第 55 卷　第 4 期
2019 年 7 月

南京大学学报(自然科学版)
JOURNAL OF NANJING UNIVERSITY
(NATURAL SCIENCE)

Vol. 55, No. 4
July, 2019

DOI:10.13232/j.cnki.jnju.2019.04.017

RSboFMC:提高数据可用性和负载均衡性的碎片矩阵缓存策略

齐小刚[1,3],强　敏[1*],刘立芳[2,3]

(1.西安电子科技大学数学与统计学院,西安,710071;
2.西安电子科技大学计算机科学与技术学院,西安,710071;
3.西安电子科技大学宁波信息技术研究院,宁波,315200)

摘　要:保证动荡环境下数据可被访问概率对数据存储网络十分重要,其可行方法之一是设计合理的存储策略,提高网络的数据可用性.将存储策略分为复制策略和放置策略进行设计,提出了基于碎片矩阵和缓存的存储策略RSboFMC(Replication Strategy based on Fragment Matrix and Cache),提高动荡环境下的数据可用性.其以重建效率和存储开销为目标,设计缓存机制和基于碎片矩阵的数据分块机制优化复制策略;以负载均衡为目标,设计基于分区和顺逆序的分发机制优化放置策略.仿真结果表明,RSboFMC在数据可用性和负载均衡性方面均优于其他策略,且具有良好的扩展性.

关键词:数据分块机制,数据可用性,缓存机制,数据分发机制

中图分类号:TP302　　**文献标识码:**A

RSboFMC:Replication strategy based on fragment matrix and cachefor improving data availability and load balance

Qi Xiaogang[1,3], Qiang Min[1*], Liu Lifang[2,3]

(1. School of Mathematics and Statistics, Xidian University, Xi'an, 710071, China;
2. School of Computer Science and Technology, Xidian University, Xi'an, 710071, China;
3. Xidian-Ningbo Information Technology Institute, Ningbo, 315200, China)

Abstract: It is critical for data storage network to provide expected possibility of data accessed under churn. One of possible method is to design proper storage strategy for increasing data availability of networks. RSboFMC(Replication Strategy based on Fragment Matrix and Cache) was proposed based on the fragment matrix and the cache to raise data availability under churn. It divided the storage strategy into replication and placement strategies, then the cache mechanism and the data partitioning mechanism which was based on fragment matrix were designed to optimize the replication strategy in consideration of reconstruct effectiveness and storage cost, as well as the distributing mechanism based on zone partition and opposite sequences was designed to optimize the placement strategy, considering the

基金项目:国家自然科学基金(61877067,61572435),教育部-中国移动联合基金(MCM20170103),西安市科技创新项目(201805029YD7CG13-6),宁波市自然科学基金(2016A610035,2017A610119)

收稿日期:2019－02－03

*通讯联系人,E-mail:mqiang@stu.xidian.edu.cn

loadbalance. Simulation results show RSboFMC outperforms other strategies in terms of the data availability and the load balance. Besides, it has good scalability.

Key words: data partitioning mechanism, data availability, cache mechanism, data distributing mechanism

互联网的普及使得网络存储应用于越来越多的场合,且由于传统集中式存储结构扩容和I/O访问效率的限制,目前多采用分布式存储结构存储海量数据以提高网络抗毁性、数据可用性,同时降低访问失败率、系统故障率[1]. 分布式存储系统通过扩展集群服务器规模提高系统存储容量、计算和性能的能力,具有良好的扩展性和中心节点失效的无效性,在近些年来受到了广泛的关注. 现有分布式存储结构根据实现目标和部署环境大致可分为两类:P2P存储系统和数据中心存储系统,它们具有运行和并行运算的优势,可解决大规模应用问题[2]. 数据存储网络的应用从MP3下载被人们所"认识",随后在文件共享、视频直播与点播、即时通讯(网上电话)、网络聊天、网络存储、网格计算等领域发展迅速,但随着如今数据井喷式的发展,它越来越多地被用于区块链[3]、分布式在线社交网络[4]、无线网络[5]及物联网[6]中. 设计合适的数据存储策略是提高数据存储网络容错性的可行方法之一,良好的存储策略可以提高数据访问效率,并尽可能均衡负载. 现有的存储系统如Amazon和Google通常使用全复制策略提高网络动荡情况下的数据可用性[7],但随着数据量的增大,存储开销和带宽开销可能也呈线性或指数型增长. 编码策略可以很好地降低存储开销[8],但通常所用的基于范德蒙矩阵的编码策略需要操作几何级数矩阵且具有较高的算法复杂度. 综上,需要提出一种新的存储策略,使得降低存储开销而无需通过操纵复杂矩阵,同时保持较高的动荡容忍性.

一般来说,存储策略主要由复制策略和放置策略组成,其中复制策略是生成文件备份并存储在不同服务器中,避免服务器崩溃造成的数据丢失[9],主要分为全复制或分块复制;放置策略是节点集选择问题,是提高系统容错性的方法之一[10]. 放置策略大致分为顺序放置和随机放置. 顺序放置采用固定的映射方式将数据各个副本放置到相应节点上,简单、易于实施,但容易引发级联失效. 而随机映射副本数据到相应节点上的方式,具有较高的容错性,但数据访问的本地性较弱,对系统的性能影响较大. 本文提出一种基于碎片矩阵和缓存的全新存储策略RSboFMC(Replication Strategy based on Fragment Matrix and Cache),主要由数据分块机制、分发机制和缓存机制组成,在降低存储开销提高数据访问效率的同时,尽可能达到负载均衡. 碎片矩阵主要是用于数据分块机制中,无需对复杂矩阵进行操作便可对文件进行类似编码的处理,有效地减少了存储开销. 不可否认,全复制策略在高动荡情况下具有较好的数据可用性,所以提出存在数据访问时触发的缓存机制,在主节点上重建文件进行缓存,提高存储网络的动荡容忍性. 基于分区且考虑了各区域最大副本节点数量的顺逆序分发机制具有良好的扩展性和负载均衡性,其可理解为一种特殊的随机放置策略. 为验证RSboFMC的策略性能,不同于二项分布下定义的部分场景适应性数据可用性指标,本文基于超几何分布提出和网络规模、存储节点集等相关的数据可用性评价指标,分别以冗余因子、网络攻击强弱和网络规模为自变量,评价RSboFMC策略的数据可用性性能;再根据实际负载值和理想负载值之差设定负载波动公式,在此基础上得出偏差分布图,全面评价RSboFMC策略的负载均衡性性能.

