视频点播中视频服务器节目替换算法研究

魏 维[1,2],罗时爱[2],刘凤玉[2]
WEI Wei[1,2],LUO Shi-ai[2],LIU Feng-yu[2]

1.成都信息工程学院 计算机系,成都 610225
2.南京理工大学 计算机科学与技术学院,南京 210094
1.Department of Computer,Chengdu University of Information Technology,Chengdu 610225,China
2.China School of Computer Science and Technology,Nanjing University of Science and Technology,Nanjing 210094,China
E-mail:weiwei@cuit.edu.cn

WEI Wei,LUO Shi-ai,LIU Feng-yu.Study of program replacement algorithm for video server in video on demand system.Computer Engineering and Applications,2008,44(2):245-248.

Abstract: A distributed VoD architecture and two program replacement algorithms for video server are proposed.Improved Least Frequency and Recently Used algorithm (LFRU) is suitable for VoD initialization.After system initialization,least weighted period algorithm is used for video server program replacement.The experiment results comparing to other algorithms indicate the algorithms are effective.

Key words: system architecture of VoD;improved LFRU algorithm;video server;replacement algorithm

摘 要:提出了一种分布集群式的视频点播体系结构,针对视频点播系统特点设计了两种适用于系统不同的运行阶段的视频服务器节目替换算法。在视频点播系统初始化时间段内使用改进的 LFRU 算法进行节目替换;系统达到稳定状态后使用最小加权周期频率替换算法进行节目替换。对比实验表明两种替换算法适合分布集群式视频点播系统,其替换效率较高。

关键词:视频点播系统体系结构;改进的 LFRU 算法;视频服务器;替换算法
文章编号:1002-8331(2008)02-0245-04 **文献标识码**:A **中图分类号**:TP391

1 引言

传统 Cache 机制(即缓冲机制)是通过对较快存储器中的数据进行访问来实现的。分布式的视频点播系统提供流媒体视频服务时,可将分布的各个视频服务器看作整个中央存储节目的高速缓存(Cache)。视频点播系统的 Cache 策略可以在传统 Cache 替换算法基础上进行改进。

传统的磁盘替换算法主要有 FIFO、SOF、LOF、LFU(Least Frequency Used)和 LRU(Least Recently Used)及 LRU 的变种 LRFU 和 LRU-K 几种。在视频点播系统初始运行阶段,节目访问不稳定、具有一定的震荡性。此时,Cache 对访问时间特性比较敏感,替换算法应该结合节目访问的时间信息和频率信息来增强适应性。比如,文献[1]和文献[2]采用 LFRU(Least Frequency and Recently Used)算法。本文对 LFRU 算法进行了改进,将综合考虑时间信息、频率信息和服务节点三因素来判断是否需要 Cache 替换。

LFU 算法中使用的是过去对数据对象所有访问的统计次数。当数据不再被访问时,过去累计的使用信息会造成"陈旧"的数据滞留在磁盘 Cache 中,就是所谓的 Cache"污染"。为了解决这一问题,本文用最近一周之内的频率信息来屏蔽过去陈旧的统计数据, 由此引入了一个加权周期点播频率的概念,形成最小加权周期频率替换算法。

这两种算法适用于 VoD 系统不同的运行阶段。在初始化阶段及稳定运行阶段分别采用改进的 LFRU 算法和最小加权周期频率替换算法进行节目 Cache 替换。

2 视频点播系统体系结构

本文的视频点播系统采用如图 1 所示的分布式集群结构。从整体的层次上来看,该系统分为三层:客户端、管理服务器和视频服务器(Video Server,VS);从服务的层次来看,系统是一个两级的层次服务结构:视频服务器和中央视频服务器,VS 提供其上存储的热门节目的请求服务,而中央视频服务器也能够提供所有节目的请求服务(在视频服务器不能为用户的请求提供服务时);从存储的层次来看,系统是一个三级存储的结构[3]:内存、VS 上的磁盘和中央存储。而中央存储采用三层在线分级存储的管理结构。

整个系统采用的是集中管理、分布式服务和分布存储的思想[4]。系统服务端由 Web 服务器、管理服务器、数据库服务器、(分布)视频服务器、中央视频服务器和中央存储组成。

