一种对象存储设备中自适应动态预取策略*

龚 玮, 冯 丹, 覃灵军
(华中科技大学 计算机学院 信息存储系统教育部重点实验室, 湖北 武汉 430074)

摘 要: 在对象存储系统中, 对象存储设备具有很高的智能和处理能力, 它负责对象及其属性的数据组织和管理, 向外提供基于对象的访问接口。每个对象均具有属性, 它反映了对象的某些特征。通过为每个对象自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, 从而实现一种高效的、自适应的动态预取算法, 可以提高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。

关键词: 对象存储设备; 对象属性; 自适应预取
中图法分类号: TP393 文献标识码: A 文章编号: 1001-3695(2006)10-0054-03

Adaptive Dynamic Prefetching Policy on Object-Based Storage Device

GONG Wei, FENG Dan, QIN Ling-jun
(Key Laboratory of Data Storage System, Ministry of Education, Huazhong University of Science & Technology, Wuhan Hubei 430074, China)

Abstract: Object-Based Storage Device(OBSD) has intelligence and processing ability, which is responsible for managing objects as well as their attributes, and provides object-based interface to the storage system. Every object has attributes which reflects some of its characteristics. OBSD can learn users' accessing regularity through defining a "prefetch" attribute page for every object. An adaptive prefetching algorithm with great efficiency is achieved, and prefetch hit rate is increased, ultimately OBSD's capability is improved.

Key words: Object-Based Storage Device(OBSD); Object Attribute; Adaptive Prefetching

现在计算机系统的计算能力和网络传输能力均有了显著提高, 而存储系统却成为计算机进一步发展的瓶颈。传统的存储系统体系结构暴露出的各种问题使得存储系统在性能、可靠性和安全性上很难再有更大的提高。对象存储系统(Object-Based Storage System, OBSS)概念[1,2]的提出在一定程度上缓解了这种压力。

传统的存储系统均以数据块或文件为访问接口提供给用户使用, 它们都在不同程度上存在着缺陷。以数据块为基本传输单元, 在存储管理和数据共享方面代价比较大; 以文件为基本传输单元, 在性能上受限制于文件服务器。对象存储系统以对象为基本传输单元。其继承了以数据块为访问接口时的高性能和以文件为访问接口时的安全性及跨平台的可操作性。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。对象还具有属性, 用于描述对象的特征, 如多媒体数据对象的 QoS属性描述了该对象的网络延迟要求[3]。对象存储系统的实现得益于集成电路技术的高速发展和嵌入式系统的流行, 它使存储设备本身具有很高的处理能力和智能成为可能。

预取策略是提高存储系统性能的主要方法之一。传统的存储系统中, 存储设备本身很难实现高效的预测, 这主要有以下两方面的原因: ①以数据块为访问接口的系统(如 SAN)中, 数据块的管理相当复杂, 而且主要依赖于中心管理器, 存储设备本身只能被动响应中心管理器的访问请求, 在存储设备端很难实现高效的预取算法; ②以文件为访问接口的系统(如 NAS)中, 文件服务器本身负载就很大, 而预取策略需要文件服务器直接参与, 其效率很难提高。在对象存储系统中, 存储设备本身具有很强的处理能力和智能, 在它之上实现对象一级的预取策略对提高存储设备的性能有很大帮助。

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, 提出了一种对象存储设备(Object Based Storage Device, OBSD)中高效的自适应预取策略, 并分析了影响预取策略的各种因素。

1 对象存储系统(OBSS)和对象存储设备(OBSD)

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(OBSD)连接起来。OBSS实现一个基于对象的分布式文件系统; MDS提供全局名字空间, 管理文件到对象的映射, 提供身份验证等安全机制; OBSD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 OBSD直接交互; OBSD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OBSD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。OBSD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

收稿日期: 2005-08-05; 修返日期: 2005-09-28
基金项目: 国家"973"计划资助项目(2004CB318201)

及对象的属性[1,2]。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 OBSD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[4]。

对象的属性可以描述对象的各种特征,加上 OBSD上集成处理能力[5],我们可以合理地利用对象的属性在 OBSD端实现高效的自适应预取策略。后面将详细描述如何利用对象属性在 OBSD上实现的一种高效的、自适应的预取策略。

