基于阵列码的校验链重组及数据填充技术研究

李萧言[1] 张志东[1] 安 阳[1]
LI Xiaoyan ZHANG Zhidong AN Yang

摘 要 阵列码因具有较低的编解码复杂度而被广泛应用于存储系统，从而保证数据的可靠性。随着数据量的快速增长，存储系统需要采用与其编码方式相适应的技术来扩大规模，但现有的扩容技术缺乏对扩容后系统性能的考虑。在 Cross-Scale 扩容方案的基础上，针对扩容后数据分布不均衡、访问性能降低的问题，提出了混合式校验链重组的优化方法 hybrid cross-scale（HCS），在保证其编码正确性的前提下，将条带中的空白块划分到不同带宽的校验链中，并为优化后的编码阵列设计了数据填充方法，且能够进行故障恢复。通过数学分析和实验验证，证明所提出方法有效提高了扩容后的数据访问性能，维持了原编码的性能优势，且在写密集型存储系统中优化效果更为明显。

关键词 阵列码；存储系统；扩容技术；系统性能；混合式校验链重组；数据填充

doi：10.3969/j.issn.1672-9528.2024.06.022

0 引言

在大数据时代下，数据量快速增长且数据所蕴含的价值也越来越高，数据中心普遍采用 RAID 技术对数据进行条带化管理 [1]，提高数据存储性能，并保障数据安全。纠删码技术属于存储系统中的数据恢复体系，可在数据出现丢失或损坏时进行恢复 [2-3]，其中阵列码采用了简单的异或运算编码而得到了广泛应用 [4-5]。用户数据量的快速增长对数据中心的存储能力和 I/O 带宽提出了更高的要求 [6]，因此设计出维持原编码性能优势且高效可靠的扩容方法成为了重要的研究内容。

现有的扩容方法主要面向扩容过程而设计，以缩短大量数据迁移带来的“脆弱窗”时间为目标 [7]。例如，刘靖宇等人 [8] 提出一种基于 RAID6 阵列码 N-Code 的扩容方案 Cross-Scale，该方案定义了扩容阈值，其根据阵列中的剩余空间选择合适的扩容方案，大幅度提高了扩容效率。在此方案的基础上，李萧言等人提出了在扩容过程中发生故障后的数据恢复方法 [9]，保障了数据安全性与完整性。但同其它扩容方案一样，该方案并未考虑扩容后数据的访问方式，随着时间的推移，会出现负载不均衡、访问性能降低的问题 [10-11]。因此，在提高扩容效率的基础上充分考虑扩容后的性能具有必要性和实际意义。

针对以上问题，本文在 Cross-Scale 的基础上提出了优化方法 hybrid cross-scale（HCS），将条带中的空白块重新划分校验链，并为优化后的存储系统设计了数据填充方法，有效提高了扩容后的数据访问性能。

1. 山西工程技术学院 山西阳泉 045000

[基金项目] 山西省教育厅项目（J20231468）

1 问题分析及优化思路

N-Code[12] 是一种具有 MDS[13] 性质的 RAID6 阵列码，单个条带中的数据布局为 $(p-1)\times(p+1)$，其中 p 为大于 2 的质数。在基于该编码的扩容方案 Cross-Scale 中提出了扩容阈值 CT，当阵列中的空数据行数量超过 CT 时，可在扩容过程中保留全部原始校验链，不仅消除了校验值的更新，从而提高了扩容效率，而且减少了磁盘操作的次数，提高了数据的安全性。

然而，在扩容过程中产生了较多空白块，且空间分布不均衡，影响存储系统的性能。如图 1 所示，从 8 块扩容至 12 块磁盘的一个条带中，由于加入了新的数据行，导致空白块较多，各磁盘中的剩余容量不同。若扩容后采用基本的轮询方式写入新数据，剩余空间较少的磁盘将被更快写满，导致部分存储器过载；另外，写入数据较多时，部分磁盘存在写入冲突，数据平均并行度降低，数据访问更加集中，影响数据读写效率。

