文章编号:1007-5321(2021)06-0116-06 **DOI**:10.13190/j.jbupt.2021-026

哈夫曼树的异构部分重复码构造

余春雷[1,2], 王 静[3], 杨成福[1], 彭小利[1,2]
(1. 四川文理学院 智能制造学院, 达州 635002; 2. 智能制造产业技术研究院, 达州 635002;
3. 长安大学 信息工程学院, 西安 710064)

摘要: 针对分布式存储系统中数据被访问频率的不同,提出一种基于哈夫曼树的可变重复度的异构部分重复(HVFR)码,将不同访问频率的数据块作为哈夫曼树带有确定权值的叶子节点,构造哈夫曼树并确定数据块的重复度,利用成对平衡设计构造异构部分的重复码,能够提高热数据的并行访问速度和系统存储效率. 性能分析和实验结果表明,与里所码以及简单再生码相比,HVFR 码可以显著减少故障节点的修复时间及修复局部性,提高热数据的并行访问速度,达到负载均衡,且计算复杂度低.

关 键 词: 分布式存储; 冷热数据; 部分重复码; 哈夫曼树
中图分类号: TP391.9 **文献标志码**: A

Construction of Heterogeneous Fractional Repetition Codes of Huffman Tree

YU Chun-lei[1,2], WANG Jing[3], YANG Cheng-fu[1], PENG Xiao-li[1,2]
(1. Intelligent Manufacturing Institute, Sichuan University of Arts and Science, Dazhou 635002, China;
2. Intelligent Manufacturing Industry Technology Research Institute, Dazhou 635002, China;
3. School of Information Engineering, Chang'an University, Xi'an 710064, China)

Abstract: Considering access frequency differences of data in distributed storage systems, a heterogeneous variable fractional repetition (HVFR) code based on Huffman tree is proposed. First, taking the data blocks with different access frequencies as the weighted leaf nodes of the Huffman tree, the Huffman tree is constructed and the duplication of the data blocks is determined. Then, the pairwise balanced design is used to construct heterogeneous fractional repetition codes. Performance analysis and experimental results show that, compared with reed-solomon codes and simple regenerating codes, HVFR codes can significantly reduce the repair time and repair locality of the failed nodes, improves the parallel access speed of the hot data, and achieves load balance with low computation complexity.

Key words: distributed storage; hot and cold data; fractional repetition code; Huffman tree

随着互联网技术的问世,面对信息技术每天产生的大量数据信息,如何高效、可靠地存储数据是当前研究的热点. 传统的数据存储一般采用集中式存储,但存在购买设备昂贵而且不易扩展等缺陷. 分布式存储系统具有扩展性好等优势被广泛研究. 复制策略和纠删码策略[1-2]通常应用在分布式存储系

收稿日期: 2021-03-04
基金项目: 国家自然科学基金项目(62001059); 陕西省重点研发计划项目(2021GY-019); 智能制造产业技术研究院开放基金项目(ZNZZ2106)
作者简介: 余春雷(1992—), 男, 硕士生.
通信作者: 王 静(1982—), 女, 教授, E-mail:jingwang@chd.edu.cn.

统中保证数据的可靠性和有效性[3].

三副本复制策略最为常见[4]，但复制策略的存储开销比较大. 纠删码策略有效地降低了系统的存储开销，但是纠删码策略的修复带宽开销和修复局部性过大. 目前典型的纠删码主要包括里所(RS, reed-solomon)码和磁盘阵列码等，如 ocean store, total recall 系统采用的冗余策略就是纠删码[5-6]. 针对上述冗余策略存在的局限性，Zhang 等[7]提出了再生码的概念.

因部分重复(FR, fractional repetition)码[8]在修复故障节点的过程中拥有较低的修复局部性和修复带宽开销以及能够对故障节点进行无编码快速修复的特性，进而受到广泛关注. 但传统 FR 码主要针对静态分布式存储系统，节点存储容量和数据块重复度保持不变. 为改善传统 FR 码存在的缺陷，提出一种基于哈夫曼树的可变重复度的异构部分重复(HVFR, heterogeneous variable fractional repetition)码.

1　文件热度计算

对首次存储和非首次存储的文件，采用不同的计算方式分别定义其文件热度.

