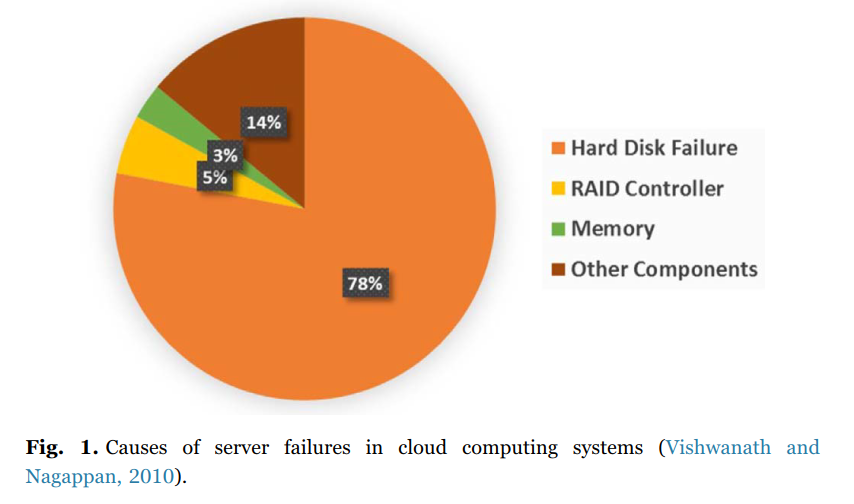
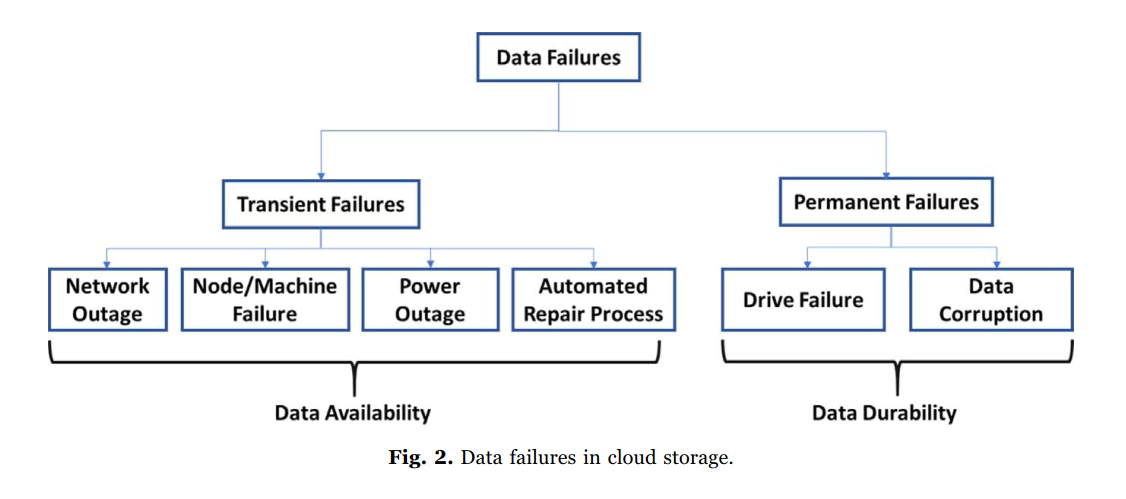
块存储：在块存储中，文件被分成块，并为每个块分配一个地址。 应用程序可以访问和组合块的地址。 存储应用程序保留元数据并使用它来共享数据。 它没有任何文件服务器来授权I / O，客户端可以使用元数据直接访问存储设备。 块存储具有良好的性能但不能保证数据传输的安全性。

对象存储：在对象存储中，文件和元数据被封装为一个对象，对象被赋予一个对象ID。 该对象可以是任何类型和地理分布的。 可以为每个对象分配唯一的元数据，例如关联的应用程序对象类型，保护级别，复制次数和地理位置。 该方式的存储管理可以涵盖从应用程序到存储设备的范围。 对象存储支持使用元数据进行直接的安全的客户端数据链接。 它提供了出色的可伸缩性来支持大数据应用程序。 对象存储除了主从备份之外，还支持有效的纠删码技术。

大数据的多样性，体积和速度特性，与云存储系统的分布式，虚拟化和可扩展的特性相适应（Kune，2016）。 O'Reilly（2016）详细讨论了大数据文件系统的优点和缺点。 HDFS，GFS，Lustre，ClusterFS，Ceph，OpenStack Swift，Quantcast和PVFS是其他的一些支持大数据应用的其他文件系统。 GFS和HDFS广泛应用于云存储，Vijayakumari等人对这些文件系统进行了比较。（2014）。

## 数据失效

在云存储系统中，很多因素都可能导致数据故障。 数据故障同样也会导致云服务故障。 Sharma等人 （2016）对云服务故障进行了详细的调查。 云数据故障的主要原因是硬件，软件，网络和电源故障（Rajasekharan，2014）。 磁盘是基于云存储的核心元素（Brewer，2016），也是最常见的故障组件（Hughes，2002）。 Vishwanath和Nagappan分析了大型云计算基础设施的硬件可靠性（Vishwanath和Nagappan，2010）。 如图1所示（2010年Vishwanath和Nagappan收集的数据），78％的服务器故障是由于硬盘造成的，5％是由于廉价磁盘阵列（RAID）控制器造成的，5％是由于内存造成的， 其余14％由于其他因素。 硬盘是最常被更换的组件，它们是服务器故障的最常见原因（Vishwanath和Nagappan，2010）。

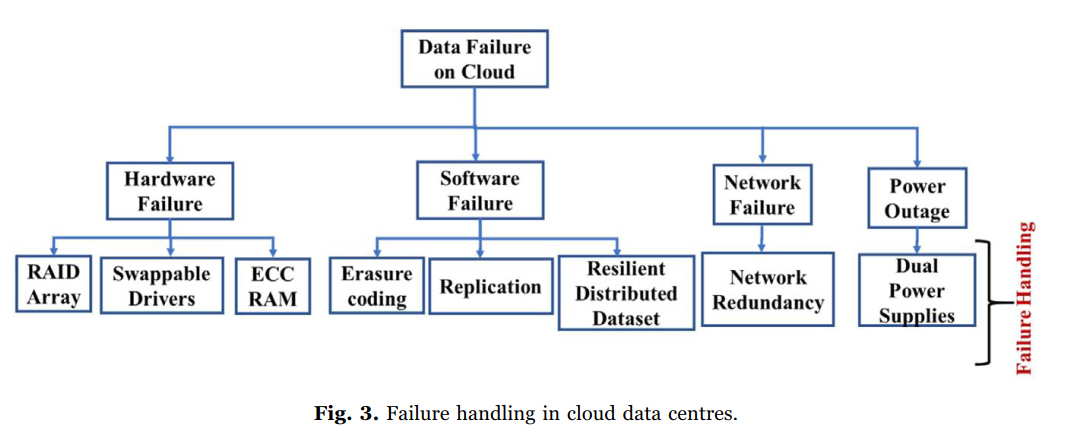
如图2所示，数据失败可能是暂时的或永久的。 由于网络中断，节点/机器故障，停电和自动修复过程而导致的数据故障是短暂的，不会导致永久的数据丢失（Rajasekharan，2014）。 从数十个谷歌存储单元收集的数据（每个谷歌存储单元在一年内有1000-7000个节点）显示，只有不到10％的故障事件在15分钟之内发生节点失效（Ford，2010）。 由于硬盘故障或数据损坏导致的数据故障会导致数据的永久丢失。

Pinheiro等人 （2007）使用9个月内收集的监测数据，详细分析了大型磁盘驱动器的故障行为。 他们发现故障概率与驱动器首次扫描错误，重新分配，分配和查看计数高度相关。 Ford等人（2010）证明了相关故障建模对可用性预测的重要性。 他们表明，不考虑节点故障会导致可用性被高估。 数据可用性增加1.5％，磁盘故障率降低10％。 但是，节点故障率降低10％，可用性提高18％。 Ma等人（2015）分析了大量备份系统的磁盘故障，以显示重新分配的扇区和某些特定类型的扇区错误对磁盘可靠性有很大的影响。 他们设计了针对单个和多个磁盘故障的主动保护。

