基于非均匀循环编码的分组修复码构造

王 静[1*]，雷 珂[1]，李家仪[2]，田松涛[1]，王相隆[1]

(1. 长安大学信息工程学院 西安 710064；2. 山东科技大学数学与系统科学学院 山东 青岛 266590)

【摘要】考虑到实际分布式存储系统中节点故障情况的多样性，该文提出一种基于非均匀循环编码的分组修复码 (GRC-NCC)，使高故障率节点得到更有效的保护。具体地，根据节点故障率对存储节点进行非均匀分组，将数据块依次存入长度递增的节点分组，再使用跨条带循环编码的思路生成组编码块和全局校验块。性能分析以及实验仿真表明，GRC-NCC 虽然具有高于 RS 码的存储开销，但与 RS 码和重叠分组修复码相比，该方法在故障节点修复过程中具有较低的修复带宽开销和修复局部性，且在多节点故障修复过程中性能更优，容错性更好。

关 键 词 分布式存储； 分组修复码； 修复带宽开销； 修复局部性

中图分类号 TN911.2 **文献标志码** A doi:10.12178/1001-0548.2021013

Construction of Group Repairable Codes Based on Non-Uniform Cyclic Coding

WANG Jing[1*], LEI Ke[1], LI Jiayi[2], TIAN Songtao[1], and WANG Xianglong[1]

(1. School of Information Engineering, Chang'an University Xi'an 710064;
2. College of Mathematics and Systems Science, Shandong University of Science and Technology Qingdao Shandong 266590)

Abstract Considering the diversity of node failures in the actual distributed storage systems, group repairable codes based on non-uniform cyclic coding (GRC-NCC) are proposed in this paper, to protect the nodes with high failure rate more effectively. Specifically, storage nodes are grouped non-uniformly according to the node failure rate, data blocks are sequentially stored into node groups with increasing length, and then group coding blocks and global check blocks are generated by using cross-band cyclic coding. Performance analyses and experimental simulations show that GRC-NCC has lower repair bandwidth overhead and repair locality during repairing the failed nodes, and better fault tolerance in the process of multi-node fault repair compared with reed-solomon (RS) codes and rotated group repairable codes (RGRC), although GRC-NCC has higher storage overhead than RS codes.

Key words distributed storage; group repairable codes; repair bandwidth overhead; repair locality

随着信息技术的快速发展，海量数据存储引起了广泛关注。当存储 PB 级以及更高量级数据时，传统的集中式存储系统在存储容量、存储成本和扩展性等方面存在诸多瓶颈，尤其是价格昂贵的设备开销使其无法适应当前存储需求。分布式存储系统以其海量存储能力、高可用性、高可扩展性和低成本等优势成为海量数据的有效存储手段[1]。分布式存储系统将大量文件存储在多个廉价存储设备中，随着系统规模的扩大，存储节点故障时有发生，因此需引入冗余存储来提高节点故障时的存储可靠性。常用的冗余存储技术包括复制策略和纠删码策略[2-5]。

复制策略不需要编解码运算，操作简单易于实现，但存储开销过大。相比复制策略，纠删码策略虽然具有较低的存储开销和较强的容错性能，但在修复故障节点时需要下载文件大小的数据量，修复带宽开销过高。针对上述两种冗余存储技术的不足，文献 [6] 创造性地将网络编码技术应用于分布式存储，提出了再生码的概念，降低了故障节点的修复带宽开销。文

收稿日期：2021－01－11；修回日期：2021－11－02
基金项目：国家自然科学基金 (62001059)；陕西省重点研发计划 (2021GY-019)
作者简介：王静 (1982－)，女，博士，教授，主要从事网络编码及分布式存储编码等方面的研究.
*通信作者：王静，E-mail：342573224@qq.com

献 [7-9] 发现节点存储开销和修复带宽开销之间的最优折中曲线，以及该曲线上的两个极值点，达到这些极值点的再生码分别称为最小存储再生 (minimum storage regenerating, MSR) 码和最小带宽再生 (minimum bandwidth regenerating, MBR) 码。在再生码的修复过程中，新生节点需要从存活节点中连接d个节点完成修复，磁盘 I/O 开销过高。为此，文献 [10-12] 提出了局部修复码 (locally repairable code, LRC)，通过将原始数据块分组生成组编码块，降低了磁盘 I/O 开销；文献 [13-14] 进一步分别研究了局部修复码的构造以及协作局部修复码。文献 [15] 针对局部修复码在修复全局校验块时成本过高以及组内多节点故障时无法修复的问题，提出分组修复码 (group repairable codes, GRC)。文献 [16] 在分组修复码的基础上，基于跨条带重叠编码的思想提出重叠分组修复码 (rotated group repairable codes, RGRC)，获得了更好的数据修复性能。