2　自适应动态预取策略

在对象存储设备中,提供给用户的是以对象为单位的逻辑视图[5]。而对象的属性能反映对象的某些特性,如对象的总访问次数、最后一次访问时间等。这就为在对象存储设备中实现高效的预取策略提供了基本条件。在对象存储系统中,对象本身是一个有结构和特定意义的数据载体,它可以是个数据库中的表,描述一些字段之间的联系;也可以是一段精彩的多媒体视频,描述一部电影中最精彩的部分[2]。这使得用户访问 OBSD中的对象时具有一定的特征,他们在一段时间内访问的对象会存在一定的联系。比如,当用户访问一个保存在数据库中一张表的对象时,用户可能会很快就访问与此表相关的其他对象;当用户访问一段电影片断时,用户可能会很快访问这部电影的下一个片断[6,7]。基于这一特性,本文提出了一种具有自我学习能力的自适应预取策略。根据不同的应用,适当调整预取策略中对预取属性页的属性设置和对属性页的各种操作,以便满足不同的应用需求。

在对象存储系统中,用户除了能对 OBSD中的对象进行读写外,还可以对对象的属性进行操作[2]。对象的属性除了标准中定义的公共属性外,还可以自定义属性[2],他们均作为对对象某些特征的描述。用户可以为了满足特殊应用的需要而为每个对象自定义一些标志对象特征的属性。比如在流媒体中,就可以为每个对象定义一个 QoS属性,标志对象中所包含的流媒体数据需要的最小延迟和网络带宽等。为了实现高效的智能预取,我们在对象存储设备中为每个对象定义一个预取属性页。属性页中有 L 条属性。每条属性记录一个对象 ID、访问次数和最后一次访问时间,其中第一条属性记录该属性所属对象的 ID号、访问次数和最后一次访问时间。每当访问对象 A 时,预取属性页中第一条属性的访问次数加 1,最后一次访问时间更新为当前时间,该属性页后面的 $L-1$ 条属性遵循以下规则进行刷新:

(1)当系统访问完对象 A 后,接着访问 B,则在预取属性页中查找是否存在对象 B 的 ID。如果存在,则将属性中的访问次数加 1,最后一次访问时间改为当前时间;如果不存在,则添加一条新的属性,属性中的 ID号为对象 B 的 ID号,访问次数为 1,最后一次访问时间改为当前时间。

(2)属性页中 L 条属性没有填满时,该属性页中的记录依次按照规则(1)进行添加;属性页中 L 条记录均填满后,则根据公式:

$$Attr\_out = k \cdot \frac{n}{T_{last\_to\_now}}$$

计算每个属性的 Attr_out值,将值最小的那条属性删除,并按照规则(1)添加新的属性。其中 n 为属性中对象的访问次数,$T_{last\_to\_now}$ 为对象的最后一次访问时间与当前时间的时间差,k 为常数。

按照上述规则,当用户对 OBSD进行了若干次访问后,OBSD中的每个对象均创建了相应的如表 1所示的预取属性页,根据预取属性页便可以实现如下的预取算法。

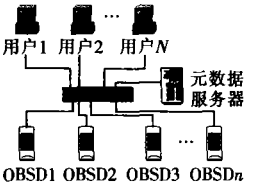

图 1　对象存储系统体系结构

表 1　对象的预取属性页

对象 ID	访问次数	最后一次访问时间
0x0001	2	2005-06-01
0x0002	3	2005-06-02
0x0005	5	2005-06-04
⋮	⋮	⋮
L	…	…

设对象 A 在某个时间段内的访问次数(即对象 A 的预取属性页的第一个属性中的访问次数)为 C_A,用户访问完对象 A 后接着访问对象 B 的次数(即对象 A 的预取属性页中对象 ID为 B 的对象 ID属性中的访问次数)为 C_{AB},根据如下公式计算对象 B 的访问概率[7,8]:

$$\begin{cases} P(B|A) = \dfrac{C_{AB}}{C_A} & \text{对象 } B \text{ 在 } A \text{ 的预设属性页} \\ P(B|A) = 0 & \text{对象 } B \text{ 不在 } A \text{ 的预设属性页} \end{cases}$$