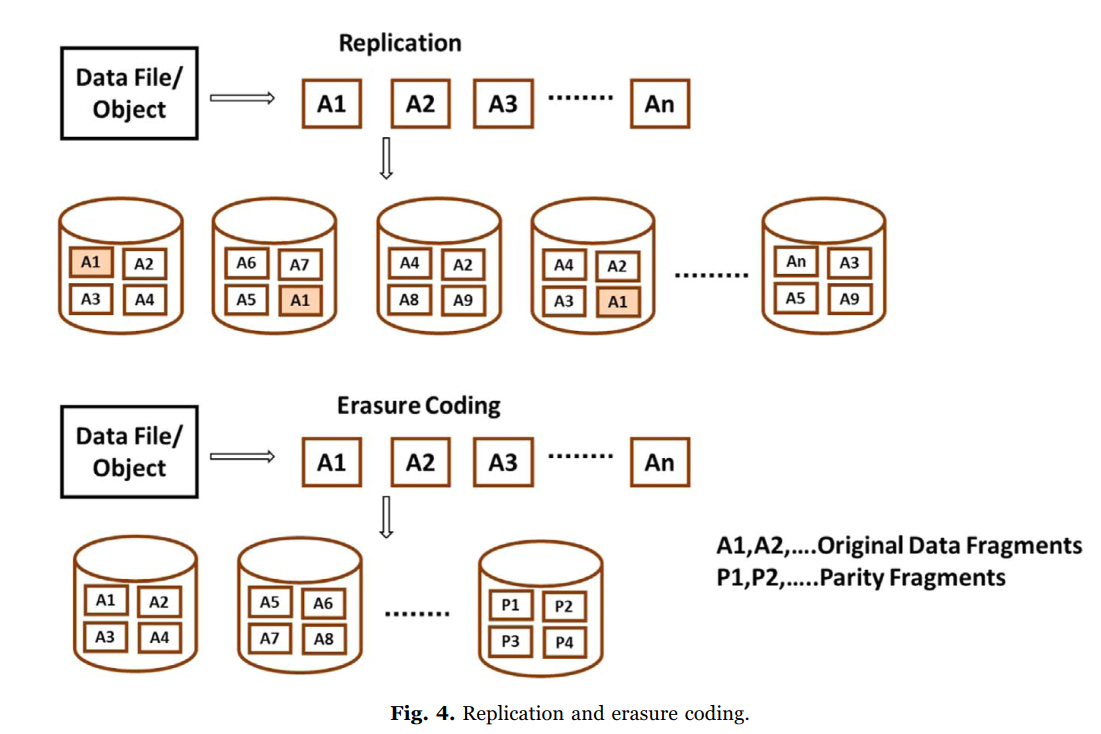
云存储系统中的各种组件故障导致永久或暂时的数据丢失。 磁盘是云存储系统中最重要的组件。 如果处理不当，磁盘故障会导致永久数据丢失。 云存储系统中的大多数其他组件故障只会导致暂数据可恢复性中断。 一些中断可能持续数小时，造成巨大的财务损失（AbuLibdeh，2010）。 上述讨论对于考虑各个组件故障来提高耐久性和可用性提供了一些启示。 下一节讨论云存储系统中使用的各种数据可靠性的保证机制。

## 数据可靠性

数据可靠性包括数据的耐用性和可用性的最大化。 高持久性意味着永久性故障发生次数较少，高可用性意味着瞬时故障发生次数较少。

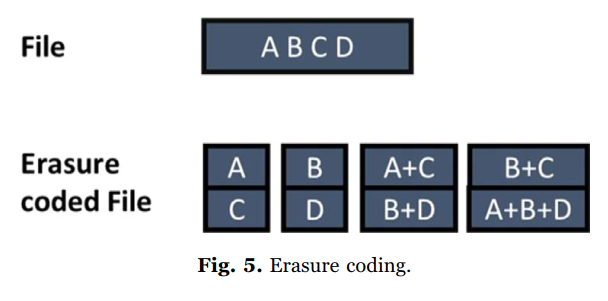
如图3所示，云数据中心采用了各种机制来提高存储系统的容错能力。 RAID阵列，交换驱动程序和纠删码存储器（ECC RAM）可以缓解硬件故障的影响。 RAID阵列是由多个磁盘组成的逻辑单元，用于存储带有条带，镜像和奇偶校验的数据。 交换驱动程序允许管理员在系统保持运行模式时交换故障或即将故障的驱动器。 ECC RAM用于通过将奇偶校验位与每个二进制代码相关联来检测和纠正单个位错误。 网络故障和停电则通过网络冗余和双电源供电来解决。

由于任何软件层面（包括云存储灾难）的故障都通过纠删码，主从备份和弹性分布式数据集（RDD）进行处理（Zaharia，2010）。 主从备份和纠删码用于处理主要的数据丢失。 RDD用于保护大数据应用程序生成的中间数据（Zaharia，2010）。

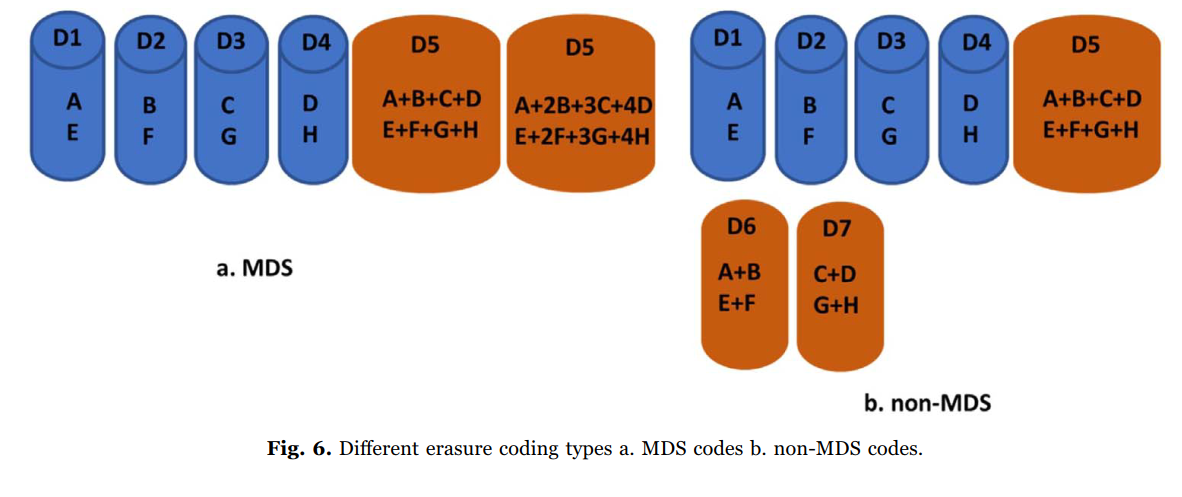
纠删码（Huang，2012; Sathiamoorthy，2013）和主从备份（Li，2011）是云存储中最流行的可靠性机制。 图4是主从备份和纠删码技术的说明。 在主从备份中，数据文件/对象被分成块并在存储系统上多次存储。 如果请求的数据在一个磁盘中失效，则数据由下一个存有有效数据的磁盘提供（Plank，2013）。 在纠删码中，数据文件/对象被分成块。 奇偶校验数据与原始数据一起被创建和存储，使得如果所请求的数据失效，则在奇偶校验数据的帮助下重建并提供所请求的数据。

尽管云提供商利用各种可靠性技术来提高对各种组件故障的容错能力，但主从备份和纠删码却因冗余数据的分布模式而脱颖而出。 因此，主从备份和纠删码支持包括灾难丢失在内的任何类型的数据丢失。 接下来的两节讨论纠删码和主从备份。

# 纠删码

纠删码在大规模存储系统中的数据失效规避方面发挥着关键性的作用（Plank，2013）。在云计算出现之前，纠删码技术主要用于通信和存储过程中的数据查错和纠正（Vins，2014）。在（n，k）纠删码存储系统中，一个大小为B的文件将被分成k个相同的块，并且添加n-k个奇偶校验块，使得除了k块以外的任意一个块都可以恢复原始文件。例如，图5表示（4,2）能够解决任意两个分块失效的纠删码机制。用于计算奇偶校验数据的算法可以是标准算法或伽罗瓦域算法（Plank，2013）。在标准算法中，加法是通过执行二进制的“异或”操作而进行的，而乘法则是二进制“与”操作。如果一个字符中的比特数为1，则执行标准算术。而当字符中的比特数大于1时，使用伽罗瓦域算法进行奇偶校验。在Galois域GF（2n）中，算术运算运行范围为从0到2n-1的有限数字集内;加法按位XOR；而乘法则更为­复杂，因为乘法运算取决于硬件，存储器和字长（Plank ，2013）。

纠删码可以分为最大距离可分码（MDS）和最大距离不可分码non-MDS。 在m个磁盘上设置奇偶校验数据并且能够应对任意不超过m个磁盘组合的数据丢失的纠删码机制，被称为MDS; 如果m个磁盘专用于保存校验数据，则non-MDS纠删码只能处理部分m个磁盘组合的数据失效。 例如，在图6a中，磁盘D5和D6专用于奇偶校验，因此该系统可以处理任意两个磁盘故障，这是MDS的方式。 在图6b中，D5，D6和D7专用于奇偶校验，但不能处理任意三个磁盘故障。 例如，如果D1，D5和D6同时失效，则D1中的数据将不会被恢复。 这就是所谓的non-MDS编码。

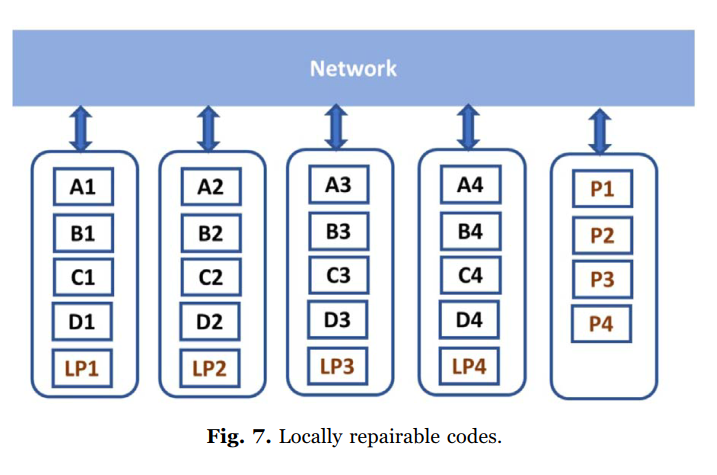
简单的纠删码的例子是RAID-6编码，阵列编码和Reed-Solomon编码（Plank，2013）。 RAID-6编码是MDS，它为数据块创建两个奇偶校验块，以便处理两个磁盘的数据故障（Jin，2009）。数组编码是用标准运算符（即XOR运算）实现的。在数组编码中，奇偶校验块是以系统数据（原始数据）的不同线性组合来计算的。行对角奇偶校验（RDP）（Xiang等人，2010），EVENODD（Blaum等人，1995），Blaum-Roth（Yixian，1994）和自由编码（Plank，2009）是RAID 6的阵列编码，最多可以处理任意两个磁盘的数据故障。星型编码是一个数组编码，它可以处理三个磁盘任意组合的数据故障（Huang和Xu，2008）。 Cauchy Reed-Solomon，广义EVENODD和广义RDP这三种数组编码的方式可以指定任意的k和m值。近期关于纠删码这一领域的最新进展，是减少了采用Reed-Solomon编码方式时的Galois域运算的CPU负担。除此之外，目前对任意k和m值，我们都能够简单快速的定义一个对应的Reed-Solomon编码机制。因此，与其他编码方式相比，目前Reed-Solomon编码更加突出（普朗克，2013年）。