定义 1　定义文件的冷热分类为 $\{F_c, F_h\}$，其中 F_c, F_h 分别代表冷文件和热文件，系统中所有文件包含其中.

对首次存储的文件，其计算周期内文件热度为

$$H = \sum_{i=1}^{n} \upsilon_i \chi \tag{1}$$

其中：υ_i 为指定计算周期内系统文件中数据块 i 的影响因子，χ 为数据块 i 的访问次数，n 为系统文件分割的编码数据块的总数. 若在指定周期内数据块 i 被用户访问，υ_i 较大.

对于非首次存储的文件，通过对编码数据块的引用量来计算文件热度，其计算周期内文件热度为

$$H = \sum_{i=1}^{r} Z_t \tag{2}$$

其中：Z_t 为系统文件中编码数据块的引用量，r 为总的编码数据块. 文件的冷热性可以利用阈值函数来判断，根据强度不变性质设置阈值为

$$f(w) = \left(\prod w\right)^{1/n} \tag{3}$$

首次存储文件时，w 为数据块的访问次数($w = \chi$)；非首次存储文件时，w 为数据块的引用量($w = Z_t$)；n 表示系统文件编码数据块的总数，通过设定阈值，比阈值高的则为 F_h，反之为 F_c.

2　基于成对平衡设计的部分重复码

2.1　成对平衡设计

假设 v 与 λ 为正整数，$\mathbf{N}$ 为正整数集合，V 与 φ 为任意集合，又 $\psi = (V, \varphi)$ 为有限关联结构，若满足以下条件：

1) $|V| = v$；

2) φ 是 V 的一些子集的集合，且对任意区组 $B \in \varphi$，都有 $|B_i| = \mathbf{N}$；

3) V 中任意一对不同的点都恰好同时包含在 λ 个区组中；

则称 $\psi = (V, \varphi)$ 为一个成对平衡设计(PBD, pair-wise balanced design)[9]，记作 $(v, \mathbf{N}, \lambda)$-PBD. 这里 v 称为成对平衡设计的阶，λ 称为相遇数，当区组的大小都相同且都为 s 时，即 $\mathbf{N} = \{s\}$，这种特殊的成对平衡设计就是平衡不完全区组设计(BIBD, balanced incomplete block design)，记作 (v, s, λ)-BIBD.

设 $V = \{1,2,3,4\}$，根据成对平衡设计生成区组的集合，有 $\varphi = \{B_1 = \{1,2,3\}, B_2 = \{1,2,4\}, B_3 = \{1,3,4\}, B_4 = \{2,3,4\}\}$. 可以看出，$V$ 中任意 2 个不同的元素恰好存在 2 个区组中，每个区组的大小为 3，因此 $\psi = (V, \varphi)$ 是一个 $(4,3,2)$-BIBD 平衡不完全区组设计.

2.2　基于成对平衡设计的部分重复码

首先采用 (v, s, λ)-BIBD 平衡不完全区组设计，其中 $\varphi = \{B_1, B_2, \cdots, B_n\}$；然后可以根据平衡不完全区组设计构造一个重复度 $\rho = s$ 的 FR 码. 平衡不完全区组设计满足：

$$\lambda \binom{v}{2} = n \binom{s}{2} \tag{4}$$

重复度 $\rho = s$ 的 FR 码的构造方式为

$$N_j = \{i : j \in B_i\},\ j = 1, 2, \cdots, v \tag{5}$$

该 FR 码中每个数据块 $i(1 \leqslant i \leqslant n)$ 的重复度 $\rho_i = s$，例如 $V = \{1,2,\cdots,7\}$，根据成对平衡设计生成区组的集合为 $\varphi = \{B_1 = \{1,2,4\}, B_2 = \{1,3,7\}, B_3 = \{1,5,6\}, B_4 = \{2,3,5\}, B_5 = \{2,6,7\}, B_6 = \{3,4,6\}, B_7 = \{4,5,7\}\}$. 可以看出，$V$ 中任意 2 个不同的元素恰好存在一个区组中，而且每个区组的大小相同，均为 3，因此 $\psi = (V, \varphi)$ 是一个 $(7,3,1)$-BIBD 平衡不完全区组设计. 根据式(5)可以构造一个 FR 码，FR 码各存储节点中存储的数据块为 $N_1 = \{1, 2, 3\}, N_2 = \{1, 4, 5\}, N_3 = \{2, 4, 6\}, N_4 = \{1, 6,$

7}, $N_5=\{3,4,7\}$, $N_6=\{3,5,6\}$, $N_7=\{2,5,7\}$. 可以看出,构造的FR码可以存储7个数据块,且每个数据块复制3倍.

