

Ceph 云存储中基于混合文件系统的读写性能优化方法

刘辉勇¹, 王 勇^{1,2,3}, 俸 皓^{2,3}

(1 桂林电子科技大学 信息与通信学院, 广西 桂林 541004;

2 桂林电子科技大学 广西云计算与复杂系统高校重点实验室, 广西 桂林 541004;

3 桂林电子科技大学 广西云计算与大数据协同创新中心, 广西 桂林 541004)

摘 要: 针对 Ceph 云存储底层 OSD 节点使用不同单一文件系统会造成集群在读、写不同大小文件性能上的较大差异这一问题, 本文提出一种在 Ceph 集群中混合使用文件系统方法来优化集群读、写性能。通过采用层次分析法综合考虑不同文件系统在读、写不同大小文件时候各自的优势, 构造层次结构模型, 给出每个层次的判断矩阵并进行一致性检验, 通过求解判断矩阵最大特征值对应的归一化特征向量, 最后确定了底层 OSD 节点使用不同文件系统的比例。在真实环境中对所提出的方法进行实验测试, 结果表明所提出的方法能够较好的改善当前 Ceph 集群使用单一文件系统带来的不足, 优化了集群读、写不同大小文件速度性能。

关键词: Ceph; 云存储; 文件系统; 层次分析法

中图分类号: TP302

文献标识码: A

文章编号: 1000—7180(2018)05—0027—08

Optimization of Ceph Reads and Writes Performance Based on Hybrid File System

LUI Hui-yong¹, WANG Yong^{1,2,3}, FENG Hao^{1,2}

(1 School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China;

2 Guangxi Colleges and Universities Key Laboratory of Cloud Computing and Complex Systems, Guilin University of Electronic Technology, Guilin 541004, China; 3 Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: The cluster of Ceph cloud storage will have a greater performance differences in reading and writing different sizes of documents, when the underlying OSD nodes use different single file systems. To solve this problem, this paper proposes a hybrid file system approach in the Ceph cluster to optimize the performance of reading and writing. Through the use of analytic hierarchy process (AHP), this method considers the advantages of different file systems in reading and writing different size files, constructs a hierarchical model, gets the judgment matrix of each level and carries out the consistency test. Then, the normalized feature corresponding to the maximum eigenvalue of the judgment matrix is worked out to determine the proportion of the underlying OSD nodes using different file systems. By testing in the real environment, the result shows that this method can make up the shortcomings of the current Ceph cluster using a single file system, achieve the purpose of optimizing the speed performance in reading and writing different size of files.

Key words: Ceph; cloud storage; file system; analytic hierarchy process

收稿日期: 2017—08—15; 修回日期: 2017—09—07

基金项目: 国家自然科学基金(61662018, 61163058); 广西自然基金(2016GXNSFAA380153); 广西高校云计算与复杂系统重点实验室研究课题(14103, 15208); 广西云计算与大数据协同创新中心研究课题(YD16303, YD16512)

1 引言

云存储专注于解决云计算中海量数据的存储,是目前大部分数据中心所采用的存储形式.云存储系统的优良性能离不开分布式文件系统的支持.作为最近几年热门研究的分布式文件系统,Ceph 不仅具有高扩展性、高性能、高可靠性的特点,而且实现了集群真正意义上的无中心节点,此外它可以提供对象存储,文件系统存储和块存储^[1],因而受到了极大的关注.

Ceph 分布式文件系统由监控节点(Monitor):用于监控整个集群状,保证集群数据一致性;存储节点(OSD):用于集群中所有的数据与对象存储;元数据服务器(MDS):为 Ceph 文件系统提供元数据计算、缓存与同步.当前 Ceph 集群里面 OSD 节点采用的文件系统主要是 Xfs、Ext4 和 Btrfs 文件系统,然而 OSD 节点在选择不同文件系统时候对 Ceph 集群读、写不同大小文件性能会有较大影响.

为解决 Ceph 集群存在这一问题,本文提出一种 Ceph 集群底层 OSD 混合使用 Ext4 文件系统和 Btrfs 文件系统方法.

2 相关工作

Ceph 的底层核心组件是 RADOS^[2] (Reliable Autonomic Distributed Object Store).它使用 CRUSH^[3] (Controlled Replication Under Scalable Hashing,一种伪随机数据分布算法)算法维护存储对象与存储服务器的对应关系,实现了高性能、大规模性和可靠性的集群存储.