基于性能指标对比不同复制策略,证明了RSboFMC在数据可用性和负载均衡方面具有较好的性能,本文的主要贡献如下:

(1)本文所提出的存储策略中数据分块机制,是基于碎片矩阵对原文件作碎片化处理,可

看作一种特殊的编码矩阵,但无需操作类似于范德蒙矩阵的复杂矩阵;而缓存机制的提出,使得在一定条件下的RSboFMC策略具有全复制策略的相关性能.

(2)RSboFMC存储策略的分发机制,是基于冗余度对系统空间进行分区,并根据文件块编号顺序选择合适的文件碎片,基于顺逆序和区域最大副本节点集存储到相关区域的合适节点上.因此,此算法具有良好的负载均衡性和可扩展性,不仅可应用于仿真环境设计下的网络规模中,也可应用于更大型的系统中.

(3)基于超几何分布提出全新的性能评价指标—重新定义的数据可用性f_{Avail},评估算法在动荡网络环境下的数据可用性性能;提出负载均衡波动指标$C_{\text{balance}_{\text{Load}}}$评估不同分发机制下的负载均衡性.

1　相关工作

由不可避免的原因造成的随机或蓄意攻击会引起数据损坏,但即使没有蓄意攻击,在一段时间之后,数据仍可能被损坏,因此对一个存储系统来说,提高网络相关性能的存储策略十分必要,其可从数据被存储形式、存储分发机制等方面进行优化策略设计以提高网络相关性能.数据被存储形式主要分为全复制和分块复制,Zhang et al[11]通过建立固定长度的list表识别流行文件和冷门文件,再将存储节点集分为冷热节点对数据进行存储,只需保证热节点处于活跃状态以确保系统性能,并以此应对全复制带来的高存储开销问题.Zhu et al[12]在FR编码的基础上提出基于分组设计的HFR数据分块机制,将文件分为正常块和奇偶校验块.在冗余因子R确定的情况下,每个正常块生成R份,校验块各生成$R-1$份.仿真结果表明,其在随机访问情况下可以优化存储空间并缩短数据重构时间,同时在节点失效情况下,具有多个修复组合方案.应用数据分块机制的存储优化策略,经常需要考虑改善数据块被重建效率,其有效方法之一是建立缓存.Inacio and Dantas[13]通过大量实验观察14个参数对Orange文件系统响应时间和吞吐量的影响,提出粗粒度页面缓存多元分析模型,提高并行文件系统的读写性能.存储分发机制与网络性能也有十分重要的关系,良好的分发机制可以提高数据可用性,增加系统动荡容错性.Lakshman and Malik[14]提出应用于Cassandra系统的随机分发机制,可以有效减小级联失效的概率,但可能造成节点负载不均衡.Silva et al[15]提出基于哈希表的全局目录式存储分发方法和基于希尔伯特空间填充曲线(Hilbert Space Filling Curves,HSFC)的查询机制,通过仿真结果证明可应用于网络规模、节点数量或存储信息量动态变化的环境中.

2　设计概述

良好的复制策略可以提高数据存储网络的相关性能,如数据可用性、网络可靠性等.基于不同侧重点,复制策略的约束条件也不相同,达到的性能也不同.本文以文件被存储性质和综合性能为约束条件,设计相关复制策略.

2.1　RSboFMC:基于碎片矩阵和缓存的复制策略　一般来说,在网络动荡性较小的情况下,文件碎片化存储的可用性性能高于全复制存储,但当网络节点大规模失效的情况下,全复制存储的数据可用性性能较高但所需存储空间也相对较大,所以全编码或全复制模式都不是最好的.在保持一定数据可用性的同时,应降低存储开销和访问时延,所以本文提出基于碎片矩阵和缓存的复制策略RSboFMC,利用碎片矩阵分块机制降低存储开销,利用缓存机制提高获取效率,还可解决相应的热点问题.从一定程度上来讲,RSboFMC可以看作一个特殊的分块复制和全复制策略的混合复制策略.当数据文件被外界请求时,存储于不同节点上的文件碎片在某一主节点上进行重建并缓存,使得数据碎片化的同时在网络中保持数据的一份完整副本,不仅降低热点文件的重建开销、提高服务质量,同时使得应用此策略的系统具有较高的动荡容忍性.

2.1.1 基于碎片矩阵的数据分块机制 考虑到较高的存储成本和数据更改的不频繁性,不需要以全复制的形式存储数据的多个副本,且需要消耗较高带宽的碎片矩阵分块机制不会频繁运作. 基于冗余度因子R和分块因子F构建碎片矩阵 fragmentMatrix:

(1)构造一个F阶基础矩阵$B(b_{ij})_{F\times F}$,其中$b_{i,j}=\{0,1\}$,当$i=0,j\in[0,F)$时,$b_{i,j}$值与均匀随机分布函数有关且$\sum b_{i,j}=R$.