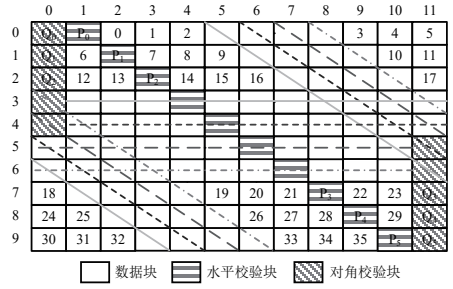

图 1 扩容后的编码阵列布局

若以 m 表示原存储系统的磁盘数量，n 为增加的扩容磁盘数量，将编码阵列中的空白块数记为 B_{bla}，则其数量为：

$$B_{bla} = 2mn + n^2 - 4n \quad (1)$$

扩容后的一个编码阵列中包含了 $2n$ 个空白校验块，其中有 n 个水平校验块和 n 个对角校验块。若使每个空白校验块所在的校验链包含相同数量的空白块，则每条校验链中的空白块数为 B_{ble} 为：

$$B_{ble} = B_{bla}/2n = m - 2 + n/2 \quad (2)$$

增加扩容磁盘之前，存储系统的并行度为 m-2，由公式（2）可知，将空白块平均划分为校验链后，每条校验链相当于在原编码阵列的并行度基础上稳定提高了 n/2 个并行度。所以优化的思路是将空白块重新划分校验链，形成一种具有混合式校验链的编码阵列。

2 校验链重组和数据填充方法

根据上述问题陈述和优化的思路，提出了具体的校验链重组方法，并为优化后的阵列设计了数据填充方法。

2.1 混合式校验链重组方法

本小节介绍在编码阵列中将空白块划分为混合式校验链的具体算法，涉及两个相关概念。

直接关联校验值：将 n 个空水平校验链和 n 个空对角校验链单独划分，每个校验链中保留一个校验值，将这个校验值称为该校验链的直接关联校验值。

冲突块：在图 2 所示的一个编码阵列中，用不同的直线标记出校验链，一部分空白块上的直线出现了相交的情况，说明该部分空白块属于两条校验链，将其称为冲突块。

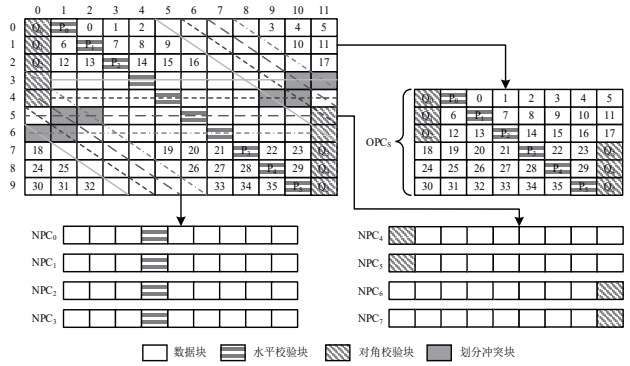

图 2 混合式校验链划分

每个编码阵列中的冲突块数量均为 n^2，若在起始状态将所有的冲突块先归入水平校验链，则需要将一部分冲突块划分到对角校验链中，才能使每个直接关联校验块所在校验链中的数据块数量相等。将 n^2 个冲突块平均分配到 $2n$ 条校验链中时，每条校验链分配到 n/2 个冲突块，所以需要在每条水平校验链中划分出 n/2 个对角校验块到与之产生冲突的水平校验链中。

扩容后形成的编码阵列中，空白水平校验链均由空白数据行拼接而来，所以冲突块均位于拼接数据行中。为了使每条对角校验链划分到相同数量的冲突块，将数据行组对称地分为上下两部分，以阶梯状划分出每一水平校验链中的 n/2 个冲突块到对应的对角校验链中，即图 2 中的纯色数据块，在上下两部分划分出的冲突块均符合 N-Code 的中心对称分布方式。

如图 2 所示，扩容后的一个编码阵列产生了 4 条水平校验链和 4 条对角校验链，这些校验链中的空白块包含了 16 个冲突块，初始状态将这些冲突块都归入水平校验链，再将每条水平校验链中划分出 2 个冲突块到对角校验链中。若以二元组（数据行号、磁盘号）表示块的位置，那么将位于 (3,10)、(3,11)、(4,9)、(4,10)、(5,1)、(5,2)、(6,0)、(6,1) 位置上的冲突块分别划分到与之对应的对角校验链中。当完成一个编码阵列的校验链划分后，扩容窗口会向后移动，每个窗口中都具有相同的划分方式。该过程最终产生了三种不同的校验链的分布方式。

