

纠删码存储系统更新方法综述

周 海 冯 丹 胡 燊 翀

(华中科技大学计算机科学与技术学院 武汉 430074)

摘 要 大数据时代下,数据量持续增长,存储系统中数据丢失的风险也随之不断提升。传统副本技术虽能保障数据可靠性,却带来极高的存储开销。纠删码技术则能以极低存储开销,实现与副本技术相当的可靠性,有效节省存储资源。但纠删码技术在进行数据更新时消耗大量系统计算和网络资源,任何数据块的更新需同步更新条带内所有校验块,从而维护纠删码条带中数据块和校验块的编码一致性。本文系统梳理了近年来纠删码数据更新方法的研究现状;首先,阐述纠删码的基本概念与更新基础,明确其在当前存储系统中面临的核心科学挑战;其次,从磁盘 I/O 降低、网络流量降低、并行传输优化及系统更新优化四个维度,综述相关研究进展,重点分析各类方法的关键技术特征,对具有代表性的更新方法的性能指标做了对比分析,总结其优势与不足;最后,梳理该领域尚未解决的研究挑战,并展望未来潜在的研究趋势及适配的技术路径。

关键词 纠删码;存储系统;数据更新;磁盘 I/O;网络传输

中图法分类号 TP393

DOI 号 10.11897/SP.J.1016.2026.02132

A Survey of Update Methods for Erasure-Coded Storage Systems

ZHOU Hai FENG Dan HU Yu-Chong

(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract In the era of big data, data volume continues to grow, and the risk of data loss in storage systems is also increasing accordingly. Although traditional replication technology can ensure data reliability, it incurs extremely high storage overhead. Erasure coding technology, by contrast, can achieve reliability comparable to that of replication technology with extremely low storage overhead, effectively saving storage resources. However, erasure coding technology consumes substantial system computing and network resources during data updates—any update to a data block requires synchronous updates to all parity blocks within the stripe, thereby maintaining the coding consistency between data blocks and parity blocks in the erasure coding stripe. This paper systematically reviews the research status of erasure coding data update methods in recent years: first, it elaborates on the basic concepts and update fundamentals of erasure coding, clarifying the core scientific challenges it faces in current storage systems; second, it summarizes the relevant research progress from four dimensions, namely disk I/O reduction, network traffic reduction, parallel transmission optimization, and system update optimization, focusing on analyzing the key technical characteristics of various methods, conducting a comparative analysis of the performance indicators of representative update methods, and summarizing their advantages and disadvantages; finally, it sorts out the unsolved research challenges in this field and looks forward to potential future research trends and adaptive technical paths.

收稿日期:2025-09-14;在线发布日期:2026-03-23。本课题得到国家自然科学基金面上项目(62272185)和国家自然科学基金青年学生基础研究项目(博士研究生)(623B2037)资助。周 海,博士,博士后,中国计算机学会(CCF)会员,主要研究领域为纠删码。E-mail: haizhou@hust.edu.cn。冯 丹(通信作者),博士,教授,中国计算机学会(CCF)会员,主要研究领域为计算机系统结构、大规模存储系统、非易失性内存。E-mail: dfeng@hust.edu.cn。胡燊翀(通信作者),博士,教授,中国计算机学会(CCF)会员,主要研究领域为云存储、纠删码。E-mail: yuchonghu@hust.edu.cn。

Keywords erasure codes; storage systems; data updates; disk I/O; network transmission

1 引言

随着信息技术飞速发展与互联网深度普及,数据呈爆发式增长,海量新数据持续产生并需持久化存储。国际数据研究机构 Statista 预测,2030 年全球数据总量将达 612 ZB,2035 年更将增至 2142 ZB。这些数据源于云存储、物联网终端日志、企业核心数据库、实时推荐系统等多元场景,不仅需要稳定持久的存储载体,更需具备抵御各类故障风险的能力。在大数据趋势下,存储系统多采用大规模服务器集群以降低服务成本。例如,Amazon 云在全球维护着数百万台廉价服务器,阿里和腾讯云的存储集群节点数已突破百万级。然而,服务器面临着断电、磁盘物理损坏以及网络中断等因素的影响,导致数据丢失风险显著上升。在此背景下,数据可靠性已成为制约存储系统向大规模发展的核心瓶颈。

保障数据存储可靠性的传统方案主要依赖数据冗余技术,以规避数据丢失的风险。其中,典型的代表性技术包括副本、独立磁盘冗余阵列(Redundant Array of Independent Disks, RAID)等。副本技术通过将同一文件的多个副本分散存储在不同节点上,实现数据容错,但由此带来的存储开销显著偏高。存储开销指冗余数据与原始数据的大小比率,例如三副本策略的存储开销为 3。现代数据中心需存储数十亿字节的数据,副本技术带来的硬件成本、能耗及运维压力,已令现有系统难以承受。早期分布式存储系统(如 HDFS^[1]、Ceph^[2])以及 Amazon S3 的初期版本,普遍采用三副本配置。然而,高达原始数据 3 倍的存储占用,迫使“降低存储开销”成为系统优化的核心目标。纠删码技术的引入,有效打破了“高可靠性与高存储开销”之间的固有矛盾。纠删码的核心优势在于:能够以极低的存储开销,实现与副本技术相当的容错能力。

纠删码的基本原理如下:首先将文件分割为 k 个原始数据块,通过编码算法生成 m 个校验块,随后将 $k+m$ 个块分布式存储于 $k+m$ 个节点。即便有 m 个节点同时故障,也可借助剩余 k 个块恢复全部原始数据。里德-所罗门编码(Reed-Solomon code, RS)^[3]是目前应用最广泛的纠删码技术:如 RS($k=4, m=2$)码仅需 1.5 倍存储开销,便可达到三副本的容错水平;Hadoop 分布式文件系统采用

的 RS($k=9, m=6$)码、Windows Azure 与 Facebook f4 BLOB 存储使用的 RS($k=10, m=4$)码,均是 RS 码在实际工业界的典型应用。

纠删码虽能以较低存储开销实现与副本相当的可靠性,但也给存储系统带来两大新挑战——数据修复与数据更新。随着存储数据规模持续扩大,断电、硬件故障等因素导致数据修复事件频发,目前已有大量研究^[4-9]致力于提升数据修复性能。然而,在真实数据中心里,数据更新的频率与影响远超过数据修复^[10]。雅虎数据中心的更新操作占比已接近 50%,且这一比例仍在持续上升^[11]。与副本技术仅涉及旧副本覆盖的简单逻辑不同,纠删码技术固有的线性关联特性决定了:任意一个数据块更新时,系统需同步更新条带内所有 m 个校验块,以保障数据块与校验块的编码一致性^[12-14]。整个更新过程需消耗大量计算资源、磁盘 I/O 与网络带宽,既占用系统核心资源,又会导致业务读写延迟增加、吞吐量下降,严重影响存储系统的服务质量。

在纠删码数据修复方向,学界与工业界已积累较为丰富的研究方法及综述文献^[15-18]。在纠删码数据更新方向,相关系统性综述仍相对稀缺。当前该方向最新的综述文献^[19]距今已逾五年,难以全面反映最新的研究动态。考虑到纠删码存在多样的编码结构,且存储系统各具特性,纠删码数据更新操作面临的技术挑战也存在显著差异。本文将聚焦纠删码存储系统的数据更新操作,从四个核心优化维度展开研究综述:一是以减少磁盘读写次数与寻址延迟为目标的磁盘 I/O 降低方法;二是以降低跨节点/跨机架网络传输量为核心的网络流量降低方法;三是以提升传输效率为导向的并行传输优化方法;四是以适配系统架构特性为出发点的系统设计优化方法。在此基础上,本文进一步梳理了现有方法中尚未解决的共性挑战,并结合新型存储介质与编码技术的发展现状,展望了未来可能的研究趋势,旨在为纠删码更新方法的深入应用提供理论参考。

2 纠删码概论

2.1 基本概念

在纠删码存储系统中,每个存储节点是存储设备的抽象化表示,可对应一台服务器、一台主机或一个磁盘。纠删码技术先将文件分块,再通过编码生

成校验块,随后将原始数据块与校验块共同存储于各个存储节点,以此保障数据可靠性。如图 1 所示,纠删码存储系统由多个存储节点构成,其内部包含多个条带。其中,每个条带横跨 n 个存储节点,而原始数据块与校验块均以条带为维度进行存储^[20]。参数 k 和 m 分别对应一个条带中的原始数据块数量与编码校验块数量,故 $n = m + k$ 。需要说明的是,若某种纠删码满足 Singleton 边界条件^[21](即达到理论最优存储利用率),或任意 k 个块可恢复全部原始数据,则该编码具备最大距离可分(Maximum Distance Separable, MDS)性质。具有 MDS 性质的编码,在相同容错能力下可实现最小存储空间开销。条带是编码、解码、修复与更新过程的最小操作单位。

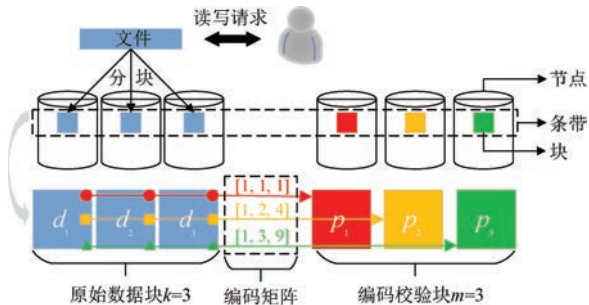


图 1 部署 RS($k=3, m=3$) 的纠删码存储系统

生成矩阵定义了原始数据块到校验块的计算规则。RS 码的生成矩阵分为上下两部分:上半部分为 $k \times k$ 的单位矩阵,下半部分为 $k \times m$ 的编码矩阵。绝大多数纠删码属于线性码^[22],通过线性组合运算将若干原始数据块生成编码校验块。如图 1 黑色虚线框内的编码矩阵所示,其对应 RS($k=3, m=3$) 码的编码过程: $k=3$ 个原始数据块通过编码矩阵生成 $m=3$ 个校验块。RS 码的编码矩阵通常采用范德蒙矩阵^[23],编码运算在伽罗华域^[24]上通过乘积线性方式完成。当条带中一个或多个数据块或校验块因节点或磁盘故障等原因丢失时,纠删码可通过条带内剩余的数据块和校验块恢复丢失的数据块或校验块,进而保障存储系统可靠性。下文将详细介绍纠删码的常用概念与基本操作。

数据块(Data Block):参与编码过程的 k 个等大小原始数据单元。如图 1 所示, d_1, d_2 和 d_3 均为数据块。

校验块(Parity Block):对 k 个数据块执行线性编码后,生成的 m 个等大小冗余数据单元。如图 1 所示, p_1, p_2 和 p_3 均为校验块。

条带(Stripe):由 k 个数据块以及对应编码生

成的 m 个校验块共同构成的数据集合。如图 1 所示, d_1, d_2, d_3, p_1, p_2 和 p_3 六个块共同组成了一个条带。

编码(Encode):在伽罗华域上对 k 个原始数据块执行乘积线性运算,进而生成 m 个校验块的过程。如图 1 所示, $k=3$ 个数据块 d_1, d_2 和 d_3 可通过异或运算生成校验块 $p_1 = d_1 \oplus d_2 \oplus d_3$ 。