(2)在B矩阵第一行值确定的基础上,按如下规则确定$F-1$阶:

①$b_{i,j}-b_{i,j+1}=1,b_{i+1,j+1}=1$.

②$b_{i,j}-b_{i,j+1}=-1,b_{i+1,j+1}=b_{i,j}$.

③$b_{i,j}-b_{i,j+1}=0$,if:$b_{i,j}+b_{i,j+1}=0$,

$b_{i+1,j+1}=b_{i,j}$;else:$b_{i+1,j+1}=1$.

④if:$b_{i,F-1}=1,b_{i+1,0}=1$;else:$b_{i+1,0}=0$.

(3)构造一维向量$F=(f_1,f_2,\cdots,f_F)$表示数据分块情况.

(4)在F基础上构造B的同型矩阵$D(d_{ij})_{F\times F}=diag(F)$.

(5)矩阵B和矩阵D作矩阵运算$FM=(-1)^i\times B\times D$生成碎片初始矩阵$FM$,$i\in 2n$.

(6)考虑到碎片初始矩阵中可能存在较多零元素,降低存储空间的利用率,类似于压缩矩阵,将F阶矩阵FM根据冗余度因子R变为$F\times R$矩阵,生成碎片矩阵 fragmentMatrix.

根据上述步骤,可以根据需要将文件分成F个文件碎片,每个碎片上包含R个数据块. 不同于简单的文件分块策略,其类似于基于范德蒙矩阵的编码机制,但更容易实现,具有较低的时间复杂度. 基于碎片矩阵的数据分块机制也有类似于编码机制的数据块和校验块. 在最坏情况下,校验块最小值为$F-R+1$. fragmentMatrix矩阵中的行向量代表一个文件碎片,根据2.2中的算法存储在网络中的一个节点上,矩阵行值确定副本节点集集合大小.

2.1.2 数据分发机制 本文设计正逆序原则和区域最大副本节点数原则以尽可能达到节点间的负载均衡和区域间的负载均衡. 基于冗余因子对网络节点进行分区操作,在对各区域节点集进行随机排序的基础上,根据2.1.1生成碎片矩阵,按照块编号顺序,依次将矩阵对应行存储于不同区域的合适节点上. 当对第一份文件完成分发时,虽然有最大副本节点数原则限制每个区域上的存储节点数量,但块编码较小的矩阵行上可能包含较大的块编码值,导致区域号较大的区域节点的负载平均值远小于区域号较小的区域节点平均负载值,故第二份文件以逆序进行分发. 以此类推,就可以很好地实现负载的均衡性能. 区域最大副本节点数原则主要是针对区域过载,因为碎片矩阵行上存储不同编码值的文件块,如果不限制区域最大副本节点数,按照块编码值进行顺逆序存储,可能会导致区域号较小和较大区域上的平均负载值远大于中间区域上的平均负载值. 故正逆序原则和区域最大副本节点数原则相辅相成,共同作用,保证了节点和区域的负载均衡性性能.

2.1.3 缓存机制 为节省存储空间降低存储开销,本文提出基于碎片矩阵的数据分块机制. 同编码机制一样,数据请求时需要进行数据重构,影响用户服务质量,所以在分块的基础上提出了缓存机制,可有效针对流行文件的资源访问,缓解热点问题,降低重建开销改善用户服务质量. 缓存机制有两种阈值标准:时间阈值TTL(Time to Live)和容量阈值CS(Cache Size). TTL标准使得存储忽略负载,本文以CS为阈值标准,基于当前最少使用原则LUC(Least Used Currently)删除溢出缓存.

合适的CS可以降低获取时延且维持较低的存储开销,CS值有两种极限结果:当其趋于最小值时,主节点上的存储开销较低但文件重建时将消耗较大的带宽;当其趋于最大值时,主节点上的存储开销达到最大但副本节点到主节点上的传输开销最小,如图1所示. 通过图1可以看出,存在合适的CS值,可以实现存储开销和重建带宽开销的相对平衡,即可构建最小化总开销模型,寻找合适的CS值,如式(1)所示.

$$\mathrm{Min}T=T_b+T_s \tag{1}$$

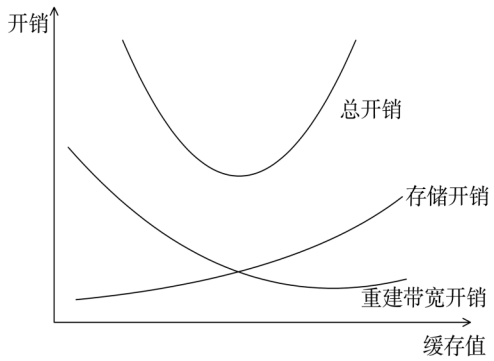

图1　开销与缓存值的关系

Fig. 1　Relationship between cost and cache size

$$\begin{aligned} s.t\ \ & T_b=\sum t_b\cdot x_i>0\\ & T_s=\sum t_s\cdot x_i>0\\ & \sum t_b\cdot x_j<\sum t_b\cdot x_i\\ & \sum t_s\cdot x_j>\sum t_s\cdot x_i\\ & x_i>0, x_j>x_i \end{aligned}$$

其中,T_b和T_s分别代表总重建带宽开销和总存储开销;t_b和t_s分别是带宽开销单价和存储开销单价;x_i代表缓存大小,即CS值.