上述编码方式均为所有数据块和校验块提供了相同等级的故障保护。而在实际的分布式存储系统中，因磁盘磨损状况和使用情况的不同，节点故障概率也会有区别。考虑到节点间不均等的故障概率，文献 [17] 在局部修复码的基础上，提出基于非均匀故障保护的局部修复码 (unequal failure protection based local reconstruction code, UFP-LRC)，将数据块划分为大小不等的分组，分配易出错的数据块到较小的分组，从而使存储系统的整体修复性能和可靠性得到显著提高，但存在组内多个数据块失效时无法局部修复以及修复全局校验块成本过高的问题。

为了对实际分布式存储系统中高故障率节点提供更高等级的保护，本文提出一种基于非均匀循环编码的分组修复码 (group repairable codes based on non-uniform cyclic coding, GRC-NCC)，通过减少高故障率节点的修复成本来改善故障节点的修复性能。具体地，根据故障率对存储节点进行分组，高故障率节点存储到小分组以提供更高等级保护。在此基础上，使用跨条带循环编码的思想生成组编码块和全局校验块，故障节点修复过程中相邻条带的编码可以提供部分解码数据，获得较低的数据传输量和修复成本。

1 分组修复码和重叠分组修复码

1.1 分组修复码

基于(n,k)最大距离可分 (maximum distance separable, MDS) 码，将文件分为大小相等的k个原始数据块，经过编码生成$n=k+m$个编码块。GRC 的构造过程为：将 MDS 码的k个数据块分为L组，记为$S_l(l=1,2,\cdots,L)$，将$m=m_0+m_1$个编码块中的前m_0个编码块作为 GRC 的全局编码块，在数据块分组$S_l(l=1,2,\cdots,L)$中生成m_1个组编码块；m_1个组编码块生成方式与 MDS 码编码块生成方式相同，只需保持本组内编码系数不变，其余组编码系数置零即可；最后将 GRC 码的m_0个全局编码块视为一组，再生成一个组编码块。

具体地，(15, 8)GRC 编码过程如图 1 所示。将 8 个原始数据块$D_1,D_2,\cdots,D_8$平均分成两组，为每个组分别生成$m_1=2$个组编码块P_1、P_2和P_3、P_4。将 (12, 8)MDS 码的$m_0=2$个编码块Q_1、Q_2作为 GRC 的全局编码块，再将其视为一组生成一个组编码块Q_3。

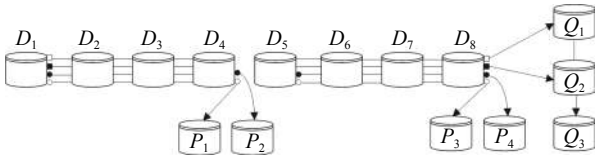

图 1 (15, 8)GRC 编码示意图

1.2 重叠分组修复码

(k,p,q,m)RGRC 基于跨条带编码的思想，k表示原始数据块的个数，p表示组编码块的个数，q表示全局编码块的个数，m表示全局组编码块的个数。(k,p,q,m)RGRC 的构造过程如下：将大小为 M 的文件按照指定的数据块大小进行划分，然后根据块数划分为M/k个条带；每次读入两个条带，将条带内数据划分为k/p个组，为每个组生成一个组编码块；再将条带数据划分为 q 组，记为$G_i(i=1,2,\cdots,q)$，其中第一个全局编码块由本条带内的数据块编码得到，第二个全局编码块由第二个条带提供G_1组和第一个条带提供剩余组数据求得，依次类推。

具体地，(4, 2, 2, 1)RGRC 的编码过程如图 2 所示。首先读取两个条带，将条带内数据划分为D_1、D_2和D_3、D_4，为每个组生成一个组编码块P_1和P_2；再将条带数据划分为两组，记为G_1和G_2，第一个全局编码块Q_1由本条带内的数据求得，第二个全局编码块Q_2由第二个条带提供G_1组和第一个条带提供G_2组数据求得；最后将Q_1、Q_2视为一组生成一个组编码块Q_3。

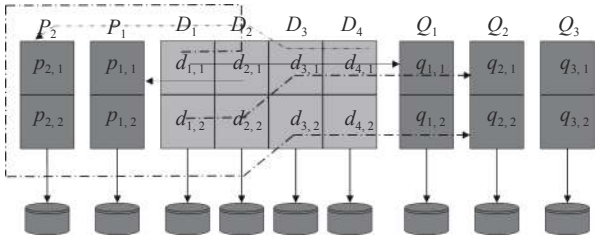

图 2　(4, 2, 2, 1)RGRC 编码示意图

2　基于非均匀循环编码的分组修复码

2.1　节点故障率

节点故障率τ是节点发生故障的频数，以每单位时间的节点故障数表示[17]。节点故障率与平均无故障时间 (mean time to failure, MTTF) 之间的关系为：

$$\tau = 1 - \mathrm{e}^{\frac{-1}{\mathrm{MTTF}}} \tag{1}$$

式中，MTTF 表示系统无故障运行的平均时间，为取所有从系统开始正常运行到发生故障之间的时间段的平均值。实际分布式存储系统中节点故障率并不相同，呈现非均匀分布的特征。