Web 服务器是用户访问视频点播系统的入口,向用户提供节目信息及登录界面等。管理服务器是整个系统的核心,其主

基金项目:四川省教育厅青年基金项目(No.2006B063);成都信息工程学院发展基金(No.KYTZ20060904)。

作者简介:魏维(1976-),男,博士,讲师,主要研究方向:视频内容分析、多媒体信息处理等;罗时爱(1982-),男,硕士生,主要研究方向:视频点播系统体系结构、视频点播系统 QoS 控制等;刘凤玉(1943-),女,教授,博导,主要研究方向:计算系统性能保持、多媒体技术等。

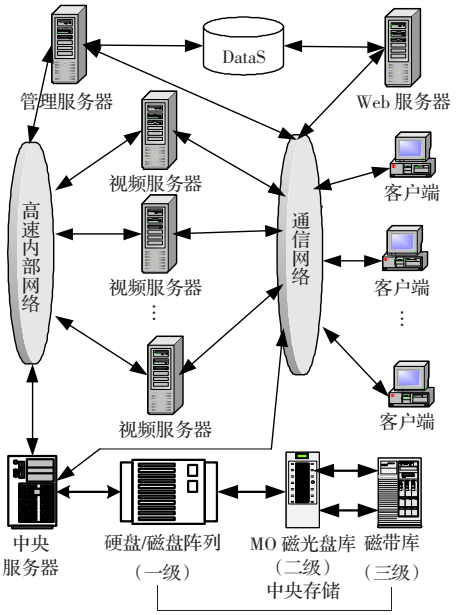

图 1 视频点播系统分布式集群体系结构图

要功能是：负责视频节目数据的管理以及视频服务器的管理、协调；利用磁盘 Cache 策略使系统的热门节目尽量合理地分布于视频服务器中；使用请求调度和流调度算法保持各视频服务器的负载平衡，提高系统的吞吐率。高效的磁盘 Cache 策略和负载平衡算法可极大地提高系统的服务效率。

数据库服务器（DataS）提供各种服务的信息，是 Web 服务器的信息来源。管理服务器根据其保存的节目信息、点播信息进行负载平衡和磁盘 Cache。视频服务器为用户提供流媒体服务。中央存储是系统的节目数据中心，可以看成是中央视频服务器的私有存储设备。中央视频服务器作为两级服务的中央级，在以下两种情况下为用户提供视频服务：一种情况是用户请求的节目是冷门节目，视频服务器磁盘上没有存储；第二种情况是用户请求的节目在视频服务器的拷贝数有限，保存有该节目的视频服务器负载已满[5]。

3 改进的 LFRU 节目替换算法

假设系统的节目库有 N 个节目，排序后节目访问概率符合 Zipf 法则，第 k 个节目的访问概率为 p_k，则节目库可表示为[3]：

$S=\{p_k \mid p_k$ 的访问概率 $f_k=\frac{c}{k^{(1-\alpha)}}, c=\frac{1}{\sum_{i=1}^{N}\frac{1}{i^{(1-\alpha)}}}, k=1,2,\cdots,N\}$。

为了解决在过渡期数据访问不稳定的问题，定义一个过渡时间区间 D。在 D 时间区间内，先使用类似 LRU 算法中的访问时间信息，提高算法对访问模式变化的适应性。当系统运行趋向于稳定后，算法使用类似于 LFU 算法中使用的频率信息。在改进的 LFRU 算法中，决定节目是否 Cache 替换的是 RFN 值，将 RFN 值最大的视频数据作为替换的牺牲对象。RFN 值是数据访问的时间信息和频率信息的加权和与节目所在服务节点的个数的乘积。

$$RFN=RF\times N \tag{1}$$

$$RF=F_D(t)\times R+[1-F_D(t)]\times F \tag{2}$$

其中，R 表示时间信息，F 表示频率信息，N 表示节目所在服务节点的个数。在 D 时间区间内，时间信息和频率信息是互补的，所以 R 和 F 的权重分别是与时间相关的函数 $F_D(t)$和 $1-F_D(t)$。

为了适应从震荡到稳定的应用情况，$F_D(t)$函数应该具有以下性质：(1)在 D 的初期，$F_D(t)$取值要大于 $1-F_D(t)$，在 D 的后期 $F_D(t)$取值要小于 $1-F_D(t)$。(2)在时间段 D 内 $F_D(t)$应该由 1 平和地趋向于 0，而 $1-F_D(t)$则由 0 平和地趋向于 1，所谓平和就是曲线的斜率不能有大的变化。这样 RF 值就从 R 趋向于 F，从具有时间适应性的访问时间信息转向具有全局优化特点的频率信息。