为了保证数据安全,文献[4]提出一种 Ceph 分布式块存储的持续数据保护(Continuous Data Protection,CDP)方法.通过监控每次块存储设备写操作,把要被写操作覆盖的数据拷贝到历史数据块设备,然后执行本次正常写操作,最后把该过程记录到日志块设备,保证了任意时间数据的安全.为了满足不同客户需求,文献[5]提出在一个 Ceph 集群创建异构存储池方法.文章主要思路是在底层节点是 Xfs 文件系统的所有 OSD 上创建一个存储池,在节点是 Btrfs 文件系统的所有 OSD 上创建另一个存储池,从而达到异构存储池目的.虽然使用了两种文件系统,但是这些具有不同文件系统的 OSD 是在不同存储池里面,这其实就是相当于两个集群环境.为了提高 Ceph 集群在读、写大文件性能,文献[6]提出一种基于多线程算法来优化集群性能.文献[7]提

出在一种面向异构存储的数据放置方法.通过利用 SSD 和 HDD 划分多种不同的副本组合,根据数据热度和读、写比例将不同的数据放到不同的副本组合上,在提升系统性能的同时有效地控制了系统容量成本.为了考虑实际集群时延问题,文献[8]通过分析集群中单节点在不同网络时延大小下与集群存储性能的关系,提出基于网络延时的 Ceph 存储性能优化方法,该方法降低了网络时延带来的影响,提高了存储性能.

上面研究虽然较好解决了 Ceph 集群中存在的不同问题,然而以上述为代表的对当前 Ceph 集群研究时,同一个集群里面底层 OSD 节点都是采用单一文件系统.因此本文研究在一个 Ceph 集群中混合使用文件系统.

3 问题描述

Ceph 引入了存储池(Pool)的概念,存储池是 Ceph 存储数据时的逻辑分区.一个存储池里面有一定数量的放置组(PG, Placement Group),放置组 PG 是 Ceph 里面数据存储、迁移和改变的最小单位,它分布在多个 OSD 中,并且可以根据集群的规模大小以及副本个数动态增减.

在 Ceph 集群中,数据存储要经历三次映射过程.首先客户端上传的文件(File)会被切割分成一定大小的对象(Objects),然后每个对象会被唯一映射到一个 PG 中,最后通过 CRUSH 算法把 PG 映射到多个 OSD 中,此过程如图 1 所示.

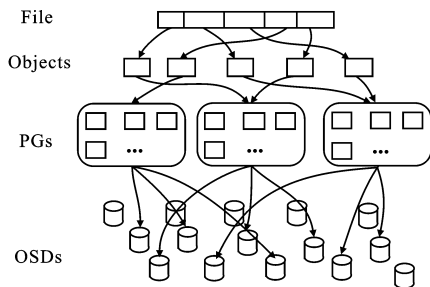


图 1 数据映射路径

为了避免数据的丢失,Ceph 通过副本模式或纠删码机制保证数据的可靠性.目前最成熟的是副本模式,默认情况下采用三种副本一致性方案.

然而在当前 Ceph 集群中底层 OSD 节点主要单独采用 Ext4 文件系统、Xfs 文件系统,或者 Btrfs 文件系统.当 OSD 节点采用不同文件系统时候,对整个集群读、写不同大小文件性能会有较大的影响.实验测试表明在相同的环境下,底层 OSD 节点单独

采用 Xfs 文件系统时集群具有较好的读性能;底层 OSD 节点单独采用 Btrfs 文件系统时集群具有较好的写性能. 这一点可以通过实验验证,具体测试描述如下:

测试环境是在相同的 Ceph 集群里面. 分别把 OSD 节点格式化为 Ext4、Xfs 和 Btrfs 文件系统,分别测试集群读、写 4 kB、64 kB、256 kB、4 096 kB 文件时候的速度大小,测试结果如表 1.

表 1 不同文件系统读、写不同大小文件性能对比						
MB/S File	Ext4		Xfs		Btrfs	
	Write	Read	Write	Read	Write	Read
4k	3. 81	30. 1	3. 1	18. 6	4. 6	21. 5
64k	11. 4	142. 2	10. 5	102. 3	13. 4	121. 6
256k	12. 9	176. 2	11. 5	168. 4	14. 2	194. 8
4096k	14. 6	233. 5	12. 6	209. 5	15. 7	288. 6

由上面的测试结果可以看出,底层 OSD 节点选择不同文件系统,带来的 Ceph 集群在读、写不同大小文件时性能上的较大差异.

4 设计方法

本文提出集群中 OSD 节点混合采用 Ext4 和 Btrfs 文件系统时,由于 Ceph 集群具有高扩展性,因此在一个集群中最主要的是确定 OSD 节点采用这两种文件系统的比例. 为具体实现采用混合文件系统优化 Ceph 存储读、写性能,本节在算法描述、算法流程方面进行展开.

4. 1 算法描述

确定 OSD 节点使用混合文件系统各自比例是一个多属性决策问题. 本文采用层次分析法来做出决策. 层次分析法(AHP)是由美国运筹学家萨蒂(T. L. Saaty)提出的一种解决多目标的复杂问题决策分析方法. 该方法把与决策有关的因素分解成目标层、准则层、方案层等层次并合理地给出每个决策方案针对每个目标的权重因子,利用权重因子求出各方案的优劣次序. 因此,本文通过定性描述各个指标的重要性,利用数学方法得到每个指标的权值,实现从定性到定量的转化. 本文使用层次分析法主要包括:建立层次结构模型;确定判断矩阵中每个权重因子;层次排序及一致性检验.