利用成对平衡设计生成的FR码存储的数据块重复度都相同,然而实际分布式存储系统对数据的访问往往是不均衡的,"热"数据经常被访问,"冷"数据很少被访问[10],因此,需要对不同类型的数据块采用重复度不同的存储机制. 进一步利用成对平衡设计也可以构造重复度不同的异构FR码,采用一个(v,S,λ)-PBD成对平衡设计,其中$\varphi=\{B_1,B_2,\cdots,B_n\}$,假定$V$中元素对应的重复度分别为$r_1,r_2,\cdots,r_v$,其中成对平衡设计满足:

$$\lambda\binom{v}{2}=\sum_{i=1}^{n}\binom{|B_i|}{2} \tag{6}$$

利用式(5)可以构造一个重复度不同的异构FR码,其中异构FR码中每个数据块$i(1\leqslant i\leqslant n)$的重复度$\rho_i=|B_i|$,并且节点$N_j$的存储容量为$r_j(j=1,2,\cdots,v)$,即节点$N_j$的存储容量对应成对平衡设计$V$中第$j$个元素对应的重复度. 例如$V=\{1,2,\cdots,5\}$,根据成对平衡设计生成区组的集合为$\varphi=\{B_1=\{3,4\},B_2=\{4,5\},B_3=\{2,3,5\},B_4=\{1,3,5\},B_5=\{1,2,4,5\},B_6=\{1,2,3,4\}\}$.

显然,$\psi=(V,\varphi)$是一个$(5,(2,3,4),2)$-PBD成对平衡设计. 根据式(5)可以构造一个异构FR码,该异构FR码的各存储节点中存储的数据块为$N_1=\{4,5,6\}$,$N_2=\{3,5,6\}$,$N_3=\{1,3,4,6\}$,$N_4=\{1,2,5,6\}$,$N_5=\{2,3,4,5\}$. 可以看出,构造的FR码可以存储6个数据块,其中数据块1、2的重复度为2,数据块3、4的重复度为3,数据块5、6的重复度为4.

3 可变重复度的异构部分重复码构造

鉴于实际的分布式存储系统大多属于异构存储系统,且分布式存储系统对数据的访问往往是不均衡的,"热"数据经常被访问,"冷"数据很少被访问. 如果在实际的异构分布式存储系统中使用传统的FR码,往往会限制存储系统某方面的性能,如系统的存储效率、数据访问吞吐量等. 为此,提出一种基于哈夫曼树的可变重复度的HVFR码,构造步骤如下.

步骤1 对过去一段时间内分布式存储系统的轨迹数据进行统计分析,得到k个数据块$d_1,d_2,\cdots,d_k$的访问频率.

步骤2 将具有不同访问频率的k个数据块$d_1,d_2,\cdots,d_k$作为哈夫曼树的叶子节点,这k个数据块$d_1,d_2,\cdots,d_k$的访问频率对应作为哈夫曼树中叶子节点的权值,根据哈夫曼算法构造哈夫曼树.

步骤3 根据构造的哈夫曼树,按照

$$\rho_i=\left\lfloor\frac{k-L_i}{\varepsilon}\right\rfloor+l,\quad l\in Z,\varepsilon\in N^* \tag{7}$$

确定访问频率不同数据块$d_i(1\leqslant i\leqslant k)$的重复度为$\rho_i(i=1,2,\cdots,k)$. 式(7)中,$L_i$为第$i$个叶子节点的路径长度,$\varepsilon$为重复度因子,$l$为修正因子.