2.2　RSboFMC算法　网络参与节点数量的变化可能会降低存储网络的性能,所以需要设计具有动荡容忍性的存储策略.RSboFMC算法主要由文件分块机制、分发机制、缓存机制构成,其基本思想是基于给定的冗余度对存储网络分区,将包含数据块1的文件碎片分发到不同网络区域的不同节点上,尽可能均衡负载,一旦存在数据访问,在主节点上重建并缓存.因网络分区只与冗余度有关,所以RSboFMC具有良好的扩展性.文件分块机制的核心思想依赖于2.1.1提出的碎片矩阵的构建,缓存机制核心是寻找式(1)的最优解,分发机制主要依赖式(2)和式(3)实现网络分区和负载均衡.算法流程图和伪代码分别如图2和算法1所示.

$$nodeValue=N/R \tag{2}$$

其中,$nodeValue$是各区域节点数量,N是网络规模,R是冗余度.

$$replicaValue=F/R \tag{3}$$

其中,$replicaValue$是各区域副本节点数的最大值,F是文件碎片数,R是冗余度.

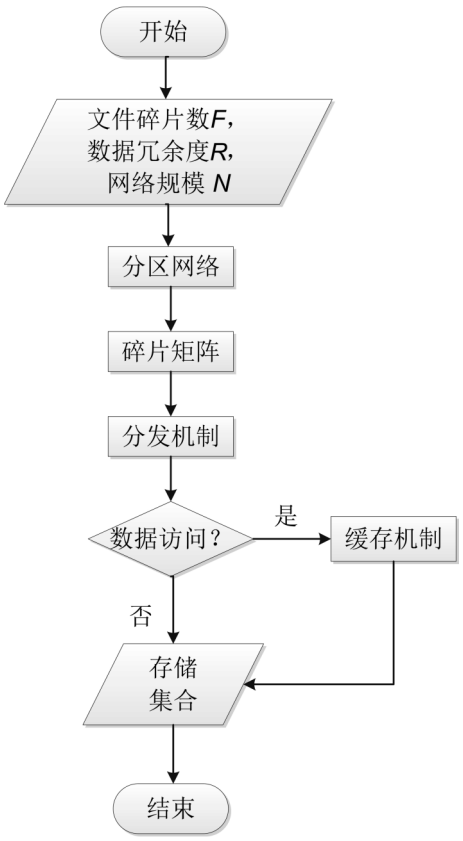

图2　算法流程图

Fig. 2　Flow diagram of algorithm

算法1　RSboFMC

输入:文件碎片数F,数据冗余度R,网络规模N

//Network partition based on R

1. int $zoneNumber=\frac{N}{R}$;
2. *roduceZoneNetwork*(N,zoneNumber);

//construct *fragmentMatrix*

3. produceFragmentMatrix(F,R);

//choose proper replicaSet

4. *balanceProtocol*(fragmentMatrix,F,R,N);
5. Boolean dataFlag=True;
6. if(dataFlag)
7. *cacheProtocol*(Δx,oldCacheSize,curCacheSize);

输出:存储表〈节点编号,碎片号〉

produceZoneNetwork():基于冗余度的分区策略使得算法具有良好的可扩展性.此函数以冗余度和网络规模为输入因子,设置区域检测值$judgeValue$判断每个节点的区域归属,建

立 list 集合 *zoneNode* 存储每个区域包含的所有节点,基于检测值循环遍历网络中所有参与节点,建立 map 集合 *zoneNetwork* 存储各区域及其对应节点集合.

constructFragmentMatrix():基于均匀分布函数生成基础矩阵 $B(b_{ij})$ 第一行,再依据 2.1.1 的(2)~(6)生成碎片矩阵 ***fragmentMatrix***,构建 list 集合 ArrayList〈ArrayList〉存储 ***fragmentMatrix***.

balanceProtocol():建立 hashMap 集合存储节点号与其对应的碎片信息,用 distriRecList.class 中的 avoidRepeatRandom()方法对各区域节点进行随机排序,避免对矩阵行列进行操作,降低时间复杂度,用 R 个 map 集合存储对应结果.为尽可能达到区域负载均衡,需要通过式(3)约束各个区域最大副本节点数.从文件块 1 开始,将所有包含其文件的碎片存储在不同区域的节点上.考虑到包含块 1 的碎片可能包含其他块值,为使当前方法中的矩阵修改操作不影响后续方法中作为输入因子的碎片矩阵且避免碎片重复存储,调用 deepCopy(list〈T〉src)进行深拷贝以建立碎片复制矩阵.在考虑节点已占用存储空间大小的基础上,通过碎片复制矩阵获得所需的文件碎片并将其存储在合适的节点上,同时将其对应位置的矩阵值置为-1.块编码值越大则其可能因位于同一碎片中较小块值检索而被存储,或由于区域最大副本数限制需要选择合适的存储区域,即可通过递归函数 replNodeLimit(map,Z)实现.特别的,若文件 A 从区域 1 开始分发,意味着当其分发结束后,区域号越小则已存储节点的集合越大,区域号越大已存储节点数量越小,所以文件 B 逆序存储,C 顺序存储,以此类推(图 3).此机制中的最大副本节点数约束均衡区域负载,文件正逆序分发均衡节点负载,两种策略共同作用,尽可能提高存储网络负载的均衡性能.