本文采用的层次分析法模型如图 2,在这个模型中,方案层(Alternative)是选择备选的 Ext4 和 Btrfs 两种文件系统. 由于 T. L. Saaty 指出在层次分析法中,一般每层比较的因素最好不超过 9 个,因

此在准则层(Criteria)里面分别选择在两种不同文件系统下集群读、写 4 kB、64 kB、256 kB、4 096 kB 文件速度作为参考指标. 目标层(Goal)是选择最终的文件系统. 最终目的就是在集群中混合使用两种文件系统,通过使用层次分析法,综合考虑各个因素影响,确定集群中 OSD 节点使用 Ext4 文件系统和 Btrfs 文件系统的比例,达到集群读、写不同大小文件性能最佳.

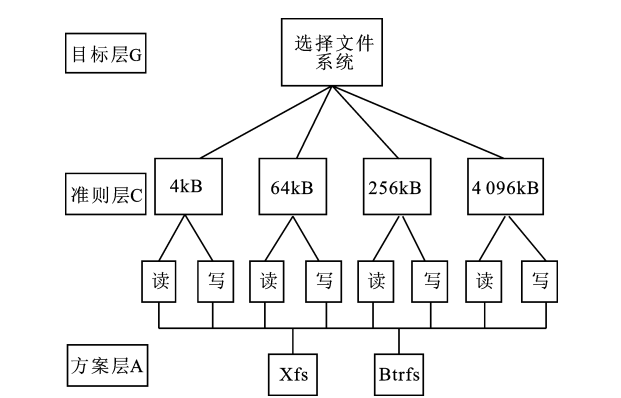


图 2 层次分析法模型

判断矩阵是表示本层里面所有因素针对上一层某一个因素的相对重要性的比较. 在构造判断矩阵前,首先要确定本层每个因素对上一层的影响程度. 如果直接给各个因素分配权重会比较困难,层次分析法做的就是两两因素之间相互比较. T. L. Saaty 提出采用 1 到 9 来表示每两个因素之间的相对重要程度.

判断矩阵的元素 a_{ij} 表示第 i 个因素相对于第 j 个因素的比较结果. 具体如下:1 表示两个因素相比,两者具有同样重要性;3 表示两个因素相比,一个因素比另一个因素稍微重要;以此类推,9 表示两个因素相比,一个因素比另一个因素极端重要. 然后 2,4,6,8 表示上面相邻判断的中间值. 因此很容易发现在判断矩阵中,对角线上都是 1. 接下来简要阐述判断矩阵中注意的两个关键因素.

定义 1:若矩阵 $A = (a_{ij})_{n \times n}$ 满足

$$a_{ij} > 0, a_{ji} = \frac{1}{a_{ij}} (i, j = 1, 2, \dots, n)$$

那么称之为正互反矩阵.

定义 2:若矩阵 $A = (a_{ij})_{n \times n}$ 满足

$$a_{ij} a_{jk} = a_{ik}, \forall i, j, k = 1, 2, \dots, n$$

那么称此正互反矩阵是一致矩阵.

在确定两个备选方案对上面准则层某一因素影响权重时候,为了使这里面我们构造的判断矩阵中各层次之间权重具有更细力度,由层次分析法设计

思路并结合表 1 测试结果,定义下面规则:

规则 1:以 Ext4 文件系统测试结果为基准,如果测试的结果基本相同就均设置为 1,如果测试的结果大于另一个结果不足 15%就设置为 2 和 1,接近 15%时候,就分别设置为 3 和 1,大于 15%但是不足 30%,就设置为 4 和 1,接近 30%就分别设置为 5 和 1,的以此类推.

一般权重值的设置为经验性设置,并不影响实际中的使用. 这里举例说明,比如在集群 OSD 单独采用 Ext4 文件系统时候,读 4 kB 文件时速度是 30.12 MB/S;在集群 OSD 单独采用 Btrfs 文件系统时候,读 4 kB 文件时速度是 21.52 MB/S,因为 30.12 是 21.52 的 1.39 倍,因此在确定集群对读 4 kB 文件影响权重时设定 Ext4 文件系统影响权重为 6,设定 Btrfs 文件系统影响权重为 1.

在确定准则层 8 个因素对目标层权重影响时,由 2007 年美国西北太平洋国家实验室布一份研究报告,该报告称在实验室文件系统中存放有 1 200 万个文件,其中低于 64 kB 的文件占 58%^[9]. 因此低于 64 kB 的文件相对来说比例就要高,又因为客户端写入的数据大小具有随机性,所以定义以下规则:

规则 2:在确定准则层对目标层构造矩阵时候,设定 4 kB 和 64 kB 文件权重为 3,大于 64 kB 的文件权重值设定为 1.