在算法中取 $F_D(t)=\frac{D-t}{D}$，$RF=\frac{D-t}{D}\times R+\frac{t}{D}\times F$。很明显，当 $t=0$，$F_D(t)=1$，$RF=R$，算法等价于 LRU 算法；当 $t=D$ 时 $F_D(t)=0$，$RF=F$，算法等价于 LFU 算法。在 D 时间区间内，t 从 0 到 D，算法也从 LRU 过度到 LFU 算法，满足上述的要求。

对节目库中每个节目（以节目 k 为例）使用如下公式计算其 RFN 值：

$$RFN_k=(\frac{D-t}{D}\times R_k+\frac{t}{D}\times F_k)\times N_k \tag{3}$$

$$R_k=t-t_k \tag{4}$$

$$F_k=\frac{\sum_{i=1}^{c_k} t-t_{ik}}{c_k} \tag{5}$$

其中，t 为系统的逻辑时钟，表示当前时间值；t_k 为节目 k 的计时器（每个节目都有一个计时器），表示对节目 k 最近一次被访问的时间值，其初始值为 t_0，是系统作为参考起点的时间值；c_k 表示对节目 k 访问的次数；N_k 表示节目 k 所在服务节点的个数；t_{ik} 表示节目 k 第 i 次访问的时间值。

式(4)用于计算节目访问的时间信息，$R_k=t-t_k$ 表示节目上一次访问距离现在的时间值。式(5)用于计算节目访问的频率信息，$F_k=(t-t_0)/c_k$ 表示节目过去所有访问距离现在的时间的平均值。RF 将时间信息和频率信息都归一化为一个时间“距离”值，用它们的加权和作为替换的一个比较因子。

本算法用于动态磁盘 Cache 策略且在过渡时间段 D 内使用，图 2 是视频点播系统中动态 Cache 节目替换的流程图。当用户请求在视频服务器上不命中时，中央视频服务器为用户提

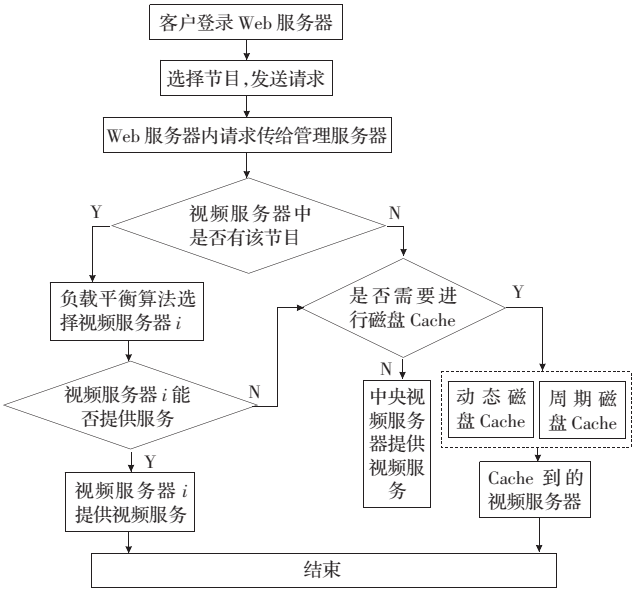

图 2 动态节目替换流程

供视频服务，如果此时点播次数达到系统规定的阀值，则使用算法计算所有节目的 *RFN* 值，如果视频服务器没有足够剩余空间则替换除请求节目外 *RFN* 值最大的节目。当然此时的选择不止一种，可以比较请求节目 p_k 的 *RFN* 值和最大的 *RFN* 值，只有当 p_k 的 *RFN* 值比最大的 *RFN* 值小时才进行替换，否则就不将节目库中的节目替换到视频服务器；为了增加新访问节目进入 Cache 的机会，也可以给请求的节目的 *RFN* 值加上一个阀值再进行比较。

