基于对象存储系统中属性管理的研究与实现*

王慧丽, 冯 丹, 覃灵军
(华中科技大学 计算机科学与技术学院 外存储教育部重点实验室, 武汉 430074)

摘 要: 针对现有属性管理方法上的缺陷和不足,提出了一种新的属性管理方法——哈希桶。哈希桶方法对对象的属性进行集中管理,不仅降低了管理存储成本,更有效地提高了系统的吞吐率。经过仿真测试表明,哈希桶对象属性管理方法性能远优于现有的属性管理方法。

关键词: 基于对象存储系统; 对象属性; 哈希桶
中图分类号: TP311 **文献标志码**: A **文章编号**: 1001-3695(2007)11-0188-03

Research and realization on attribution management of object-based storage system

WANG Hui-li, FENG Dan, QIN Ling-jun
(National Key Laboratory of Data Storage System, College of Computer Science & Technology, Huazhong University of Science & Technology, Wuhan 430074, China)

Abstract: The method of old attribute management, which stored the attributes at a fixed format, limited the extensibility of object and caused the management of complex Hash bucket, the new method of attribute management was proposed to solve the flaws and shortcomings of the old one. Hash bucket, which managed the object's attributes concentrated, reduced the storage costs on management and also improved the systems' throughout greatly. It was indicated clearly by simulation tests that hash bucket is significantly better than the old method on attribute management.

Key words: object-based storage system (OBSS); object attribute; hash bucket

引言

随着计算机技术的快速普及,人们对数据的需求呈指数级增长,存储系统已经成为制约计算机进一步发展的瓶颈。传统的存储系统,如 NAS和 SAN都难以提供完美的存储解决方案: NAS基于文件,文件级别的接口提供了安全性和跨平台的互操作性; SAN基于块,块级别接口在快速访问、高性能方面有优势。此时,基于对象的存储系统[1]在众多存储方案中脱颖而出,它在性能、可扩展、数据共享以及容错、容灾等方面有杰出表现,为人们提供一个完美的解决方案。对象存储系统的概念已经被工业界广泛认可,并由多家公司联合,由美国国家标准组织 (ANSI)下属的 T10工作组制定标准——OSD命令集 (object-based storage device commands)[2]; IBM、Panasas等公司已经推出了相关产品。目前影响较大的对象存储系统有集群文件系统公司开发的 Lustre文件系统[3]和 Panasas公司开发的 Panasas ActiveScale storage cluster[4]等。

基于对象存储系统之所以取得这么大的成功,最根本的原因在于它推出了一个全新的存储接口,即对象 (object)。对象是一些具有逻辑关系的数据的载体,它与块的固定大小不一样。对象是可变长的,可包含任何类型的数据,如文件、数据库记录、图像以及多媒体视频、音频等。至于包含何种类型数据由应用决定,对象可动态地扩大和缩小。对象分为用户对象、分区对象、集合对象和根对象四种。其中:用户对象是对象存储设备中数量最多的,它存放各种对象数据及其属性数据;把用户对象进行分区管理,构成分区对象;集合对象是一些具有相同或相近特性的用户对象或者分区对象的集合;根对象及其属性用于描述对象存储设备的一些特征,一个对象存储设备只有一个根对象。对象由一个 128 bit的标志符惟一表示。该 128 bit的标志符为 64 bit的 partition_ID和 64 bit的 user_Object_ID的组合。

本文根据当前存储系统应用的一些特点,结合对象存储设备自身的优点,提出了一种对象属性管理的新方法——哈希桶。通过哈希桶对对象属性进行集中式的管理,大幅度地提高了系统的性能。

对象存储系统和对象存储设备

对象存储系统的体系结构如图 1所示。高速网络将用户、元数据服务器 (metadata server, MDS)和对象存储设备 (OSD)[5]连接起来。OBSS实现一个基于对象的分布式网络文件系统。MDS提供全局名字空间,管理文件到对象的映射,

收稿日期: 2006-08-24; **修返日期**: 2006-11-20 **基金项目**: 国家"973"计划资助项目 (2004CB318201)

作者简介:王慧丽 (1982-),女,浙江人,硕士研究生,主要研究方向为网络存储系统研究 (jessvs@gmail.com);冯丹 (1970-),女,湖北京山人,教授,博导,主要研究方向为网络存储系统、计算机高速接口通道;覃灵军 (1975-),男,广西柳州人,博士研究生,主要研究方向为对象存储系统.