Reed-Solomon编码是目前最流行的纠删码。 它们可以被设计为数据块和奇偶校验磁块的数量任意组合。 Reed-Solomon编码是MDS编码。 编码和解码可以用Galois域运算完成。 Facebook和Microsoft Azure在他们的存储系统中实现了Reed-Solomon代码（Beaver et al.，2010; Huang et al.，2012）。 在纠错码的存储系统中，发生的任何数据故障都会触发数据重构，以应答对应的数据请求。 由于纠删码中的数据重构涉及高磁盘I / O和网络带宽，因此增加了数据重建的成本。 现在许多研究都集中在降低Reed-Solomon编码存储系统失败数据的重建成本这个问题上。

本文重点介绍了这些研究中的两个方向。 一个方向是减少数据重建时的网络带宽占用，这些编码被称为重生成编码。 另一个方向是减少重建丢失数据所需的磁盘I / O，这类编码被称为本地可修复编码（LRC）。 在下面的章节中，我们分别讨论non-MDS / LRC和重生成编码。

## Non-MDS编码/本地可修复编码

Non-MDS编码包括原始数据块的本地奇偶校验以及全局奇偶校验，以便重建时只需要最少的磁盘I / O就能完成重建。

图7表示本地可修复编码。 本地奇偶校验的优势在于使用比全局奇偶校验更少数量的数据块重建单个故障块。 全局奇偶校验则可用于重建两个或两个以上同时失败的块。添加本地奇偶校验会使编码成为non-MDS并增加存储开销。

Huang等人（2013）设计了两个新的非MDS纠错码（基础金字塔编码和通用金字塔码）。 他们根据MDS编码设计出了基础金字塔编码。 例如，金字塔编码可以由（11,8）如下MDS编码构造。 （11,8）MDS编码的八个数据块被分成两个相等大小的组。 三个奇偶校验块中的两个保持不变，称为全局奇偶校验块。 两个新的冗余块可以分别由两个相等的数据组构成，作为本地奇偶校验块。 这种技术可以显著提高读取性能，因为本地奇偶校验会减少重建丢失数据时涉及的磁盘I / O。 与（9,6）MDS码相比，（10,6）基础金字塔编码将重构读取成本降低了50％，额外存储开销为11％，不可恢复概率为5.6×10-7。 因此，在高容错能力和额外存储开销的前提下，该编码方式提高了重构性能。

通用金字塔编码不是基本金字塔代码的扩展，它的特点是具有最大可恢复（MR）性。 通用金字塔编码的奇偶校验块使用生成矩阵计算。 具有MR特性的纠删码的容错范围很大，即满足匹配条件的所有故障情况都是可恢复的。 与通用金字塔编码相比，基本金字塔编码的不可恢复性要高出45%（Huang，2013）。

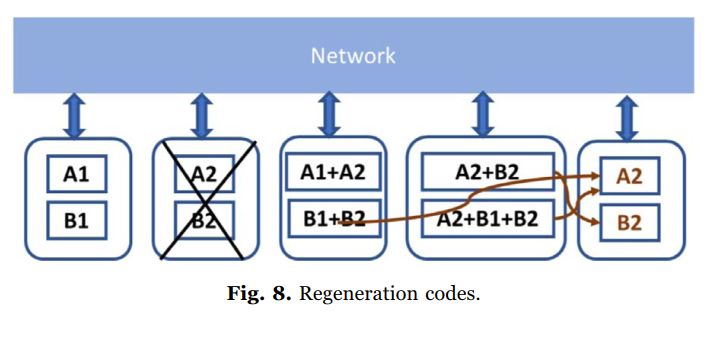
在提出金字塔编码之后，Huang等人为Microsoft Azure存储提出了一套新的非MDS纠删码（局部重构代码）（Huang，2012）。 这个代码是用（k，n，r）参数定义的。 它将k个数据片段分成n个组，并为每个组生成n个本地奇偶校验以及r个全局奇偶校验。 它可以容忍高达r + 1的故障，与Reed Solomon编码相比，这种编码方式减少了重构数据的带宽和I / O数据流，在存储开销方面则增加了1.33倍。 以解码4KB片段的平均等待时间来看：Reed-Solomon为13.2μs，LRC为7.12μs。 LRC的解码速度更快，因为它将重建所需的碎片数量减少了一半。

Sathiamoorthy等人（2013）提出了一种新的非MDS纠删码（XORing the Elephants）。 他们通过合并本地奇偶校验，在Facebook的RS（10,4）存储系统之上设计了LRC（10,6,5）编码。 他们进一步将每5个数据块设为一组，每组设置一个本地奇偶校验位，这样任何单个丢失的数据块都可以通过与该组中剩余的块进行通信来重建数据。 与Reed Solomon编码相比，它在重建时减少了大约2倍的磁盘I / O和网络流量，存储开销仅为Reed Solomon编码的14％。

Plank等人 （2013）提出了扇区 - 磁盘（SD）编码，它可以处理磁盘和扇区组合的数据故障。 这是一个非MDS编码，可以处理任何两个磁盘或条带中的任意两个字段的故障。 与其他非MDS编码相比，它具有最小的存储开销。 他们还指出，它所需要的计算和磁盘I / O相对更少。

虽然所有上述non-MDS编码在保证可靠性的同时改进了性能，但是这些编码的缺点在于会增加额外的存储开销。 此外其中的本地奇偶校验只对磁盘中的单个块故障有效。

## 重生成编码

重生成编码的特点是在占用最小的数据I/O的情况下对故障节点进行有效的修复。 以往的惯例是，通过使用任意k个可用节点进行通信和下载所有数据，然后根据下载的数据重建失效的节点数据。 Dimakis等人 （2010）证明，通过网络编码，任意d个存活节点（k≤d≤n - 1）的数据分片足够重建失效节点。 （n，k）纠删码的存储系统假定B为文件的大小，每个片段由有限域上的α个标志位组成。 根据重生成编码的定义，任意d个幸存节点的任意β<α个标志位足以修复故障节点。 因此，用于修复的下载的数据总量dβ，小于文件B的大小，如图8所示（Rashmi，2011）。 假设图中的每个数据块都是1 GB。若产生图示中的A2和B2的数据失效后，重建只需要3 GB而不是4 GB。

### 最小存储量和最小带宽重生成编码

重生成编码可以是最小存储重生成（MSR）或最小带宽重生成（MBR）。 如果α值最小，称为最小存储重生成。 如果β值最小，称为最小带宽重生成。 在MSR中，α和β可以通过首先最小化α然后最小化β来确定。 在MBR中，α和β可以通过首先最小化β，然后最小化α来确定。

数据的修复过程可以是部分的，功能性的或精确的。 在精确的重生成编码中，替换节点存储与失效节点完全相同的数据。 功能性重生成编码重建一个新的节点，虽然是MDS编码，但这个新节点会包含一些与对应的失效节点不一致的数据。 在部分重生成编码中，原始数据节点被精确修复，奇偶校验节点被功能性修复（Suh and Ramchandran，2011）。

Suh和Ramchandran（2011）提出了一个精确的MSR编码模型，其中d≥2k - 1的有限域中至少有2（n - k）个有冲突校准能力。通过采用至少n (m − k + 1)个有限域的积矩阵框架，Rashmi等人 （2011）提出了一个能够适用所有（n,k,m）值的精确MBR编码架构和一个适用所有（n,k d ≥ 2k − 2）值的MSR架构。 通过Cadambe（2010年），Papailiopoulos和Dimakis（2011年）以及Suh和Ramchandran（2010年）几人的研究，确定了精确MSR编码的（n，k，m）参数的值的多种选择。 此外，也有许多研究（Cadambe 2011; Tamo 2011; 2011）确定了混合MSR编码的多种参数选择，这种编码模式支持系统部件的精确修复和奇偶校验部件的功能修复。

MSR和MBR编码专注于存储和带宽最小化，但可能会增加磁盘I / O。 精确修复的各种参数选择仍然是一个悬而未决的问题。

### 传输修复式重生成编码

在重生成编码过程中，故障节点的替换需要连接到其余节点，并接收小于α的β个数据块，将其作为一个存有α个数据块标志位的函数。 参与修复过程的节点读取多个数据块并传递这个函数。 这个过程可能会导致磁盘I / O开销。 为了最小化I / O开销并避免数据节点执行算术运算，有学者提出了传输修复式重生成编码。

Rashmi等人（2009）提出了一个直观的传输修复式精确MBR编码模型（n，k，d = n - 1）。 功能性修复是通过传递不同（n，k，d）值的MSR码（Hu 2013；Shum and Hu 2012）来实现的。通过在Shah等人（2012年）定义的有大小为2的有限域上传输MBR码（n，k = n-2，d = n-1）来进行精确的修复。 Lin和Chung（2014）定义了一种新的修复方法，即在m = n-1个MBR点处传输精确的MBR码，这需要一个较小的有限域。 Chen和Wang揭示了k≥3，β<d - k + 1的最小存储再生（MSR）码不能通过传输修复的方式实现（Chen and Wang，2015）。