其一，保留原校验链的 OPC。在一个编码阵列中，所有的有效数据块都属于 OPC，共同组成了扩容前的 RAID6 N-Code 编码布局，仍然保持了原有的校验方式和可靠性，校验链并行度仍为 6，且校验块不需更新，例如图 2 中 OPCs 包含的各条校验链。

其二，并行度更大的水平校验链。每条校验链中都包含了一个直接关联的水平校验值，在逻辑视图中，校验值均位于校验链的第四个块中，而在物理层面则是以倾斜分散的方式分布的，形成了类似 RAID5 的布局方式。这些水平校验链相较原校验链将并行度提高了 n/2，例如图 2 中的 NPC0~NPC3。

其三，并行度更大的对角校验链。每条校验链中都包含了一个直接关联的水平校验值，分布在盘号为 0 和 m+n-1 的磁盘中，形成了类似 RAID4 的布局方式，但校验值分布在两个磁盘中，缓解了 RAID4 单独校验盘的性能瓶颈。同样，这些对角校验链相较原校验链将并行度提高了 n/2，例如图 2 中的 NPC4~NPC7。

综上所述，三种不同并行度的校验链共同构成了完整统一的存储空间，为提高新数据填充的效率提供了条件。

2.2 新数据填充及故障处理

数据填充方法通常在数据写入之前建立，并将引导下一个数据访问请求元素的位置。优化后的磁盘阵列中，根据编码阵列是否经过扩容和优化分为两种写入位置，下面分别对其进行讨论。

若写入未经扩容和优化的编码阵列，这些编码阵列中均为完整条带，则采用水平数据填充的方式写入各个磁盘中。

该方式既充分利用了并行技术，缩短数据写入时间，且采用水平数据填充可以将多个数据集中于一条校验链中，有效减少校验值的更新数量。

若写入扩容且优化后的编码阵列，则可提出一种按校验链写入数据的填充方式。该方式不改变各数据块所属的校验链，在保护数据安全的同时，采用了校验链延迟恢复策略，减少了用户层面的响应时间，且恢复编码阵列后维持原编码的容错性。具体过程如下文所示。

每个编码阵列优化后共产生了 $2n$ 条全空的校验链，其中包括 n 条水平校验链和 n 条对角校验链，每条校验链都包含了一个直接关联校验块和 m-2+n/2 个空数据块。如图 3 所示，仍然以 N_b 表示写入的数据，N_q 表示直接关联的对角校验值，N_p 表示直接关联的水平校验值。当存储系统接收到新数据写入请求时，水平校验链优先写入，以直接关联校验块所在数据行自上而下的顺序选择当前要写入的校验链。例如在写入 N_{b0}~N_{b7} 这 8 个数据时，依次写入 N_{p0} 所在的水平校验链，写入位置分别为 (3,1)、(3,2)、(3,3)、(3,5)、(3,6)、(3,7)、(3,8)、(3,9)；接着写入 N_{b8}~N_{b15} 时则写入 N_{p1} 所在的对角校验链，依此类推。当写入 N_{b32}~N_{b39} 时，已经写满所有水平校验链，所以开始依次填充对角校验链。

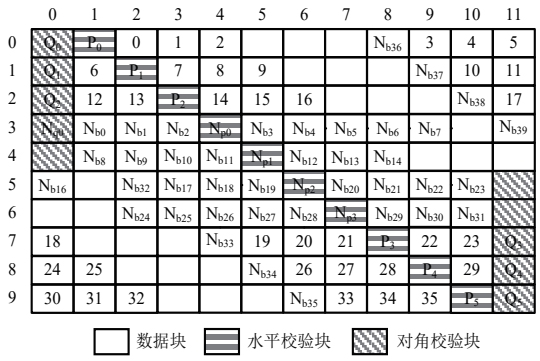

图 3 数据填充过程

更新校验值时仍采用了校验链延迟恢复策略，即暂时只更新写入数据校验链的直接关联校验值，待系统空闲或者缓存写满时，将该编码阵列中的校验值读出，与写入的新数据异或产生新的校验值，将编码阵列恢复为完整的 RAID6 系统。该情况下，在已恢复双校验的编码阵列中，仍可以提供容双盘故障的能力；在进行新数据填充的编码阵列中，若未进行校验恢复，则可以提供单盘容错。如果此时填充数据仍在水平校验链中进行，那么恢复故障盘时按照水平校验链进行数据重构；如果填充数据在对角校验链中进行，且水平校验链部分已经完成校验恢复，那么对角校验值则采用增量更新的方式，根据对角校验链进行数据块的重构；如果填充数据在对角校验链中进行，且水平校验链部分尚未进行校验恢复，那么通过水平校验链和对角校验链相结合的方式重构数据。