在构造判断矩阵时候,实际上矩阵不一定满足上面定义 2 所描述的一致性矩阵,这里给出不一致性的允许范围,具体定义如下:

一致性指标: $CI = \frac{\lambda - n}{n - 1}$ 其中 λ 是构造判断矩阵的最大特征值, n 是构造的判断矩阵的维数. 当 $CI = 0$,有完全一致性;当 CI 接近于 0,有满意一致性;当 CI 越大,不一致越严重. 为衡量 CI 的大小,引入随机一致性指标 RI ,这里直接给出 RI 取值如表 2 所示.

表 2 随机一致性指标 RI								
n	1	2	3	4	5	6	7	8
RI	0	0	0.5	0.9	1.1	1.2	1.3	1.4

给出一致性比率: $CR = \frac{CI}{RI}$, 当一致性比率 $CR < 0.1$ 时候,通过一致性检验,可用其最大特征值 λ 对应的归一化特征向量作为权向量,权向量就是最终求得的比例,否则要重新构造判断矩阵.

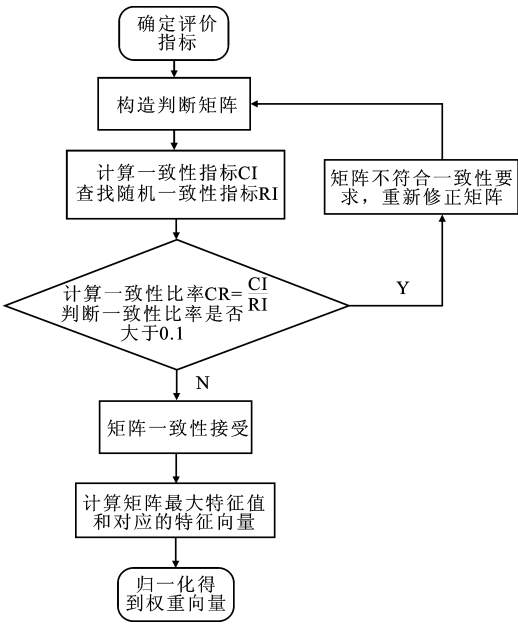


图 3 算法流程图

4.2 算法流程

基于上面所采用层次分析法来确定集群 OSD 节点混合采用不同文件系统的方法,下面具体给出设计的方法步骤:

步骤 1:根据确定的评价指标,构造出准则层对目标层的判断矩阵 C ;

步骤 2:计算判断矩阵 C 最大特征值 λ ,根据判断矩阵维数,查找随机一致性指标 RI ,并计算一致性比率 CR ,判断一致性比率 CR ,如果大于 0.1,则一致性不满足,需要转步骤 1 重新构造判断矩阵;否则转步骤 3;

步骤 3:计算最大特征值 λ 对应的归一化特征向量 $W^{(2)}$;

步骤 4:构造方案层对准则层的 8 个判断矩阵 A_1 到 A_8 ,分别计算 A_1 到 A_8 判断矩阵各自最大特征值,根据判断矩阵维数,查找随机一致性指标 RI ,并计算各自的一致性比率 CR_i ,其中 $i = 1, 2, \dots, 8$.

步骤 5:判断一致性比率 CR_i ,如果大于 0.1,则一致性不满足,需要转步骤 4 重新构造判断矩阵;否则转步骤 6;

步骤 6:计算方案层对准则层的 8 个判断矩阵各自最大特征值对应的的归一化特征向量 $W^{(3)}$;

步骤 7:由 $W^{(3)}$ 和 $W^{(2)}$ 计算结果,就确定了集群 OSD 节点采用 Btrfs 文件系统和 Ext4 文件系统的比例.

5 实验与分析

为了验证本文所提出的在 Ceph 集群中混合使用 2 种不同文件系统来优化读、写性能方法,本文搭建真实环境为:Ceph 集群有 7 台虚拟机,每台虚拟机 2G 内存,CPU 为 4 核;1 台监控节点,2 台客户端主机,4 台存储主机,每个主机上有 2 个存储节点 OSD,每个 OSD 容量大小 40G;,. 软件环境:Ceph 版本号 0.94.9,操作系统 Ubuntu 14.04.1 LTS. 实验环境结构如图 4.