解码(Decode):利用条带内至少 k 个块,通过线性运算得到至多 m 个数据块的过程,可认为是编码的逆过程。如图 1 所示,可通过 p_1, p_2 和 p_3 三个块与编码矩阵对应的逆矩阵执行线性运算,生成原始数据块 d_1, d_2 和 d_3 。

修复(Repair):当条带中某个数据块或校验块失效时,可选取条带内任意 k 个数据和校验块恢复该失效块。如图 1 所示,若数据块 d_1 因故障丢失,可通过 d_2, d_3 和 p_1 修复得到 $d_1 = d_2 \oplus d_3 \oplus p_1$ 。需说明的是,解码是修复的一部分。

更新(Update):当条带中某个数据块被重写更新时,为保障后续失效数据修复的正确性,需同步更新对应的 m 个校验块。如图 1 所示,若数据块 d_1 被更新为 d'_1 时,可通过读取旧数据块 d_1 和旧校验块 p_1 ,生成新校验块 $p'_1 = p_1 \oplus d_1 \oplus d'_1$ 。 p_2 和 p_3 的更新逻辑与 p_1 一致。

2.2 更新基础

为兼顾高容错能力与低冗余成本,学界与工业界已提出多类纠删码方案,例如侧重修复效率的局部可修复码(Local Reconstruction Codes, LRC)^[25]、旨在最小化修复带宽的再生码(Regenerating Codes, RC)^[26],以及兼具普适性与存储效率的 RS 码^[3]等。其中,RS 码因在相同容错能力下具备理论最优存储利用率,且编解码逻辑成熟、易于实现,已在 HDFS^[27]、QFS^[28]等主流存储系统中广泛应用。

从编码原理层面分析,RS 码基于线性组合运算对原始数据块实施编码,从而生成校验块。RS 码的核心逻辑可由式(1)表示:对于 k 个同等大小的原始数据块 $\{d_1, d_2, \dots, d_k\}$,RS 码在伽罗华域内通过线性加权求和,生成 m 个同等大小的校验块 $\{p_1, p_2, \dots, p_m\}$ 。其中, $x_{i,j}$ ($1 \leq i \leq k$ 且 $1 \leq j \leq m$) 是数据块 d_i 用于编码生成校验块 p_j 的系数,其取值由编码规则(如范德蒙矩阵、柯西矩阵)确定。

$$p_j = \sum_{i=1}^k x_{i,j} \cdot d_i \quad (1)$$

在存储系统实际部署时,RS 码通常将 k 个原始数据块与编码生成的 m 个校验块整合为一个完整条带。为最大化容错能力,系统将条带内的 $k +$

m 个块均匀分布到 $k+m$ 个独立存储节点。基于此分布,即便在 m 个节点同时失效的场景下,系统依然能通过剩余 k 个节点的块完整恢复全部数据。

当任意一个数据块 d_i 被更新为新数据 d'_i 时,系统需同步更新全部 m 个校验块,以保障编码一致性。其中,每个新校验块 p'_j 均通过式(2)重新生成。

$$p'_j = p_j + x_{i,j} \cdot (d'_i - d_i) \quad (2)$$

将 $d'_i - d_i$ (表示为 Δd_i) 定义为数据增量块, $x_{i,j} \cdot (d'_i - d_i)$ (表示为 Δp_j) 定义为校验增量块。进一步,若条带中 r ($1 \leq r \leq k$) 个数据块 $\{d_1, d_2, \dots, d_r\}$ 被更新为 $\{d'_1, d'_2, \dots, d'_r\}$, 则新校验块 p'_j 可通过式(3)重新生成。

$$p'_j = p_j + \sum_{i=1}^r x_{i,j} \cdot (d'_i - d_i) \quad (3)$$

当数据块中仅部分数据子块发生更新时,上述等式可推广至一般场景。RS 码借助线性编码特性,需同步更新校验块中与更新数据子块偏移地址一致的对应校验子块。

结合式(1)和(2),RS 码衍生出两种基础更新方案^[29],即读改写(Read Modify Write, RMW)与重构写(Reconstructed Write, RCW)。RMW 方案的操作逻辑为:先读取待更新数据块的旧数据,与新数据块计算得到数据增量块;同时读取所有旧校验块,将其与数据增量块进行线性运算,生成新校验块。RCW 方案则需读取条带中全部未被更新的数据块,将其与更新后的数据块共同重新编码,进而生成新的校验块。以 RS($k=4, m=2$) 编码的条带为例(如图 2 所示),假设数据块 $\{d_1, d_2\}$ 更新为 $\{d'_1, d'_2\}$ 。图 2(a) 中, RMW 先根据新旧数据块计算对应数据增量块 $\{\Delta d_1, \Delta d_2\}$; 随后,读取旧校验块 $\{p_1, p_2\}$; 最后,通过将旧校验块与数据增量块进行线性编码,从而生成新校验块 $\{p'_1, p'_2\}$ 。图 2(b) 中, RCW 先读取条带内未更新的数据块 $\{d_3, d_4\}$; 随后,将这些未更新的数据块与更新后的数据块 $\{d'_1, d'_2\}$ 重新编码,从而生成新的校验块,并最终构成新条带。

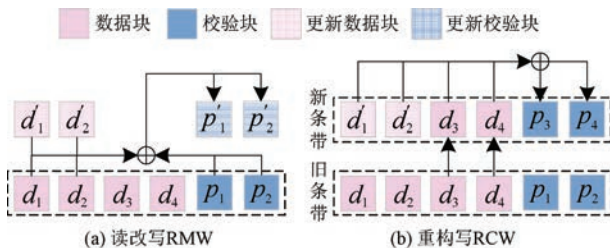


图 2 基于 RS 码的两种更新方案

本文重点聚焦单条带的更新优化研究。鉴于不同条带采用独立编码机制,且各条带间的操作彼此独立,对单条带的优化分析可向多条带场景推广。因此,本文选取单条带作为核心研究对象。

3 主要挑战

虽然将纠删码技术应用于大规模存储系统,能够以较低的成本实现高可靠性保障,从而降低存储开销;但另一方面,由其线性编码约束所引发的写放大与校验块更新问题,仍然制约系统性能。这种因结构特性导致的性能瓶颈,在很大程度上限制了纠删码技术的推广及应用。

早期芯片计算能力有限,相关研究致力于更新场景下的编解码操作优化。此类工作的核心思路是通过设计不同的编码矩阵,以期降低更新过程中伽罗华域的运算开销。这类优化方向的典型代表研究包括:基于范德蒙/柯西矩阵^[3,30]改进而来的二元分布矩阵^[31-32]、位标准矩阵^[33],以及具备最优更新复杂度的异或码^[34]与再生码^[35]。

随着 CPU 计算能力的持续提升,纠删码技术已逐渐从“理论构造”迈向“大规模商用”阶段。以 Hadoop HDFS 3.0、Amazon S3 为代表的主流存储系统,纷纷引入纠删码技术替代传统副本与 RAID 技术。

然而,受编码结构复杂性与存储系统资源有限性的制约,数据更新操作不可避免地产生大量节点内及节点间数据传输。传输过程不仅消耗节点系统资源,还需多节点协同配合。与此同时,新型存储系统架构的演进,也为纠删码的部署与落地带来了新的技术挑战。这些科学挑战共同构成了本综述后续研究的核心动机。具体而言,纠删码存储系统中数据更新操作面临的核心挑战,主要集中于磁盘 I/O、网络流量、传输调度以及系统更新四个主要维度。

磁盘 I/O: 现有基于 RS 码的纠删码存储系统主要依赖读改写 RMW 和重构写 RCW 两种基础更新方案,但二者在不同更新场景下均存在显著的读 I/O 与计算开销冗余问题,且缺乏适配场景动态变化的优化机制。一方面, RMW 方案需读取条带内待更新数据块的旧数据及所有旧校验块。因此,当条带中更新数据块数量增加时, RMW 的读 I/O 开销会呈现出线性递增趋势。另一方面, RCW 方案需读取条带内所有未更新的数据块。因此,当更新数据块的占比较低时,重新编码所有数据块的过程

会产生显著的读 I/O 开销;此外,RCW 方案的计算开销通常高于 RMW 方案。因此,当前核心挑战在于:既要针对不同的更新场景来动态选择 RMW 与 RCW 方案以降低读 I/O 开销,又需要对所选方案进行优化改进,实现读 I/O 与计算开销的动态平衡。

网络流量:受限存储系统前端应用的动态性以及节点跨地域部署的特性,网络带宽作为一种关键的稀缺资源,常处于受限状态。这意味着,实际可分配用于更新操作的带宽往往十分稀缺且不足。然而,数据更新操作往往涉及节点间的大量数据传输,这导致网络资源消耗极为显著。受此影响,网络传输已成为数据更新的核心瓶颈,并严重制约了存储系统的整体性能。从数据分布特性分析,条带内的数据块与校验块通常被分散部署于不同的存储节点乃至机架中。因此,当更新单个数据块时,系统必须同步更新条带内的全部校验块。这一过程不可避免地会产生大量的跨节点及跨机架网络流量。例如,阿里云与腾讯云存储集群的实测数据显示,更新请求引发的数据传输量分别占总传输量的 41.02% 和 41.80%^[36],印证了更新流量的显著占比。尤其在跨机架部署场景中,跨机架网络带宽的稀缺性远高于架内带宽(二者比例通常为 1:5~1:20)^[37-38],进一步加剧了更新操作引发的性能瓶颈。因此,当前核心挑战在于如何设计优化更新算法,以显著降低数据更新过程中的跨节点及跨机架网络流量开销,从而缓解带宽资源稀缺所带来的性能制约。

传输调度:为满足纠删码存储系统中的高并行更新需求,系统应构建跨层级的调度机制,旨在实现条带内及条带间更新操作的协同调度与联合优化。单个条带内存在多样化更新场景,包括部分块更新(单个数据块中部分数据更新)、全块更新(单个数据块中全部数据更新)与多块更新(条带内部分数据块更新);然而,现有调度模式受限于单点静态调度框架与无状态设计,难以根据数据块与校验块的关联性动态调整任务优先级,进而引发更新过程中的数据重传冗余。数据重传不仅占用额外网络与计算资源,更会严重制约单条带更新的速度与整体性能。与此同时,存储系统中的更新操作往往涉及多个条带,即便单条带更新性能已达最优,简单叠加多个单条带最优更新方案仍会引发网络拥塞,且无法实现多条带更新的全局最优。因此,当前核心挑战在于:一是如何构建面向纠删码的条带内和条带间并行更新模型,二是如何设计高并行动态任务生成器。通

过在复杂更新场景下实现有状态的细粒度高并行调度,旨在实现更新任务的高效分配与资源的合理利用,从而提升纠删码存储系统的更新性能。

系统更新:系统更新主要基于纠删码的读改写 RMW 和重构写 RCW 等基础方案。该过程通过适配存储系统架构、负载特征或存储介质特性,进而优化存储系统更新性能。不同存储系统中,更新方法的适配性问题尤为突出;在内存键值系统的高频小 I/O 场景中,RMW 方案因高 I/O 开销导致更新性能受限,而 RCW 方案则存在内存开销过大的问题;在对象存储跨区域部署场景中,通常采用异地更新方案以消除更新过程中旧数据块和校验块的读 I/O 开销,但该方案会导致旧数据块失效,需配套设计垃圾回收机制回收无效空间。然而,跨区域网络带宽本就稀缺,且垃圾回收过程需要迁移传输大量数据,回收过程仍会制约系统更新效率。因此,当前核心挑战在于如何将纠删码更新方案与存储系统架构深度耦合。其关键是针对性地适配不同场景下的特性与约束条件,从而系统性地优化纠删码的更新性能。