虽然 LFRU 算法和 LRFU 算法都是对 LFU 和 LRU 算法的折衷，但它们还是有很大的差别[2]。如前面所述 LRFU 算法是对访问时间乘上一个和频率有关的权重，它是基于访问时间的替换算法。而 LFRU 算法则是给访问频率乘上一个和访问时间有关的加权值，是基于访问频率的。另外，LRFU 算法中的权重函数和时间无关，参数选择后算法也固定下来，没有适应性。LFRU 算法中的权重函数则随时间发生变化，可以从 LRU 算法过度到 LFU 算法。它的折衷性在设定的时间段内起作用并且是变化的，这和 LRFU 算法完全不同。改进的 LFRU 算法更是继承了原 LFRU 算法的优点，并同时考虑了节目所在视频服务节点的个数这个因素。

4　最小加权周期频率替换算法

本文提出的最小加权周期频率替换算法以节目的点播频率为依据，引入了一个加权周期点播频率的概念。视频点播系统服务对象是人，而人的生活是有时间规律的。通常人们白天工作学习，晚上才有空进行休息和娱乐，到了深夜人们还得进行睡眠为第二天的工作学习养精蓄锐。每个礼拜的周末又有两天的休息时间，这段时间人们可以尽情地休闲、娱乐。根据人的生活规律，本文得到点播请求的时间规律性：每天的点播请求大多在晚上至深夜这段时间，而每周的周末两天又是点播请求比较集中的时段，也就是点播请求基本上是以星期循环的，每个星期的请求分布具有时间上的相似性。

根据前面的分析，本算法考虑用当前时间前一周的点播频率来预测未来节目的点播频率，同时把一周按自然天分成 7 天，节目的预测点播频率就是节目在这 7 天的平均点播频率的加权和，称之为加权周期点播频率。根据数据访问的局部性原理，最近访问的数据更可能随后被访问，因此这 7 天的点播频率信息对预测值的影响应该随时间的远近而有所区别，这种区别就用加权的方式实现。节目的平均点播频率就是每个节目拷贝在某一天的平均点播次数，假定某天节目 i 在视频服务器上有 n 个备份且总共有 m 个点播请求，则该节目该天的平均点播频率为 m/n。

加权周期点播频率用 *WDF* 表示，WDF_i 就是第 i 个节目的加权周期点播频率。节目的加权点播频率用公式(6)和(7)来计算。

$$WDF_j=\sum_{j=1}^{7}W_j\times ADF_{ij} \tag{6}$$

$$ADF_{ij}=\frac{\sum_{k=1}^{n}DF_{ijk}}{N_{ij}} \tag{7}$$

其中，i 表示视频服务器(中央服务器除外)上所有节目的第 i 个节目；j 表示时间段，j 从 1 到 7 分别表示倒数第 1 天到倒数第 7 天；ADF_{ij} 表示第 i 个节目在倒数第 j 天的平均点播频率；W_j 表示倒数第 j 天的加权值，W_j 从 W_1 到 W_7 依次递减；WDF_i 表示第 i 个节目的加权周期点播频率；n 表示系统总的视频服务器数(中央服务器除外)；k 表示第 k 个视频服务器；DF_{ijk} 表示第 i 个节目倒数第 j 天在第 k 台服务器上的点播次数；N_{ij} 表示第 i 个节目倒数第 j 天在所有视频服务器中的拷贝份数。

以上计算节目加权周期点播频率的方法可以不作修改用于周期磁盘 Cache 中，利用了所有能利用的信息。在动态的磁盘 Cache 中，因为动态磁盘 Cache 的时间不像周期 Cache 正好限定在天与天的交界处，它可以发生在一天中的任何时刻，所以还需要加以修改。因此用于动态磁盘 Cache 的用于计算加权周期点播频率的式(6)和(7)应该改成公式(8)和(9)

$$WDF_i=\sum_{j=1}^{7}WW_j\times ADF_{ij}+ADF_{i0} \tag{8}$$

$$ADF_{ij}=\frac{\sum_{k=1}^{n}DF_{ijk}}{N_{ij}} \tag{9}$$

其中，ADF_{i0} 为节目 i 当天的平均点播频率。加权周期点播频率的计算方法通过对前 7 天的平均点播频率进行一定的加权统计(用于动态 Cache 的计算方法还加上当天的平均点播频率)，计算得到的值对节目的未来点播请求有很好的预测性，不仅没有 LRU 单考虑时间特性的缺点，也解决了 LFU 带来的“Cache 污染”问题。加权周期点播频率只是磁盘 Cache 替换的依据之一，当然也是最重要的依据，基本上是优先替换加权周期点播频率最小的节目，但是视频点播系统是一个复杂的系统，还需要考虑“大片初上映”和“节目抖动”的两种特殊情况。