2.2　基于非均匀循环编码的分组修复码构造

基于非均匀循环编码的分组修复码构造算法具体步骤如下。

1) 根据节点故障率对存储节点进行非均匀分组，记为$G_j(j=1,2,\cdots,l)$。

2) 将 MDS 码$k=\sum_{i=0}^{l-1}(k_0+2i)$个数据块依次放入不同分组，$G_1$由故障率最高的节点组成，存入$k_0$个数据块；$G_2$由故障率次高的节点组成，存入$k_0+2$个数据块；依次类推，$G_l$由故障率最低的节点组成，存入$k_0+2(i-1)$个数据块。

3) 将多个条带组合成条带集，MDS 码编码块个数为$m=m_0+m_1$。

4) 为λ个高故障率分组生成m_1个组编码块，其中第一个组编码块由本条带内数据块计算得到，剩余组编码块由本条带与相邻条带各提供一部分数据块计算得到；同时，m_1个组编码块生成方式与 MDS 码编码块生成方式相同，只需保持本组内编码系数不变，其余组编码系数置为零即可。

5) 对于$l-\lambda$个低故障率分组，分别异或组内数据生成一个组编码块。

6) 对 MDS 码的前m_0个编码块，保持第一个全局编码块不变，依旧由本条带内的数据求得，剩余的全局编码块则由上一条带提供G_1组和本条带提供剩余组数据求得，依次类推，最后再将全局编码块视为一组生成一个组编码块。

$(k,\lambda m_1,l-\lambda,m_0+1)$GRC-NCC 由$l$个大小不等分组内的$k=\sum_{i=0}^{l-1}(k_0+2i)$个数据块、$\lambda m_1+(l-\lambda)+1$个局部校验块、$m_0$个全局校验块组成，编码后共生成$n=k+\lambda m_1+(l-\lambda)+m_0+1$个编码块。具体地，(6, 2, 1, 3)GRC-NCC 编码过程如图 3 所示。

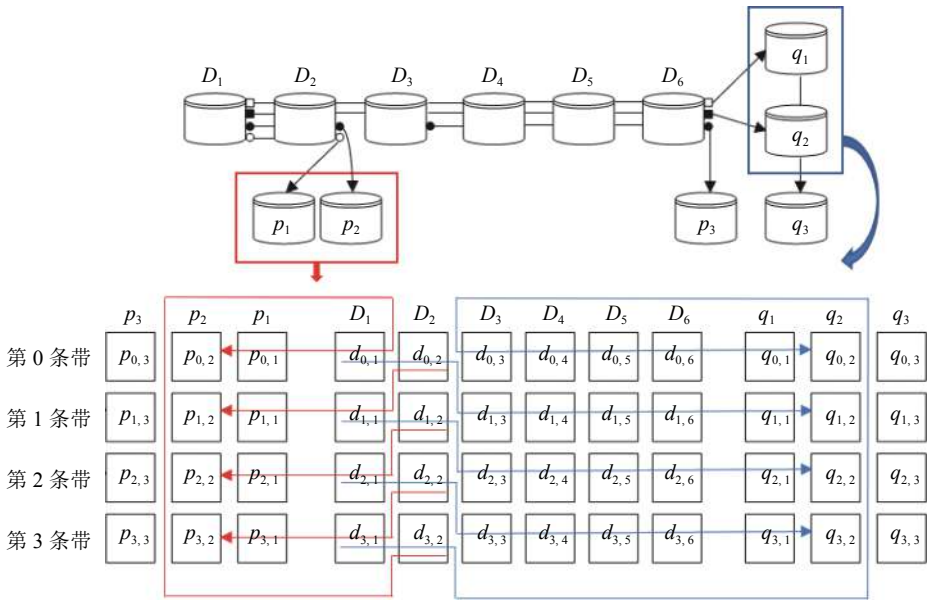

图 3　(6, 2, 1, 3)GRC-NCC 编码示意图

根据节点故障率将存储节点划分为大小不等的$l=2$个分组，记为G_1和G_2。将 MDS 码中$k_0=2$个数据块存入第 1 个分组G_1，剩余$k_0+2=4$个数据块存入第 2 个分组G_2。再将 4 个条带组合成条带集，为$\lambda=1$个高故障率分组生成$m_1=2$个组编码块p_1和p_2，其中p_1由组内数据计算得到，p_2由组内前半段数据与上一条带中后半段数据计算得到。对由D_3、D_4、D_5、D_6组成的$l-\lambda=1$个低故障率分组，异或组内数据生成一个组编码块p_3。对 MDS 码的前$m_0=2$个编码块，保持第一个全局编码块q_1不变，由本条带内的数据求得。剩余的全局编码块q_2由上一条带提供G_1组和本条带提供G_2组数据求得。再将全局编码块q_1和q_2视为一个分组，生成一个组编码块q_3。

2.3 故障节点修复

2.3.1 单节点故障

在 GRC-NCC 编码方案中，单节点故障均可进行组内修复。下面分别以图 3 中高故障率分组节点D_1、低故障率分组节点D_3以及全局校验节点q_1故障为例，对采用 GRC-NCC 编码的单节点故障修复进行说明。