4 纠删码更新方法

当前存储系统中,数据更新操作极为常见且频繁。当数据块被更新时,纠删码需依据编码规则同步更新条带内全部 m 个校验块,以保障更新后数据块与校验块的编码一致性^[12-14]。纠删码的数据更新过程涉及磁盘读写操作与跨节点网络传输。在频繁更新场景下,更新过程会产生显著的读写开销与带宽消耗,进而成为制约存储系统整体性能的关键瓶颈。如表 1 所示,近年来相关研究从不同优化维度出发,致力于提升更新性能并降低更新对存储系统的影响。这些研究的核心优化方向包括磁盘 I/O 降低、网络流量降低、传输调度优化及系统更新优化四大类。本节将围绕上述四大方向,对近年来针对纠删码数据更新的代表研究工作展开系统梳理与总结。

4.1 磁盘 I/O 降低

纠删码的数据更新过程涉及磁盘读写操作。这一过程不仅消耗大量的内存与 CPU 资源,更因频繁的磁盘寻道操作而显著增加更新延迟。鉴于此,研究人员提出了一系列优化方案,旨在通过降低磁盘 I/O 开销,有效提升纠删码存储系统的更新性能。

表 1 纠删码数据更新相关的代表工作

优化角度	研究工作	主要方法
磁盘 I/O 降低	UCODR ^[39]	基于数据块与所有校验块间编码关联度,自适应选择重构写或读改写方案
	ECMU ^[40]	基于数据块更新比例与校验块复用性,自适应选择重构写或读改写方案
	PLR ^[10]	在校验块地址后预留一块连续空间充当日志,降低随机读取开销
	PARIX ^[41-42]	提出了预测部分写,减少重复更新数据块的磁盘读取开销
	PARD ^[43]	在日志中预留连续存储空间,将增量块和新数据块顺序写入日志
网络流量降低	LASC ^[44]	基于更新请求局部性特征,推测性缓存更新块对应部分或全部旧数据块
	CoRD ^[45]	对多个偏移地址相交的更新数据块选择性聚合编码,避免数据重传
	CAU ^[46]	根据机架间块的布局有选择地执行不同更新方案
	RackCU ^[47-48]	选择更新数据块和校验块最多的机架作为收集机架
	ESDU ^[49]	根据新旧数据块的分布动态调整编码条带的大小
并行传输优化	PUM-P, PDN-P ^[50]	将更新的计算子任务尽可能放在距离数据近的节点上处理
	T-Update, TA-Update ^[51-52]	采用树结构并行化更新时的网络传输过程
	CPU ^[53]	禁忌搜索算法寻找最优传输路径,并采用流水线技术优化传输过程
	MDTUpdate ^[54]	在多个更新的数据块和校验块之间分别构建树型传输路由
	ACOUS ^[55]	利用蚁群优化算法优化数据增量块收集和校验增量块分发的传输路由
系统更新优化	GRPU ^[56]	构建相关性图来分布数据块,感知机架间网络距离构建树型传输路由
	HybridPL ^[57]	数据块和异或校验块采用就地更新,非异或校验块采用校验日志更新
	BCStore ^[58]	采用异地更新并设计批量更新算法,将更新数据块批量编码为新条带
	GOOD ^[59]	采用异地更新降低读 I/O 开销,图算法分布数据块以提高并行读性能
	MetaEC ^[60]	解耦式数据结构实现高效元数据管理,校验日志一致性更新协议降低延迟

4.1.1 RMW 与 RCW 策略优化

文献[39]中,Shen 等人提出更新码在线数据重构算法(Update Code-Online Data Reconstruction, UCODR),其核心设计是基于块元素距离对更新任务优先级进行排序。具体而言,对每个更新的数据块,算法计算其与条带内所有校验块的二进制异或差异值——该距离越小,表明校验块与数据块的编码依赖关系越紧密。基于这一特性,UCODR 优先选择距离最小的校验块作为更新起点,利用纠删码的线性特性替代部分数据块读取操作;当块元素距离小于预设阈值时,UCODR 采用重构写 RCW 方案,否则选用读改写 RMW 方案。实验结果表明,相较于单一采用 RCW 或 RMW 的传统方案,UCODR 可平均减少 33%~50% 的磁盘读 I/O 开销。然而,该算法仅适用于单块更新场景,且更新过程中会产生大量异或计算开销。针对 UCODR 的不足,Liu 等人^[40]提出混合写策略的动态选择机制(Erasure-Code Multi-Block Update, ECMU)。该机制以数据块更新比例与校验块复用性为决策依据,在多数数据块更新场景下,通过动态选择重构写 RCW 或读改写 RMW 方案降低校验块更新过程中的磁盘 I/O 开销。当多个数据块被更新时,ECMU 利用重构写 RCW 方案直接计算新校验块,避免旧数据块和旧校验块的冗余读取;当仅单个数据块更新时,ECMU 采用 RMW 读改写方案仅读取旧数据和校验块,减少读 I/O 次数。此外,ECMU 还通过公共

异或操作复用机制,进一步降低 40%~60% 的计算开销。

4.1.2 混合更新策略优化

针对数据更新密集场景下的磁盘 I/O 开销优化问题,Zhang 等人^[43]提出保留空间和数据增量的校验日志更新方法(Parity Logging with Reserved Space and Data Delta, PARD),其核心思路是结合带保留空间的校验日志方法(Parity Logging with Reserved Space, PLR)^[10]与推测性部分更新方法(Speculative Partial Write, PARIX)^[41-42]的技术优势。如图 3 所示,当数据块首次更新时,数据节点先读取原始旧数据块并计算数据增量块,随后 PARD 将该数据增量写入校验节点的日志预留空间(例如校验块末尾保留的 4KB 连续区域);为避免校验块更新过程中的随机磁盘寻道开销,PARD 沿用 PLR 的核心优化策略——在数据文件中预分配日志预留空间,将新数据按时间顺序写入该连续区域,实现从随机写向顺序写的转换。在后续重复更新场景中,对于已更新过的数据块,PARD 仅需写入新增数据内容,无需重复读取原始数据块;校验节点则基于日志中存储的历史数据增量块与新增数据块,直接计算校验增量块,从而避免传统更新中逐次“读-改-写”的冗余操作。实验结果表明,在小写入密集型负载下,PARD 平均寻道时间减少 82%。4MB 块大小场景下,PARD 的更新吞吐量提升 186%。在更新密集型负载下,PARD 的优化效果更为显著。

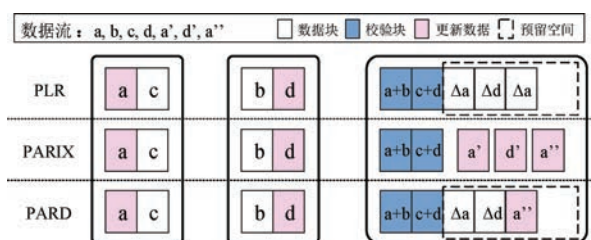


图3 保留空间和数据增量的校验日志更新方法 PARD

4.1.3 数据缓存策略优化

在一些存储系统工作负载中,小 I/O 更新请求占据主导地位^[61-62],这类请求通常引发部分块更新(即单个数据块中仅部分数据被更新)^[42,63]。数据增量的计算需从磁盘读取旧数据并缓存至内存。采用全块缓存旧数据块的策略,虽能通过一次性读取避免后续的重复 I/O,但会因加载大量未更新数据而导致内存资源严重浪费;反之,若仅缓存更新涉及的旧数据,虽可显著降低内存占用,却需在每次更新时触发磁盘访问,在频繁更新场景下,密集的磁盘寻道操作将成为制约系统性能的关键瓶颈。针对这一缓存策略权衡问题,文献[44]首次提出一种局部性感知推测性缓存方法(Locality-Aware Speculative Cache, LASC),旨在降低并平衡更新过程中的磁盘 I/O 与内存开销。LASC 对“局部性”的定义为:若两个更新请求在特定时间窗口内访问同一数据块,则认为两者具有更新局部性。已有研究^[46,64]证实,此类更新局部性在存储系统中普遍存在。LASC 通过感知更新请求流的局部性高低,动态调整其缓存粒度:在局部性较高时,优先采用全块缓存策略以最小化磁盘访问频次;而在局部性较低时,采用部分块缓存策略,仅保留必要的旧数据片段,以降低内存开销。LASC 借助一个滑动窗口计算窗口内更新流的局部性。滑动窗口按预设步长在更新流中移动,通过统计窗口内更新请求对数据块的访问频次与分布,从而量化局部性。然而,LASC 仍存在明显局限性:滑动窗口长度对更新性能的影响尚未得到系统评估,且窗口滑动过程会引发数据更新延迟,进而影响更新数据的可靠性。

总结:磁盘 I/O 开销降低是纠删码数据更新优化的核心方向之一,其核心目标在于解决传统读改写 RMW 与重构写 RCW 两种基础更新方案在不同场景下的 I/O 冗余与计算过载问题,同时降低磁盘寻道延迟对更新性能的影响。该方向的整体研究思路围绕不同更新场景展开,具体结合更新块数量(单块/多块)、更新数据比例(全块/部分块)及负载特征(小 I/O 密集型/分散型负载)等关键因素,设计动

态优化策略。为适应复杂多变的存储系统环境,现有研究通过分析对比不同更新方案的优劣特性,采用多方案混合设计思路,在特定更新场景下自适应选择最优更新方案,最终降低磁盘 I/O 开销。

当前分布式存储系统中,InfiniBand、RDMA、CXL 等支持高速互连的网络技术正快速发展,磁盘 I/O 性能已逐渐取代网络性能^[65],成为更关键的优化方向。但面对频繁且复杂的更新场景,磁盘 I/O 性能的提升依然有限。现有研究多聚焦于优化纠删码的基础更新方案,却忽略了从编码结构与系统架构两个维度开展联合优化。未来研究可通过设计高效编码结构以降低更新造成的磁盘 I/O 开销,同时针对不同编码结构和存储架构协同设计更新方法,进而提升磁盘 I/O 性能。

4.2 网络流量降低

当条带内任意一个数据块发生更新时,校验块更新过程涉及跨节点或跨机架网络数据传输。因此,部分研究工作主要致力于降低更新过程中的跨节点及跨机架网络流量,进而提升纠删码存储系统的数据更新性能。

4.2.1 跨节点网络流量优化

针对跨节点网络流量优化这一问题,Zhou 等人^[45]选取了阿里云^[66]与腾讯云^[67]的两组块级 I/O 负载进行分析。基于对更新数据在块和条带中分布特征的关联研究,结果揭示了部分块更新(块内局部数据更新)与部分条带更新(条带内局部块更新)模式的普遍存在性。现有更新方案主要分为基于块与基于增量两类,但两类方案均为特定场景设计,难以在多样化更新场景下有效降低跨节点网络流量。基于块的方案通过聚合编码多个更新数据块来更新校验块,但需传输包含未更新部分的完整数据块。在部分块更新场景下,基于块的方案会产生显著网络资源消耗。基于增量的方案仅传输更新数据子块与原始数据子块间的增量部分,并独立更新每个校验子块。当块内更新比例较高或条带校验块数量较多时,基于增量的方案在部分条带更新场景下仍会引发大量网络流量。为此,Zhou 等人设计了基于块和增量的混合更新方法(Combination of Raid-Based and Delta-Based, CoRD)。如图 4 所示,CoRD 利用增量方案传输更新部分的增量,降低未更新数据的冗余传输;同时利用块方案,对多个偏移地址相交的增量选择性聚合编码,避免增量数据重传,从而降低网络流量。此外,该方法通过合理选取聚合收集节点,并迭代组织编码多个增量数据,进一步降低网络