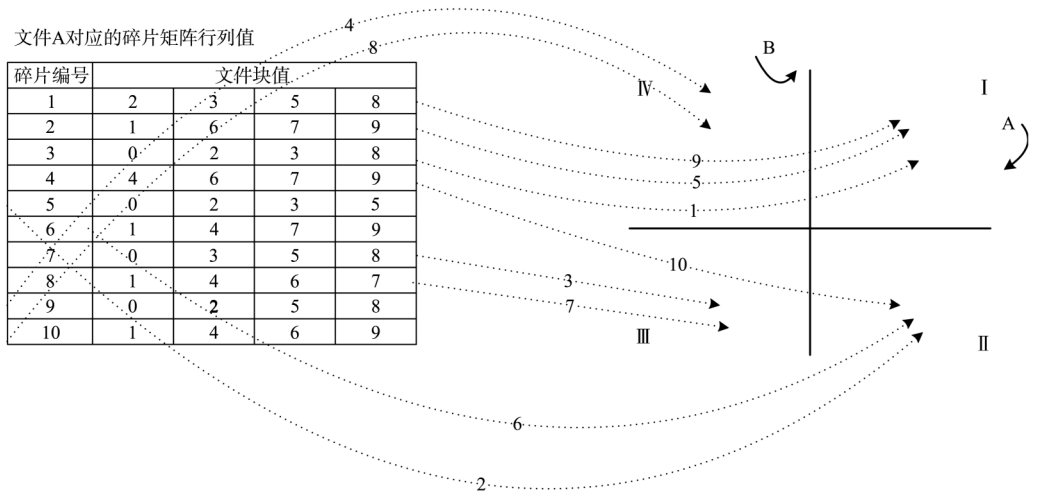

图 3 分发机制示意图

Fig. 3 Diagram of distributing mechanism

图 3 是 $F=10$,$R=4$ 时的分发机制示意图.左图的表格代表一个构建完成的 A 文件碎片矩阵,采用顺序进行文件碎片分发.找到包含文件块 0 的 4 个碎片编号 3,5,7,9,依次放入不同区域的合适的节点上;接着包含文件块 1 的碎片 2,6,8,10 完成存储,同时文件块 3,5,8,9,10 也相应完成存储;将包含文件块 2 的碎片 1 分配到区域Ⅰ上的节点后,区域Ⅰ可存储节点集达到极值,所以剩余包含文件块 4 的碎片 4 只能在区域Ⅱ或后续区域寻找合适节点,具体需要通过 replNodeLimit(map,Z)实现.

cacheProtocol():当存在数据访问时,此机制被触发.由 2.1.3 分析可知,此方法的核心是找到一个合适的缓存大小,在保持较低存储开

销的同时获得低带宽开销下的较高数据获取效率.根据最小化开销模型可通过简单的启发式算法得到合适的CS值,但需要根据不同时间段访问的变化情况不断地进行微调,所以提出自适应算法获取CS值,如算法2所示.当缓存文件内存值大于CS值时,根据LUC原则判断过期缓存并删除.

算法2　自适应CS值

输入:缓存单次增加值Δx,更新前缓存值oldCacheSize,更新后缓存值curCacheSize

1. for data access,do
2. compute T_s+T_b,x_i=oldCacheSize;
3. compute $T_s^{'}+T_b^{'}$,x_j=curCacheSize=$x_i+\Delta x$;
4. end for
5. $T=T_s+T_b$;
6. $T^{'}=T_s^{'}+T_b^{'}$;
7. if ($T^{'}>T$)
8. if ($x_j>x_i$)
9. $x_{j^-}=\Delta x$;
10. else
11. $x_{j^+}=\Delta x$;
12. end if
13. else
14. if ($x_j>x_i$)
15. $x_{j^+}=\Delta x$;
16. else
17. $x_{j^-}=\Delta x$;
18. end if
19. end if
20. $x_i=x_j$

3　性能评估

存储策略一般由复制策略和放置策略组成.本文以全复制方法(allReplica)和码率为1/2的编码策略(coded)作为复制策略组的对照,以随机放置策略(random)和顺序放置策略(sequential)作为放置策略组的对照,以动荡容忍下可用性和负载均衡性为评价指标,不同方法为参照,进行多次仿真实验,证明RSboFMC策略相关性能的优越性.

假设系统初始由N个节点构成,所有节点加入网络,并完成数据分发机制.每次实验由初始载入阶段开始,为使实验结果更具现实意义,将仿真环境设置为攻击下的动荡网络.攻击情况下节点失效在数学计算上可看作是一种特殊的网络动荡,故在后续的仿真结果中,均以动荡率表示攻击程度.

数据可用性提高意味着数据可以被访问的概率提高.现有文献大多以二项分布定义节点失效概率,再用此概率计算节点有效性,评估不同网络环境下的数据可用性[16].由于二项分布式中只有节点这一变化因子,此情况下计算的可用性只与节点自身能力有关,如节点在t时段内x时刻平均在线时长.针对特定网络,如社交网络,根据其用户在线时长数据,用节点有效性衡量网络数据可访问效率是有意义的,即此评价指标没有普适性.由于网络规模已知,即总体样本容量已知,本文基于超几何分布提出一个重新定义的数据可用性评价指标(式(4)).假设节点不可预测情况下失效概率相互独立,用离散概率描述一定网络规模下n个节点在攻击情况下不可预测性失效,其中包含有k个存储节点的概率,不仅可以评价节点有效性,还可评估存储节点正常运行遭遇攻击时,数据在不可预测情况下丢失时网络中的数据可用性.

$$f_{\mathrm{Avail}}(k,n,m,N)=P(x=n)=1-\frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}} \tag{4}$$

其中,k是存储节点失效数量,n是失效节点数量,m是存储节点集大小,N是网络规模.

3.1　数据可用性性能

3.1.1　冗余度因子影响　在动荡率恒定(0.5)的情况下改变副本数量,通过本文提出的数据可用性评估副本数量对算法的影响.众所周知,副本数量与动荡情况下鲁棒性成正比,但考虑到其与存储空间、维护开销成反比,设定副本数分别为2,4,6,8.