步骤4 对访问频率不同的k个数据块$d_1,d_2,\cdots,d_k$进行最大距离可分码(MDS, maximum-distance-separable)编码产生p个校验块,将第$j(1\leqslant j\leqslant p)$个校验块的重复度用符号$\rho'_j$标记,而且$\rho'_j$满足条件$\min(\rho_i)\leqslant\rho'_j\leqslant\max(\rho_i)-1, i=1,2,\cdots,k$. 将得到的$p$个校验块添加到访问频率不同的$k$个数据块,得到$k+p$个数据块及其对应的重复度.

步骤5 采用(v,S,λ)-PBD成对平衡设计,构造$k+p$个区组$B_1,B_2,\cdots,B_{k+p}$,记为$\varphi=\{B_1,B_2,\cdots,B_{k+p}\}$. 将第$i$个区组$B_i$的大小设置为$k+p$个数据块中第$i$个数据块对应的重复度为$\rho_i$,即满足$|B_i|=\rho_i(i=1,2,\cdots,p+k)$.

步骤6 根据式(5)构造可变重复度的异构FR码.

根据轨迹数据分析,在过去一段时间内$k=8$个数据块d_1, d_2, …, d_8的访问次数分别是{10, 20, 50, 60, 70, 90, 150, 200}. 然后将这8个具有不同访问频率的数据块d_1, d_2, …, d_8作为哈夫曼树的叶子节点,根据哈夫曼算法构造哈夫曼树,且所构造的哈夫曼树中叶子节点的权值对应数据块的访问次数,如图1所示.

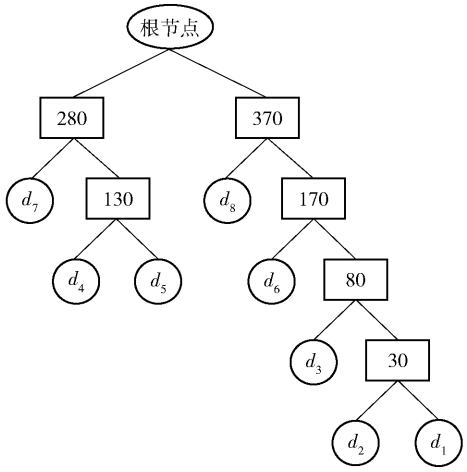

图1 哈夫曼树构造示意图

根据图 1，按照式(7)可求出不同访问频率数据块的重复度(见表 1)，且重复度因子 ε 取 2，修正因子 l 取 1. 对 $k=8$ 个数据块 d_1, d_2, …, d_8 进行 MDS 编码，生成 2 个校验块 p_1 和 p_2，重复度分别取 3 和 2.

表 1　数据块重复度

数据块	路径长度	重复度
d_8	2	4
d_7	2	4
d_6	3	3
d_5	3	3
d_4	3	3
d_3	4	3
d_2	5	2
d_1	5	2

根据满足式(6)的条件，考虑采用(6, (2, 3, 4), 2)-PBD 成对平衡设计，其中 $V=\{1,2,\cdots,6\}$, $S=\{2,3,4\}$, $\lambda=2$，即 V 中任意 2 个元素存在 2 个区组中. 在(6,(2, 3, 4), 2)-PBD 成对平衡设计中构造 $k+p=10$ 个区组，生成的区组集合 $\varphi=\{B_1, B_2,\cdots,B_{10}\}$，具体为 $\varphi=\{B_1=\{3,6\}, B_2=\{4,6\}, B_3=\{2,3,4\}, B_4=\{3,4,5\}, B_5=\{2,5,6\}, B_6=\{1,4,6\}, B_7=\{1,3,5,6\}, B_8=\{1,2,4,5\}, B_9=\{1,2,3\}, B_{10}=\{2,6\}\}$

根据式(5)可以得到一个新的异构 FR 码，其中异构 FR 码中各存储节点存储的数据块为 $N_1=\{6, 7, 8, 9\}$, $N_2=\{3, 5, 8, 9, 10\}$, $N_3=\{1, 3, 4, 7, 9\}$, $N_4=\{2, 3, 4, 6, 8\}$, $N_5=\{4, 5, 7, 8\}$, $N_6=\{1, 2, 5, 6, 7, 10\}$

最后将具有不同重复度的数据块和校验块存放到新构造的 FR 码中，并存储到对应的节点中，可得到一个$(n,k,(\rho))$HVFR 码. 其中 n 为存储节点，k 为原始数据块，(ρ)为重复度的集合. 基于哈夫曼树的可变重复度的异构 FR，即(6,8,(2,3,4))HVFR 码的编码方案如图 2 所示.