流量。针对跨块更新场景, CoRD 还通过翻转条带内部分数据块的存储逻辑, 进一步提升更新性能。然而, CoRD 仍存在明显局限性: 当更新数据部分之间无地址交集时, 仍需采用增量更新方案。

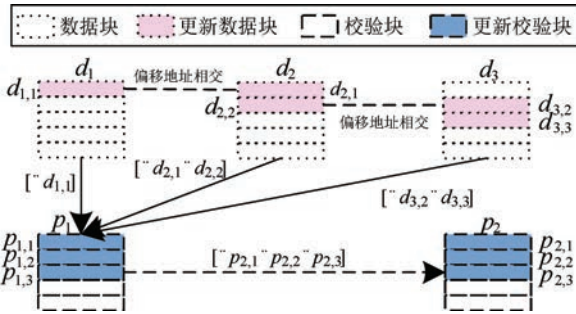


图 4 基于块和增量的混合更新方法 CoRD

4.2.2 跨机架网络流量优化

为了优化存储资源的管理, 现有分布式存储系统通常将大量存储节点分散部署于多个机架之中。其中, 位于同一机架内的所有节点, 均物理接入至同一台交换机。研究表明^[38, 68], 跨机架网络带宽仅为机架内带宽的 $1/5 \sim 1/20$, 其稀缺性已成为制约数据更新性能的核心瓶颈。针对这一问题, Shen 等人^[46]提出跨机架感知更新方法 (Cross-Rack Aware Update, CAU)。CAU 感知机架内数据块与校验块的分布布局, 自适应选择更新方案以降低跨机架网络流量。CAU 的核心思路为: 当某机架内更新的数据块数量大于更新的校验块数量时, 优先传输校验增量块, 否则传输数据增量块。CAU 的传输方式可以降低跨机架网络流量。此外, CAU 通过将更新的数据块迁移至校验块所在机架, 集中更新操作于同一机架内, 进一步减少跨机架数据传输。作为 CAU 的进阶优化方案, 机架协作更新方法 (Rack-Coordinated Update, RackCU)^[47-48] 既保障了更高的可靠性, 又最小化了跨机架网络流量。与 CAU 的延迟更新机制不同, RackCU 在数据块更新后立即触发校验块更新操作, 确保更新数据的一致性与可靠性。如图 5 所示, RackCU 通过两个阶段完成校验块更新。首先, RackCU 选取更新数据块或校验块数量最多的机架作为收集机架。图 5(a) 展示了数据增量块收集阶段, 收集机架下载其他机架的更新数据增量块。图 5(b) 展示了选择性校验块更新阶段, 收集机架将下载的数据增量块线性编码生成 m 个校验增量块, 并发送至校验块所在机架。RackCU 首次从理论上证明, 选择单个收集机架可最小化跨机架网络流量, 为纠删码跨机架更新优化提供了坚实的理论基础。

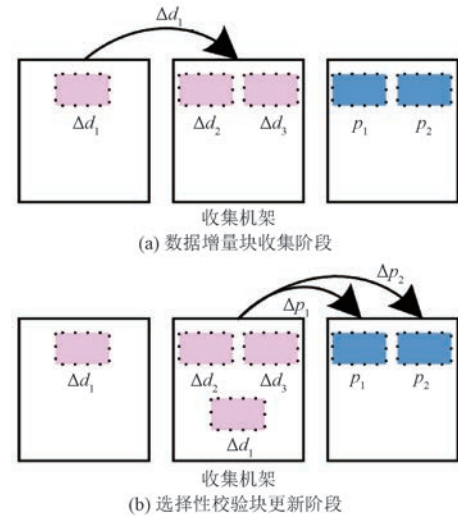


图 5 机架协作更新方法 RackCU

4.2.3 跨数据中心网络流量优化

现有基于增量更新方法存在两方面显著局限: 一是难以动态调整编码条带尺寸, 二是未针对跨数据中心环境优化网络传输, 这使得更新过程中造成大量跨数据中心网络传输量, 更新性能受到制约。

针对这一问题, Bao 等人^[49]提出一种基于弹性条带的跨数据中心纠删码增量更新方法 (Elastic Stripe-Based Delta Update Method for Cross Data Center Erasure Code, ESDU)。ESDU 的核心技术包括弹性条带划分、两阶段重复数据包定位与更新拓扑优化三个模块。首先, ESDU 通过两阶段重复数据包定位获取新旧数据的重复匹配信息, 再基于该信息动态调整条带尺寸, 以此避免插入或删除操作引发的后续条带偏移问题。同时, 该方式可以最大化匹配数据包的占比, 减少不匹配数据包与增量数据包的传输开销。ESDU 借助非重叠树算法生成最优条带划分方案, 能够确保后续条带的数据包偏移保持稳定, 进而降低跨数据中心场景下增量数据的传输量。此外, ESDU 还结合存储节点的地理位置信息与编码矩阵的线性特性, 构建跨数据中心流量最优的更新拓扑, 从而降低跨数据中心网络流量。ESDU 为跨数据中心纠删码存储系统提供了高效的校验更新方法, 能够显著降低跨数据中心的网络流量, 缩短更新延迟, 适用于代码仓库备份、实时数据同步等高频数据更新场景, 同时兼顾低冗余特性与高容错能力, 实现了性能与可靠性的动态平衡。

总结: 本节介绍的相关研究致力于降低纠删码更新环节中存在的跨节点、跨机架乃至跨数据中心的传输流量, 从而提高更新性能。

考虑一个条带采用 $RS(k, m)$ 编码方案, 数据块

大小为 B 。假设该条带中有 r 个数据块发生更新,各更新块对应的更新部分大小为 $b_i (1 \leq i \leq r)$ 。采用 CoRD 方法对 r 个更新数据块进行聚合编码后,传输数据量大小为 $B' (B' \leq B)$ 。表 2 从优化场景、更新方法及传输流量等维度,对上述四种更新方法进行了对比分析。其中,CoRD 面向一般分布式存储架构,CAU、RackCU 与 ESDU 则针对更高层级的跨机架或跨数据中心架构,四种方法均采用就地更新方案。网络流量传输方面,CoRD 因仅传输更新的数据部分,实现了理论最优的传输流量;CAU 和 RackCU 的跨机架网络流量上下限保持一致;ESDU 的网络传输流量取决于更新的数据块和校验块数量。

表 2 网络流量降低方法对比

更新方法	优化场景	更新方案	传输流量
CoRD ^[45]	跨节点	就地更新	$\sum(b_i) + B' \cdot (m-1)$
CAU ^[46]	跨机架	就地更新	最低: $B \cdot \min(r, m)$ 最高: $B \cdot (r+m-1)$
RackCU ^[47-48]	跨机架	就地更新	最低: $B \cdot \min(r, m)$ 最高: $B \cdot (r+m-1)$
ESDU ^[49]	跨数据中心	就地更新	$B \cdot (r + \min(r, m))$

因此,针对一般跨节点场景,优先选用 CoRD 方法可实现更低的网络流量。在大型数据中心场景中,跨机架或跨数据中心的网络带宽更具稀缺性且成本更高,系统可选择 CAU、RackCU 与 ESDU 方法。

4.3 并行传输优化

此外,存在一类研究工作依据存储系统的网络带宽情况,采用了并行化的优化思路。通过设计不同结构的传输拓扑,该类研究工作能够显著加速数据更新的进程。

4.3.1 单数据块更新优化

基于更新管理器的数据和校验块更新方法 (Parity Blocks by a Updating Manager and the Parity Nodes, PUM-P) 和 (Parity Blocks by the Data Nodes and Parity Nodes, PDN-P)^[50] 采用典型的星型传输架构。该方法将计算子任务优先分配至数据就近节点执行,从而降低数据传输延迟。树结构更新方法 (Tree-structured Update, T-Update)^[51] 与自适应树结构更新方法 (Tree-structured Adaptive Update, TA-Update)^[52] 则采用树型传输路由。如图 6(a) 所示,两种方法在更新的数据块与所有校验块之间构建更新树。通过采用 Prim 算法求解总传输时间最小的最优树结构, T-Update 和 TA-Update 避免星型架构中心节点传输负载过重所导致

的性能瓶颈。近年来,大量纠删码更新优化的研究工作,均围绕上述两种传输架构展开。这些研究旨在进一步提升并行传输的效率,从而显著优化系统的更新性能。

Wu 等人^[53] 进一步采用并行流水线技术提升单块更新性能,并设计一种跨机架流水线更新方法 (Cross-Rack-Aware Pipelining Update, CPU)。如图 6(b) 所示,该方法的核心思路在于,将完整的更新数据块切分为若干个等大小的子块,并使每个子块沿优化后的传输路径在多个节点间进行并行传输。这一机制能够充分利用节点间的空闲网络带宽,从而显著提升更新效率。针对路径优化所面临的 NP-hard 挑战, CPU 采用禁忌搜索算法,通过最大化最小链路带宽的策略来确定最优传输拓扑;并结合离线学习机制自适应地调整数据块切片大小,从而在传输效率与资源开销之间达成动态平衡。在实际传输过程中,数据块经由切分处理后形成定长切片,并通过流水线路径并行传输,进而更新校验块。然而, T-Update 与 CPU 均侧重于对单数据块更新场景进行优化。当条带涉及多个数据块的并行更新时,如何借助多块协同机制以高效完成校验块的更新,仍然是当前亟待解决的技术挑战。

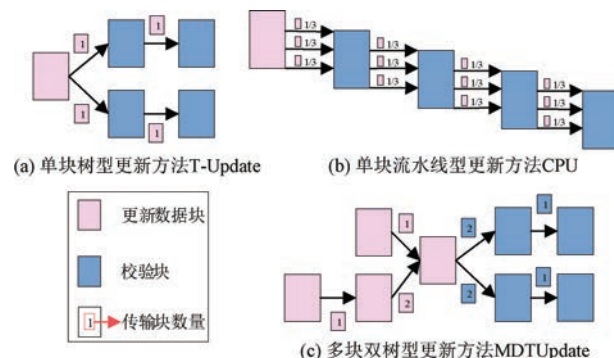


图 6 基于树型和流水线技术的并行传输优化方法

4.3.2 多数据块更新优化

Zhou 等人^[54] 对微软存储工作负载 MSR^[69] 展开分析,发现部分负载中的大 I/O 更新请求与延迟更新策略^[57,59,70] 会导致单个条带中出现多个更新的数据块。 T-Update 与 CPU 在应对多数据块更新场景时,需要构建多个更新树或多条流水线。该方式会造成部分节点网络拥塞,进而降低更新性能。针对这一问题, Zhou 等人在 T-Update 基础上,设计了适用于一般存储系统架构的多块双树结构快速更新方法 (Multi-Block Double Tree Update, MDTUpdate)。如图 6(c) 所示, MDTUpdate 的核心设计在于,分别在多个更新数据块与校验块之间构建

