文章编号: 1005－0523(2009)02－0062－05

基于内存文件系统的分布式文件服务器 Cache系统

程宏波[1], 伦 利[1], 郑宗校[2]

(1.华东交通大学 电气与电子工程学院,江西 南昌 330013; 2.浙江大学 信息科学与工程学院,浙江 杭州 310014)

摘要:分布式文件服务器在虚拟机平台上的应用越来越普遍,但现有的文件服务器 Cache系统不适合在虚拟机上使用。利用内存文件系统管理 Cache中的信息,并设置 Cache中文件的大小阈值,使 Cache系统间歇性使用,从而有效地解决了这一问题。试验结果表明,基于该设计思想实现的 Cache系统性能优异,维护简便,可以达到预期的使用效果。

关 键 词: Cache;内存文件系统;分布式文件服务器

中图分类号: TP391 **文献标识码**: A

分布式文件服务器将连接在网络上的多个文件服务器通过某种方式管理以对外提供统一的文件服务功能。为了提高分布式文件服务器的性能, Cache机制被引入其中。目前应用较多的 Cache系统中的管理对象主要有两种[1]:一是文件,二是存储块。

现阶段采用 J2EE架构实现的文件服务器由于其开发快捷,维护方便,且具有跨平台的优势,得到越来越广泛的应用[2,3]。但在这种基于虚拟机的文件服务器上实现 Cache又遇到了新问题:在虚拟机上不能直接操作存储块。因此,在内存中实现 Cache就不能再采用存储块作为管理对象。

文[4]使用内存文件系统实现了 Cache但其管理对象仍然是基于存储块的。也有使用文件作为管理对象,在内存中实现 Cache的[5]。但该方案建立在 Linux平台上,且采用了内存磁盘作为载体,带来了系统的复杂性、安全性和经济性等多方面问题,再加上 Linux平台的限制,这种 Cache的实现方式也不足以解决上述问题。

针对这一现状,本文提出使用内存文件系统管理 Cache中的信息,并通过限制 Cache中文件的大小从而使 Cache系统间歇性工作的设计思想,最后给出原型系统及其试验结果。

1 内存文件系统和在 Cache中的应用

1.1 内存文件系统的设计

内存文件系统作为应用程序的一部分,随着应用程序的启动而初始化。如图 1所示,内存文件系统主要由文件节点和文件管理器两个类组成。文件节点包含必要的文件信息以及文件内容。文件管理器负责内存文件系统的初始化,以及文件读写接口和备份接口的实现。