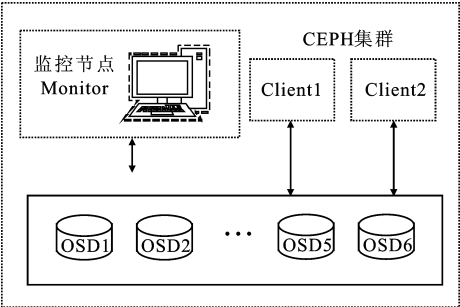


图 4 实验环境结构

首先需要确定 OSD 节点使用两种文件系统的比例. 由上面 4.1 节定义的规则 2 分析可以构造本实验使用的准则层各因素对目标层的判断矩阵如下 C:

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 1/3 & 1/3 & 1/3 & 1/3 & 1 & 1 & 1 & 1 \\ 1/3 & 1/3 & 1/3 & 1/3 & 1 & 1 & 1 & 1 \\ 1/3 & 1/3 & 1/3 & 1/3 & 1 & 1 & 1 & 1 \\ 1/3 & 1/3 & 1/3 & 1/3 & 1 & 1 & 1 & 1 \end{bmatrix}$$

通过计算可以知道该矩阵最大特征值为 8,其对应的归一化特征向量:

$$W^{(2)} = (0.19, 0.19, 0.19, 0.19, 0.06, 0.06, 0.06, 0.06)^T$$

通过分析可以知道这个判断矩阵的一致性指标 $CI = 0$,一致性比率 $CR = 0$,可知这一判断矩阵通过一致性检验.

其次,在确定两个方案层对准则层中各个因素的影响权重时候,由上面第 3 节测得的数据和 4.1 节定义的规则 1,构造的 8 个判断矩阵 A_1 到 A_8 如下:

$$A_1 = \begin{bmatrix} 1 & 6 \\ 1/6 & 1 \end{bmatrix} \quad A_2 = \begin{bmatrix} 1 & 1/4 \\ 4 & 1 \end{bmatrix}$$

$$\begin{array}{ll} 4 \text{ k 读} & 4 \text{ k 写} \\ A_3 = \begin{bmatrix} 1 & 3 \\ 1/3 & 1 \end{bmatrix} & A_4 = \begin{bmatrix} 1 & 1/3 \\ 3 & 1 \end{bmatrix} \\ 64 \text{ k 读} & 64 \text{ k 写} \\ A_5 = \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix} & A_6 = \begin{bmatrix} 1 & 1/2 \\ 2 & 1 \end{bmatrix} \\ 256 \text{ k 读} & 256 \text{ k 写} \\ A_7 = \begin{bmatrix} 1 & 1/4 \\ 4 & 1 \end{bmatrix} & A_8 = \begin{bmatrix} 1 & 1/2 \\ 2 & 1 \end{bmatrix} \\ 4 \text{ 096 k 读} & 4 \text{ 096 k 写} \end{array}$$

这 8 个判断矩阵它们对应的最大特征值如下:

特征值	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
$\lambda_{\max}$	2	2	2	2	2	2	2	2

最大特征值对应的归一化特征向量如下:

$$W^{(3)} = \begin{pmatrix} 0.86 & 0.20 & 0.75 & 0.25 & 0.25 & 0.33 & 0.20 & 0.33 \\ 0.14 & 0.80 & 0.25 & 0.75 & 0.75 & 0.67 & 0.80 & 0.67 \end{pmatrix}$$

可以分析这 8 个判断矩阵均通过一致性检验. 最后计算决策结果如下:

$$W = W^{(3)} W^{(2)} = (0.45, 0.55)^T$$

计算出决策结果就是一个集群里面 OSD 节点设置为 Btrfs 文件系统比例为 0.55,设置为 Ext4 文件系统性比例为 0.45,集群在读、写不同大小文件时候能性能最佳. 在实际情况下选择这两种文件系统的 OSD 个数取整,比如本实验中 OSD 节点为 8,那么设置为 Btrfs 文件系统的 OSD 个为 $8 \times 0.55 = 4.4$ 取整后设置为 4,设置为 Ext4 文件系统个数也就是 4. 在实际集群搭建部署中,由于 Ceph 集群具有高扩展性,那么在新增 OSD 节点格式为这两个文件系统的比例时服从本文方法计算结果即可.

在本文实验测试时,OSD 节点分别选择 Ext4 文件系统、Btrfs 文件系统和本文混合两种文件系统. 由上面分析得出的结论,本文实验情况下选择混合使用文件系统时两种文件系统的 OSD 个数均为 4 个.

首先在搭建好的集群里面创建一个池 Pool,因为有 8 个 OSD 节点,设置 pg 大小 256,然后在 Pool 池里面创建 2 个虚拟块 Rbd 大小均为 30G,最后通过虚拟机挂载 Ceph 集群提供的块存储设备并对块存储设备进行不同大小文件读、写来测试存储集群的性能.