由于系统的 CPU, Memory等资源有限,我们设定一个动态的预取极限 H ($0 \leqslant H < 1$),当用户访问对象 A 时,OBSD就根据对象 A 的预取属性页计算出访问概率大于预取极限 H 的若干个对象,并将其预取到对象存储设备的内存中。这样,当用户对 OBSD进行了若干次的访问后,对象的预取属性页就能很好地反映用户对 OBSD中对象访问在先后关系上的兴趣。以流媒体为例,OBSD中存放的对象均是多媒体数据,当用户对 OBSD进行了一定时间的访问后,对象的预取属性页就能反映出用户访问完一段视频后,下一个极有可能访问的对象。与此同时,随着用户对 OBSD的不断访问,对象的预取属性页也要进行相应的更新,它将随着用户对 OBSD中对象访问兴趣的变化而变化。这就使得此预取策略有一定的自我学习的智能。预取对象时,与哪些系统资源有关,预取多少对象最为合适将在下一节进行详细分析。

3　算法分析

为了分析方便,我们假设对象存储设备中的对象保存在磁盘上的一个连续区域中,这样当访问某个对象时,磁盘定位到对象起始位置后便可以连续读取该对象。我们利用代价函数对算法进行分析。设 α_w 为设备访问对象时定位到该对象的起始位置所付出的代价,它与磁盘的寻道速度,对象在磁盘上存放的位置有关;α_r 为将单位大小的数据从磁盘介质上读到内存中(或将内存中的数据写到磁盘上)的代价,它与磁盘的读写速度,系统可利用的 CPU、内存等资源有关;b 为 OBSD网络带宽,它与当前的网络负载有关;a_t 为在单位时间内通过网络传输数据所需的代价,它与 OBSD的处理能力以及用户要求 OBSD对对象作何处理有关。下面将对用户访问 OBSD时,预取策略的相关代价函数进行分析。

用户对某个 OBSD进行了一段时间的访问后,当用户请求一个对象而此对象并没有被 OBSD预取时,用户访问此对象的

代价就与设备的系统负载以及网络传输有关,设其代价为

$$C_1 = \alpha_w + \alpha_r \cdot S + \alpha_t \cdot t_t = \alpha_w + \alpha_r \cdot S + \alpha_t \cdot \frac{S}{b} \quad (1)$$

其中 S 为对象的大小。如果用户请求的对象已经被存储设备预取到内存中,则用户请求的对象只需要从对象存储设备的内存中传输给用户即可,这时的代价为

$$C_2 = \alpha_t \cdot \frac{S}{b} \quad (2)$$

根据以上的分析,我们现在假设用户请求一个新的对象,此对象的属性页中有 m 个对象的访问概率 P 大于 0。这时若要满足用户的下一个请求,在没有预取的情况下,其平均代价 C_3 为

$$C_3 = \sum_{i=1}^{m} P_i \cdot \left[\alpha_w + \alpha_r \cdot s_i + \alpha_t \cdot \frac{S_i}{b_i} \right] \quad (3)$$

其中,P_i 为对象 i 的访问概率,S_i 为对象 i 的大小,b_i 为传输对象 i 时的网络传输能力。如果在 m 个对象中,有 n 个对象已经在设备端进行了预取,那么这时的平均代价 C_4 为

$$C_4 = \sum_{i=1}^{n} \alpha_t \cdot \frac{S_i}{b_i} + \sum_{i=n+1}^{m} P_i \cdot \left[\alpha_w + \alpha_r \cdot S_i + \alpha_i \cdot \frac{S_i}{b_i} \right] \quad (4)$$

比较式 (3)和式 (4)可以得到,要想 C_4 的值最小,就必须预取满足以下条件的对象:

$$P > \frac{1}{\frac{b}{\alpha_t} \cdot \left(\frac{\alpha_w}{S} + \alpha_r \right) + 1} \quad (5)$$

我们定义

$$H = \frac{1}{\frac{b}{\alpha_t} \cdot \left(\frac{\alpha_w}{S} + \alpha_r \right) + 1} \quad (6)$$

为预取极限。从式 (6)中可以看出,随着 α_w 和 α_r 增大,预取极限 H 就越小,OBSD就可以预取更多的对象,有助于提高命中率。这说明,提高 OBSD上磁盘的读写速度,以及尽量使对象存放在磁盘上的连续区域从而减少磁盘在读取对象时的定位时间可以提高 OBSD的预取效率;另一方面,OBSD的网络带宽也影响 OBSD的预取。合理地利用网络带宽,并在多个 OBSD之间进行负载平衡,将有利于 OBSD的预取效率。预取极限 H 还与 α_t 有关,OBSD本身的处理能力越强,其预取极限 H 越小,因此 OBSD大多采用高性能的嵌入式系统进行设计与开发。