3 实验结果及分析

通过对比不同磁盘 I/O 访问记录的平均响应时间可以反应不同数据布局的访问性能。实验将优化形成的数据布局分别与具有相同磁盘数目的标准 N-Code 编码布局以及仅采用 Cross-Scale 扩容的数据布局进行对比，目的在于分析新的数据分布方式是否可以维持原编码的访问性能优势，同时也在于分析其是否能够缓解 Cross-Scale 方案带来的访问性能损失。

表 1 所示为测试访问性能时所运行的三种 I/O 访问记录的读写请求特征。通过对比不同 I/O 访问记录的平均响应时间，可以反映不同数据布局的访问性能。对比算法均采用轮询方法填充空白块，HCS 方案采用按校验链写入数据的填充方法和校验链延迟恢复策略。

表 1 I/O 访问记录特征

I/O 访问记录	读请求占比 / %	写请求占比 / %
Trace1	23.21	76.79
Trace2	82.30	17.70
Trace3	97.98	2.02

图 4 为运行 Trace1 时的平均响应时间，相对标准 N-Code 和 Cross-Scale，HCS 方案的平均响应时间分别降低了 1.78% ～ 3.33% 和 5.76% ～ 33.64%。Trace1 的写请求所占比例较高，HCS 采用了校验链延迟恢复策略，在系统空闲或者缓存写满之前暂时只计算直接关联的校验值，因此降低了用户层面的响应时间。

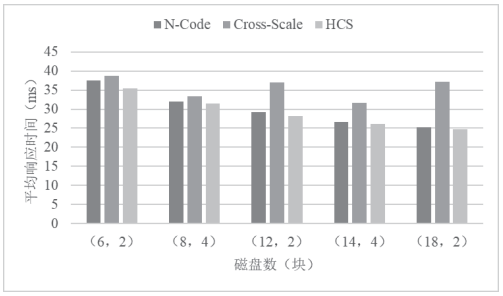

图 4 运行 Trace1 时的平均响应时间

图 5 为运行 Trace2 时的平均响应时间，HCS 方案与标准 N-Code 相比仅在小范围内波动；相对 Cross-Scale 降低了 9.11% ～ 20.79%，有效提高了访问性能。Trace2 中的读请求比例提高，采用 Cross-Scale 扩容时，当读请求命中新数据填充区域时，数据分布相对集中，轮询写入的数据在连续读取时会产生磁盘争用，因此平均响应时间较长。采用 HCS 方案优化后，其校验链与标准 N-Code 相比并行度减少了 n/2，因此响应时间略有提升；而按校验链写入的数据填充方法和校验值延迟更新策略能使其性能仍高于优化前的数据布局。

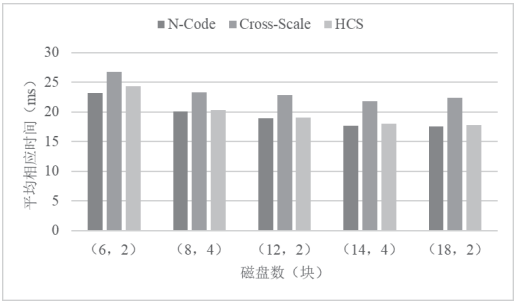

图 5 运行 Trace2 时的平均响应时间

图 6 为运行 Trace3 时的平均响应时间，HCS 方案相对 Cross-Scale 降低了 0.16% ～ 3.4%。由于 Trace3 中的读请求占比高，而优化策略对读请求响应时间影响较小，且在混合式校验链中寻址带来了更多时间消耗，所以优化后的性能接近于标准的 N-Code，维持了原编码的优势。