提供身份验证等安全机制。OSD则向外提供对象接口,以对象作为存取单元。用户访问文件时先向元数据服务器发送请求,获取文件的信息(如文件由哪些对象组成以及对象所在的设备等)及访问证书,然后客户与OSD直接交互。OSD收到客户请求后,对其身份进行认证,然后执行客户的对象读写请求。在这里客户向OSD发送的I/O请求与基于块的I/O请求不同,仅包括对象ID、对象的偏移地址以及长度。

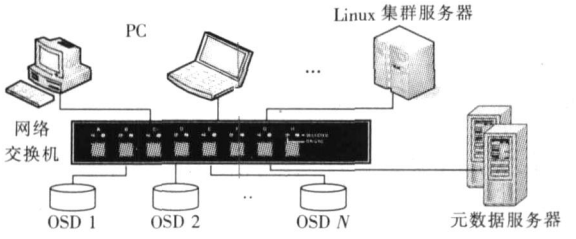

图1　对象存储系统体系结构

OSD是一个智能化的设备,包括CPU、memory、网络接口以及块设备接口,管理对象存储空间的分配、数据组织以及对象的属性。对象存储系统的最大好处之一就是将底层的数据组织和同步操作交由OSD管理,这就大大减轻了用户端和元数据服务器的负担,同时也提高了整个系统的并行性和可扩展性。

对象属性

对象属性是对象特点的描述或历史行为的记录,充分利用对象的属性特征能有效地提高基于对象存储系统的整体性能。对象可以表示任何类型的应用数据。对于不同的应用,数据的属性是不同的。如果为所有的对象定义一种统一的属性表示方式,这种表示方式会占用大量的存储空间。为简化和扩展对象属性的表示,本文采用一种属性页(attribute page)的表示方法,每个属性页上记录相关或相近的属性。每个属性页用属性页号(attri_Page_ID)来表示;属性页中具体的一个属性用属性索引号(attri_Index_ID)来索引。因此,对象的一个属性以(partition_ID, user_ID, attri_Page_ID, attri_Index_ID)惟一表示。

对于每个对象都具有的公共属性则定义为公共属性页(其ID号为0),如每个对象都具有创建时间、最近一次修改时间、对象大小等属性。每个对象至少有一张属性卡片,那就是公共属性卡片,由系统定义、系统方法和用户自定义方法均可解释存取。对于与应用有关的属性,则可定义为用户属性页,如多媒体对象与数据库表格对象具有完全不同的特点:多媒体对象有QoS[6]、帧速率、视频图像大小等属性,可定义为多媒体属性卡片;数据库表格对象的属性可能有记录的条数、每条记录的格式等内容,可定义为数据库表格属性卡片。不同的对象使用了不同的自定义的属性卡片。

属性页的使用为表示新对象类型的相应特征提供了方便,只要根据新对象的特点定义针对该对象特点的属性卡片,就可对属性进行扩展,从而表示新对象的特征。如为每个对象定义一个预取属性页,对用户访问对象存储设备的规律进行智能的自我学习,提高了对象存储设备的预取命中率,从而提高对象存储设备的性能。

属性管理可分为分布式和集中式管理两种。分布式的属性管理将每个对象的所有属性存放到一个前缀名与对象相同的文件中,如对象ID为123,对象的属性文件命名为123.attri。这样一个对象就对应数据文件和属性文件两个文件。笔者采用文件方法实现了分布式属性管理。文件方法虽然简单易行,但是其存储管理开销大,在实际实现时性能不是很理想。集中式属性管理,即将所有对象的属性集中存放管理,采用哈希桶方法予以实现。最终实验数据表明该方法不仅降低了管理成本,而且提高了整个系统的吞吐率。

对象属性的存放策略

. 哈希桶

属性是对象存储系统中最大的特色,用户在实际应用中需要进行大量的查找、增添、删除属性的操作。如何提高属性的存取速度和效率,对于提高整个对象存储系统的性能有着重要的意义。因此,本文提出了哈希桶的方法来集中管理属性的存放。