树型传输结构。这一方法不仅充分利用了节点间的网络带宽资源,还能够借助多个更新数据块间的协同机制高效完成校验块更新,从而降低总更新时间。为加速更新树的构建进程,MDTUpdate 采用了一种快速贪心算法。该算法依据节点间网络带宽大小,能够迅速确定树构建过程中的节点选取顺序及其插入位置。此外,MDTUpdate 还从理论上证明了:基于该贪心算法构建的更新树,能实现总更新时间的最小化。

Li 等人^[55]采用相似优化思路,针对多块更新场景提出基于蚁群优化算法的多数据节点更新方案(Ant Colony Optimization Algorithm based Multiple Data Nodes Update Scheme, ACOUS)。该方案通过蚁群优化算法高效完成数据增量收集与校验增量分发过程,进一步优化多节点协作更新的过程。然而,MDTUpdate 与 ACOUS 均未采用流水线技术优化传输过程。

针对多块更新场景与跨机架存储架构,Jia 等人^[56]设计了基于图的跨机架并行更新方法(Graph-based Cross-Rack Parallel Update, GRPU)。GRPU 通过优化数据布局并利用并行传输技术,减少跨机架网络流量并实现机架间负载均衡。通过设计相关性图驱动的数据布局策略,GRPU 构建相关性图描述数据块间的更新相关性,将相关性最高的数据块重组至同一条带且分配到同一机架。如图 7 所示,两个数据块的相关性定义为特定时间窗口内的共同更新次数,次数越高则相关性越大。图 7(a)展示了九个数据块间的相关性并构建相关性图;图 7(b)将相关性图分割为多个子图,每个子图包含 k 个相关性最高的数据块;图 7(c)对每个子图中的数据块重新编码组织,生成新的纠删码条带。通过上述布局优化,GRPU 可将更新的数据块集中于同一条带,减少校验块的更新次数,进而降低更新过程中产生的网络传输量。

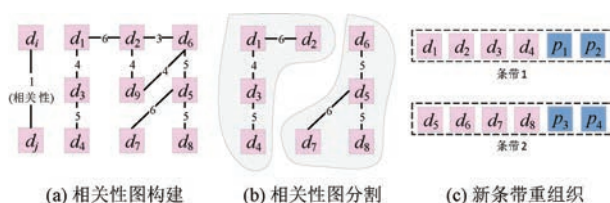


图 7 基于相关性图的条带重组织方法

此外,GRPU 利用机架间的网络距离构建并行传输树,选择网络距离最短的传输路径并动态分配传输任务,最终实现机架间的负载均衡与跨机架链

路的并行传输。相较于 MDTUpdate 与 ACOUS 仅聚焦于节点间传输路由优化,GRPU 不仅利用数据相关性降低了多数据块更新场景下的网络流量,还实现了机架间的并行传输,进一步提升了跨机架存储架构的更新性能。

总结:并行传输优化旨在通过构建高效的传输拓扑及任务调度机制,在提升多块与多条带更新并行效率的同时,避免节点过载与链路拥塞问题。该研究的整体思路是:依据更新规模及网络架构特性,设计高效的并行传输结构,同时借助智能算法优化传输路径与任务调度。

现有研究依据更新场景的差异展开了分类优化。针对单块更新场景,T-Update 和 TA-Update 采用树型传输拓扑取代了传统的星型架构,从而有效避免了中心节点负载过大的问题;CPU 方案则进一步引入流水线切片技术,通过将待更新数据块拆分并沿优化路径并行传输,以充分利用节点间的空闲带宽。而针对多块更新场景,MDTUpdate 设计出双树协同传输结构,不仅在多更新块与校验块间实现了协同传输,还结合快速贪心算法构建了最优更新树;ACOUS 则采用蚁群算法,对数据增量的收集与分发过程进行优化;GRPU 则聚焦于跨机架更新优化,其通过相关性图驱动数据布局,将高关联性数据块聚合至同一机架,并基于机架间网络距离构建并行传输树,最终实现跨机架层面的负载均衡。

然而,上述研究均依赖于一个核心前提,即网络带宽环境是稳定的。虽然已有诸多研究工作^[71-72]能够快速获取实时网络带宽值,旨在降低带宽波动对更新时间的影响。然而,在实际存储系统中,节点带宽始终呈现出动态变化的特征,且节点间的上下行带宽往往存在非对称性;加之现有研究所采用的网络模型多为对实际拓扑的简化,上述局限性致使当前研究工作难以直接适配动态复杂的真实网络环境。未来研究应进一步结合实际网络拓扑结构来设计优化算法,从而提升算法对复杂网络环境的适配能力。同时,研究还需完善跨节点、机架、区域的多层级协同更新调度机制,以期有效应对真实存储系统中的网络动态性与拓扑复杂性。

4.4 系统更新优化

最后,现有研究工作虽能够优化纠删码的数据更新性能,但往往会对系统的其他性能产生潜在影响,涵盖数据编码、修复及读写性能等方面。为此,部分研究工作从系统整体架构出发,在保障系统原有核心性能不受损的前提下,针对性优化纠删码的

数据更新性能,实现更新性能与系统性能的动态平衡。

4.4.1 键值存储系统更新优化

针对内存键值存储系统中频繁数据更新引发的大量磁盘 I/O 开销与高内存占用问题,Cheng 等人^[57]提出混合就地更新和校验日志更新的架构(Hybrid of In-Place Update and Parity Logging, HybridPL),旨在提升纠删码内存键值存储系统的修复与更新性能,同时降低内存占用开销。HybridPL 采用分层存储设计,对更新数据块与异或校验块采用就地更新方式^[12,73-74],以降低内存占用开销;对非异或校验块则采用校验日志更新方式^[63,75],以提升更新性能。在存储节点分配上,HybridPL 将数据块与异或校验块存储于 DRAM 节点,借助 DRAM 的高带宽特性保障单节点故障修复性能;将非异或校验块存储于磁盘节点,以降低频繁更新场景下的网络传输流量。此外,HybridPL 还设计了基于合并的日志缓冲技术与校验日志节点辅助修复技术,进一步提升日志节点的更新性能与系统的多块修复性能。

另有一类研究采用异地更新方式^[76]来消除旧数据的读操作。当数据发生更新时,该方式不会在原始存储位置用新数据覆盖旧数据,而是将新数据写入新的存储空间中。异地更新方式虽能避免对旧数据块与旧校验块的磁盘读操作,但原始旧数据会随之失效。因此,异地更新方式需要设计相应的垃圾回收机制回收原始条带中的无效存储空间。例如,文献^[58]提出一种键值存储下基于批量编码的带宽高效更新方法(Bandwidth-Efficient In-Memory KV-Store with Batch Coding, BCStore)。BCStore 采用异地更新方式,通过批量缓存新数据块,并重新编码新数据块来生成新校验块,避免旧校验块的逐次更新开销。为回收无效的旧数据块,BCStore 设计了基于块迁移的垃圾回收机制,将有效数据块从无效块比例高的条带迁移至无效块比例低的条带,同时释放无效块比例高的条带所占用的存储空间。针对更新数据大小各异的场景,BCStore 引入虚拟编码条带机制。该机制可通过固定虚拟空间对齐数据块,支持不同大小的对象填充至同一物理条带,减少块迁移过程中的空间碎片问题,提升垃圾回收效率。理论分析证明,即使在最坏场景下(每个条带仅含一个无效块),BCStore 的总流量开销仍不超过传统就地更新方式。此外,BCStore 还根据键的访问热度对更新请求进行排序,将

访问热度高的对象(热对象)集中存储于少数条带。该方式可降低垃圾回收过程中的块迁移开销,进一步减少网络带宽消耗并提升系统整体性能。

4.4.2 分布式存储系统更新优化

Deng 等人^[59]提出一种图辅助的异地更新方法(Graph-Assisted Out-of-Place Update, GOOD),旨在解决纠删码存储系统中更新操作引发的高 I/O 延迟与负载不均衡问题。GOOD 采用异地更新方式,借助更新缓冲区暂存待更新数据块。当缓冲区缓存数据量达到预设阈值时,GOOD 将其批量重组为新条带,以此避免频繁 I/O 操作带来的性能损耗。此外,GOOD 还构建加权二分图模型,结合文件访问热度与元数据修改难度,从而为二分图中的边分配权重。该方法采用最大权匹配算法,对更新数据块与存储节点间的匹配过程进行了优化,旨在确保更新后的文件块分布保持尽可能均匀。通过维持文件读取的并行访问能力,该策略有效避免了因同一文件数据块集中于少数节点而引发的并行读取性能下降问题。最后,GOOD 根据条带中无效块的比例,优先对无效块比例高的条带实施回收,同时保持块分布均衡性。

4.4.3 分离式内存系统更新优化

分离式内存系统通过解耦计算与内存资源,有效提升整体资源利用率。中科大 Li 等人^[60]发现,分离式内存系统环境中的 KV 存储系统,在数据更新过程中面临两大核心技术难点。其一,纠删码元数据管理效率低下。传统基于多层索引与指针的数据结构,在分离式内存系统中需跨内存节点执行远程访问。然而,内存节点计算能力较弱,多轮网络往返会导致索引查询延迟显著增加。其二,更新一致性保障与并发冲突问题。分离式内存系统中内存节点计算能力有限,难以支持多版本校验块管理。传统两阶段提交协议的跨节点传输开销较大,不仅易引发更新延迟,还可能带来数据一致性风险。针对上述难点,研究团队提出一种在分离式内存上构建的纠删码键值存储系统(Erasure-Coded KV-Store on Disaggregated Memory, MetaEC)。MetaEC 通过解耦式纠删码元数据结构,设计三类 8 字节对齐的元数据格式,适配 RDMA 操作以减少远程访问轮次。同时,MetaEC 利用基于校验日志的一致性更新协议,更新过程中直接修改 KV 对,并将校验增量日志存储于内存节点,避免旧校验块的读取开销。借助写锁与读锁机制实现条带级并发控制,MetaEC 有效解决多任务更新冲突问题。然而,MetaEC 仍

存在明显局限性:一方面,其未集成自适应编码机制,难以动态调整冗余参数以适配数据热度的动态变化;另一方面,校验日志长期积累易产生存储碎片,且系统缺乏高效的日志合并策略,影响系统运行性能。

总结:系统更新优化从存储系统整体架构出发,核心目标是实现更新性能与其他系统性能的动态平衡。如表 3 所示,近年来的相关研究围绕一般分布式集群、内存键值存储及分离式内存三类典型存储系统展开,所采用的更新涵盖就地更新、异地更新及日志更新三类基本方式。针对内存键值存储系统,HybridPL 采用就地更新与日志更新的混合方式,实现更新过程中网络流量与内存开销的平衡;BCStore 则采用异地更新方式,需配套设计垃圾回收机制以回收无效数据。针对一般分布式存储系统,GOOD 在采用异地更新方式时,还兼顾了新数据迁移后的并行读取性能与元数据管理开销问题。最后针对分离式内存系统的优化,MetaEC 有效解决了元数据远程访问延迟及更新过程中的并发读写冲突问题。

表 3 系统更新优化方法对比

更新方法	存储架构	更新方案	优化指标
HybridPL ^[57]	内存键值	就地+日志更新	网络流量 内存开销
BCStore ^[58]	内存键值	异地更新	网络流量 垃圾回收
GOOD ^[59]	分布式	异地更新	并行读 元数据管理
MetaEC ^[60]	分离式内存	日志更新	元数据管理 并发读写