1.2 在 Cache中使用内存文件系统

内存文件系统在 Cache系统中的使用如图 2所示。 Cache中使用的内存文件系统将采用和文件服务

收稿日期: 2008－12－12

作者简介:程宏波(1979－),男,湖北随州人,讲师.研究方向为牵引供电系统及微机保护装置可靠性。

器中磁盘文件系统相同的目录结构。无论文件服务器采用何种方式管理用户文件，只要是基于树状目录结构的，都可以用内存文件系统实现同样的组织方式。

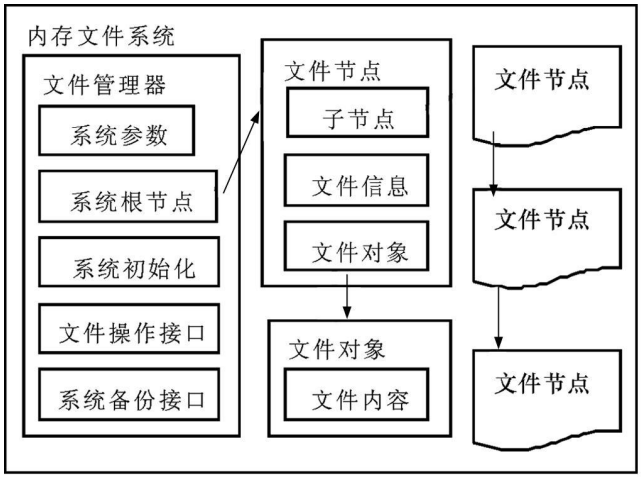

图 1　内存文件系统组成结构图

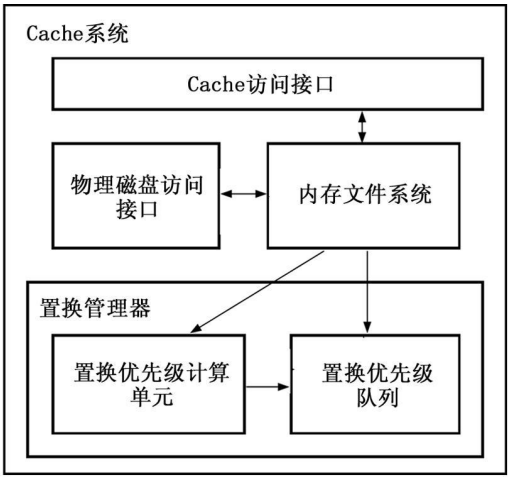

图 2　基于内存文件系统的 Cache系统架构图

设物理磁盘文件系统中的每一个文件 (或目录)为一个文件节点，用 Nn表示 (n为文件磁盘上的文件总数)，则物理磁盘的文件系统中所有文件可以用集合 $A=\{N_1, N_2, \cdots, Nn\}$表示。设内存文件系统中的文件数为 m，文件节点集合为 B。在内存文件节点和磁盘文件节点间建立映射规则 f则当 m＝n时，$b=f(a)$，$a\in A$，$b\in B$，所以有集合 $B=\{b: b=f(a),\ a\in A\}$。因为内存的实际空间有限，通常 $m\neq n$，因此内存文件系统中的文件集合实际上为 B'，$B'\subset B$。

设物理磁盘文件系统中的文件间的关系满足 R(a，b)，其中 a，b∈ A。则对于内存文件系统中的文件间的关系 R′(a′，b′)有

R(a，b) ＝R′(a′，b′)，其中 $a'=f(a)$，$b'=f(b)$。

2　间歇使用型 Cache系统

为了解决大文件带来的诸多问题，本文提出间歇性使用 Cache的设计思想。Cache系统的架构如图 2所示。Cache系统与文件服务器的集成如图 3所示。图中，写缓存和读缓存都有一条避开 Cache直接访问磁盘的数据流，因而 Cache系统的使用不再是自始至终，而是间歇性的。

2.1　大文件的处理

按照 Cache缓存文件的思想，即便访问的文件很大，也应该把这些巨大的文件放入 Cache中。但现实中这样做是很困难的。

为了解决这个问题，当文件大小超过某个值时，不再使用 Cache。即可以采用间歇使用型 Cache系统。决定是否放入 Cache的文件大小的临界值称为阈值。通过对假想模型的估算，可以得到阈值的范围，并且可以证明间歇性使用 Cache系统不会损失性能。间歇使用型 Cache系统是平衡 Cache性能与空间的一种有效途径。

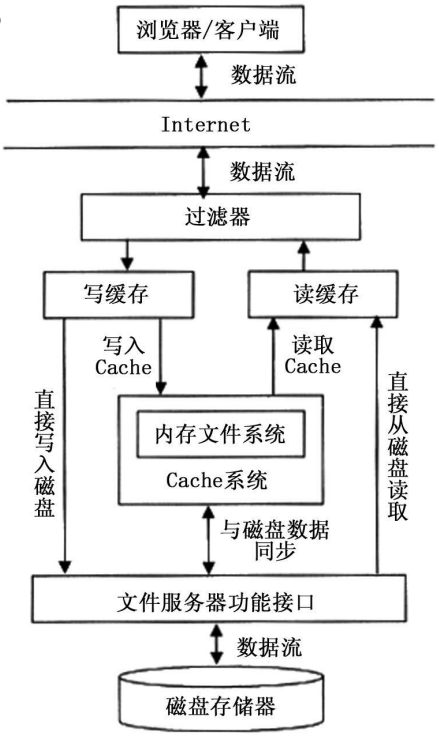

图 3　Cache系统与文件服务器的集成图

2.2 阈值的计算

首先建立如下的计算模型：设网络带宽为 B_n，磁盘带宽为 B_d，磁盘的平均寻道时间为 t_d，内存带宽远远大于网络和磁盘的带宽，网络带宽大于磁盘带宽，Cache的命中率为 H，使用 Cache时文件操作的总耗时为 T_1，未使用 Cache时文件操作的总耗时为 T_2，当传送的文件大小为 S时，不考虑网络延时，有

$$T_1 = H \cdot \frac{S}{B_n} + (1-H)\left(\frac{\beta(S)S}{B_d} + t_d\right) \tag{1}$$

$$T_2 = \frac{S}{B_d} + t_d \tag{2}$$

其中 $\frac{\beta(S)S}{B_d} + t_d$ 是使用 Cache但未命中的情况下，从磁盘读取文件的耗时。

如果使用相对减少的耗时 $F = \frac{T_2 - T_1}{T_2}$ 来表示使用 Cache的性能提升，则有

$$\begin{aligned} F &= \frac{T_2 - T_1}{T_2} = \frac{\frac{S}{B_d} + t_d - H \cdot \frac{S}{B_n} - (1-H)\left(\frac{\beta(S)S}{B_d} + t_d\right)}{\frac{S}{B_d} + t_d} \\ &= \frac{S\left[1-(1-H)\beta(S) - H \cdot \frac{B_d}{B_n}\right] + HB_d t_d}{S + B_d t_d} \\ &= \frac{S\left[1-\beta(S) + H \cdot \left(\beta(S) - \frac{B_d}{B_n}\right)\right] + HB_d t_d}{S + B_d t_d} \end{aligned} \tag{3}$$