传输修复式重生成编码仅占用最小的磁盘I / O，并具有MSR和MBR代码的所有优点。 但是，只有一些特定的（n,k,d）参数能够使用这种方法。

### 协作修复重生成编码

Hu等人（2010）首先提出了一种用于多节点故障的相互协作恢复（MCR）机制。在这种机制中，要修复的节点可以在相互之间交换数据，以提高存储和带宽的利用率。 基于新节点的带宽消耗，Kermarrec（2011）和Shum（2011）提出了合作重构编码模式。Shum和Hu（2011）提出了（n，k，d = k，t = n-k）参数下的精确MBCR的构造，其中t是为重构而通信的新节点的数量。 Wang和Zhang（2013）表明，对于（n，k，d，t）的所有可能值，至少有n个精确MBCR编码能实现。 Le Scouarnec（2012）研究了了当d≥k= 2时的几种能够实现精确MSCR的参数选择。Pei et al（2015）提出了基于树结构（CTREE）的协同重生成数据修复模型，在多次数据故障的情况下，该模型优化了修复需要的网络流量和时间。他们提出CExchange来降低网络流量成本。 提出了ED-TREE和PTransmission来减少维修时间和提高数据传输效率。以上所有编码均没有通用性，只能适用于部分参数。

以下工作集中于优化重建所需的磁盘I / O，降低I / O恢复的成本，不会像non-MDS那样产生任何存储开销。 该算法支持任何基于XOR的纠删码（即数组码）。 Xiang等人 （2010）针对RDP编码中的单个磁盘故障提出行对角线最佳恢复（RDOR），并显著降低了数据恢复的I / O成本。 他提出了单磁盘故障的I / O最佳数据恢复模型。 Khan等人（2012）提出了一种基于标志（每个磁盘中的块的分区），实现最小化重建数据所需磁盘I / O的算法。 该算法支持小于3的任意奇偶校验块。

以下编码旨在减少带宽和磁盘I / O，而不会产生任何存储开销。 Rashmi等人（2014）提出了Hitchhiker代码，该代码使用构建在RS Code之上Hop-and-couple（磁盘布局）的“Piggybacking”框架。 它支持系统和奇偶校验数据片段的任意组合。 尽管Hitchhiker将重建期间读取数据所需的时间缩短了32％，重建期间的计算时间减少了36％，网络传输和磁盘I / O减少了35％，这增加了编码时间。

# 主从备份

主从备份是云数据中心最常用的可靠性机制，以便以低延迟和最小的带宽消耗来提高可用性和持久性（Bonvin等，2009）。 数据失效时，为了保持持久性，需要在活动磁盘中恢复失效数据的副本。 这种恢复可以被动地或主动地执行。 在反应式主从备份中，副本将在失败后创建。 在主动备份中，即使在发生故障之前也会创建副本。 用于主从备份的常用方法是静态备份和动态备份。

## 静态备份

在静态备份中，副本的数量及其位置是固定的（Bonvin等，2009）。无论用户行为如何变化，副本都是手动创建和管理的。随机备份是HDFS，RAMCloud，GFS和Windows Azure中最常用的备份技术。在这种技术中，数据被备份到不同机架上随机选择的节点上。随机块复制可以处理并发失败，因为块被放置在不同的机架上。该方法的缺点是，当所有副本都丢失时，整个备份系统就无法起作用。此外，备份丢失的数据涉及与定位和恢复丢失的数据这两个高消耗的操作。西顿等人（2013）建议副本集备份。它将节点分成与备份数相关的副本集，并把这种映射关系队列化。副本放置在其中一个副本集中。这样数据丢失只会发生在某些副本集的所有节点同时失败的情况下。这种做法增加了数据的持久性，没有显著的存储开销，并且具有与随机备份相同的性能。

Liu和Shen（2016）提出了多重故障弹性复制（MRR）来提高云存储的可用性。作者根据数据的访问频率为每个对象定义了不同数量的复制。节点被分成不同的组，使得组可以处理不同数量的复制，每个节点由来自不同数据中心的节点组成。它降低了一致性维护成本低的数据丢失的可能性。 Long等人（2014）提出了云存储的多目标优化管理（MOM）算法。 MOM根据5指标的数学模型（即不可用性，服务时间，负载变化，能耗和延迟）来决定副本的数量和副本的位置。在模型的定义中已经考虑了参数大小，文件访问率，失败概率，传输速率和容量数据节点。作者表示，该算法提高了文件的可用性，负载平衡，减少了服务时间，延迟和能源消耗。

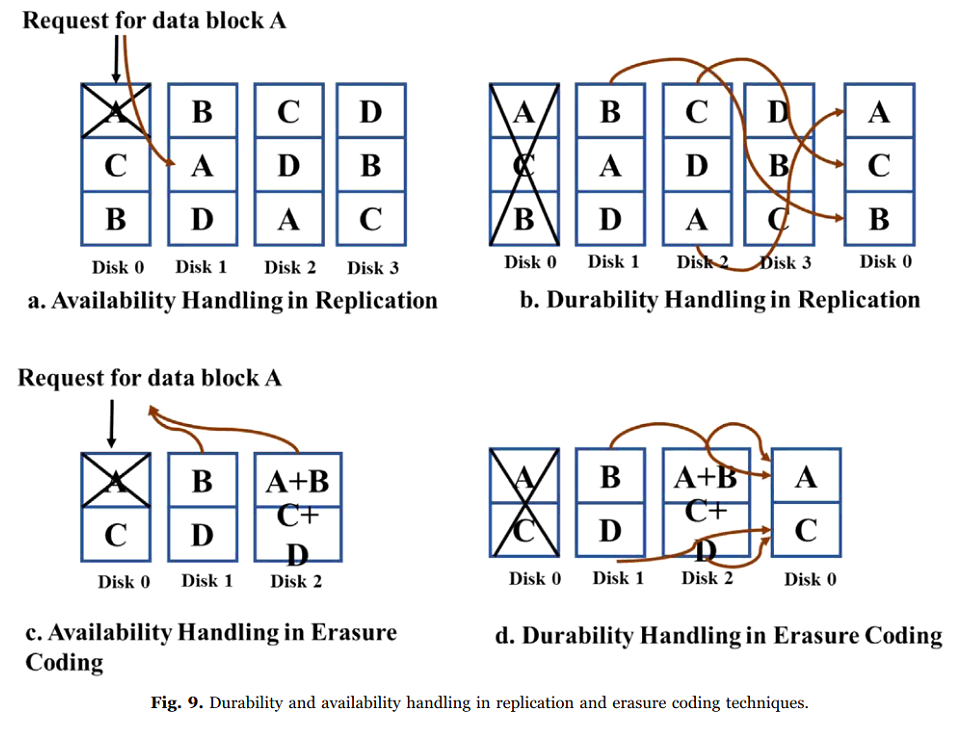
## 动态备份

在动态备份中，副本是动态创建和删除的。为了提高耐用性，可用性，成本，存储效率，带宽，延迟，能源和执行时间，根据用户行的为自动进行副本的创建，位置，管理和删除等操作。 Bonvin等人（2009）提出了基于到地理多样性的动态消耗的，保持高可用性和低延迟的云备份。他提出了一个键值对存储Skute，它决定了备份的成本效益。Sun等人（2012）定义了系统可用性与备份数量之间关系的数学模型。他们提出了动态备份策略，以确定要备份哪些数据，备份时间，副本数量和新副本的位置，以提高读取性能和可用性。Qu和Xion（2012）提出了弹性、容错和高效的算法来实现云存储系统的高可用性。它通过考虑流量负载，节点存储容量和副本带宽来动态平衡节点间的工作负载。这种方法在高存储效率的同时提高了数据可用性。

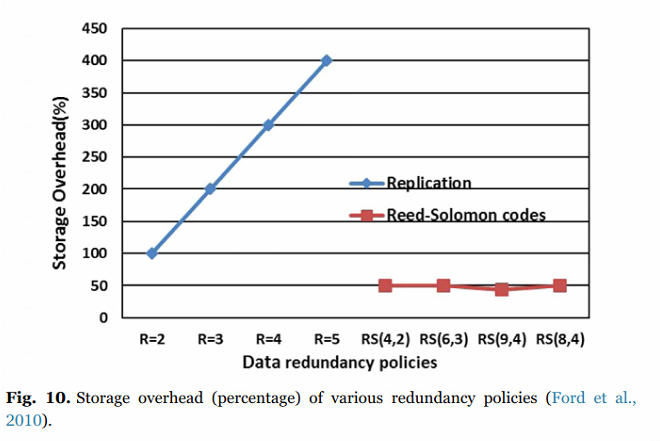
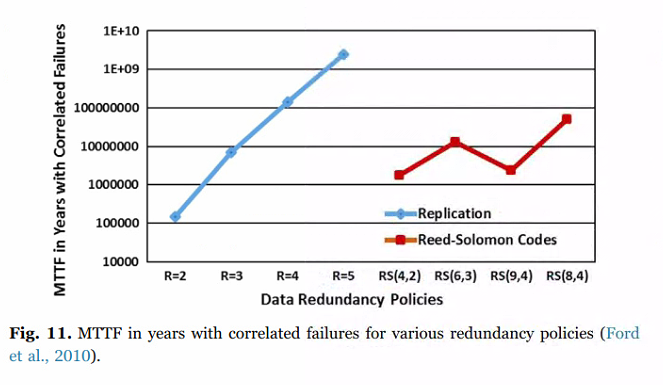
Hussein和Mousa（2012）也提出动态备份策略。基于数据请求和时间序列技术的历史，该策略预测未来的数据访问频率。如果预测频率超过阈值，则选择数据块进行备份。在这个过程之后决定副本的数量和副本的位置。实验结果表明，无论是增加任务数量还是提高系统的可靠性，该策略都能保持响应时间的稳定。 Boru等人（2015）提出了一种数据备份技术来优化云数据中心的能耗、网络带宽和通信延迟。他们定义了能耗和带宽需求的模型，并提出了基于此模型的节能备份­策略，可以减少通信延迟。Li等人（2011）提出在云存储上大数据应用的高成本效益的备份策略，这是一个在备份的基础上的云数据可靠性模型。他们使用一种算法来确定最小数量的副本，并保证了数据的可靠性。为了保证数据的可靠性和最小的备份量，他们使用了一个通用数据可靠性模型来预测数据的可靠性。数据可靠性在此期间一直使用主动备份算法进行维护：该算法检测副本状态并在需要时触发数据恢复过程。