上述系统更新优化方案结合当前主流存储系统的自身特性与纠删码更新过程展开联合优化,实现了更新性能与系统整体性能的提升。然而,现有方案仍存在两方面显著局限:其一,当前系统更新优化方案高度依赖特定存储架构的特性,相关算法与技术缺乏跨不同系统架构的通用适配性,难以迁移至不同类型的存储系统中;其二,当前方案大多基于静态系统参数进行设计,难以有效适配存储系统中数据特征、业务负载及硬件资源的动态变化,适配性与鲁棒性不足。

5 研究挑战与趋势

近年来,纠删码存储系统中的数据更新方法仍面临不少的研究挑战,这些挑战构成了后续研究趋

势。同时,随着纠删码逐步拓展至各类新型存储设备与系统,纠删码在新型应用场景中势必衍生出更多新的问题与挑战,这也为未来研究趋势指明了重要方向。

5.1 研究挑战

目前,纠删码的数据更新方法大多聚焦于单条带场景的优化;此外,异地更新方式中的垃圾回收策略尚未得到深入研究与设计;最后,存储系统中网络带宽的动态变化特性,也为数据更新过程带来了额外挑战。

多条带更新:为降低频繁更新场景下的更新开销,现有方案多采用附加-提交^[46]与缓冲区日志^[57,59,70]技术。这类技术将更新数据暂存于日志缓冲区,直至缓冲区达到预设阈值后触发批量更新。这一机制易导致更新触发时存在多条待更新条带,未来研究工作需重点研究以下两个关键问题。第一,条带更新优先级的动态选择策略。部分条带可能在多轮更新触发中被多次更新,系统需要重复多次更新校验块。可通过合并此类重复更新请求,系统仅执行一次校验块更新,可显著降低整体更新成本;同时,未来研究工作需结合存储系统当前的可用资源状态(如网络带宽、节点算力),动态分配条带更新优先级,确保资源高效利用。第二,多条带并行更新的高效调度机制。当单个批次内多条条带同时更新时,这一过程易造成节点间流量负载不均;加上网络带宽资源稀缺,这会导致更新过程中网络拥塞,进而严重制约多条带更新性能。以 RackCU 更新方案为例,该方案为每条条带指定一个收集器节点,负责数据增量块的收集与校验增量块的分发。如图 8 所示,通过合理选择各条带的收集器节点,可实现网络流量的负载均衡。因此,如何通过条带内收集器节点的优化选择实现节点间负载均衡,并高效调度条带间并行更新,是未来的研究重点。

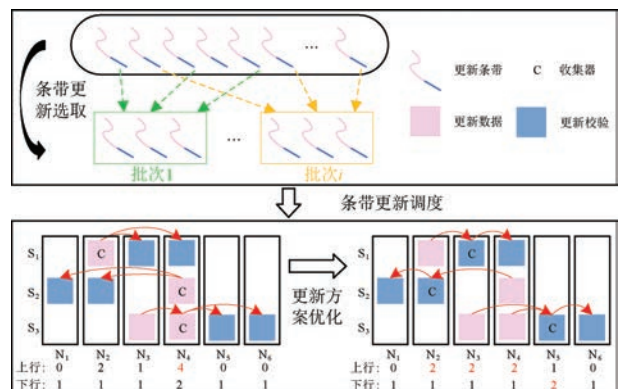
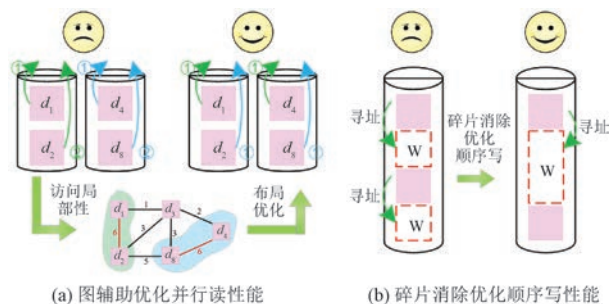


图 8 多条带更新下的条带选取和负载均衡

无效数据回收:异地更新方式会将更新后的数据写入新的存储空间,使得原始旧数据变为无效数据。无效数据会占用存储空间,造成存储资源浪费。然而,存储系统无法直接删除这类无效数据,原因是需维持同一条带中剩余有效数据块的编码一致性。回收无效数据会产生大量磁盘 I/O 与网络流量,抢占系统总线带宽及网络带宽资源,进而影响系统前端正常应用的服务性能(包括延迟与吞吐量指标)。针对这一问题,需重点研究以下两方面内容:一是如何设计回收策略,以降低回收过程中的资源开销?云存储场景下应用负载复杂多样,这使得存储数据呈现出复杂多变的特征,包括数据状态和分布等。现有纠删码存储系统的回收算法^[58-59]存在设计场景单一、资源开销大、易影响系统服务性能等问题,尚未针对回收数据的多样化特征来设计回收方案。二是如何保障回收操作后的系统数据读写性能?回收过程中需对原始有效数据块执行迁移操作,这可能导致访问相关性强的数据块被迁移至同一节点,造成读并行性能降低。如图 9(a)所示,若将具有强访问相关性的数据块分散存储于不同节点,可有效提升读并行性能。此外,回收过程中,有效数据的迁移与无效数据的释放会产生大量空闲存储空间,此类空间后续将被写入新数据。如图 9(b)所示,通过在回收期间将此类空闲空间进行合并,可降低新数据写入所需的磁盘寻址,进而提升新数据写性能。因此,未来研究工作需优化数据块迁移布局与空闲空间管理策略,实现回收后读写性能的保障。



导致特定机架或区域的网络流量骤增,基于增量的更新方案则受限于最慢机架/区域的网络传输速度。因此,系统设计者亟需针对跨机架、跨区域场景,设计适配大条带特性的高效数据更新算法,平衡网络负载。此外,纠删码本身具有高计算复杂度,其编码过程需依赖 FPGA、ASIC 等专用硬件加速,然而,大条带包含更多数据块,进一步提升了硬件计算能力的要求。在此背景下,大规模数据编码产生的编码开销正逐渐成为制约更新性能的核心瓶颈。因此,针对纠删码的数据编码问题,从编码方案的理论构造、编码加速库的选型与设计,到系统级的工程实现,构成了后续潜在的研究趋势。

针对大条带场景下多数据块同时更新的核心挑战,可采用借鉴的技术手段如下:对于大条带中更新数据块数量较多的问题,可对数据块进行分组。每个分组采用 MDTUpdate^[54] 更新方法,即组内数据块与校验块分别构建树型传输结构。该方式可避免单个收集器传输过载。针对跨机架或跨区域更新场景,可在机架或区域内部先完成数据增量的线性聚合,并根据实时网络负载状态动态选择基于块或基于增量的更新方案来均衡更新时机架或区域间网络传输负载。最后,采用软硬件结合思路来加快更新过程中的编码进程。通过协作 Intel 指令集、GPU 和 FPGA 对编码进行加速。同时,系统可基于多核并行架构设计流水线编码算法,通过任务拆分与并行调度进一步优化编码速度。

5.2.2 再生码更新

当前纠删码数据更新领域的研究多聚焦于应用最为广泛的 RS 码,对再生码(如 Butterfly 码^[83] 和 Clay 码^[84] 等)的更新优化研究相对匮乏。再生码凭借低于 RS 码的修复开销,已吸引大量研究聚焦其修复机制^[85-87]。再生码的编码结构具有特殊性,每个数据块会被分割为多个子块。这种特性会使得再生码的编码复杂度显著高于 RS 码。此前针对 RS 码设计的更新方法,难以直接迁移应用至再生码场景。在算法优化层面,研究者需深入探索更高效的编解码算法,以大幅降低再生码更新过程中的计算复杂度。其次,再生码在修复阶段通过精心设计的编码结构实现修复带宽的降低,但再生码的更新操作往往会削弱这一核心优势。当多个数据块同时更新时,系统需同步调整数据块与校验块的线性组合关系,进一步增加了更新难度。此外,再生码的编码结构对数据块的分布具有强依赖性,更新操作易引发条带碎片化的问题。同时,数据块更新频率的差

异可能导致节点间更新负载的不均衡。综上,再生码的更新优化需在计算效率提升、修复与更新性能权衡、数据分布均衡三大核心挑战上展开研究,为构建高效可靠的存储系统提供支撑。

针对再生码更新的核心挑战,可采用借鉴的技术手段如下:其一,针对再生码计算效率偏低的问题,设计者可探索借助信息流图模型构建逼近计算开销下界的编码结构。在分析过程中需兼顾所设计的结构对修复性能的影响,尝试通过机器学习技术快速高效地搜索不同需求约束(如更新性能、修复性能等)下的最优编码结构。其二,为避免更新过程中节点间网络传输负载失衡的问题,设计者可对数据块与校验块进行重新分布与迁移,以均衡节点间的传输负载,降低网络拥塞风险,进一步提升更新性能。

5.2.3 新型存储介质

早期纠删码技术多应用于机械硬盘 HDD 存储系统,对存储数据执行编码保护。当前以华为和浪潮为代表的多家企业已逐步布局研发“去 HDD 化”的全闪存存储集群,该类集群以固态硬盘 SSD 为核心存储介质。此前面向 HDD 集群设计的纠删码更新方法,难以直接迁移至大规模 SSD 全闪集群,原因在于 SSD 的独特硬件特性催生了新的技术挑战。传统更新方法未充分考虑 SSD 的磨损均衡需求与写放大问题。因此,如何充分利用 SSD 高吞吐量与写放大特性加速纠删码更新过程,同时兼顾 SSD 寿命保障设计高效更新算法,成为未来的研究方向。

针对新型存储介质下的更新挑战,可采用借鉴的技术手段如下:针对现有纠删码更新研究未充分兼顾 SSD 磨损均衡这一核心问题,设计者需深入分析更新操作对 SSD 磨损的影响。设计面向纠删码的磨损均衡感知数据分发与校验块迁移算法,以均衡节点内及节点间的 SSD 磨损状态,从而延长 SSD 使用寿命。针对现有更新方案易加剧 SSD 写放大的问题,探索基于更新合并和基于分层 SSD 架构(含三层存储单元 TLC SSD 与四层存储单元 QLC SSD)的纠删码高效更新技术,通过对多个更新请求进行合并来减少 SSD 写入次数,同时将冷热更新数据分层存储来延长 SSD 使用寿命,实现更新效率与存储介质耐久性的协同优化。

6 总 结

信息技术的飞速发展推动数据量呈爆发式增长,分布式存储系统凭借良好的可扩展性与高可用

性,已成为海量数据存储的核心载体。随着存储系统规模持续扩大,硬件故障频发问题日益凸显,低冗余成本的纠删码技术因能在保障数据可靠性的同时降低存储开销,被广泛应用于存储系统中。在基于纠删码的存储系统中,需同时满足两大核心需求:一是为前端用户提供高效的数据访问服务,二是通过后端纠删码更新操作维护数据一致性。然而,纠删码更新操作往往会产生大量磁盘 I/O 开销与网络传输流量,且会占用大量计算和网络等系统资源,进一步影响更新性能。鉴于此,本文首先系统分析了影响纠删码数据更新性能的核心因素,随后从四大技术维度,对近年纠删码数据更新领域的相关研究工作进行了全面综述与深入探讨:

磁盘 I/O:这类研究依据更新请求的分布特征与数据访问规律,选择合适的更新策略,核心目标是减少更新过程中的磁盘读写 I/O 与寻址延迟;