由图4可知冗余度因子与数据可用性成正比,且冗余度越小,越能体现不同算法的数据可

用性性能,因为当网络规模一定时,随着冗余度的增加,高数据可用性算法提升空间较小,低数据可用性算法提升空间较大. 仿真环境设置攻击强度为 0.5,属于高动荡情况,但此时 RSboFMC 与全复制策略的数据可用性基本持平,尤其是在副本因子较低时,其数据可用性优于全复制策略. 除此之外,综合考虑数据可用性和存储开销,即 $valueOrder$=AVERAGE($s\times T_s+(1-s)\times f_{\text{Avail}}$),算法性能优越性排序为:

RSboFMC>allReplicaRandom>codedRandom>codedSequential

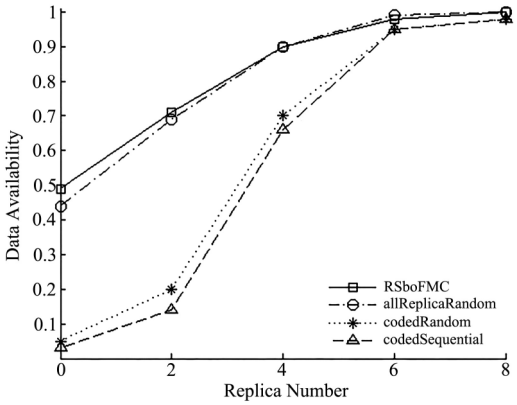

图 4 不同冗余因子下的数据可用性

Fig. 4 Data invulnerability with different replica factors

3.1.2 攻击强弱影响 不同攻击情况下造成的动荡强弱可以影响存储系统性能,而增大副本节点集合可以减弱这些影响. 因此固定副本数为 4,研究不同动荡(0.1,0.2,0.3,0.4,0.5,0.6,0.7)对不同算法的性能影响.

在相对稳定的网络环境中不同策略的数据可用性均可达到 70% 及以上,但随着失效节点数(动荡比)的增加,不同存储策略的数据可用性呈不同的递减趋势. 由图 5 可知,RSboFMC 在低动荡下有最优的数据可用性,在高动荡情况下其可用性接近于全复制策略,说明类似于编码策略的 RSboFMC 节省存储空间的同时,还具有类似于全复制高动荡下高数据可用性性能. 除此之外,在攻击强度大于 40% 后采用不同放置策略的编码策略出现骤降,而 allReplicaRandom 和 RSboFMC 较平稳下降,且其变化角度不超过 60°. 若考虑副本数或动荡率以等差数列增加对数据可用性的影响,由图 4 和图 5 可以得出数据可用性对副本数的变化更为敏感.

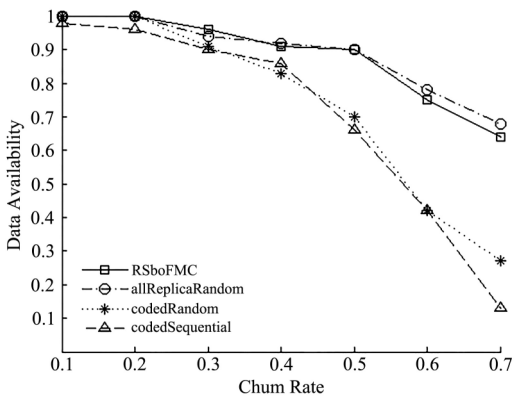

图 5 不同攻击强度下的数据可用性

Fig. 5 Data invulnerability with different churn rate

3.1.3 网络规模影响 在冗余度因子为 4,动荡率为 0.5 的情况下,扩大网络规模(100,200,…,1000,2000,3000,4000),观察数据可用性受影响变化的趋势(图 6).

对于固定的副本数和动荡率,由仿真结果可得,RSboFMC 的算法性能与网络规模无关,即 RSboFMC 算法不止对搭建环境时设定的网络规模有效,也对更大级别的网络规模有效,特别是当网络规模较大时,RSboFMC 算法能维持最高的数据可用性性能.

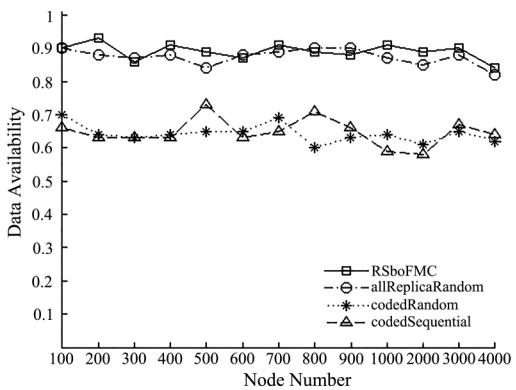

图 6 不同网络规模下的数据可用性

Fig. 6 Data invulnerability with different network size

3.2　负载均衡性能　为证明本文基于分区且考虑顺逆序和区域最大副本数的分发机制具有良好的负载均衡性,选取上述所说的放置策略组为对照组,将4300份文件在网络规模N=500的存储系统中进行分布式存储,结果如图7、图8和图9所示.直观上讲,由图7可知本文提出的分发机制使得节点间负载波动较平缓,由图8可知由于存储节点的随机选择导致节点间负载极大的不均衡性,顺序放置的数据分发机制可以看成一种特殊的随机放置数据分发机制,由图9可知其均衡性强于随机放置下的性能.除此之外,考虑到三种分发机制中平均负载相同,利用式(5)量化衡量各机制下的负载波动情况,RSboFMC相较于随机放置,将负载波动减弱58.58%;对于顺序放置则减弱了53.05%,即将负载波动较于传统分发机制至少减弱了50%以上.

$$C_{\text{balance Load}}=\sqrt{\frac{1}{N}\left[(c_{1f}-\overline{c_f})^2+(c_{2f}-\overline{c_f})^2+\cdots+(c_{Nf}-\overline{c_f})^2\right]} \tag{5}$$