高故障率分组节点D_1发生故障时，可通过节点D_2与p_1完成修复，总共传输 8 个数据块。进一步地，节点D_1也可通过节点D_2与两个组编码节点p_1、p_2混合修复，如图 4 所示，只需传输 6 个数据块，且混合修复算法的优势随着条带数的增多更为明显。当低故障率分组节点D_3故障时，可通过节点D_4、D_5、D_6和组编码节点p_3完成修复。当全局校验节点q_1故障时，可通过剩余的全局校验节点q_2和组编码节点q_3进行局部修复，从而减少修复成本。

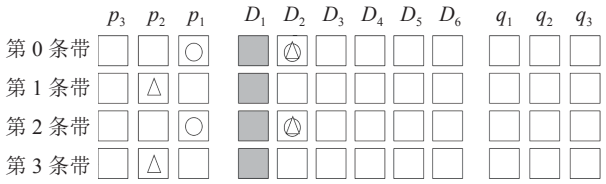

图 4 (6, 2, 1, 3)GRC-NCC 单节点故障修复

2.3.2 多节点故障

在 GRC-NCC 编码方案中，多节点故障修复可分为组内修复与全局修复两种，修复原则是先组内修复，组内不可修复时再全局修复。下面分情况进行讨论：

1) 多个故障节点分别位于不同修复组内时，可分别在不同修复组内进行单节点修复。

2) 多个故障节点均位于高故障率分组内时，若故障节点数目不大于组编码节点数目m_1，则可在组内完成修复；否则，将使用全局修复。全局修复中，若故障节点数目不大于编码节点 (包括组编码节点与全局编码节点) 数目，使用全局完成修复，否则无法成功修复。

3) 多个故障节点均位于低故障率分组内时，此时故障节点数目大于组编码节点数目，则在该故障情况下无法进行组内修复，只能采用全局修复。若故障节点数目不大于编码节点数目，使用全局修复，否则无法成功修复。

4) 其他多节点故障修复情况。对可使用组内完成修复的修复组先使用组内修复，再使用全局修复。全局修复完成后，修复条件可能会发生变化，此时若还可使用组内修复则继续进行组内修复，组内修复不可进行时再使用全局修复。重复以上过程，直至所有故障节点均修复成功。

以图 3 中高故障率分组节点D_1、D_2、p_1、p_2全部故障以及全局编码节点q_1、q_2发生故障为例，说明采用 GRC-NCC 编码方案无法成功修复故障节点的情况。首先在高故障率分组中，节点D_1、D_2以及组编码节点p_1、p_2全部故障，无法进行组内修复；其次全局校验分组中的全局编码节点q_1、q_2故障，故障节点数目大于未故障组编码节点数目，也不能进行组内修复；最后考虑全局修复，由于全局编码节点q_1、q_2故障，则节点D_1、D_2不能被修复，故在该故障情况下无法成功修复。

图 3 中高故障率分组发生两节点故障时，仍可进行组内修复，减少修复成本；当高故障率分组节点全部故障时，可使用全局修复完成所有故障节点的修复，为高故障率节点提供了更高等级的保护。低故障率分组发生两节点故障，修复过程中相邻条带的编码可为对方提供一部分解码数据，减少数据的传输量。如图 5 所示低故障率分组节点D_3和p_3故障时，首先通过剩余数据节点与全局校验节点q_2修复第一条带与第三条带中对应节点D_3的数据块，所用数据块以Δ表示；再通过剩余数据节点与全局校验节点q_1修复第 0 条带与第 2 条带中对应节点D_3的数据块，所用数据块以○表示；最后通过已修复的节点D_3和节点D_4、D_5、D_6实现组编码节点p_3的修复。可见，GRC-NCC 编码方案为高故障率节点提供更有效保护的同时，也考虑了使用跨条带循环编码的思想以减少低故障率节点的修复成本。

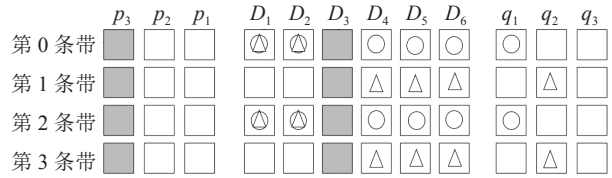

图 5　(6, 2, 1, 3)GRC-NCC 中低故障率分组两节点故障修复

3　性能分析

本节主要分析 GRC-NCC 的存储开销、修复局部性、修复带宽开销和容错能力，并与 RS 码和 RGRC 进行比较。由于修复带宽开销和修复局部性依赖于节点故障，故分别讨论了单节点故障和多节点故障的情况。