设文件修改的概率为 P，P为常数，则 $\beta(S)$可近似写成 $1+PS$。对一般的 PC机而言，$B_n = 133$ MB/s，$B_d = 100$ MB/s，$t_d = 0.012$ s。对于一般的网络应用，理想的 Cache的命中率 $H \geqslant 35\%$，这里取 $H = 35\%$。一般文件更新的几率要远远小于未更新的几率，不妨取 $p = 0.1$。将以上各参数的值代入(3)式，可得

$$F = \frac{-0.065S^2 + 0.0875S + 0.42}{S + 1.2} \tag{4}$$

对(4)式求导可得，当 $S = 1.69$ MB时，F有最大值 13.2%，且只有当 $0 < S < 3.3$ MB时，$F > 0$。S的这一取值范围就是阈值的取值范围。由此可知，文件大小超过阈值的文件不应当放入 Cache中，否则会影响系统的性能。

2.3 Cache系统对分布式的支持

基于内存文件系统的 Cache可以方便地部署到原来的无 Cache机制的分布式文件服务器架构中。

由于 Cache系统提供了与磁盘文件系统完全类似的文件操作接口，只要在原有的架构中插入 Cache层即可向上无缝连接客户端接口。向下连接文件服务器接口则通过 Cache系统内建的物理磁盘访问接口实现。由于每台分布式的文件服务器单元独立维护自身的 Cache，Cache中缓存的只是当前服务器单元的数据，因此只需考虑 Cache数据与当前服务器单元中数据的一致性即可。

3 原型系统的性能测试

3.1 测试环境简介

原型系统的测试在常见的 PC机上进行。该 PC机的 CPU为 PⅣ 1.8 GHz，内存为 512 MB，硬盘为 7 200转的 ATA100接口高速硬盘。为了模拟足够的网络带宽，采用单机运行模式，即客户端与服务器位于同一台计算机，且客户端不实际占用磁盘 I/O。

操作系统采用 Windows 2000 server，Java虚拟机为 1.5.0版。客户端模拟随机存取访问。Cache系统

采用简单的 SIZE置换算法和回写一致性算法。服务器上预先保存有 650个文件，总计 985 MB。其大小基本满足对数正态分布。

3.2 使用 Cache后文件服务器的直接性能提升

计算提升比率所采用的计算式为

$$R=\frac{\text{不使用 cache所需的时间}-\text{使用 cache所需的时间}}{\text{不使用 cache所需的时间}}$$

测试结果如图 4所示。从图中可以看出，使用 Cache后访问同一个文件所需的时间大大缩短，性能提升在 50%以上。同时也可以看到，Cache的性能提升与被访问文件的大小有关。当文件较小时，由于 Cache系统本身调度所需的时间占了较大比重，因此使用 Cache时与不使用时访问时间基本一致；但随着文件的增大，提升比率有所增加。当文件足够大时，Cache系统的调度时间和磁盘等待时间在文件访问中的比例变小，提升的比例就接近于内存带宽比磁盘带宽的提升率，而趋于稳定。