在使用 Fio 测试集群的读、写性能时候,有几个重要参数:一、libaio 表示异步 I/O 操作;二、direct

=1 表示测试过程绕过机器自带的 buffer,使测试结果更加真实;三、iodepth 表示队列深度,它指的是一次给队列多少个 I/O 请求. 为了提高集群的读、写性能,需要提高 iodepth. 由于测试时候队列深度 iodepth 的偏小会导致集群的读、写性能没有达到满负载,因此首先要确认队列深度这个参数的大小. 下面测试在不同队列深度情况下,集群 OSD 为 Ext4 文件系统时候,对 4 k 大小文件的读、随机读、写和随机写的性能,测试结果如图 5.

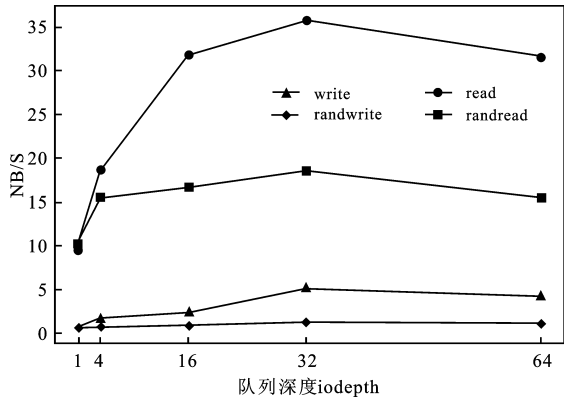


图 5 不同 iodepth 下集群读、写性能

从图 5 中可以看出,随着队列深度 iodepth 的增加,集群的读、写性能都会开始增加. 尤其是 4 k 文件读性能增加明显. 当队列深度增加到 32 左右,集群的读、写性能基本达到最大值. 因此这就确定了在该实验环境下,队列深度 iodepth 大小为 32.

在上面确定 iodepth 大小后,使用 Fio 测试工具生成顺序读、顺序写请求来对实验环境中集群进行测试. 读写文件大小分别为 1 k、4 k、16 k、32 k、64 k、128 k、256 k、512 k、1 024 k、4 096 k.

图 6 表示集群底层 OSD 节点文件系统选择不同策略时候,写不同大小文件时 IOPS 性能对比. 图 7 表示集群底层 OSD 节点文件系统选择不同策略时候,读不同大小文件 IOPS 性能对比.

从图 6 中可以看出随着写入的文件逐渐增大,集群写入 IOPS 性能快速下降. 这是因为文件越大,写入单个文件消耗时间就越长,因此 IOPS 就会下降. 可以看出当集群里面节点单独使用 Btrfs 文件系统时候与单独使用 Ext4 文件系统时候相比,集群在写入不同大小文件时候都有较好的 IOPS 性能,最高能够相差 30%左右;本文提出的方法在写不同大小文件时候,虽然整体性能没有比单独使用 Btrfs 文件系统带来的性能好,但是整体上比单独使用 Ext4 文件系统带来的性能均有提高,最大可以提高

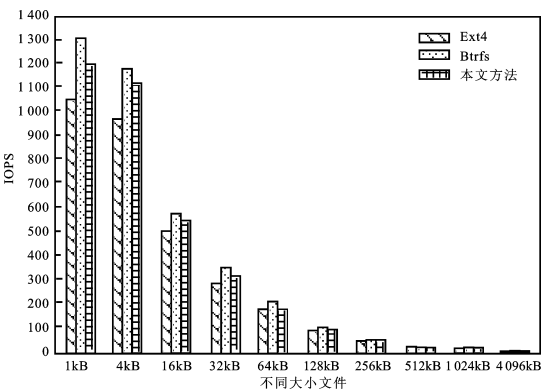


图 6 写不同大小文件 IOPS 性能对比

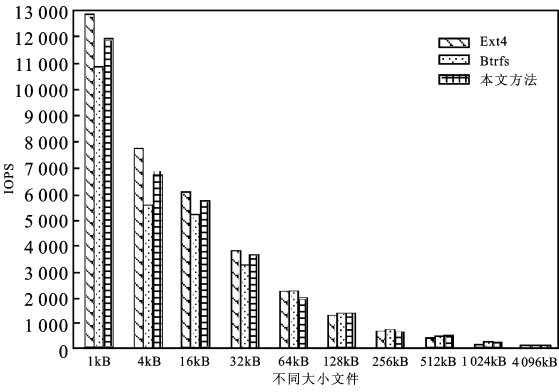


图 7 读不同大小文件 IOPS 性能对比

20%左右. 因此本文提出的方法就很好的弥补了 OSD 单独使用 Ext4 文件系统带来写不同大小文件时 IOPS 性能不足.