哈希桶的整体设计思想是:属性用〈key, value〉的记录方式表示,并以桶(bucket)为单位进行存放。用户只需知道相应属性的key,根据哈希映射关系便可迅速找到属性的value,而不需要按照传统的文件方法,先根据属性文件名查找对应的属性文件,再在该属性文件中查找对应的具体属性。

属性的key用〈partition_ID, user_ID, attri_Page_ID, attri_Index_ID〉表示。由于partition_ID、user_ID惟一表示了一个对象,这样的组合就惟一确定了每一个对象的每一条具体属性。Key对应的value中存放相应的属性数据。桶用来记录查找属性的相关信息(用struct bucket_element表示,包括属性的哈希值(hash_value)、该属性记录在磁盘中的位置(data_pointer)、属性的key长度(key_size)、属性的value长度(data_size)等)。桶的大小可以由用户自己设定,默认为1 024 Byte;桶中可装载的bucket_element的个数用max_count表示。当桶中的记录装满时,即count=max_count,该桶就裂变成两个桶。为了迅速地定位到所需要的桶,还引进了目录表(dir_table),记录了所有桶对应的在磁盘上的存放地址信息。

. 属性查找

下面来看一下属性查找的过程(图2),具体用obs_findkey(uint64_t partition_ID, uint64_t user_ID, uint32_t attri_Page_ID, uint32_t attri_Index_ID, …)函数来实现。

a)根据属性key中的〈partition_ID, user_ID〉,通过哈希计算得到相应31位的hash_value。

b)取hash_value的高N位作为目录表(dir_table)的索引,dir[N]中的值指向存放该属性信息的桶。根据该值从磁盘读入相应的桶。

c)Hash_vale % max_count,得到的值作为所查找属性在桶中的索引(index)位置。

d)在桶中第index个bucket_element中记录了属性的相关信息,从中得到date_pointer中记录了该属性在磁盘上的具体

存放地址。

e)根据 date_pointer值从磁盘中读入该属性记录,比较该属性的 key值与要查找属性的 key值。如果相同,该属性就是所查找的属性,查找成功;反之, index值加 1,跳转步骤 d)。

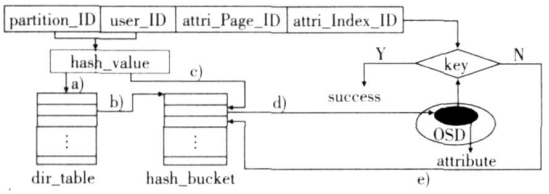

图 2 属性的查找过程

. 属性删除

在哈希算法中,哈希冲突是不可避免的。同样,在哈希桶的方法中,由于采用了哈希算法,哈希值相同的属性由于哈希冲突必然会相邻地存放在同一个桶里。在之前的属性查找过程中,提到根据属性的〈partition_ID, user_ID〉来计算 hash值,而没有用精确度更好、哈希冲突较少的整个 key值〈partition_ID, user_ID, attri_Page_ID, attri_Index_ID〉来计算哈希值,正是利用哈希冲突的特性很好地解决了属性删除这个难点。在删除一个对象时,不仅要删除对象的数据,还要删除对象所有的属性。如果按整个 key计算哈希值的话,由于哈希值的不定性,该对象的属性将会分布到多个桶中,这样在删除属性时,就需要读入多个与之相关的桶。在最坏的情况下,需要遍历所有的桶,大大降低了速度。而且,用户还需要了解对象的所有属性 key值,造成使用上的诸多不便,在某些场合还会直接导致操作失败。根据〈partition_ID, user_ID〉计算 hash值,同一个对象的所有属性,由于其〈partition_ID, user_ID〉相同,所得的 hash值也必定相同;而根据解决哈希冲突的处理方法, hash值相同的属性顺次相邻存放,因此该对象所有的属性必定存放到同一个桶中。在删除对象属性时,只需要从磁盘上读入该桶,查找桶中所有〈partition_ID, user_ID〉相同的属性,便可完全删除该对象的所有属性,从而实现对象快速、高效的删除。

性能评估

为了测试两种属性管理方法的优劣,笔者编写程序生成一个综合负载[7]。该负载中对象属性大小分布在 1 KB～8 MB,且服从正态分布,属性的请求随机到达。在本测试的 trace中,读属性操作占 60%,写属性与删除属性操作占 40%。实验平台采用 Intel P4 3 GHz处理器, 512 MB内存, ATA 120 GB 2 MB缓存的磁盘。在单个 OSD设备上进行仿真测试。