我们以流媒体服务器为例对以上算法进行验证。在 OBSD中存放 1 000个多媒体影片对象,并让用户对 OBSD进行长时间的访问。编写程序实现以上描述的自适应预取算法,并将用户对 OBSD中对象的访问次数和预取的命中率进行记录。为了证明上述算法的智能性,我们记录了用户对 OBSD不同访问次数下的预取命中率,在测试平台上设置了不同的预取极限,得到的结果如图 2所示。

从图 2中我们很明显地看到,当用户的访问次数达到一定值后,OBSD对用户的兴趣进行了有效的学习,预取的命中率能维持在一个稳定的水平上。这就充分证明,以上描述的算法具有很高的智能和自我学习能力。预取命中率的大小取决于预取极限 H 的值,因此,提高 OBSD本身的处理能力有助于提高 OBSD的命中率。

比较普通的预取算法和自适应预取算法的命中率。普通的预取算法一般都是根据目前访问的对象,预取紧接着的若干个对象,它是一种静态的预取。我们使用了同样的测试平台,OBSD中存放 1 000个多媒体影片对象,让用户按照同样的访问方式对两个 OBSD进行访问,分别记录下两个 OBSD中随着用户对 OBSD访问次数的增加,其预取对象的命中率。图 3是对两种算法的比较结果。

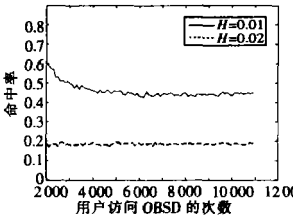

图 2 不同预取极限时的命中率比较

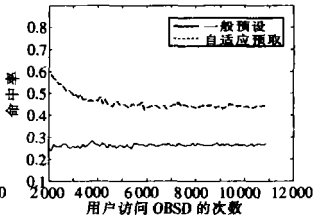

图 3 不同的预取算法之间的比较

从图 3中可以看出,对于流媒体服务器而言,自适应的动态预取策略能够很好地学习用户访问多媒体对象的兴趣。随着用户对 OBSD访问次数的增加,预取的命中率能稳定在一个较高的水平,它比静态的预取相邻对象预取策略的命中率高出近一倍,其动态的自我学习能力体现得相当明显。

4 结论

在对象存储系统中,对象的属性能反映对象的某些特征。自定义的对象属性能针对用户不同的应用描述用户访问 OBSD中对象的访问兴趣。借助自定义预取属性的预取策略,能够根据用户对 OBSD的访问情况进行实时的自我学习,从而实现高效的智能预取。它较普通的预取策略在命中率上有较大的提高。

参考文献:

[1] M Mesnier, G R Ganger, E Riedel. Object-based Storage[J]. IEEE Communications Magazine, 2003, 41(8): 84-90.

[2] J Satran. Object-based Storage Device Commands[EB/OL]. http://www.t10.org/draft/osd, 2004.

[3] Lu Y, Du D H C, Ruwart T. QoS Provisioning Framework for an OSD-based Storage System[C]. Proceeding of the 22nd IEEE/the 13th NASA Goddard Conference on Mass Storage Systems and Technologies, 2005. 28-35.

[4] Richard Hedges, Bill Loewe. Parallel File System Testing for the Lunatic Fringe: The Care and Feeding of Restless I/O Power Users[C]. Proceeding of the 22nd IEEE/the 13th NASA Goddard Conference on Mass Storage Systems and Technologies, 2005. 3-17.

[5] Dan Feng, Ling-jun Qin, Ling-Fang Zeng, *et al*. A Scalable Object-based Intelligent Storage Device[C]. Proceeding of the 3rd International Conference on Machine Learning and Cybernetics, 2004. 387-391.

[6] Philip A, Bernstein, Shankar Pal. Context-based Prefetch for Implementing Objects on Relations[C]. Proceeding of the 25th VLDB Conference, 1999. 102-107.

[7] Zhimei Jiang. An Adaptive Network Prefetch Scheme[C]. IEEE International Conference on Communications, 1998. 8-12.

[8] Zheng Zhao, Jie Yang. Measurement Based Intelligent Prefetch and Cache Technique in Web[C]. Proceeding of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering, 1999. 9-12.

作者简介:

龚玮(1981-),男,湖北鄂州人,硕士研究生,主要研究方向为网络存储系统;冯丹(1971-),女,湖北金山人,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。