3.1　存储开销

本文采用文献 [18] 中定义的存储开销，存储编码数据块需要的存储空间与存储原始数据块存储空间的比值。依据上述存储开销的定义，(n,k)RS 码和$(k,\lambda m_1,l-\lambda,m_0+1)$GRC-NCC 的存储开销分别为：

$$S_{\mathrm{RS}}=\frac{n}{k} \tag{2}$$

$$S_{\mathrm{GRC-NCC}}=\frac{k+\lambda m_1+l-\lambda+m_0+1}{k} \tag{3}$$

根据$(k,\lambda m_1,l-\lambda,m_0+1)$GRC-NCC 构造过程得$m_0+m_1=n-k$，则可直接比较 RS 码和 GRC-NCC 的存储开销。对于$(k,\lambda m_1,l-\lambda,m_0+1)$GRC-NCC，其存储开销为：

$$S_{\mathrm{GRC-NCC}}=\frac{(\lambda-1)(m_1-1)+l}{k}+S_{\mathrm{RS}}=\eta+S_{\mathrm{RS}} \tag{4}$$

因$\lambda\geqslant 1$，$m_1\geqslant 1$且$l\geqslant 2$，可推断出：

$$\eta=\frac{(\lambda-1)(m_1-1)+l}{k}>0$$

因此，相比于 RS 码，GRC-NCC 没有达到最优的存储开销。

3.2　修复单故障节点

修复单故障节点时，RS 码的修复局部性和修复带宽开销分别为k和n。

除 RS 码外，RGRC 和 GRC-NCC 的构造使得故障节点可能位于局部组或全局校验组，因此综合考虑两种情况，讨论 RGRC 和 GRC-NCC 在单节点故障情况下的修复局部性和修复带宽开销。首先分析 RGRC 和 GRC-NCC 在单节点故障情况下的修复局部性。修复局部性是指故障节点修复过程中连接的存活节点数目。

在$(k,\lambda m_1,l-\lambda,m_0+1)$GRC-NCC 中，故障节点可以位于局部组，包括高故障率分组和低故障率分组，也可以位于全局校验组。当故障节点位于第$i(i=0,1,\cdots,l-1)$个局部组时，修复单故障节点需要连接k_0+2i个存活节点，其中$k=\sum_{i=0}^{l-1}(k_0+2i)$；当故障节点位于全局校验组时，该故障节点的修复局部性为m_0，则 GRC-NCC 的修复局部性为：

$$\frac{(k_0+2i)(k+\lambda m_1+l-\lambda)+m_0(m_0+1)}{k+\lambda m_1+l-\lambda+m_0+1}$$

对于(k,p,q,m)RGRC，当单故障节点位于局部组时，修复局部性为k/p；当单故障节点位于全局校验组时，修复局部性为q/m，所以 RGRC 的修复局部性为：

$$\frac{k(k+p)/p+q(q+m)/m}{k+p+q+m}$$

为方便比较，设定 RGRC 和 GRC-NCC 中分组数目及全局校验节点个数均为 2。图 6 为单节点故障时，RS 码、RGRC 和 GRC-NCC 的修复局部性曲线。从图 6 可以看出，随着存储节点数目k的增加，RS 码、RGRC 和 GRC-NCC 的修复局部性也随之线性增加，且提出的 GRC-NCC 的修复局部性最小，RS 码的修复局部性最大。

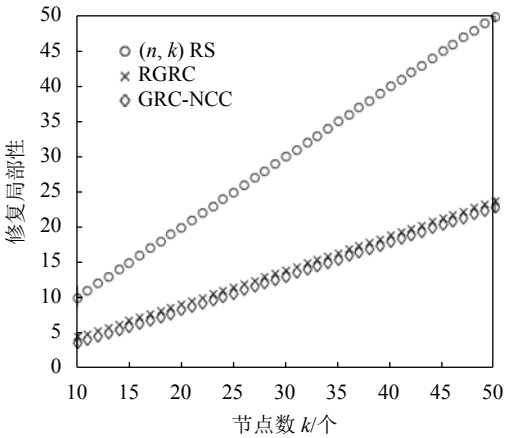

图 6　单节点故障情况下的修复局部性

进一步分析修复单节点故障的带宽开销。修复带宽开销是修复故障节点时需要下载的数据量的大小。当 GRC-NCC 的单故障节点位于局部组时，修复带宽开销为$w_1=(k_0+2i)n/k$；当单故障节点位于 GRC-NCC 的全局校验组时，修复带宽开销为$w_2=m_0n/k$；考虑到$k=\sum_{i=0}^{l-1}(k_0+2i)$，$n=k+\lambda m_1+l-\lambda+m_0+1$，则 GRC-NCC 的修复带宽开销为：

$$B_{\mathrm{GRC\text{-}NCC}}=\frac{w_1(k+\lambda m_1+l-\lambda)+w_2(m_0+1)}{k+\lambda m_1+l-\lambda+m_0+1} \tag{5}$$

下面讨论 RGRC 的修复带宽开销。当 RGRC 的单故障节点位于局部组时，修复带宽开销为$w'_1=(k+p+q+m)/p$；当单故障节点位于全局校验组时，修复带宽开销为$w'_2=q(k+p+q+m)/km$。因此，RGRC 的修复带宽开销为：

$$B_{\mathrm{RGRC}}=\frac{w'_1(k+p)+w'_2(q+m)}{k+p+q+m} \tag{6}$$

取$n=k+4$，假定 RGRC 和 GRC-NCC 中分组数目及全局校验节点个数均为 2，RS 码、RGRC 和 GRC-NCC 的修复带宽开销的性能如图 7 所示。由图可知，GRC-NCC 的修复带宽开销最小，RGRC 次之，RS 码的修复带宽开销最大。