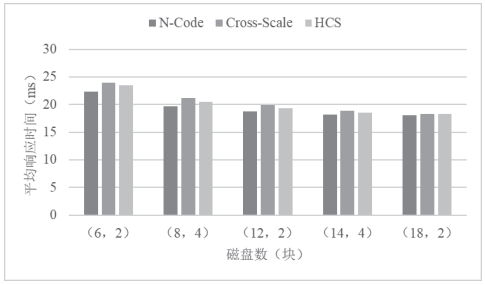

图 6 运行 Trace3 时的平均响应时间

综上所述，HCS 可以维持原编码所具有的读写性能优势，对读写性能的影响仅在小范围中波动。与 Cross-Scale 相比，HCS 明显降低了平均响应时间，特别是在写密集型场景中，平均响应时间最高降低了 33.64%。

4 结语

本文基于阵列码 N-Code 的扩容方案 Cross-Scale 展开研究，针对扩容后数据分布不均衡、访问性能降低等问题，设计了优化方法 HCS，将编码阵列中的空白块重新划分，形成混合式条带，并根据优化后的阵列特点，提出一种按校验链写入数据的填充方法，并在数据写入时采用了校验链延迟恢复策略，提高了数据写入性能并使磁盘阵列具有一定的容错能力。经过测试，HCS 方案有效提高了扩容后的数据访问性能，维持了原编码的优势，且在写请求密集的场景中更具优势。后续将继续围绕阵列码的设计和安全性开展研究，结合固态硬盘阵列的特点，在实现高效扩容的同时兼顾固态硬盘的磨损均衡。

参考文献：

[1] 李牧原，谢平，高原．一种高效的 RAID6 在线扩容机制 [J]. 现代电子技术，2022,45(18):31-36.

[2] 辛尚德，侯宇．大数据应用中的数据安全保障技术研究 [J]. 电子元器件与信息技术，2023,7(3):205-207.

[3] 郑美光，化韬斐，张心宇，等 .xStripeMerge: 基于纠删码存储的高效宽条带生成方法 [J]. 通信学报，2023,44(11):213-224.

[4] 洪铁原，唐聃，熊攀，等．存储系统中的局部修复阵列码模型 [J]. 计算机应用研究，2024,41(1):193-199.

[5] 解峥，王子豪，唐聃，等．低编译复杂度的双容错阵列码 [J]. 计算机应用，2023,43(9):2766-2774.

[6]LIN Z, GUO H, WU C, et al.Rack-scaling:an efficient rack-based redistribution method to accelerate the scaling of cloud disk arrays[C]//2021 IEEE International Parallel and Distributed Processing Symposium,[v.1].Piscataway: IEEE, 2021: 892-901.

[7] 元铸，谢平，耿生玲 .RAID 系统扩容方案研究综述 [J]. 电子学报，2019,47(11):2420-2431.

[8] 刘靖宇，李萧言，李浩鹏，等．一种提高 N-Code 可扩展性的数据重组方案 [J]. 小型微型计算机系统，2023,44(2):442-448.

[9] 李萧言，安阳．基于扩容阈值的磁盘阵列故障恢复分析 [J]. 信息记录材料，2024,25(1):191-193.

[10] 李志鹏．基于纠删码的容错存储系统中数据布局优化 [D]. 合肥：中国科学技术大学，2019.

[11]YUAN Z, YOU X, LV X, et al.SS6:online Short-Code RAID-6 scaling by optimizing new disk location and data migration[J].The computer journal,2021,64(10):1600-1616.

[12]XIE P, YUAN Z, HUANG J, et al.N-Code:an optimal RAID-6 MDS array code for load balancing and high I/O performance[C]//Proceedings of the 48th International Conference on Parallel Processing.New York:ACM,2019:331-340.

[13]FANG W, LV J, CHEN B, et al.New constructions of MDS array codes and optimal locally repairable array codes[J].IEEE transactions on information theory,2024,70(3):1806-1822.

【作者简介】

李萧言（1996—），女，山西阳泉人，硕士，助教，研究方向：网络存储、大数据技术。

张志东（1979—），男，山西阳泉人，硕士，副教授，研究方向：人工智能、大模型技术。

安阳（1990—），男，河南驻马店人，硕士，讲师，研究方向：大数据技术及应用。

（收稿日期：2024-04-10）