HVFR 码既满足了现实异构分布式存储的实际需求，又满足了冷热数据的存储机制，提高了系统热数据的访问效率，而且降低了存储系统存储的成本，在修复故障节点的过程中可以无编码修算，降低了修复复杂度.

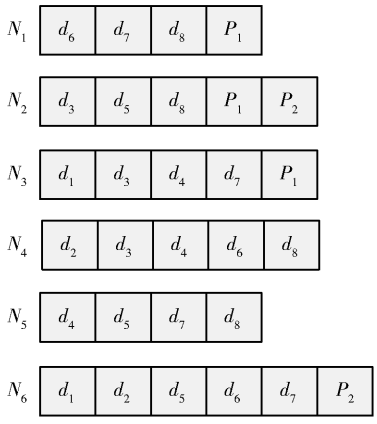

图 2　(6,8,(2,3,4))HVFR 码的编码方案

4　性能分析

下面对 HVFR 码进行性能分析，主要讨论其修复局部性、存储效率、修复复杂度和修复时间，并与最常见的 RS 码、SRC(simple regenerating codes)以及传统 FR 码进行对比.

4.1　修复局部性

修复局部是指节点故障修复时需要连接的节点数，当单节点故障时，对于(n,k)RS 码，修复单节点故障时需要连接 k 个存活节点. 因此，RS 码单故障节点的修复局部性为 k；对于 SRC，原文件分为 f 个子文件，则修复单故障节点需要从 $2f$ 个节点下载数据块进行修复，SRC 的修复局部性为 $2f$；对于 HVFR 码，因为每个节点 N_j 的存储容量是 r_j $(j=1,2,\cdots,v)$，而且 HVFR 码从存活节点下载的数据块可能大于 1，所以 HVFR 码第 j 个单节点故障的修复局部性 d_j 小于等于该节点的节点容量，即 $d_j \leqslant r_j$. 对于 RS 码和 SRC，系统出现两节点故障时均需要连接 k 个存活节点才能修复故障节点，则此时 RS 码和 SRC 的修复局部性均为 k；对于 HVFR 码，任意 2 个节点 N_m 和 N_y 故障时，修复局部性 $d \leqslant ((r_m + r_y) - |N_m \cap N_y|)$.

假设 $k=8$，基于哈夫曼树可变重复度的异构 FR 码外部采用(10,8)MDS 码. 当出现单节点故障时，与图 3 所示的情况相同，(10,8)RS 码修复单节点故障的修复局部性为 8；当 f 取 3 时，(10,8,3) SRC 的修复局部性为 6；采用图 2 示例中 HVFR 码综合修复局部性为 2.5. 当系统出现 2 个节点故障时，(10,8)RS 码和 SRC 的修复局部性均为 8，采用(6,8,(2,3,4))HVFR 码的修复局部性为 3.

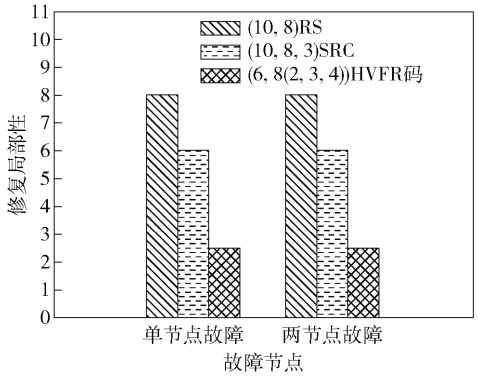

图3 修复局部性对比

4.2 存储效率

传统FR码对每个数据块都采用同样的重复度进行复制,而笔者提出的HVFR码采用的是可变的重复度. 对于热数据采用较高的重复度,对于冷数据采用较低的重复度,这样不但可以提高热数据的并行读取效率,还可以提高系统的存储效率. 假设原文件包含 k 个原始数据块,HVFR码把原始数据块划分为重复度为 $\rho_1,\rho_2,\cdots,\rho_n$ 的不同数据块,其中 $\rho_1>\rho_2\cdots>\rho_n$,不同重复度的数据块分别占原始数据块总数的 $T_1\%,T_2\%,\cdots,T_n\%$. 传统FR码的重复度一般取 $\rho=3$,则共需要存储 $3k$ 个数据块,采用HVFR码则需要存储 $k\sum_{i=1}^{n}T_i\%\rho_i$ 个数据块.