网络流量:针对更新数据在块内、节点间的分布特性,这类研究结合就地更新、异地更新等不同更新方式,优化节点间、机架间及跨区域的数据传输路径与数据量,降低网络传输开销,进而提升整体更新性能;

并行传输:借助树型传输结构、流水线并行等技术,这类研究并行化更新数据的传输与处理过程,降低更新时间,加速更新进程;

系统更新:这类研究聚焦整个存储系统的更新性能提升,包括更新与修复性能的协同优化、数据并行读取和顺序写入性能优化等多个方向,从系统设计层面提升更新性能。

最后,本文结合近年来纠删码数据更新领域的最新研究动态,深入剖析了现有更新方法存在的不足与核心技术挑战,并对未来的研究趋势进行了展望,为后续相关研究提供参考。

参 考 文 献

- [1] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system//Proceedings of the 26th Symposium on Mass Storage Systems and Technologies. Incline Village, USA, 2010:1-10
- [2] Weil S A, Brandt S A, Miller E L, et al. Ceph: a scalable, high-performance distributed file system//Proceedings of the 7th Symposium on Operating Systems Design and Implementation. Berkeley, USA, 2006:307-320
- [3] Reed I S, Solomon G. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 1960,8(2):300-304
- [4] Vajha M, Ramkumar V, Puranik B, et al. Clay codes: moulding MDS codes to yield an MSR code//Proceedings of the 16th USENIX Conference on File and Storage Technologies. Oakland, USA, 2018:139-154
- [5] Zhang G, Huang Z, Ma X, et al. RAID+: deterministic and balanced data distribution for large disk enclosures//Proceedings of the 16th USENIX Conference on File and Storage Technologies. Oakland, USA, 2018:279-294
- [6] Subrata M, Rajesh P, Moo R, et al. Partial-Parallel-Repair: a distributed technique for repairing erasure coded storage//Proceedings of the European Conference on Computer Systems. London, UK, 2016:1-16
- [7] Li R, Li X, Lee P P C, et al. Repair pipelining for erasure-coded storage//Proceedings of the USENIX Annual Technical Conference. Santa Clara, USA, 2017:567-579
- [8] Lin S, Gong G, Shen Z, et al. Boosting full-node repair in erasure-coded storage//Proceedings of the USENIX Annual Technical Conference. Virtual, 2021:641-655
- [9] Li X, Cheng K, Tang K, et al. ParaRC: Embracing sub-packetization for repair parallelization in MSR-Coded storage//Proceedings of the USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2023:17-32
- [10] Chan J C W, Ding Q, Lee P P C, et al. Parity logging with reserved space: towards efficient updates and recovery in erasure-coded clustered storage//Proceedings of the 12th USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2014:163-176
- [11] Cheng Liang-Feng. Research on efficient erasure coding technologies for in-memory key-value stores. Wuhan: Huazhong University of Science and Technology, 2022(in Chinese)
(程良锋. 面向内存键值存储系统的高效纠删码技术研究. 武汉:华中科技大学,2022)
- [12] Aguilera M K, Janakiraman R, Xu L. Using erasure codes efficiently for storage in a distributed system//Proceedings of the International Conference on Dependable Systems and Networks, Yokohama, Japan, 2005:336-345
- [13] Goodson G R, Wylie J J, Ganger M K, et al. Efficient Byzantine-tolerant erasure-coded storage//Proceedings of the International Conference on Dependable Systems and Networks. Florence, Italy, 2004:135-144
- [14] Frolund S, Merchant A, Saito Y, et al. A decentralized algorithm for erasure-coded virtual disks//Proceedings of the International Conference on Dependable Systems and Networks. Florence, Italy, 2004:125-134
- [15] Luo Xiang-Hong, Shu Ji-Wu. Summary of research for erasure code in storage system. *Journal of Computer Research and Development*, 2012,49(1):1-11(in Chinese)
(罗象宏,舒继武. 存储系统中的纠删码研究综述. 计算机研究与发展,2012,49(1):1-11)
- [16] Wang Yi-Jie, Xu Fang-Liang, Pei Xiao-Qiang. Research on erasure code-based fault-tolerant technology for distributed storage. *Chinese Journal of Computers*, 2017,40(1):236-255

- (in Chinese)
(王意洁, 许方亮, 裴晓强. 分布式存储中的纠删码容错技术研究. 计算机学报, 2017, 40(1): 236-255)
- [17] Yang Song-Lin, Zhang Guang-Yan. Review of data recovery in storage systems based on erasure codes. *Journal of Frontiers of Computer Science and Technology*, 2017, 11(10): 1531-1544(in Chinese)
(杨松霖, 张广艳. 纠删码存储系统中数据修复方法综述. 计算机科学与探索, 2017, 11(10): 1531-1544)
- [18] Shen Z, Cai Y, Cheng K, et al. A survey of the past, present, and future of erasure coding for storage systems. *ACM Transactions on Storage*, 2025, 21(1): 1-39
- [19] Zhang Yao, Chu Jia-Jia, Weng Chu-Liang. Survey on data updating in erasure-coded storage systems. *Journal of Computer Research and Development*, 2020, 57(11): 2419-2431(in Chinese)
(张耀, 储佳佳, 翁楚良. 纠删码存储系统数据更新方法研究综述. 计算机研究与发展, 2020, 57(11): 2419-2431)
- [20] Plank J S. A tutorial on reed-solomon coding for fault-tolerance in RAID-like systems. *Software: Practice and Experience*, 1997, 27(9): 995-1012
- [21] Massey J L. The theory of error-correcting codes. *Proceedings of the IEEE*, 1980, 68(1): 185-186
- [22] Candes E J, Tao T. Decoding by linear programming. *IEEE Transactions on Information Theory*, 2005, 51(12): 4203-4215
- [23] Yao Qiao-Ri. Design and implementation of efficient erasure-coded wide-stripe generation mechanism based on conversion-by-scaling. Wuhan: Huazhong University of Science and Technology, 2022(in Chinese)
(姚乔日. 基于扩展转换的纠删码大条带高效生成机制的设计与实现. 武汉: 华中科技大学, 2022)
- [24] Wang Qin, Li Jun-Ru, Shu Ji-Wu. Survey on in-network storage systems. *Journal of Computer Research and Development*, 2023, 60(11): 2681-2695(in Chinese)
(汪庆, 李俊儒, 舒继武. 在网存储系统研究综述. 计算机研究与发展, 2023, 60(11): 2681-2695)
- [25] Huang C, Simitci H, Xu Y, et al. Erasure coding in windows azure storage//*Proceedings of the Annual Technical Conference*. Boston, USA, 2012: 15-26
- [26] Dimakis A G, Ramchandran K, Wu Y, et al. A survey on network codes for distributed storage. *Proceedings of the IEEE*, 2011, 99(3): 476-489
- [27] Rashmi K, Shah N B, Gu D, et al. A solution to the network challenges of data recovery in erasure-coded distributed storage systems: a study on the Facebook warehouse cluster//*Proceedings of the USENIX Workshop on Hot Topics in Storage and File Systems*. Berkeley, USA, 2013: 1-5
- [28] Ovsianikov M, Rus S, Reeves D, et al. The quantcast file system//*Proceedings of the VLDB Endowment*. New York, USA, 2013: 1092-1101
- [29] Thomasian A. Reconstruct versus read-modify writes in RAID. *Information Processing Letters*, 2005, 93(4): 163-168
- [30] Roth R M, Lempel A. On MDS codes via cauchy matrices. *IEEE Transactions on Information Theory*, 2002, 35(6): 1314-1319
- [31] Blaum M, Brady J, Bruck J, et al. EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures. *IEEE Transactions on Computers*, 1995, 44(2): 192-202
- [32] Myna V, Vinayak R, Nikhil K, et al. Explicit rate-optimal streaming codes with smaller field size. *IEEE Transactions on Information Theory*, 2024, 70(5): 3242-3261
- [33] Plank J S, Simmerman S, Schuman C D. Jersure: a library in C/C++ facilitating erasure coding for storage applications. USA: University of Tennessee, Technology Report: 23, 2008
- [34] Anthapadmanabhan N P, Soljanin E, Vishwanath S. Update-efficient codes for erasure correction//*Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing*. Monticello, USA, 2010: 376-382
- [35] Han Y S, Pai H T, Zheng R, et al. Update-efficient regenerating codes with minimum per-node storage//*Proceedings of the International Symposium on Information Theory*. Istanbul, Turkey, 2013: 1436-1440
- [36] Li J, Wang Q, Lee P P C, et al. An in-depth comparative analysis of cloud block storage workloads: Findings and implications. *ACM Transactions on Storage*, 2023, 19(2): 1-32
- [37] Shen Z, Lin S, Shu J, et al. Cluster-aware scattered repair in erasure-coded storage: design and analysis. *IEEE Transactions on Computers*, 2020, 70(11): 1861-1874
- [38] Hu Y, Li X, Zhang M, et al. Optimal repair layering for erasure-coded data centers: From theory to practice. *ACM Transactions on Storage*, 2017, 13(4): 1-24
- [39] Shen J, Zhang K, Gu J, et al. Efficient scheduling for multi-block updates in erasure coding based storage systems. *IEEE Transactions on Computers*, 2017, 67(4): 573-581
- [40] Liu Y, Wei B, Wu J, et al. Erasure-coded multi-block updates based on hybrid writes and common XORs first//*Proceedings of the 39th International Conference on Computer Design*. Storrs, USA, 2021: 472-479
- [41] Li H, Zhang Y, Zhang Z, et al. PARIX: Speculative partial writes in erasure-coded systems//*Proceedings of the USENIX Annual Technical Conference*. Santa Clara, USA, 2017: 581-587
- [42] Zhang Y, Li H, Liu S, et al. PBS: an efficient erasure-coded block storage system based on speculative partial writes. *ACM Transactions on Storage*, 2020, 16(1): 1-25
- [43] Zhang Zi-Lin, Liu Duo, Tan Yu-Juan, et al. An erasure-coded data update method for distributed storage clusters. *Journal of Computer Research and Development*, 2022, 59(11): 2451-2466(in Chinese)
(章紫琳, 刘铎, 谭玉娟等. 一种适用于分布式存储集群的纠删码数据更新方法. 计算机研究与发展, 2022, 59(11): 2451-2466)
- [44] Zhou H, Hu Y, Feng D, et al. Locality-aware speculative