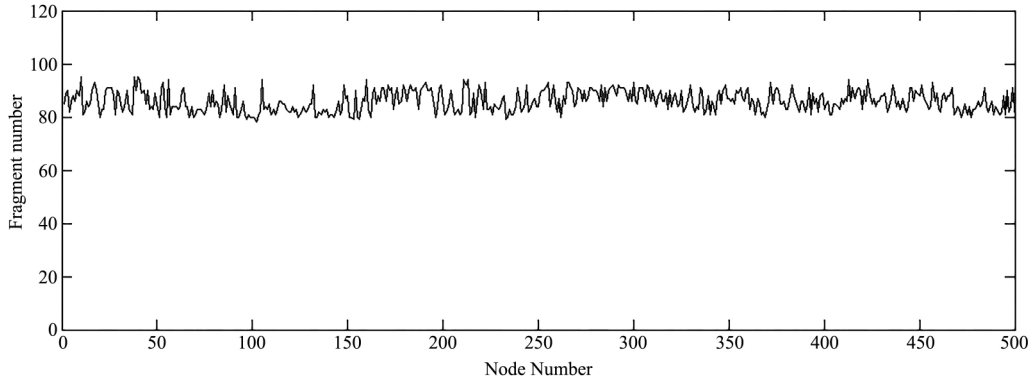

图7　基于分区且考虑顺逆序和区域最大副本数的分发机制下节点负载情况

Fig. 7　Nodes load of distributing mechanism based on partition, considering opposite sequences and maximum replicas in zone

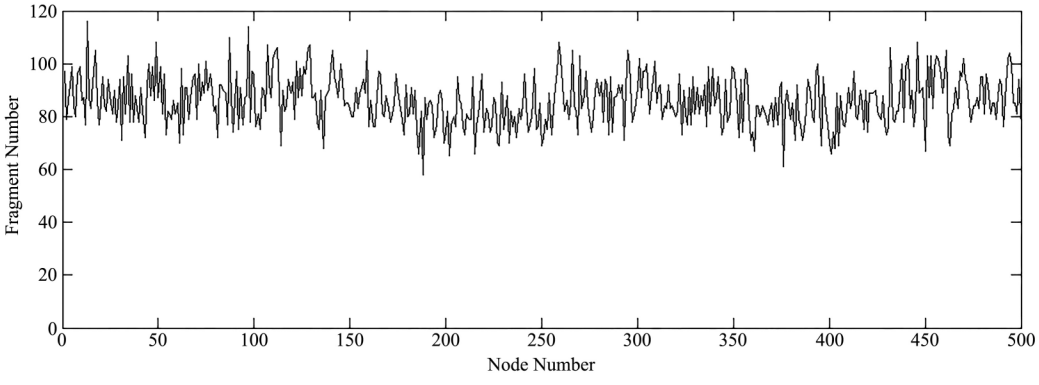

图8　基于随机放置的分发机制下节点负载分配情况

Fig. 8　Nodes load based on random distributing mechanism

其中,$C_{\text{balance Load}}$是负载波动值,N是网络规模,c_{if}是节点i上存储的文件碎片数量,即节点i的负载情况,$\overline{c_f}$是平均负载值.

为进一步说明本文提出的基于分区且考虑顺逆序和区域最大副本数分发机制的优越性,在图7、图8和图9的基础上,以节点理想负载

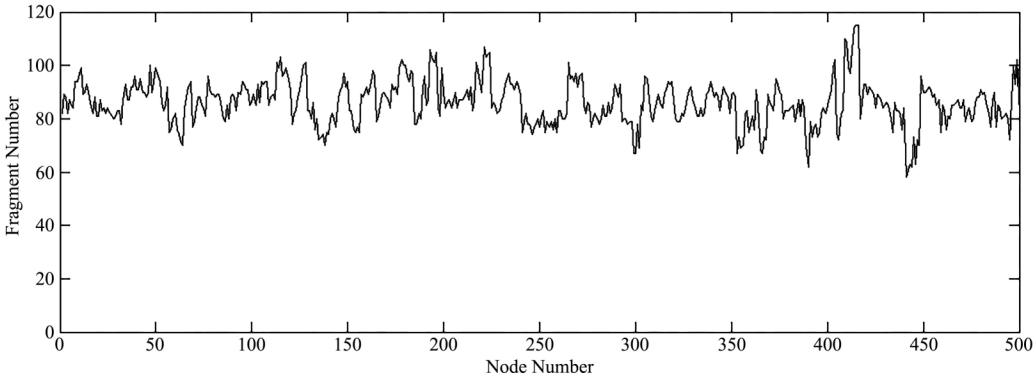

图 9 基于顺序放置的分发机制下节点负载分配情况

Fig. 9 Nodes load based on sequential distributing mechanism

为基准值,求出各策略中节点实际负载值和理想值的偏差概率分布曲线(图 10).

由图 10 可以得出,RSboFMC 策略节点实际负载和理想负载偏差值多集中在[0,|6|]之间(94.8%),少量在[|7|,|9|]间;而基于随机放置的分发机制节点实际负载和理想负载偏差值多集中在[|1|,|13|]之间(82.8%),部分在[|14|,|17|]和 0 上,少量在[|18|,|30|]上;对于基于顺序放置的分发机制,其节点实际负载和理想负载偏差值多集中在[|1|,|8|]和|11|上(74.6%),部分在[|9|,|10|],[|13|,14‖]和 0 上,少量在[|15|,|30|]和|12|上. 由上述分析可知,RSboFMC 策略具有最小的负载平均偏差值,即具有良好的负载均衡性性能.