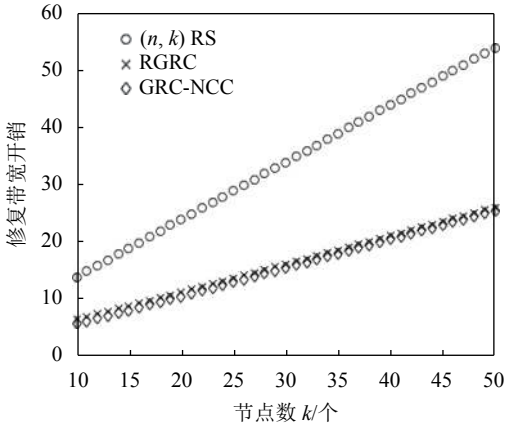

图 7 单节点故障情况下的修复带宽开销

3.3 修复多故障节点

修复多故障节点时，RS 码的修复局部性和修复带宽开销仍分别为k和n。对于 RGRC 和 GRC-NCC，当多个故障节点分别位于不同分组时，均可通过组内完成修复，因此主要讨论同组发生多节点故障的情况。

RGRC 无论是修复局部组还是全局校验组的多故障节点，都必须采用全局修复，因此 RGRC 的修复局部性为k，修复带宽开销为$k+p+q+m$。

对于 GRC-NCC，当高故障率分组发生多节点故障，且假设故障节点个数不大于m_1时，仍可进行组内修复，修复局部性为$k_0+2i(i=0,1,\cdots,\lambda-1)$；当低故障率分组或全局校验组发生多节点故障时，此时需采用全局修复，修复局部性为k。所以 GRC-NCC 的修复局部性为：

$$\frac{(k_0+2i)\left(\sum_{i=0}^{\lambda-1}(k_0+2i)+\lambda m_1\right)+k\left(\sum_{i=\lambda}^{l-1}(k_0+2i)+l-\lambda+m_0+1\right)}{k+\lambda m_1+l-\lambda+m_0+1}$$

取 RGRC 和 GRC-NCC 中分组数目及全局校验节点个数为 2 时，得到图 8 所示的多节点故障时的修复局部性和存储节点数k的关系。从图中得知，GRC-NCC 修复多节点故障时的修复局部性最小。

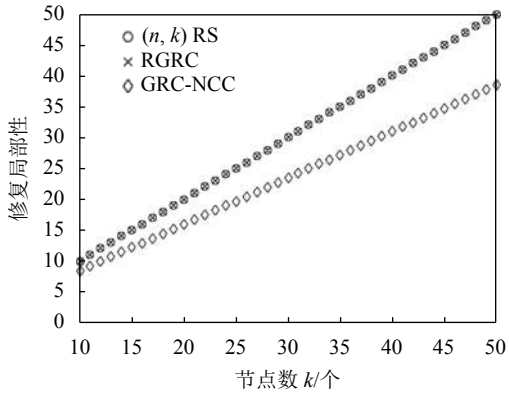

图 8 多节点故障情况下的修复局部性

当高故障率分组发生多节点故障时，GRC-NCC 的修复带宽开销为$w_1=(k_0+2i)n/k$；当低故障率分组或全局校验组发生多节点故障时，其修复带宽开销为$w_2=n$。所以 GRC-NCC 的修复带宽开销为：

$$B_{\mathrm{GRC-NCC}}=\frac{w_1\left(\sum_{i=0}^{\lambda-1}(k_0+2i)+\lambda m_1\right)+w_2\left(\sum_{i=\lambda}^{l-1}(k_0+2i)+l-\lambda+m_0+1\right)}{k+\lambda m_1+l-\lambda+m_0+1} \tag{7}$$

设定$n=k+4$，RGRC 和 GRC-NCC 中分组数目及全局校验节点个数为 2，多节点故障时几种码的修复带宽开销和存储节点数k的关系如图 9 所示，其中 GRC-NCC 修复多节点故障时的修复带宽开销最小。

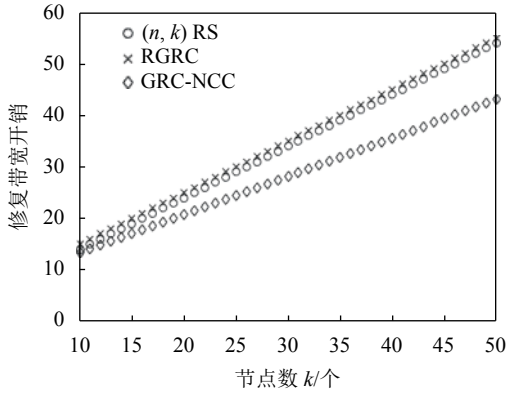

图 9 多节点故障情况下的修复带宽开销

实际分布式存储系统中，多节点故障导致数据不可用占据了较少数情况。而对于 GRC-NCC 而

言，多个故障节点同时位于低故障率分组的可能性更小，因此若只考虑高故障率分组发生多节点故障的情况，则 GRC-NCC 的修复局部性为$k_0+2i(i=0,1,\cdots,\lambda-1)$，修复带宽开销为$(k_0+2i)n/k$。图 10 和图 11 分别给出了此种情况下各种码的修复局部性和修复带宽开销随存储节点数k的变化曲线。从图中可以得知，GRC-NCC 具有更优的修复局部性和修复带宽开销。