# 主从备份和纠删码之间的比较

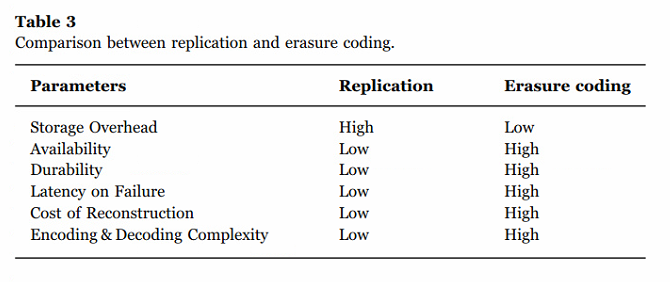
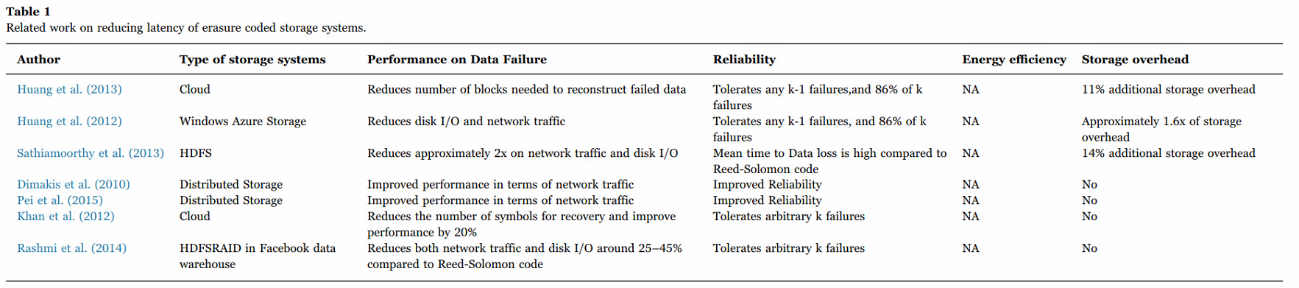
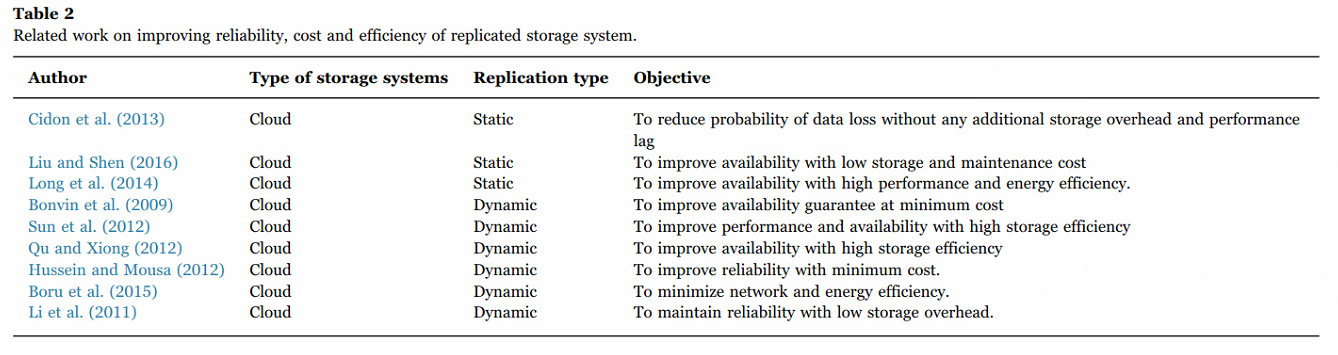
主从备份和纠删码是云数据中心使用的重要的可靠性机制，可以保护数据免遭破坏。 了解这些技术在云存储系统中实施最佳技术的优点和缺陷，以此节省成本来提高可靠性是一件非常重要的事。 下面详细分析这些技术在各个方面的参数。

图9示出了如何在主从备份和纠删码存储系统中处理对失效数据的读取请求。 它还显示了如何在瞬间和永久数据丢失的情况下重建数据。 通过简单地将请求重定向到下一个可用的副本来请求在备份存储系统中失效的数据。 另一方面，在纠删码的存储系统中，失效数据通过重建来自下一个可用磁盘的数据来服务请求。 纠删码存储中的重构涉及比在主从备份存储中更多的磁盘I / O。 例如，在图9c中，块A的重构涉及从两个不同的磁盘读取两个数据块。 与主从备份相比，这增加了纠删码存储系统中的读取请求的等待时间。

在纠删码系统中永久失效时的磁盘重建涉及比复制更多的磁盘I / O。 例如，在图9b中，故障磁盘的重建只访问三个磁盘就能重建三个数据片段。 然而，在图9d中，故障磁盘的重建需要访问四个磁盘以重建两个碎片。 这增加了纠删码系统中的重建成本。

图10为存储开销对比图，图11为针对两种冗余策略的相同数据故障的年平均故障时间（MTTF）对比图。 这两幅图是福特等人的观测到的数据（2010）所描述出来的。 这些数据表明，与主从备份相比，纠删码具有更低的存储开销，可靠性更高。 在大规模存储系统中，用纠删码替换主从备份可以显著节约成本。

纠删码的存储效率比主从备份更高，但存在性能问题（Cook，2014）。 在纠删码的存储系统中编码数据是耗时操作，而在主从备份中，对失效数据的请求可以在没有延迟的情况下被重定向到下一个可用副本（Cook等人，2014）。 在纠删码系统中，则是从后面的可用磁盘中重构失效数据，这个操作增加了读取请求的延迟（Cook，2014）。 而且，在带宽和磁盘I / O方面，重建数据的成本相对较高（Li和Li，2013）。

一些研究将主从备份和纠删码做了对比（Weatherspoon等，2002; Lin等，2004; Rodrigues和Liskov，2005; Cook等，2014; Ford等，2010）。 这些比较假定参数之间是独立的。 例如与主从备份相比，纠删码提供了显著的存储效率，或者提高了耐久性和可用性。 但是，纠删码不能明显的提高存储效率和可靠性。 我们在表3中对两种技术进行了了比较。评价“高”和“低”用于表示一种技术相对于另一种技术是优还是劣。

# 大数据应用的云存储可靠性技术的最新进展

由于云存储系统的故障频繁，云存储系统采用数据冗余来处理故障。 主从备份是提高数据可靠性的简单解决方案。 但是数据如果到了TB和PB这个数量级，主从复制会大大增加存储开销。 目前纠删码技术正在不断发展，因为它在节省了大量存储空间的同时保证了可靠性和耐用性。 但是，恢复丢失的数据所涉及的重建成本抵消了节省出来的大量存储空间。 与主从备份相比，Reed-Solomon编码每比特需要大约十倍的修复开销。 本节的其余部分将讨论在云存储系统中使用大数据应用的冗余技术所面临的挑战。

一些研究（表1）侧重于减少网络流量，减少纠删码存储系统中重建失败数据的磁盘I / O。 有些纠删码模型试图通过额外的存储开销来提高重构数据时的性能，但是没有一个性能能达到像主从备份的水平。

一些研究（表2）侧重于最大限度地减少主从备份存储系统中的冗余数量，以提高存储效率。 与没有牺牲可靠性的纠删码相比，并不能显著减少存储开销。 实现高可靠性，高存储效率和高性能的主从备份和纠删码存储系统尚未实现。

混合可靠性机制可能是未来数据中心的发展方向。混合可靠性机制结合了主从复制和纠删码。表4中列出了混合可靠性机制目前所做的非常有限的工作.Araujo et al（2011）提出了基于混合编码的双重编码。这种编码的思路是在一个节点中保留一个完整的数据副本，并在网络中传播纠删码的片段。在双重编码中，原始数据片段和奇偶校验片段的副本被安排在网络中的不同节点中。虽然这种做法在重建时节省了带宽，但还是影响了存储效率。Ma等人（2013）提出了一个名为CAROM的主从备份和纠删码结合的新方案。他们的方法将数据写入请求后的整个文件缓存，用以服务于后续的读写请求。它还会在读取请求时缓存请求的块，以便为后续读取服务。与主从备份相比，它可以在保持低延迟的同时节省60％的存储成本和43％的纠删码的带宽成本。但是，当请求的数据不在内存中时，仍然需要在块不可用时重建数据。Li等人（2016）提出了主动纠删码（ProCode），它根据驱动器故障预测自动调整数据的备份权重 。它将读取延迟降低了63％，重建时间减少了78％。ProCode在由闪存驱动器组成的存储系统中没有任何影响，可交换的驱动程序可以更有效地处理驱动器故障。