假设原始数据块的数量为 k,HVFR码中 $\rho_1=4,\rho_2=3,\rho_3=2$. 其中 $T_1=20,T_2=20,T_3=60$,传统FR码的重复度 $\rho=3$. 在原始数据块 k 为100,200,300,400的情况下,比较了2种编码的存储效率,传统FR码比HVFR码需要多存 $2k/5$ 个数据块,如图4所示. 因此,HVFR码比传统FR码的存储效率提高了13.33%. HVFR码在提高存储效率的同时,也保证了热数据的并行读取,比传统FR码更适用于实际的分布式存储系统,而且花费的存储成本更低.

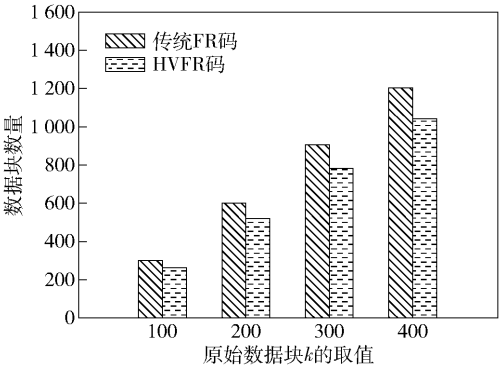

图4 存储效率性能比较

假设文件的大小为 M,t 为单位时间内的寻址时间,分别采用RS编码、HVFR码、三副本和SRC进行存储,几种编码方案的性能如表2所示.

表2 几种编码方案的性能

性能参数	三副本	SRC	RS	HVFR
存储开销	M	$(f+1)M/k$	M/k	Mr_j/k
修复复杂度	0	$(f-1)(f+1)$	k^2+k	0
寻址时间	tM	$(f-1)(f+1)tM$	$(k^2+k)tM$	tM

4.3 故障节点修复复杂度

对于RS码,修复单故障节点时首先需要从所有存活节点中任意连接 k 个存活节点来重构原文件;然后再编码生成需要修复的故障节点. 所有的运算都是在有限域 GF(q) 中进行的,因此,RS码的修复过程需要 k^2+k 次有限域乘法运算. 对于SRC,修复一个数据块系统需要完成 $f-1$ 次异或运算操作. 由于SRC中每个节点存储 $f+1$ 个数据块,SRC的修复过程需要 $(f-1)(f+1)$ 次异或运算. 对于HVFR码,修复故障节点时,不需要任何解码译码操作,直接从其他存活节点下载数据,通过复制进行修复,修复过程异或运算次数为0. 由此可见,HVFR码的修复复杂度相较于RS码和SRC明显较低.

4.4 修复时间

采用Hadoop分布式文件系统,对笔者提出的HVFR码与RS码、SRC进行了故障节点修复时间对比. 实验中,系统服务器的CPU配置为Intel Xeon E5-2680 2.5 GHz,内存大小为24 GB,在VMWare软件上搭建一个伪分布式集群,操作系统为RHEL7(Linux内核3.10.0),整体框架中将虚拟机作为数据的存储节点. 每台虚拟机需装有Nginx进行通信,并通过配置服务脚本设置为开机自启,搭建配置相同的实验环境.

实验1 单节点故障修复时间测试. 设定存储节点的数量 $n=10,k=8$,分别在实验环境中部署(10,8)RS、(10,8,3)SRC以及(6,8,(2,3,4))HVFR码. 在分布式存储系统中,每个节点的存储容量分别设置为200,400,600 MB,对3种编码在单节点失效修复时间进行测试. 在相同条件下多次测试,取平均测试值,测试结果如图5所示. 单节点故障时,HVFR码与RS码和SRC相比大幅度降低了故障节点的修复时间.