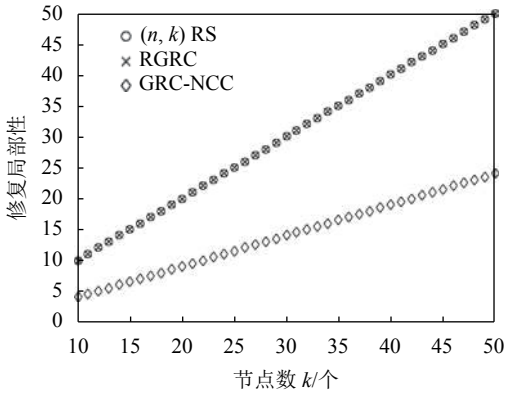

图 10　高故障率分组多节点故障情况下的修复局部性

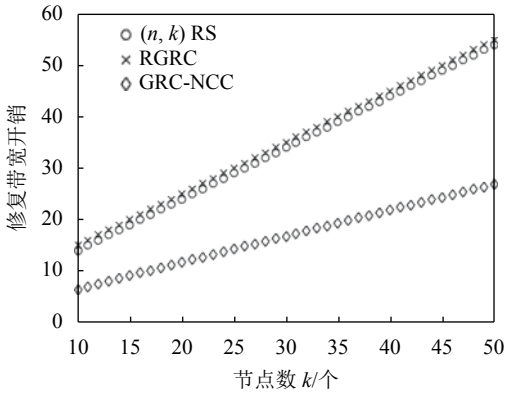

图 11　高故障率分组多节点故障情况下的修复带宽开销

3.4　容错能力

容错能力是指分布式存储系统可以容忍节点故障且保证数据不丢失的能力，是分布式存储系统的一个重要因素[19]。具体地，一个 (10, 6)RS 码可以容忍任意 4 个节点故障。对于 (6, 2, 2, 1)RGRC 和 (6, 2, 1, 3)GRC-NCC 来说，两种编码均能恢复任意 3 错，因此下面主要讨论故障节点数大于 3 时，两种编码方式的容错能力。

当故障节点数为 4 时，统计 (6, 2, 2, 1)RGRC 不可修复情况的数目为$2(C_4^4+C_4^2C_2^2)$，计算其容 4 错能力为：

$$P_{\text{RGRC},4}=1-\frac{C_2^1(C_4^4+C_4^2C_2^2)}{C_{11}^4}=0.956 \tag{8}$$

(6, 2, 1, 3)GRC-NCC 容 4 错能力为：

$$P_{\text{GRC-NCC},4}=1-\frac{C_5^4+C_5^2C_2^2}{C_{12}^4}=0.970 \tag{9}$$

当故障节点数为 5 时，(6, 2, 2, 1)RGRC 和 (6, 2, 1, 3)GRC-NCC 容 5 错能力分别为：

$$P_{\text{RGRC},5}=1-\frac{C_2^1C_7^1+C_2^1C_4^3C_3^2+C_2^1C_4^2}{C_{11}^5}=0.892 \tag{10}$$

$$P_{\text{GRC-NCC},5}=1-\frac{C_5^5+C_5^4C_7^1+C_5^3C_3^2+C_5^2C_3^3}{C_{12}^5}=0.904 \tag{11}$$

(6, 2, 2, 1)RGRC 最多能容忍 5 个节点故障，其容 6 错能力为 0；而 (6, 2, 1, 3)GRC-NCC 容 6 错能力为：

$$P_{\text{GRC-NCC},6}=1-\frac{C_5^5C_7^1+C_5^4C_7^2+C_2^1C_5^3C_3^3+C_5^2C_3^3C_4^1+C_5^2C_4^4}{C_{12}^6}=0.852 \tag{12}$$

图 12 给出了 (10, 6)RS 码、(6, 2, 2, 1)RGRC 和 (6, 2, 1, 3)GRC-NCC 在不同故障节点数目情况下的容错能力，其他k和故障节点数目情况下同理。通过上述分析对比可得，相比于其他两种编码方式，GRC-NCC 的容错能力更好，同时可以容忍更多的节点故障。