从图 7 中可以看出在读不同大小文件时候,当文件大小在 1 k 到 64 k 左右,OSD 节点分别使用 Ext4 文件系统、本文混合文件系统和 Btrfs 文件系统,集群在读文件时 IOPS 性能依次下降. 当文件大小在 128 k 时候,三种情况下带来的性能效果几乎没什么差别,而当文件大于 128 k 时候,OSD 节点分别使用 Ext4 文件系统、本文混合文件系统和 Btrfs 文件系统,集群在读文件时 IOPS 性能依次上升. 由此可以看出 Ext4 文件系统可以提高集群在读小文件的 IOPS 性能,当文件大于 128 k 时候,Btrfs 文件系统带来的读文件性能超过了单独使用 Ext4 文件系统. 本文提出的方法可以解决在读不同大小文件时候,由于单独使用 Ext4 文件系统或者 Btrfs 文件带来的性能不足问题.

图 8 和图 9 分别表示集群底层 OSD 节点文件系统选择不同策略时候,写入不同大小文件速度性能比较和读取不同大小文件速度性能比较.

从图 8 可以看出在写入不同大小文件时候,

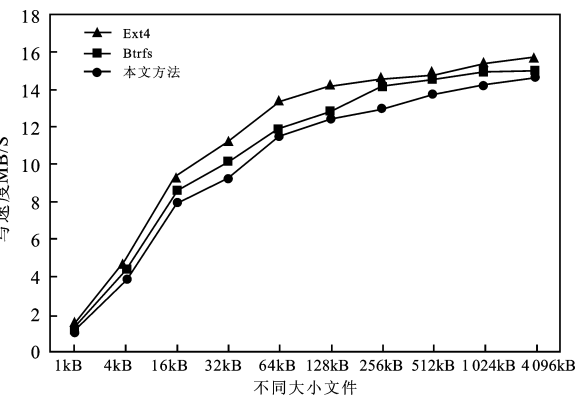


图8 写不同大小文件性能对比

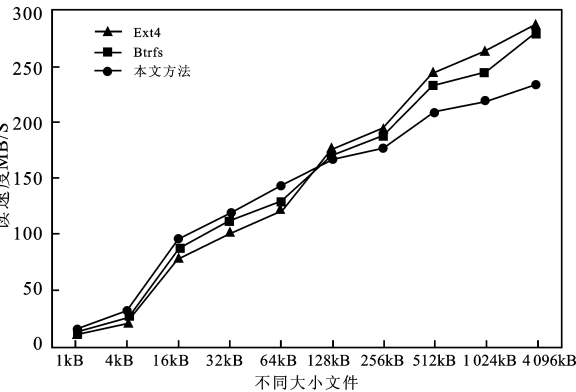


图9 读不同大小文件性能对比

OSD节点单独使用Btrfs文件系统或Ext4文件系统时候,带来的性能差异比较大.整体来说单独使用Btrfs文件系统带来的写性能更加有优势.当文件大小超过16k时候,单独使用Btrfs文件系统比单独使用Ext4文件系统带来的性能整体上高了10%以上.当文件大小是32k时候,两者带来的差异更加明显,单独使用Btrfs文件系统比单独使用Ext4文件系统带来的性能整体提高了22.5%.本文提出的方法虽然没有达到单独使用Btrfs文件系统所带来的性能,但是在文件大于256k时候带来的性能也已经接近单独使用Btrfs文件系统带来的性能,并且整体比单独使用Ext4文件系统带来的写性能提高5%~10%.

从图9可以看出在读取不同大小文件时,当文件小于128k时候,单独使用Ext4文件系统带来的读性能更加有优势,采用混合文件系统带来的性能次之,单独使用Btrfs文件系统性能最差.当文件大于128k时候,采用Btrfs文件系统读性能最佳,并且比单独使用Ext4文件系统带来的性能要高20%以上.采用混合文件系统的性能虽然没有达到单独使用Btrfs文件系统带来的性能,但是也非常接近.

由上面的实验结果显示,Ceph集群在写入不同大小文件时候,底层OSD节点单独使用Btrfs文件系统比单独使用Ext4文件系统带来的性能要好,最高可以相差22.5%.但是在读取不同大小文件时候,当文件小于128k时候底层OSD节点单独使用Ext4文件系统比单独使用Btrfs文件系统带来的性能要好,当文件大于256k时候,情况相反,底层OSD节点单独使用Btrfs文件系统比单独使用Ext4文件系统带来的性能要好.本文提出的在一个Ceph集群中混合使用两种不同文件系统方法,在读写不同大小文件时候均在可以比较好的解决集群采用单一文件系统带来的性能不足.

6 结束语

本文在Ceph集群平台下,针对底层OSD选择不同文件系统给集群带来读、写不同大小文件性能上的差异,提出了一种优化Ceph存储读写性能方法.通过混合使用Ext4文件系统和Btrfs文件系统,并采用层次分析法综合考虑两种文件系统在读、写不同大小文件上各自的优势,确定了Ceph集群底层OSD节点使用这两种文件系统的比例.在实际条件下实验表明本文提出的方法能够较好的弥补目前Ceph集群使用单一文件系统带来的不足,优化了集群在读、写不同大小文件上速度性能.