## 学术挑战

云存储系统中大数据应用的数据可靠性领域面临着许多开放的挑战。 在本节中，我们将探讨提高云存储系统中大数据应用程序的数据可靠性所面临的挑战。

### 储存效率

主从备份的数据可靠性直接与存储开销成正比。 因此，降低存储开销而不牺牲可靠性是复制中最大的挑战。 尽管纠删码可以大大节省存储空间并且具有相当的可靠性，但是在出现故障的情况下，它会增加大数据应用程序的网络流量和延迟。 本次调查针对主从备份的各个方面进行了探讨，尝试使用动态备份策略来节省存储空间，但是该种策略牺牲了部分可靠性。 调查也表明，目前对纠删码的研究集中在减少网络传输和延迟两方面。 实现动态冗余的自动化，比如改变纠删码数据中的副本数量以支持故障处理，也可以进一步提高存储效率。

### 带宽效率

网络带宽始终是分布式存储中的稀缺资源。带宽的使用量与分布式存储中传输的数据量成正比。在主从备份和纠删码存储系统中，修复失效数据会消耗相当多的网络带宽。传统的纠删码比主从备份涉及更多的带宽消耗。触发云数据中心数据重构的事件主要是节点故障，它增加了纠删码存储系统中的网络流量（Sathiamoorthy et al，2013）。在纠删码存储系统中减少网络流量的各种工作已是目前业界的中点关注对象。但是，这些工作并不能减少主从复制存储系统的网络流量。虽然最近对纠删码的算法改进工作取得了一定进步，但数据重构时的网络传输问题仍然是目前最重要的挑战之一。此外的重要挑战还有在不牺牲性能和能耗的前提下提高带宽效率。对故障的预测技术也是一个重要的研究方向。在故障发生之前，采用故障预测机制进行动态备份能够减少纠删码存储器中的重构次数，从而减少网络流量。

### 能量效率

存储系统是云计算中最重要的能源消耗组件之一（Sharma等，2016）。数据中心使用的节能方法节省了运营成本，并有助于保护环境（Butt等，2014）。存储系统的能源效率很大程度上取决于读写延迟（Kumar，2014）。 Pinheiro等人（2006）介绍了一种称为转向访问的技术，它将存储系统中不同磁盘上的原始数据和冗余数据分开。这种技术使包含冗余数据的磁盘处于空闲状态，直到出现磁盘故障大量出现。这项技术已经被证明可以节省20-61％的磁盘能耗。 Harnik等人（2009）提出了一种在低功耗模式下使用辅助节点（附加节点池和额外的数据副本）的全覆盖方法。纠删码存储系统的节能潜力在低功率模式下受到限制，但是当n与k之间的比率增加时，其节能潜力得到改善。 Butt等人（2014）提出了能源可靠性产品（ERP）度量标准，以比较不同设计与数据中心存储系统的能源效率和可靠性之间的关系。 Greenan等人（2008）提出了功率感知编码，并提出了一种用于在功率感知纠删码存储系统中读取，写入和激活器件的通用技术。他们还表示，激活非活动磁盘会增加功耗。Li等人（2011）提出了基于主从备份的云存储系统中能量有效数据传输的链路速率控制数据传输（LRCDT）策略。

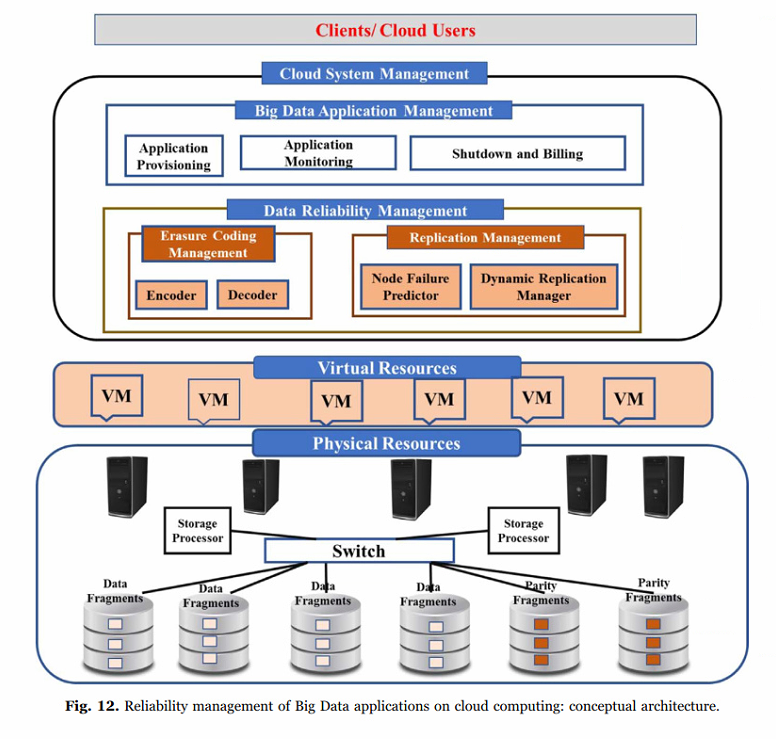
就储存来说，节能的重点关注于纠删码的数据重构过程，对主从备份而言也是一样。 因此，在不牺牲存储效率的情况下改进数据重建的能耗是一个挑战。 通过减少纠删码中的重构次数，可以减少网络流量，提高能源效率，通过对所选数据块进行季度性备份，可以减少重建次数。

### 数据访问延迟

在纠删码的存储系统中，当出现故障时，必须执行译码操作以重建原始数据。 因此，数据访问延迟是纠删码存储系统中最重要的挑战之一。 在主从备份存储系统中，通过选择最佳的复制位置可以显著降低访问延迟。 在低延迟的条件下，大数据应用程序的能耗和性能都会大幅度提高。 减少延迟的同时减少存储开销对研究人员来说是一个挑战。 通过基于纠删码中的访问历史记录和失败日志激活主动动态备份，可以减少存储开销，从而减少访问延迟。

## 概念性架构

我们对云存储系统中的大数据应用提出了以下混合技术，以减少网络流量，并以较少的存储开销作为代价提高数据恢复的性能。 基于节点故障预测，这种混合技术应用了制纠删码数据的主动式数备份。 这种混合技术将显著减少网络流量，并在较少存储开销的前提下获得较高的大数据应用的性能。 图12描述了混合技术提高可靠性，降低存储开销性能的概念架构。

云系统管理是所提出架构的中心点。 所有关于应用程序管理和数据可靠性决策的管理都将在这里进行。 它分为两大部分：大数据应用管理和数据可靠性管理。

大数据应用管理负责为应用程序提供资源，监视应用程序的运行状况，并关闭和清理资源以及计费。 数据可靠性管理负责提高数据的可靠性、性能和减少大数据应用的网络流量。 它分为如下两个部分：

•纠删码管理：该组件具有编码器和解码器。 编码器将根据适配的纠删码将数据划分为原始和奇偶校验片段。解码器将重建丢失的数据。

•主从备份管理：这是这个概念架构中最重要的组成部分。这是我们提出的，基于节点纠删码数据故障的访问历史和故障历史记录的主动处理组件。云存储系统中的每个配置节点都会根据节点故障历史记录运行守护程序，以预测节点故障。节点故障预测器组件周期性地收集节点的故障状态（Agrawal，2015）。当状态显示节点预测失败时，将检查该节点的数据访问历史记录。基于节点访问历史，可以推导出数据块的访问模式，并且可以预测很快可能访问的数据块（Dai，2014）。这些故障预测节点中的数据块应该被动态备份管理器主动地复制到最优的下一个可用节点中。这将减少发生数据重建的可能性。另一方面，它将减少网络流量，提高大数据应用的可靠性和性能。该方法也会增加存储效率，因为它只复制哪些预测到会有故障并且接下来很有可能会被请求的数据。不可预测的故障数据块将通过典型的解码器重构来提供数据。

## 未来的研究方向

在大数据时代，提高对数据失效的容错带来了各种挑战，例如如何减少存储开销; 如何减少网络带宽和/或网络流量; 如何最小化延迟；以及最后是如何最大限度地减少能源消耗。 为了解决这些问题，我们列出了如下一些研究方向：

•在主从备份中减少存储开销而不牺牲数据的持久性和可用性的新技术：应该研究主动式故障处理和动态备份的技术，以减少备份的存储开销，同时不牺牲大数据应用程序的可靠性。

•大数据优化的纠删码：纠删码有助于减少存储开销，同时保持持久性和可用性。 但是，这是以增加网络流量、增加磁盘I / O和增大延迟为代价的。 因此，需要一种新的纠删码技术来减少网络流量、磁盘I / O和延迟，同时减少大数据的存储开销。

•提高数据可靠性的混合技术：由于主从备份和纠删编码都具有各自的优势和局限性，通过混合技术提高可靠性，充分利用每种技术优势，这是一个很有前景的研究课题。 在这个方向，混合技术的概念架构在下一节中被提出。

•基于故障预测的片段动态备份：虽然纠删码可以降低存储系统的能耗，但这是由故障发生时重建数据所需的能耗所决定的。 因此，如果算法可以预测由于故障而更可能变得不可用的数据，则如果预测项目被主动地备份，则该能量消耗就可以节省，因为在这种情况下可以避免重构。

此外，更加长远的一些研究方向还有：

•根据用户可靠性要求优化主从备份和纠删编码的配置参数的方法。

•在为大数据应用定义数据可靠性这一指标的基础上，定义云存储的地理多样性。

# 总结

云计算是大数据应用的重要支撑，因为它提供了具有成本效益的按需服务。云计算使得存储和计算资源可以根据需求快速增加和减少。随着云存储系统中的故障逐渐成为常态，云存储系统采用了各种容错机制来提高数据的可靠性。由于复制涉及巨大的存储开销，纠删码在大数据应用的云存储系统中的应用被深入推进。但是，如果发生数据故障，重建丢失的数据会涉及到许多影响应用程序性能的资源。这是的纠删码在云储存中产生了性能瓶颈。在本文中，我们讨论了这两种技术的最新研究。在纠删码的存储系统中，提出了各种技术来减少重建所需要的资源。在主从备份存储系统中，提出以最小备份量来达到高数据可靠性。本文介绍了现有的两种技术的混合的尝试，并提出了一种基于动态备份的纠删码存储系统混合技术。所提出的技术和概念架构能够有效地处理纠删码的重构问题，并且在较少的存储开销和能耗的条件下具有高可靠性。

参考文献

Abu-Libdeh, H., Princehouse, L., Weatherspoon, H., 2010. Racs: a case for cloud storage

diversity. In: Proceedings of the 1st ACM symposium on Cloud computing. ACM, pp.