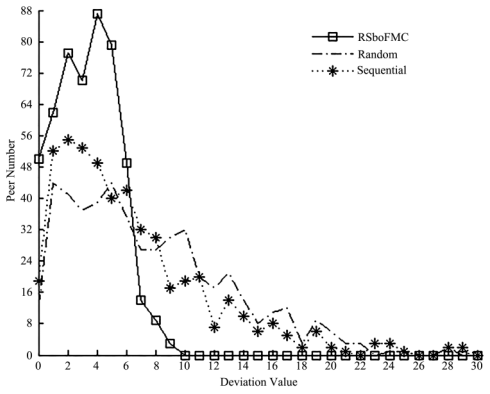

图 10 不同策略实际负载和理想负载偏差值分布图

Fig. 10 Deviation value between real load and ideal load with different strategies

4 结 论

数据存储网络存储策略的设计难点之一是动荡情况下数据的可用性性能和负载均衡性性能. 本文首先研究了提高网络性能的不同复制方法,然后基于优化复制策略和放置策略的思想提出一种新型存储策略:基于碎片矩阵和缓存的存储策略(RSboFMC). 仿真结果显示 RSboFMC 策略在数据可用性和负载均衡性方面优于其他策略. 从网络规模来看,RSboFMC 策略不受规模大小的影响,具有良好的可扩展性. 一个完整的存储过程主要由良好的拓扑结构、合适的副本节点集、合理的路由协议和完善的一致性维护策略构成,大多拓扑优化存储策略侧重于递归产生具有较小网络直径和高吞吐量的存储网络[17-18],而高吞吐量又依赖于节点的分配模式和传播模式[19-20],进而可推测其与数据放置策略有关,故未来研究中,考虑将基于拓扑优化的存储策略与基于数据放置优化的存储策略相结合,优化存储过程的初始状态,提高存储相关的初始性能.

参考文献

[1] Zhang X X, Xu F. Survey of research on big data storage//Proceedings of the 2013 12th International Symposium on Distributed Computing and Applications to Business, Engineering & Science.. Kingston upon Thames, United Kingdom: IEEE, 2013:76-80.

[2] Yoon J W, Hong T Y, Choi J W, et al. Evaluation of P2P and cloud computing as platform for exhaustive key search on block ciphers. Peer-to-Peer Networking and Applications, 2018, 11(6): 1206－1216.

[3] Karafiloski E, Mishev A. Blockchain solutions for big data challenges: a literature review//IEEE EUROCON 2017－17th International Conference on Smart Technologies. Ohrid, Macedonia: IEEE, 2017: 763－768.

[4] Chen K, Shen H Y, Sapra K, et al. A social network based reputation system for cooperative p2p file sharing. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(8), 2140－2153.

[5] Waluyo A B, Taniar D, Rahayu W, et al. Trustworthy data delivery in mobile p2p network. Journal of Computer and System Sciences, 2017, 86: 33－48.

[6] Park S H, Park J K. IoT industry and security technology trends. International Journal of Advanced Smart Convergence, 2016, 5(3): 27－31.

[7] Ghemawat S, Gobioff H, Leyng S T. The Google file system//Proceedings of the 19th ACM Symposium on Operating Systems Principles. Bolton Landing, NY, USA: ACM, 2003: 29－43.

[8] 艾潇. 面向数据中心的容错存储和数据访问优化问题的研究. 硕士学位论文. 哈尔滨:哈尔滨工业大学, 2015. (Ai X. Research on fault tolerant storage and data access optimization in data center networks. Master Dissertation. Harbin: Harbin Institute of Technology, 2015.)

[9] Khan O, Burns R, Plank J, et al. Rethinking erasure codes for cloud file systems: minimizing I/O for recovery and degraded reads//Proceedings of the 10th USENIX Conference on File & Storage Technologies. Berkeley, CA, USA: USENIX Association, 2012: 14－17.

[10] Qin Y, Ai X, Chen L J, et al. Data placement strategy in data center distributed storage systems//2016 IEEE International Conference on Communication Systems (ICCS). Shenzhen, China: IEEE, 2017: 1－6.

[11] Zhang L W, Deng Y H, Zhu W H, et al. Skewly replicating hot data to construct a powerefficient storage cluster. Journal of Network and Computer Applications, 2015, 50: 168－179.

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

[13] Inacio E C, Dantas M A R. A coarse-grained page cache aware multivariate analytical model for the storage performance of a parallel file system. Concurrency and Computation, Practice and Experience, 2018, 30(8): e4389.

[14] Lakshman A, Malik P. Cassandra: a decen-tralized structured storage system. ACM SIGOPS Operating Systems Review, 2010, 44(2): 35－40.

[15] Silva T, Kamienski C, Fernandes S, et al. A flexible DHT-based directory service for information management. Peer-to-Peer Networking and Applications, 2015, 8(3): 512－531.

[16] Sit E, Haeberlen A, Dabek F, et al. Proactive replication for data durability//Proceedings of IPTPS. Santa Barbara, CA, USA: IPTPS, 2006: 1－6.

[17] Azizi S, Hashemi N, Khonsari A. HHS: an efficient network topology for large-scale data centers. The Journal of Supercomputing, 2016, 72(3): 874－899.

[18] Li Z H, Guo Z Y, Yang Y Y. BCCC: an expandable network for data centers. IEEE/ACM Transactions on Networking, 2016, 24(6): 3740－3755.

[19] Leong D, Dimakis A G, Ho T. Distributed storage allocation problems//Proceedings of the 2009 Workshop on Network Coding, Theory, and Applications. Lausanne, Switzerland: IEEE, 2009: 86－91.

[20] Bhattacharya H, Chattopadhyay S, Chattopadhyay M, et al. A novel intelligent modeling of storage and bandwidth constraints in distributed storage allocation//Mandal J, Dutta P, Mukhopadhyay S, et al. Computational Intelligence, Communications and Business Analytics. Springer Singapore, 2017: 336－346.

(责任编辑　杨可盛)