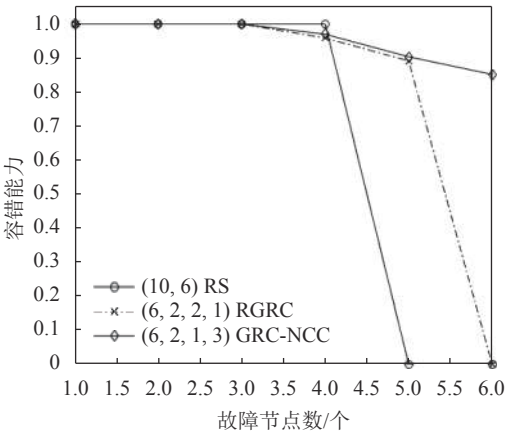

图 12　容错能力对比

4　结 束 语

实际分布式存储系统中节点故障率不同，而现有的纠删码大多都是为节点提供相同等级的保护，这使得高故障率节点没有得到更高等级的保护。针对该问题，本文提出一种基于非均匀循环编码的分组修复码 GRC-NCC，根据节点故障率对存储节点

进行非均匀分组，同时使用跨条带循环编码的思想，减少故障修复过程的数据传输量。与 RS 码和 RGRC 相比，GRC-NCC 具有更好的修复局部性和修复带宽开销性能，且在多节点故障修复过程中性能更优，同时容错性能也更好。

参 考 文 献

[1] CALDER B, WANG J, OGUS A, et al. Windows azure storage: A highly available cloud storage service with storage consistency[C]//Proceedings of ACM SOSP '11. Cascais, Portugal: SIGOPS, 2011: 143-157.

[2] BSOUL M, ABDALLAH A E, ALMAKADMEH K, et al. A round-based data replication strategy[J]. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(1): 31-39.

[3] WANG Y J, XU F L, PEI X Q. Research on erasure code-based fault-tolerant for distributed storage[J]. Chinese Journal of Computers, 2017, 40(1): 236-255.

[4] VAISHAMPAYAN V A. Lattice erasure codes of low rank with noise margins[C]//IEEE International Symposium on Information Theory (ISIT). Vail, CO: IEEE, 2018: 961-965.

[5] LEE O T, KUMAR S D M, CHANDRAN P. Erasure coded storage systems for cloud storage-challenges and opportunities[C]//The 3rd International Conference on Data Science and Engineering (ICDSE). Cochin: IEEE, 2016: 52-58.

[6] DIMAKIS A G, GODFREY P B, WU Y N, et al. Network coding for distributed storage systems[J]. IEEE Transactions on Information Theory, 2010, 56(9): 4539-4551.

[7] HUANG K, PARAMPALLI U, XIAN M. On secrecy capacity of minimum storage regenerating codes[J]. IEEE Transactions on Information Theory, 2017, 63(3): 1510-1524.

[8] RASHMI K V, SHAN N B, KUMAR P V. Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction[J]. IEEE Transactions on Information Theory, 2011, 57(8): 5227-5239.

[9] MAHDAVIANI K, KHISTI A, MOHAJER S. Bandwidth adaptive & error resilient MBR exact repair regenerating codes[J]. IEEE Transactions on Information Theory, 2019, 65(5): 2736-2759.

[10] PAPAILIOPOULOS D S, DIMAKIS A G. Locally repairable codes[C]//2012 IEEE International Symposium on Information Theory Proceedings. Cambridge, MA: IEEE, 2012: 2771-2775.

[11] RAWAT A S, PAPAILIOPOULOS D S, DIMAKIS A G, et al. Locality and availability in distributed storage[J]. IEEE Transactions on Information Theory, 2016, 62(8): 4481-4493.

[12] GOPALAN P, HUANG C, SIMITCI H, et al. On the locality of codeword symbols[J]. IEEE Transactions on Information Theory, 2012, 58(11): 6925-6934.

[13] HAO J, XIA S T, KENNETH W, et al. Bounds and constructions of locally repairable codes: Parity-check matrix approach[J]. IEEE Transactions on Information Theory, 2020, 66(12): 7465-7474.

[14] WANG J, YAN Z Y, LI K C, et al. Local codes with cooperative repair in distributed storage of cyber-physical-social systems[J]. IEEE Access, 2020(8): 38622-38632.

[15] 林轩. GRC: 一种适用于多节点失效的高容错低修复成本纠删码[J]. 计算机研究与发展, 2014, 51(S): 172-181.
LIN X. GRC: A high fault-tolerance and low recovery-overhead erasure code for multiple losses[J]. Journal of Computer Research and Development, 2014, 51(S): 172-181.

[16] 李承欣. 基于 LRC 的数据重构优化研究[D]. 武汉: 华中科技大学, 2019.
LI C X. Research on data reconstruction optimization based on LRC[D]. Wuhan: Huazhong University of Science & Technology, 2019.

[17] HU Y P, LIU Y H, LI W J, et al. Unequal failure protection coding technology for cloud storage systems[C]//2016 IEEE International Conference on Cluster Computing. Taipei, Taiwan, China: IEEE, 2016: 231-240.

[18] KAMATH G M, PRAKASH N, LALITHA V, et al. Codes with local regeneration and erasure correction[J]. IEEE Transactions on Information Theory, 2014, 60(8): 4637-4660.

[19] FU Y, SHU J, LUO X. A stack-based single disk failure recovery scheme for erasure coded storage systems[C]//IEEE International Symposium on Reliable Distributed Systems. Nara: IEEE, 2014: 136-145.

编 辑 税 红