- cache for fast partial updates in erasure-coded cloud clusters//Proceedings of the 41st International Conference on Computer Design. Washington, USA, 2023:142-149
- [45] Zhou H, Feng D, Hu Y, et al. CoRD: combining raid and delta for fast partial updates in erasure-coded storage clusters//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Atlanta, USA, 2024:1-14
- [46] Shen Z, Lee P P C. Cross-rack-aware updates in erasure-coded data centers: design and evaluation. *IEEE Transactions on Parallel and Distributed Systems*, 2020, 31 (10): 2315-2328
- [47] Gong G, Shen Z, Wu S, et al. Optimal rack-coordinated updates in erasure-coded data centers//Proceedings of the IEEE Conference on Computer Communications. Vancouver, Canada, 2021:1-10
- [48] Gong G, Shen Z, Chen L, et al. Optimal rack-coordinated updates in erasure-coded data centers: design and analysis. *IEEE Transactions on Computers*, 2023, 72(7):1871-1885
- [49] Bao H, Wang Y. ESDU: An elastic stripe-based delta update method for erasure-coded cross-data center storage systems. *Journal of Parallel and Distributed Computing*, 2022, 167: 173-186
- [50] Zhang F, Huang J, Xie C. Two efficient partial-updating schemes for erasure-coded storage clusters//Proceedings of the Seventh International Conference on Networking, Architecture, and Storage. Xiamen, China, 2012:21-30
- [51] Pei X, Wang Y, Ma X, et al. T-update: A tree-structured update scheme with top-down transmission in erasure-coded systems//Proceedings of the 35th Annual IEEE International Conference on Computer Communications. San Francisco, USA, 2016:1-9
- [52] Wang Y, Pei X, Ma X, et al. Ta-update: An adaptive update scheme with tree-structured transmission in erasure-coded storage systems. *IEEE Transactions on Parallel and Distributed Systems*, 2017, 29(8):1893-1906
- [53] Wu H, Du W, Gong P, et al. CPU: Cross-rack-aware pipelining update for erasure-coded storage. *IEEE Transactions on Cloud Computing*, 2020, 10(4):2424-2436
- [54] Zhou H, Feng D, Hu Y. MDTUpdate: A multi-block double tree update technique in heterogeneous erasure-coded clusters. *IEEE Transactions on Computers*, 2023, 72 (10): 2808-2821
- [55] Li Qian, Hu Yu-Peng, Ye Zhen-Yu, et al. An ant colony optimization algorithms based data update scheme for erasure-coded storage systems. *Journal of Computer Research and Development*, 2021, 58(02):305-318(in Chinese)
(李乾, 胡玉鹏, 叶振宇, 等. 基于蚁群优化算法的纠删码存储系统数据更新方案. *计算机研究与发展*, 2021, 58(2): 305-318)
- [56] Jia R, Deng H, Gu Y, et al. GRPU: An efficient graph-based cross-rack parallel update scheme for cloud storage systems//Proceedings of the 40th International Conference on Computer Design. Olympic Valley, USA, 2022:154-161
- [57] Cheng L, Hu Y, Ke Z, et al. LogECMem: Coupling erasure-coded in-memory key-value stores with parity logging//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. St. Louis, USA, 2021:1-15
- [58] Li S, Zhang Q, Yang Z, et al. BCStore: Bandwidth-efficient in-memory KV-store with batch coding//Proceedings of the 33rd International Conference on Massive Storage Systems and Technology. New York, USA, 2017:1-13
- [59] Deng H, Jia R, Wu C. A graph-assisted out-of-place update scheme for erasure coded storage systems//Proceedings of the 50th International Conference on Parallel Processing. Lemont, USA, 2021:1-10
- [60] Li Q, Lyu M, Liu T, et al. MetaEC: An efficient and resilient erasure-coded KV store on disaggregated memory. *ACM Transactions on Architecture and Code Optimization*, 2025, 1-24
- [61] Shen Z, Lee P P C, Shu J, et al. Correlation-aware stripe organization for efficient writes in erasure-coded storage systems//Proceedings of the 36th Symposium on Reliable Distributed Systems. Hong Kong, China, 2017:134-143
- [62] Shen Z, Lee P P C, Shu J, et al. Correlation-aware stripe organization for efficient writes in erasure-coded storage: algorithms and evaluation. *IEEE Transactions on Parallel and Distributed Systems*, 2019, 30(7):1552-1564
- [63] Stodolsky D, Gibson G, Holland M. Parity logging overcoming the small write problem in redundant disk arrays. *ACM SIGARCH Computer Architecture News*, 1993, 21(2):64-75
- [64] Shen Z, Lee P P C. Cross-rack-aware updates in erasure-coded data centers//Proceedings of the 47th International Conference on Parallel Processing. Eugene, USA, 2018:1-10
- [65] Cheng K Y, Li G, Li X, et al. Less is more for I/O-efficient repairs in erasure-coded storage//Proceedings of the USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2026:1-11
- [66] Wang Q, Li J, Lee P P C, et al. Separating data via block invalidation time inference for write amplification reduction in log-structured storage//Proceedings of the 20th USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2022:429-444
- [67] Zhang Y, Huang P, Zhou K, et al. OSCA: An online-model based cache allocation scheme in cloud block storage systems//Proceedings of the USENIX Annual Technical Conference. Boston, USA, 2020:785-798
- [68] Shen Z, Shu J, Huang Z, et al. ClusterSR: Cluster-aware scattered repair in erasure-coded storage//Proceedings of the IEEE International Parallel and Distributed Processing Symposium. New Orleans, USA, 2020:42-51
- [69] Narayanan D, Donnelly A, Rowstron A. Write off-loading: practical power management for enterprise storage. *ACM*

- Transactions on Storage, 2008,4(3):1-23
- [70] Ousterhout J, Agrawal P, Erickson D, et al. The case for RAMClouds: Scalable high-performance storage entirely in DRAM. ACM SIGOPS Operating Systems Review, 2010,43(4):92-105
- [71] Hu N, Steenkiste P. Evaluation and characterization of available bandwidth probing techniques. IEEE Journal on Selected Areas in Communications, 2003,21(6):879-894
- [72] Le Thanh Man C, Hasegawa G, Murata M. A new available bandwidth measurement technique for service overlay networks//Proceedings of the IEEE International Conference on Management of Multimedia Networks and Services. Berlin, Germany, 2003:436-448
- [73] Cheng L, Hu Y, Lee P P C. Coupling decentralized key-value stores with erasure coding//Proceedings of the ACM Symposium on Cloud Computing. Santa Cruz, USA, 2019:377-389
- [74] Chen H, Zhang H, Dong M, et al. Efficient and available in-memory KV-store with hybrid erasure coding and replication. ACM Transactions on Storage, 2017,13(3):1-30
- [75] Jin C, Feng D, Jiang H, et al. RAID6L: A log-assisted RAID6 storage architecture with improved write performance//Proceedings of the 27th Symposium on Mass Storage Systems and Technologies. Denver, USA, 2011:1-6
- [76] Menon J. A performance comparison of RAID-5 and log-structured arrays//Proceedings of the Fourth IEEE International Symposium on High Performance Distributed Computing. Washington, USA, 1995:167-178
- [77] Guo J, Chang Z, Wang S, et al. Who limits the resource efficiency of my datacenter: An analysis of alibaba datacenter traces//Proceedings of the International Symposium on Quality of Service. Phoenix, USA, 2019:1-10
- [78] Yao Q, Hu Y, Tu X, et al. PivotRepair: Fast pipelined repair for erasure-coded hot storage//Proceedings of the 42nd International Conference on Distributed Computing Systems. Bologna, Italy, 2022:614-624
- [79] Wang Z, Zhang G, Wang Y, et al. Dayu: Fast and low-interference data recovery in very-large storage systems//Proceedings of the USENIX Annual Technical Conference. Renton, USA, 2019:993-1008
- [80] Cai Y, Lin S, Shen Z, et al. ChameleonEC: Exploiting tunability of erasure coding for low-Interference repair//Proceedings of the International Symposium on High Performance Computer Architecture. Las Vegas, USA, 2025:15-28
- [81] Hu Y, Cheng L, Yao Q, et al. Exploiting combined locality for wide-stripe erasure coding in distributed storage//Proceedings of the 19th USENIX Conference on File and Storage Technologies. Berkeley, USA, 2021:233-248
- [82] Kadekodi S, Silas S, Clausen D, et al. Practical design considerations for wide locally recoverable codes (LRCs). ACM Transactions on Storage, 2023,19(4):1-26
- [83] Pamies-Juarez L, Blagojevic F, Mateescu R, et al. Opening the chrysalis: on the real repair performance of MSR codes//Proceedings of the 14th USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2016:81-94
- [84] Vajha M, Ramkumar V, Puranik B, et al. Clay codes: Moulding MDS codes to yield an MSR code//Proceedings of the 16th USENIX Conference on File and Storage Technologies. Oakland, USA, 2018:139-154
- [85] Li X, Cheng K, Tang K, et al. ParaRC: Embracing sub-packetization for repair parallelization in MSR-coded storage//Proceedings of the 21st USENIX Conference on File and Storage Technologies. Berkeley, USA, 2023:17-32
- [86] Ma B, Hu Y, Feng D, et al. Repair I/O optimization for clay codes via gray-code based sub-chunk reorganization in Ceph//Proceedings of the International Conference on Massive Storage Systems and Technology. New York, USA, 2024:1-15
- [87] Gan C, Hu Y, Zhao L, et al. Revisiting network coding for warm blob storage//Proceedings of the 23rd USENIX Conference on File and Storage Technologies. Santa Clara, USA, 2025:139-154



ZHOU Hai, Ph. D. His research interests focus on erasure code.

FENG Dan, Ph. D., professor. Her research interests include computer system architecture, large-scale storage systems, non-volatile memory.

HU Yu-Chong, Ph. D., professor. His research interests include cloud storage, erasure code.

Background

This study focuses on the data update challenge in erasure-coded distributed storage systems, a core issue in the field of distributed storage and data reliability. With the explosive growth of big data, traditional replication strategies (e.g., triple replication with $3\times$ storage overhead) for ensuring data reliability have become unsustainable due to ex-

cessive storage costs. Erasure coding (EC) emerges as a superior alternative, as it maintains comparable fault tolerance (e.g., tolerating 2 node failures) with minimal storage overhead (e.g., $1.5\times$ for RS(4,2) code). However, EC introduces a critical bottleneck: updating any single data block requires synchronously updating all parity blocks in the stripe,

leading to high overhead in disk I/O, network traffic, and computation—this severely degrades system performance, especially in high-update-frequency scenarios (e. g., 50% update ratio in Yahoo’s data centers).

Internationally, research on EC has long prioritized data repair (e. g., reducing repair bandwidth via regenerating codes or local reconstruction codes), while systematic studies on EC-based data updates remain limited. Existing update optimizations are fragmented and scenario-bound: disk I/O reduction methods adapt to specific workloads but lack flexibility; network traffic optimizations reduce cross-node/rack traffic but fail to handle dynamic bandwidth fluctuations; parallel transmission techniques enhance parallelism but ignore heterogeneous network topologies; system-level designs fit specific architectures but lack generality. These gaps hinder EC’s application in diverse storage systems.

This review addresses these limitations by systematically synthesizing EC data update methods over the past five years. It categorizes optimizations into four directions (disk I/O reduction, network traffic reduction, parallel transmission optimization, system update optimization), clarifies their key features, advantages, and drawbacks, identifies unresolved challenges (e.

g., multi-stripe scheduling, invalid data recycling, dynamic network adaptation), and proposes future trends (e. g., wide-stripe EC, regenerating code updates, SSD-aware designs). By filling the gap in systematic reviews of EC updates, it provides a comprehensive road map for advancing EC’s industrial utility in high-performance storage systems.

This work is supported by the National Natural Science Foundation of China under Grant No. 62272185 and 623B2037. The project aims to research data recycling techniques for erasure-coded cloud storage to ensure low storage overhead, improve recycling and updating performance, and guarantee user service efficiency. The work of our research team is to enhance the performance of parity updates brought by migration coverage in data recycling. Works related to this have been published in international conferences and journals, such as ACM/IEEE SC’24, ACM/IEEE SC’25, IEEE TPDS2026, IEEE TC2025, ACM TACO2024, etc. Erasure coding, as a technology that achieves high storage efficiency and fault tolerance, has attracted attention from both industry and academia. This review helps us achieve a thorough grasp of the pros and cons of erasure coding, which in turn propels our research forward.