229–240.

Agrawal, B., Wiktorski, T., Rong, C., 2015. Analyzing and predicting failure in hadoop

clusters using distributed hidden markov model. In: Proceedings of International

Conference on Cloud Computing and Big Data in Asia. Springer, pp. 232–246.

Araujo, J., Giroire, F., Monteiro, J., 2011. Hybrid approaches for distributed storage

systems. Data Manag. Grid Peer-to-Peer Syst., 1–12.

AWS, 2016. Summary of the Amazon DynamoDB Service Disruption and Related

Impacts in the US East Region. URL 〈https://aws.amazon.com/message/5467D2〉.

Baesens, B., 2014. Analytics in a Big Data World: The Essential Guide to Data Science

and Its Applications. John Wiley & Sons.

Beaver, D., Kumar, S., Li, H.C., Sobel, J., Vajgel, P. et al., 2010. Finding a needle in

haystack: Facebook’s photo storage. In: Proceedings of OSDI., vol. 10. pp. 1–8.

Blaum, M., Brady, J., Bruck, J., Menon, J., 1995. Evenodd: an eﬃcient scheme for

tolerating double disk failures in raid architectures. IEEE Trans. Comput. 44 (2),

192–202.

Bonvin, N., Papaioannou, T.G., Aberer, K., 2009. Dynamic cost-eﬃcient replication in

data clouds. In: Proceedings of the 1st Workshop on Automated Control for

Datacenters and Clouds. ACM, pp. 49–56.

Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., Zomaya, A.Y., 2015. Energy-eﬃcient

data replication in cloud computing datacenters. Clust. Comput. 18 (1), 385–402.

Brewer, E., Ying, L., Greenﬁeld, L., Cypher, R., Tso, T., 2016. Disks Fordata Centers.

White Paper for FAST 1(1), p. 4.

Butt, A.R., Bhattacharjee, P., Wang, G., Gniady, C., 2014. Exploring trade-oﬀs between

energy savings and reliability in storage systems. Green Comput. Book: Tackling

Energy Eﬃc. Large Scale, 149.

Cadambe, V.R., Huang, C., Jafar, S.A., Li, J., 2011. Optimal repair of mds codes in

distributed storage via subspace interference alignment. arXiv preprint arXiv:1106.

1250.

Cadambe, V.R., Jafar, S.A., Maleki, H., 2010. Distributed data storage with minimum

storage regenerating codes-exact and functional repair are asymptotically equally

eﬃcient. arXiv preprint arXiv:1004.4299.

Chen, C.P., Zhang, C.-Y., 2014. Data-intensive applications, challenges, techniques and

technologies: a survey on big data. Inf. Sci. 275, 314–347.

Chen, Y., Wang, Y., 2015. On the non-existence of minimum storage regenerating codes

with repair-by-transfer property. IEEE Commun. Lett. 19 (12), 2070–2073.

Cidon, A., Rumble, S.M., Stutsman, R., Katti, S., Ousterhout, J.K., Rosenblum, M., 2013.

Copysets: Reducing the frequency of data loss in cloud storage. In: Proceedings of

Usenix Annual Technical Conference. pp. 37–48.

Cook, J.D., Primme, R., de Kwant, A., 2014. Compare cost and performance of

replication and erasure coding. Hitachi Rev. 63, 304.

Dai, D., Chen, Y., Kimpe, D., Ross, R., 2014. Provenance-based object storage prediction

scheme for scientiﬁc big data applications. In: 2014 IEEE International Conference

on Big Data (Big Data). IEEE, pp. 271–280.

Dimakis, A.G., Godfrey, P.B., Wu, Y., Wainwright, M.J., Ramchandran, K., 2010.

Network coding for distributed storage systems. IEEE Trans. Inf. Theory 56 (9),

4539–4551.

Ford, D., Labelle, F., Popovici, F.I., Stokely, M., Truong, V.-A., Barroso, L., Grimes, C.,

Quinlan, S., 2010. Availability in globally distributed storage systems. In:

Proceedings of Osdi., vol. 10. pp. 1–7.

Gantz, J., Reinsel, D., 2012. The digital universe in 2020: big data, bigger digital

shadows, and biggest growth in the far east. IDC iView: IDC Anal. Future 2007

(2012), 1–16.

Greenan, K.M., Long, D.D., Miller, E.L., Schwarz, T.J., Wylie, J.J., 2008. A spin-up saved

is energy earned: achieving power-eﬃcient, erasure-coded storage. In: Proceedings

of HotDep.

Groenfeldt, T., 2012. Big Databig Money Says it is a Paradigm Buster.

Gunawi, H. S., Do, T., Joshi, P., Hellerstein, J.M., Arpaci-Dusseau, A.C., Arpaci-Dusseau,

R.H., Sen, K., 2010. Towards automatically checking thousands of failures with

micro-speciﬁcations. In: Proceedings of HotDep.

Harnik, D., Naor, D., Segall, I., 2009. Low power mode in cloud storage systems. In:

Proceedings of IEEE International Symposium on Parallel & Distributed Processing,

2009. IPDPS 2009. IEEE, pp. 1–8.

Hu, Y., Xu, Y., Wang, X., Zhan, C., Li, P., 2010. Cooperative recovery of distributed

storage systems from multiple losses with network coding. IEEE J. Sel. Areas

Commun. 28, 2.

Hu, Y., Lee, P.P., Shum, K.W., 2013. Analysis and construction of functional regenerating

codes with uncoded repair for distributed storage systems. In: INFOCOM, 2013

Proceedings IEEE. IEEE, pp. 2355–2363.

Huang, C., Xu, L., 2008. Star: an eﬃcient coding scheme for correcting triple storage

node failures. IEEE Trans. Comput. 57 (7), 889–901.

Huang, C., Chen, M., Li, J., 2013. Pyramid codes: ﬂexible schemes to trade space for

access eﬃciency in reliable data storage systems. ACM Trans. Storage (TOS) 9 (1), 3.

Huang, C., Simitci, H., Xu, Y., Ogus, A., Calder, B., Gopalan, P., Li, J., Yekhanin, S., et al.,

2012. Erasure coding in windows azure storage. In: Proceedings of Usenix Annual

Technical Conference. Boston, MA, pp. 15–26.

Hughes, G.F., Murray, J.F., Kreutz-Delgado, K., Elkan, C., 2002. Improved disk-drive

failure warnings. IEEE Trans. Reliab. 51 (3), 350–357.

Hussein, M.-K., Mousa, M.-H., 2012. A light-weight data replication for cloud data

centers environment. Int. J. Eng. Innov. Technol. 1 (6), 169–175.

Jin, C., Jiang, H., Feng, D., Tian, L., 2009. P-code: a new raid-6 code with optimal

properties. In: Proceedings of the 23rd International Conference on

Supercomputing. ACM, pp. 360–369.

Kermarrec, A.-M., Le Scouarnec, N., Straub, G., 2011. Repairing multiple failures with

coordinated and adaptive regenerating codes. In: Proceedings of the 2011

International Symposium on Network Coding (NetCod). IEEE, pp. 1–6.

Khan, O., Burns, R.C., Plank, J.S., Pierce, W., Huang, C., 2012. Rethinking erasure codes

for cloud ﬁle systems: minimizing i/o for recovery and degraded reads. In:

Proceedings of FAST. p. 20.

Kumar, A., Tandon, R., Clancy, T. C., 2014. On the latency and energy eﬃciency of

erasure-coded cloud storage systems. arXiv preprint arXiv:1405.2833.

Kune, R., Konugurthi, P.K., Agarwal, A., Chillarige, R.R., Buyya, R., 2016. The anatomy

of big data computing. Softw.: Pract. Exp. 46 (1), 79–105.

Le Scouarnec, N., 2012. Exact scalar minimum storage coordinated regenerating codes.

In: 2012 IEEE International Symposium on Information Theory Proceedings (ISIT).

IEEE, pp. 1197–1201.

Li, J., Li, B., 2013. Erasure coding for cloud storage systems: a survey. Tsinghua Sci.

Technol. 18 (3), 259–272.

Li, P., Li, J., Stones, R.J., Wang, G., Li, Z., Liu, X., 2016. Procode: a proactive erasure

coding scheme for cloud storage systems. In: 2016 IEEE Proceedings of the 35th

Symposium on Reliable Distributed Systems (SRDS). IEEE, pp. 219–228.

Li, W., Yang, Y., Yuan, D., 2011. A novel cost-eﬀective dynamic data replication strategy

for reliability in cloud data centres. In: 2011 IEEE Proceedings of the Ninth

International Conference on Dependable, Autonomic and Secure Computing

(DASC). IEEE, pp. 496–502.