参考文献:

[1] Red Hat. Home Ceph[EB/OL][2017-06-30]. <http://docs.ceph.org.cn/start/intro/>.

[2] Weil S A, Leung A W, Brandt S A, et al. Rados: a scalable, reliable storage service for petabyte-scale storage clusters[C]//Proceedings of the 2nd international workshop on Petascale data storage; held in conjunction with Supercomputing. New York, ACM, 2007: 35-44.

[3] Weil S A, Brandt S A, Miller E L, et al. CRUSH: Controlled, scalable, decentralized placement of replicated data [C]//Proceedings of the 2006 ACM/IEEE conference on Supercomputing. Hongkong, ACM, 2006: 122.

[4] 王胜杰,徐龙. 一种Ceph分布式块存储的持续数据保护方法[J]. 网络安全技术与应用,2017,16(2):84-85.

[5] Meyer S, Morrison J P. Supporting heterogeneous pools in a single ceph storage cluster[C]//17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing. Timisoara, Romania, 2016:352-359.

- [6] Ke Z, Ai H P. Optimization Of Ceph reads/writes based on multi-threaded algorithms[C]//IEEE International Conference on High Performance Computing and Communications; IEEE International Conference on Smart City; IEEE International Conference on Data Science & Systems, New York, 2016:719-725.
- [7] 刘飞,蒋德钧,张欢,等. 异构存储感知的 Ceph 存储系统数据放置方法[J]. 计算机科学, 2017, 44(6):17-22.
- [8] 陈凌剑,王勇,俸皓. 基于网络延时的 Ceph 存储性能优化方法[J]. 微电子学与计算机, 2017, 34(6):84-88.
- [9] Mackey G, Sehrish S, Wang J. Improving metadata management for small files in HDFS[C]//IEEE Inter-

national Conference on Cluster Computing and Workshops. New Orleans, LA, USA, 2009:1-4.

作者简介:

刘辉勇 男, (1992-), 硕士研究生. 研究方向为云存储.

E-mail:1126610758@qq.com.

王勇 男, (1964-), 博士, 教授. 研究方向为云计算、计算机网络技术及应用、信息安全等.

俸皓 男, (1978-), 博士研究生. 研究方向为计算机网络、无线传感器网络、软件定义网络及云计算等.

(上接第 26 页)

法;(2)为存储节点设置冗余块的最佳方法;(3)确定最低数据冗余以达到所需服务质量的机制. 由于经济收入是云存储服务的一个关键因素, 下一步工作中, 我们将对工作进行扩展, 以便在今后的工作中进行考虑.

参考文献:

- [1] Chen Minghung, Tung Yuchih, Hung Shihhao, et al. Availability is not enough: minimizing joint response time in peer-assisted cloud storage systems[J]. IEEE Systems Journal, 2016, 10(4): 1424-1434.
- [2] Yu Jia, Wang Huaqun. Strong key-exposure resilient auditing for secure cloud storage[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(8): 1931-1940.
- [3] 刘伯红, 史慧丽. 一种高效属性撤销的云存储访问控制方案[J]. 微电子学与计算机, 2016, 33(4): 41-45.
- [4] Yan Hao, Li Jiguo, Han Jinguang, et al. A novel efficient remote data possession checking protocol in cloud storage[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(1): 78-88.
- [5] Yu Jia, Ren Kui, Wang Cong. Enabling cloud storage auditing with verifiable outsourcing of key updates[J]. IEEE Transactions on Information Forensics and Secu-

ity, 2016, 11(6): 1362-1375.

- [6] 刘绍刚. 基于 NAS 的私有云存储平台的设计与实现[J]. 计算机测量与控制, 2017, 25(4): 205-208.
- [7] 谭振华, 杨广明, 王兴伟, 等. 面向云存储的多维球面门限秘密共享方案[J]. 软件学报, 2016, 27(11): 2912-2928.
- [8] Abdul Nasir Khan, Mat Kiah M L, Mazhar Ali, et al. BSS: block - based sharing scheme for secure data storage services in mobile cloud environment[J]. The Journal of Supercomputing, 2014, 70(2): 946-976.
- [9] Rashmi M Jogdand, Goudar R H, Gazal Begum Sayed, et al. Enabling public verifiability and availability for secure data storage in cloud computing[J]. Evolving Systems, 2015, 6(1): 55-65.
- [10] Pitchai R, Jayashri S, Raja J. Searchable encrypted data file sharing method using public cloud service for secure storage in cloud computing[J]. Wireless Personal Communications, 2016, 90(2): 947-960.

作者简介:

胡力文 女, (1983-), 硕士研究生, 讲师. 研究方向为计算机软件技术理论. E-mail:61661605@qq.com.

李启明 男, (1982-), 博士, 副教授. 研究方向为大数据分析、高性能计算.