3.3 不同阈值时的 Cache和文件服务器性能比较

测试结果如图 5所示。从图中可知，Cache的命中率随文件阈值的增大而增加，但当阈值超过一定值以后，命中率的提升就十分有限，命中率曲线很快趋于水平。

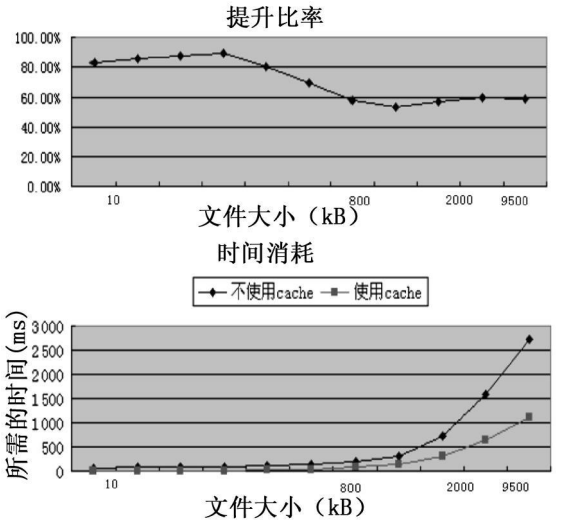

图 4　使用 Cache后性能提升图

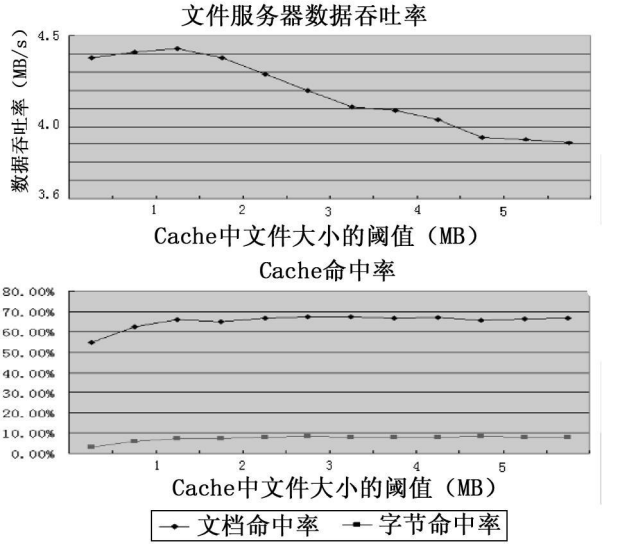

图 5　不同阈值时系统性能图

从数据吞吐率图中可知，吞吐率随阈值的增加大致呈抛物线分布。当阈值在 1.5 MB时，吞吐率达到峰值。同时也可以看到，当阈值超过 3 MB时，吞吐率滑落到 4 MB/s以下，这个吞吐率相对不使用 Cache时的吞吐率已没有优势可言。测试结果与理论计算结果十分吻合。

另一方面，从图中可见，使用基于内存文件系统的 Cache系统即便在间歇性使用的情况下也可以有效提升文档命中率，缩短大部分文件操作请求的操作时间，但相对而言磁盘 I/O开销的节省程度比较有限。即便如此，字节命中率最高时还是能达到 10%左右。因此，总的来说使用 Cache后的性能提升是相当显著的。

4 结论

基于内存文件系统的 Cache系统，实现简单，维护方便，效果显著。使用内存文件系统有利于管理 Cache中的信息；使用阈值限制 Cache中文件的大小，间歇性启用 Cache系统，不仅使内存也可以用于缓存

逻辑文件,而且提高了内存的利用率,改善了系统性能。

本文重点在于阐述基于内存文件系统的 Cache系统的思想,但在实际应用中还存在诸多影响 Cache系统性能的因素。其中最主要的参数是 Cache的大小和 Cache文件的阈值,此外还有 Cache的置换算法。因此,根据服务器运行情况动态调整参数,选用适合文件服务器访问特性的置换算法等都将使得文件服务器的性能得到较大的提升。

参考文献:

[1] 丁善镜. 缓冲技术在分布式文件系统中的应用 [J]. 计算机应用, 2002, 22(9): 115—117.

[2] 周志华,何 萍,尹建伟,等. 基于 Web的海量存储柔性分布式文件服务器设计 [J]. 计算机应用研究, 2002, 19(11): 14—16.

[3] 周志华,陈 刚,董金祥. 基于 Web的跨平台多线程分布式文件服务器设计 [J]. 计算机工程, 2003, 29(4): 41—42.

[4] 陈锡明,卢显亮,宋 杰. 分布式内存文件服务器 (DMFS)的研究和设计 [J]. 小型微型计算机系统, 2002, 21(1): 50—53.

[5] 周 婷,杨根科,吴智铭. 一种基于 Linux的 NAS文件系统设计 [J]. 计算机应用, 2003, 23(6): 92—95.

The Cache System Based on Memory File System for Distributed File Server

CHENG Hong-bo[1], LUN Li[1], ZHENG Zong-xiao[2]

(1. School of Electrical and Electronic Engineering, East China Jiaotong University, Nanchang 330013, China; 2. School of Information Science and Engineering, Zhejiang University, Hangzhou 310014, China)

Abstract: Application of virtual machine in distributed file server becomes more popular. However, the current file server Cache is not suitable for virtual machine. The paper solves this problem by utilizing information in memory-file-system-based Cache system, designing the size of Cache files to make the Cache system be used intermittently. The experiment reveals that the Cache system has good properties and convenient maintenance, achieving the expected result.

Key words: Cache; memory file system; distributed file server

(责任编辑:王建华)