实验结果如图 3所示。在属性管理的性能比较上,总体而言,哈希桶方法远远优于文件方法管理。文件方法管理由于涉及到众多 file、dentry、inode等内核数据结构之间的复杂关系[8],在大量的文件读、写、删除操作以后,性能下降很快。特别是在文件的读写属性操作时,更是因为要加互斥操作以及维护数据的一致性,使得其操作时间要高于哈希桶方法中的处理时间。在哈希桶的管理方法中,由于涉及的数据结构较少,系统的吞吐率虽然随着访问次数的增长不可避免地有所下降,但是仍大幅度地优于同等条件下的文件管理方法。

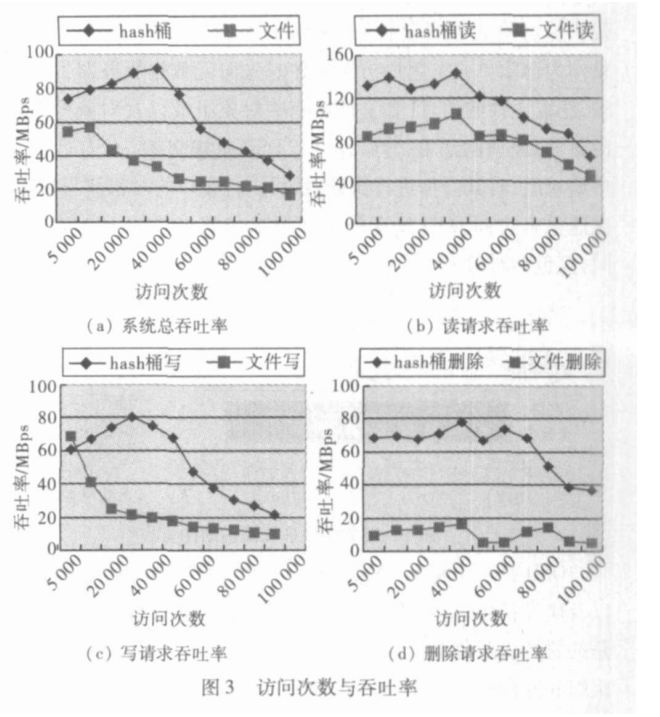

图 3 访问次数与吞吐率

结束语

本文研究了对象存储系统中的属性管理方法哈希桶,通过集中管理对象属性降低了存储管理成本,提高了管理效率。在同等条件下仿真测试得出结论,哈希桶属性管理方法的性能远远优于现有的文件属性管理方法。

参考文献:

[1] MESNIER M, GANGER G R, RIEDEL E. Object-based storage[J]. IEEE Communications Magazine, 2003, 41(8): 84-90.

[2] LOHMEYER J B, PENOKIE G O, ALOISI P D, *et al*. Information technology SCSI object-based storage device commands (OSD), Technical Council Proposal Douament T10/1355-D[R]. [S. l.]: ANSI, 2004.

[3] BRAAM P J. The Lustre storage architecture[EB/OL]. http://www.lustre.org/docs/lustre.pdf.

[4] Panasas. Object storage architecture: defining a new generation of storage systems built on distributed, intelligent storage devices[R]. [S. l.]: Storage Networking Solutions Europe, 2004.

[5] FENG Dan, QIN Ling-jun, ZENG Ling-fang, *et al*. A scalable object-based intelligent storage device[C]//Proc of the 3rd International Conference on Machine Learning and Cybernetics. Shanghai: [s. n.], 2004: 387-391.

[6] LU Ying-ping, DU D H C, RUWART T. QoS provisioning framework for an OSD-based storage system[C]//Proc of the 22nd IEEE/13th NASA Goddard Conference on Mass Storage Systems and Technologies. 2005: 28-35.

[7] WANG Feng, XIN Qin, HONG Bo, *et al*. File system workload analysis for large scientific computing applications[C]//Proc of the 21st IEEE/12th NASA Goddard Conference on Mass Storage Systems and Technologies. 2004: 129-152.

[8] BOVET D P, CESATIM. Understanding the Linux kernel[M]. 2nd ed. [S. l.]: O'Reilly, 2002: 400-465.