当一部大片刚刚上映时点播率肯定非常高。如果这部影片只存在数据中心而让磁盘 Cache 算法将其 Cache 到视频服务器上，这明显是不合理的，而应该直接在视频服务器组中拷贝多份(管理服务器应该提供这个功能)。这时如果有其他影片需要 Cache，用前面的公式计算这部大片的加权周期点播频率得到的结果可能很小，Cache 替换算法将替换这部影片。显然此时的这种替换应该被禁止，本文算法为每个节目设定一个值来解决这个问题。值表示其是否可以替换，为 1 则不能替换，为 0 则可以考虑将该节目作为替换的牺牲品。影片在直接拷贝到视频服务器组时，设置其值为 1，并可以设置值为 1 的持续时间长度，持续时间到了值就变成 0，此后 Cache 替换算法可以考虑删除该影片数据来存放其他节目。

在 Cache 替换过程中还有可能出现磁盘中所有节目的加权周期点播频率比需要 Cache 的节目在中央服务器上的点播频率高的情况，此时如果依然替换加权周期点播频率最小的节目，那么就会出现节目频繁的换入换出的现象，即所谓的节目抖动。出现这种情况则系统的资源就会被磁盘 Cache 过程占据，导致系统服务效率的下降。要防止这种现象的出现就要在 Cache 替换过程中增加对最小加权周期点播频率和要 Cache 的节目的点播频率的比较，如果前者更大则不能替换。同时这种情况说明系统的视频服务器数量已经难以胜任当前的并发用户数量，系统应该提示管理员应该增加视频服务器的数量扩展系统的吞吐量。

最小加权周期频率替换算法。

(1)使用计算公式计算磁盘中除系统规定不能替换的节目外所有节目的加权周期点播频率，如果是周期 Cache 使用公式(6)和公式(7)，如果是动态调度则使用公式(8)和公式(9)，计

算时对值为 1 的节目不进行计算;

(2)所有计算结果按加权周期点播频率从小到大排序;

(3)检查最小加权周期点播频率和需要 Cache 的节目的点播频率的大小关系,如果前者更小则转(4),否则替换结束;

(4)选择替换加权点播频率最小的节目,如果该节目存在多个视频服务器,则随机选择一个,删除选中的视频服务器上该节目;

(5)检查选中的视频服务器是否有足够的空间,如果有转(6),如果没有转(7);

(6)节目拷贝到选中的视频服务器,转(8);

(7)删除选中的视频服务器上加权周期点播频率次小的节目,转(5);

(8)替换结束。

5 实验与分析

为了验证本文改进的 LFRU 算法和最小加权周期频率替换算法的在分级存储系统中的性能,本文在视频点播系统中实现了 LFU、LRU 算法和 LFRU 算法,与改进的 LFRU 算法和最小加权周期频率替换算法程序一起进行对比实验。

实验主要硬件环境:浪潮英信服务器 NC2000(视频服务器),IBM 磁盘阵列机,Dell 小型机以及千兆有线局域网。此实验在视频点播系统中存储了 50 个节目,将其中 10 个节目看成热门节目,20 个看成冷门节目,剩余的 20 个看成不冷不热的节目。随机拷贝一些节目到视频服务器磁盘,规定视频服务器的磁盘最大空间为 3 G。实验中采用动态 Cache 策略,然后统计动态 Cache 策略中使用的各种替换算法的命中率(用户请求在视频服务器命中的比率:被调度到视频服务器的用户请求数/总的用户请求数)。