Lin, S.-J., Chung, W.-H., 2014. Novel repair-by-transfer codes and systematic exact-mbr

codes with lower complexities and smaller ﬁeld sizes. IEEE Trans. Parallel Distrib.

Syst. 25 (12), 3232–3241.

Lin, W., Chiu, D.M., Lee, Y., 2004. Erasure code replication revisited. In: Proceedings of

the Fourth International Conference on Peer-to-Peer Computing, 2004. IEEE, pp.

90–97.

Liu, J., Shen, H., 2016. A low-cost multi-failure resilient replication scheme for high data

availability in cloud storage. In: 2016 IEEE Proceedings of the 23rd International

Conference on High Performance Computing (HiPC). IEEE, pp. 242–251.

Long, S.-Q., Zhao, Y.-L., Chen, W., 2014. Morm: a multi-objective optimized replication

management strategy for cloud storage cluster. J. Syst. Archit. 60 (2), 234–244.

Ma, A., Traylor, R., Douglis, F., Chamness, M., Lu, G., Sawyer, D., Chandra, S., Hsu, W.,

2015. RAIDshield: characterizing, monitoring, and proactively protecting against

disk failures. ACM Trans. Storage (TOS) 11 (4), 17.

Ma, Y., Nandagopal, T., Puttaswamy, K.P., Banerjee, S., 2013. An ensemble of replication

and erasure codes for cloud ﬁle systems. In: INFOCOM, 2013 Proceedings IEEE.

IEEE, pp. 1276–1284.

Mesnier, M., Ganger, G.R., Riedel, E., 2003. Object-based storage. IEEE Commun. Mag.

41 (8), 84–90.

Muralidhar, S., Lloyd, W., Roy, S., Hill, C., Lin, E., Liu, W., Pan, S., Shankar, S.,

Sivakumar, V., Tang, L., et al., 2014. f4: Facebooks warm blob storage system. In:

Proceedings of the 11th USENIX Conference on Operating Systems Design and

Implementation. USENIX Association, pp. 383–398.

O'Reilly, J., 2016. Network Storage: Tools and Technologies for Storing Your Companyas

Data. Morgan Kaufmann, Volanto, USA.

Papailiopoulos, D.S., Dimakis, A.G., 2011. Distributed storage codes through hadamard

designs. In: 2011 IEEE International Symposium on Information Theory

Proceedings (ISIT). IEEE, pp. 1230–1234.

Pei, X., Wang, Y., Ma, X., Fu, Y., Xu, F., 2015. Cooperative repair based on tree structure

for multiple failures in distributed storage systems with regenerating codes. In:

Proceedings of the 12th ACM International Conference on Computing Frontiers.

ACM, p. 14.

Pinheiro, E., Bianchini, R., Dubnicki, C., 2006. Exploiting redundancy to conserve energy

in storage systems. In: ACM SIGMETRICS Performance Evaluation Review. vol. 34.

ACM, pp. 15–26.

Pinheiro, E., Weber, W.-D., Barroso, L.A., 2007. Failure trends in a large disk drive

population. In: Proceedings of FAST., vol. 7. pp. 17–23.

Plank, J.S., 2009. The raid-6 liber8tion code. Int. J. High. Perform. Comput. Appl. 23 (3),

242–251.

Plank, J.S., 2013. Erasure codes for storage systems: a brief primer. Usenix Mag. 38 (6),

44–50.

Plank, J.S., Blaum, M., Hafner, J.L., 2013. Sd codes: erasure codes designed for how

storage systems really fail. In: Proceedings of FAST. pp. 95–104.

Qu, Y., Xiong, N., 2012. Rfh: A resilient, fault-tolerant and high-eﬃcient replication

algorithm for distributed cloud storage. In: 2012 Proceedings of the 41st

International Conference on Parallel Processing (ICPP). IEEE, pp. 520–529.

Rajasekharan, 2014. Data Reliability in Highly Fault-tolerant Cloud Systems. URL

〈https://pdfs.semanticscholar.org/abe7/

7e70864a0d914365ed755cac5ce1abc3b8b0.pdf/〉.

Rashmi, K.V., Shah, N.B., Kumar, P.V., 2011. Optimal exact-regenerating codes for

distributed storage at the msr and mbr points via a product-matrix construction.

IEEE Trans. Inf. Theory 57 (8), 5227–5239.

Rashmi, K., Shah, N.B., Gu, D., Kuang, H., Borthakur, D., Ramchandran, K., 2013. A

solution to the network challenges of data recovery in erasure-coded distributed

storage systems: a study on the facebook warehouse cluster. In: 2012 Proceedings of

HotStorage.

Rashmi, K., Shah, N.B., Gu, D., Kuang, H., Borthakur, D., Ramchandran, K., 2014. A

hitchhiker’s guide to fast and eﬃcient data reconstruction in erasure-coded data

centers. In: Proceedings of ACM SIGCOMM Computer Communication Review. vol.

44. ACM, pp. 331–342.

Rashmi, K., Shah, N.B., Kumar, P.V., Ramchandran, K., 2009. Explicit construction of

optimal exact regenerating codes for distributed storage. In: Proceedings of the 47th

Annual Allerton Conference on Communication, Control, and Computing, 2009.

Allerton 2009. IEEE, pp. 1243–1249.

Rodrigues, R., Liskov, B., 2005. High availability in dhts: erasure coding vs. replication:

erasure coding vs. replication. Peer Peer Syst. IV, 226–239.

Sathiamoorthy, M., Asteris, M., Papailiopoulos, D., Dimakis, A.G., Vadali, R., Chen, S.,

Borthakur, D., 2013. Xoring elephants: novel erasure codes for big data. In:

Proceedings of the VLDB Endowment. vol. 6. VLDB Endowment, pp. 325–336.

Shah, N.B., Rashmi, K.V., Kumar, P.V., Ramchandran, K., 2012. Distributed storage

codes with repair-by-transfer and nonachievability of interior points on the storage-

bandwidth tradeoﬀ. IEEE Trans. Inf. Theory 58 (3), 1837–1852.

Sharma, Y., Javadi, B., Si, W., Sun, D., 2016. Reliability and energy eﬃciency in cloud

computing systems: survey and taxonomy. J. Netw. Comput. Appl. 74, 66–85.

Shum, K.W., 2011. Cooperative regenerating codes for distributed storage systems. In:

Proceedings of the 2011 IEEE International Conference on Communications (ICC).

IEEE, pp. 1–5.

Shum, K.W., Hu, Y., 2011. Exact minimum-repair-bandwidth cooperative regenerating

codes for distributed storage systems. In: Proceedings of the 2011 IEEE

International Symposium on Information Theory Proceedings (ISIT). IEEE, pp.

1442–1446.

Shum, K.W., Hu, Y., 2012. Functional-repair-by-transfer regenerating codes. In:

Proceedings of the 2012 IEEE International Symposium on Information Theory

Proceedings (ISIT). IEEE, pp. 1192–1196.

Suh, C., Ramchandran, K., 2011. Exact-repair mds code construction using interference

alignment. IEEE Trans. Inf. Theory 57 (3), 1425–1442.

Suh, C., Ramchandran, K., 2010. On the existence of optimal exact-repair mds codes for

distributed storage. arXiv preprint arXiv:1004.4663.

Sun, D.-W., Chang, G.-R., Gao, S., Jin, L.-Z., Wang, X.-W., 2012. Modeling a dynamic

data replication strategy to increase system availability in cloud computing

environments. J. Comput. Sci. Technol. 27 (2), 256–272.

Tamo, I., Wang, Z., Bruck, J., 2011. Mds array codes with optimal rebuilding. In:

Proceedings of the 2011 IEEE International Symposium on Information Theory

Proceedings (ISIT). IEEE, pp. 1240–1244.

Vijayakumari, R., Kirankumar, R., Rao, K.G., 2014. Comparative Analysis of Google File

System and Hadoop Distributed File System.

Vins, V.A., Umamageswari, S., Saranya, P., 2014. A Survey on Regenerating Codes.

Vishwanath, K.V., Nagappan, N., 2010. Characterizing cloud computing hardware

reliability. In: Proceedings of the 1st ACM Symposium on Cloud computing. ACM,

pp. 193–204.

Wang, A., Zhang, Z., 2013. Exact cooperative regenerating codes with minimum-repair-

bandwidth for distributed storage. In: INFOCOM, 2013 Proceedings IEEE. IEEE, pp.

400–404.

Wang, P., Dean, D.J., Gu, X., 2015. Understanding real world data corruptions in cloud

systems. In: Proceedings of the 2015 IEEE International Conference on Cloud

Engineering (IC2E). IEEE, pp. 116–125.

Weatherspoon, H., Kubiatowicz, J., et al., 2002. Erasure Coding vs. Replication: A

Quantitative Comparison. In: IPTPS., vol. 1. Springer, pp. 328–338.

Wu, Y., 2011. A construction of systematic mds codes with minimum repair bandwidth.

IEEE Trans. Inf. Theory 57 (6), 3738–3741.

Xiang, L., Xu, Y., Lui, J., Chang, Q., 2010. Optimal recovery of single disk failure in rdp

code storage systems. ACM SIGMETRICS Perform. Eval. Rev. 38 (1), 119–130.

Yixian, Y., 1994. Peiod distribution for blaum-roth code. J. Beijing Univ. Posts

Telecommun., 3.

Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I., 2010. Spark: cluster

computing with working sets. HotCloud 10 (10–10), 95.

Zeng, W., Zhao, Y., Ou, K., Song, W., 2009. Research on cloud storage architecture and

key technologies. In: Proceedings of the 2nd International Conference on Interaction

Sciences: Information Technology, Culture and Human. ACM, pp. 1044–1048.