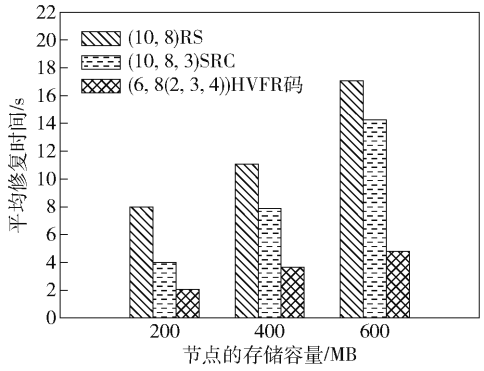

图 5　单节点故障修复时间对比

实验 2　两节点故障修复时间测试. 设定存储节点的数量 $n=10, k=8$,分别在实验环境中部署(10,8)RS、(10,8,3)SRC 以及(6,8,(2,3,4))HVFR 码. 每个节点的存储容量分别设置为 200,400,600 MB,对 3 种编码在两节点失效修复时间进行测试,多次测试,取平均测试值,如图 6 所示. 两节点故障时,与 RS 码和 SRC 相比,HVFR 码同样大幅度降低了节点失效修复时间. 而且可以看出,对于多节点故障的修复,相比于 RS 码和 SRC,HVFR 码的修复时间更快.

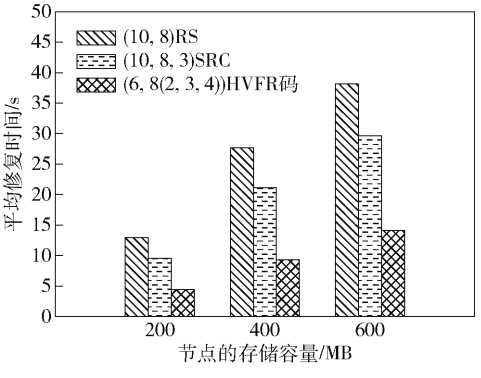

图 6　两节点故障修复时间对比

5　结束语

考虑到传统 FR 码不适应分布式存储系统中冷、热数据的存储,提出一种基于哈夫曼树的可变重复度的异构部分重复码,即 HVFR 码. 基于热度不同,数据块重复度也不同的思想,对于热数据采用重复度较高的复制,能够提升热数据的并行访问速率,同时也使得热数据的容错性能更好. 对于冷数据采用重复度较低的复制,有效地节省了存储空间,降低了存储设备的成本. HVFR 码的设计中考虑了冷、热数据的存储问题,在实际的分布式存储系统应用中具有更广泛的发展前景.

参考文献:

[1] Zhu Bing, Li Hui, Shum K W, et al. HFR code: a flexible replication scheme for cloud storage systems[J]. IET Communications, 2015, 9(17): 2095-2100.

[2] Lee O T, Akash G J, Kumar S D M, et al. Storage node allocation methods for erasure code-based cloud storage systems[J]. Arabian Journal for Science and Engineering, 2019, 44(11): 9127-9142.

[3] Wang Jing, Yan Zhiyuan, 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.

[4] Zhu Bing, Shum K W, Li Hui. Fractional repetition codes with optimal reconstruction degree [J]. IEEE Transactions on Information Theory, 2020, 66(2): 983-994.

[5] Xia Junxu, Guo Deke, Cheng Geyao. In-network block repairing for erasure coding storage systems[J]. Concurrency and Computation: Practice and Experience, 2019, 31(24): e5432.

[6] Bao Han, Wang Yijie, Xu Fangliang. An adaptive erasure code for jointCloud storage of Internet of things big data[J]. IEEE Internet of Things Journal, 2020, 7(3): 1613-1624.

[7] Zhang Huayu, Li Hui, Zhu Bing, et al. Minimum storage regenerating codes for scalable distributed storage [J]. IEEE Access, 2017, 5: 7149-7155.

[8] Nam M Y, Kim J H, Song H Y. Some constructions for fractional repetition codes with locality 2 [J]. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2017, E100. A(4): 936-943.

[9] Zhu Bing, Li Hui, Li S Y R. General fractional repetition codes from combinatorial designs[J]. IEEE Access, 2017, 5: 26251-26256.

[10] Lee K S, Park H, No J S. New binary locally repairable codes with locality 2 and uneven availabilities for hot data[J]. Entropy, 2018, 20(9): 636-644.