首先比较改进的 LFRU 算法与 LFU、LRU 算法的性能。用一个 240 个数组成的循环队列模拟用户请求,每个数表示请求的节目 ID,队列的请求在频率上服从 zipf 法则。每隔 15 s 从队列取一个数作为用户请求,队列用完后,数组的数整体向前平移模拟访问概率分布的变化以及节目访问概率的变化。对这样一个请求模型,分别使用改进的 LFRU 算法与 LFU、LRU、LFRU 算法作为动态磁盘 Cache 的替换策略进行实验,对实验结果进行统计得到 4 种算法的命中率分布如图 3 所示。图中右边从上到下四条线分别是改进的 LFRU 算法、LFRU 算法、LFU 算法、LRU 算法的请求命中率折线,四条线显示改进的 LFRU 算法命中率比 LFRU 要好一些。在算法的初期,改进的 LFRU 算法优于 LRU 算法,在算法的末期,改进的 LFRU 算法优于 LFU 算法,这和理论分析是一致的。此对比实验证明本文改进的 LFRU 算法的效果相对较好。

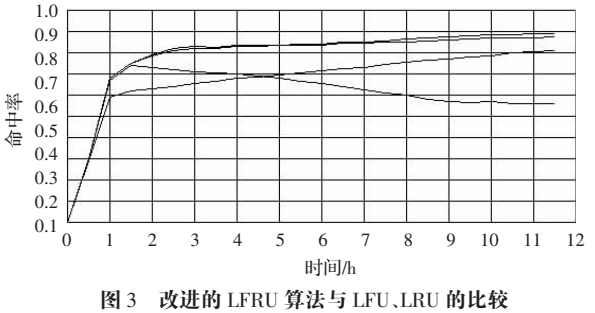

图 3 改进的 LFRU 算法与 LFU、LRU 的比较

要检测最小加权周期频率数据迁移的性能,需要考察访问模式变化时算法对陈旧数据的替换效果,并与 LFU 算法进行比较。用两个 240 个数组成的循环队列模拟用户请求(两个队列节目的概率有着巨大的差别),每个数表示请求的节目 ID,每个队列的请求在频率上服从 zipf 法则。每隔 15 s 从队列取一个数作为用户请求,队列用完后,随机从队列任一位置开始,如此重复 12 次,然后换另一个队列,也做 12 遍。通过对使用这两种数据迁移算法的实验结果进行统计,其的命中率分布如图 4,上下两条线分别是最小加权周期频率替换算法、LFU 算法的请求命中率折线。从图中实验数据分析可得:最小加权周期频率替换算法能较好地适应节目访问模式的变化,克服数据的陈旧问题。而 LFU 算法对节目访问概率的变化比较敏感,命中率在访问模式变化时波动比较大,要隔较长时间才能缓和。因此本文提出的最小加权周期频率替换算法性能相对较优。

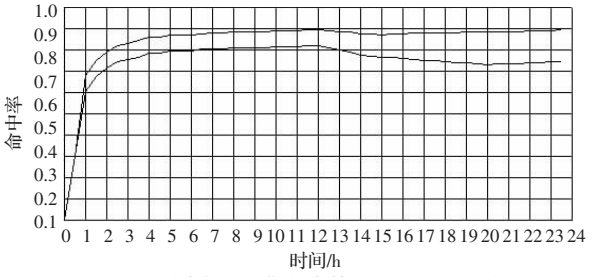

图 4 最小加权周期频率算法与 LFU 的比较

6 结语

针对分布集群体系结构的视频点播系统,本文提出两种节目替换算法。在系统的运行开始阶段点播请求不稳定,此时用改进的 LFRU 算法进行视频服务器的节目替换。系统的运行到达稳定状态后,改用最小加权周期频率替换算法。对比实验表明所提出算法较适合分布式集群结构的 VoD 系统,替换效率相对较高。(收稿日期:2007 年 8 月)

参考文献:

[1] 刘志明.并行 VOD 系统中数据迁移和存储技术研究[D].长沙:国防科技大学,2003.

[2] Li Yong,Peng Yu-xing,Chen Fu-jie.Disk cache replacement algorithms for large-scale video on demand system[J].Journal of Computer Academe and Development,2000,37(2):207-212.

[3] Wang J.BECP:the scheduling policy based on best-effort in clustered VOD servers [C]//11th International Conference on Parallel and Distributed Systems-Workshops(ICPADS'05),Fukuoka,Japan,2005:534-539.

[4] Barlas G,Veeravalli B.Optimized distributed delivery of continuous-media documents over unreliable communication links[J].IEEE Transactions on Parallel and Distributed Systems,2005,16(10):982-994.

[5] 罗时爱.视频点播系统体系结构和磁盘 Cache 策略的研究与实现[D].南京:南京理工大学,2006.