# 大规模连续媒体服务的缓存替换算法设计与实现

张 潇,吴敏强,恽 爽,陆桑璐,谢 立
(南京大学计算机软件新技术国家重点实验室,南京大学计算机科学与技术系,江苏南京 210093)

**摘 要**: 连续媒体的缓存设计是非常关键的问题,本文针对大规模连续媒体服务系统的特点,提出了 EA 缓存替换算法. 该算法充分考虑了现有用户和请求接入用户的服务需求,提高了内存使用效率. 我们的理论分析和实验模拟证明了它在性能上大大优于传统的缓存替换算法.

**关键词**: 连续媒体服务; 缓存设计; 替换算法
**中图分类号**: TP302 **文献标识码**: A **文章编号**: 0372-2112 (2003) 05-0783-03

# Caching Design for Large Scale Continuous Media Service

ZHANG Xiao, WU Min-qiang, YUN Shuang, LU Sang-lu, XIE Li
(*State Key Laboratory for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nangjing, Jiangsu* 210093, *China*)

Abstract: Continuous media caching design is a very critical problem. A caching strategy called EA caching algorithm was presented. This algorithm considers the requirements of serving existing and anticipated demands so as to improve the utilization of memory. The theoretical analysis and experimental simulation have proved that the performance of EA algorithm is far better than the traditional caching algorithm.

Key words: scontinuous media service; caching design; replacement algorithm

## 1 引言

如何实现对连续媒体服务的支持,减少读取连续媒体数据的磁盘 I/O,是连续媒体服务系统的一个关键问题. 缓存技术可以很显著的提高系统的性能,文献[1]认为在大规模服务下,缓存技术的使用可以提高服务器 20 %以上的性能.

连续媒体服务具有以下特点: (1) 通常的连续媒体文件(以下简称为 CM 文件) 都达到了 G 级的存储大小,读取一个 CM 文件的时间可能会长达 1 至 2 小时,这种现象导致用户服务会长时间占用硬盘带宽; (2) 通常对 CM 文件的操作仅限于读操作(read-only); (3) 数据的用户访问行为是不可再现的,也就是说,在一般情况下,被用户请求访问过的数据将不再被该用户访问.

本文将针对连续媒体的特点,提出适合连续媒体服务要求的缓存替换算法,以发挥缓存保证连续媒体数据读取连续性与一致性的作用,达到减少磁盘 I/O 的目的.

## 2 EA( Existing and Anticipated Caching) 替换算法

大规模连续媒体服务的缓存替换算法的设计有两个关键问题: 一是如何量度即将被未来用户访问到的数据块,并做出相应的算法调整; 二是如何量度即将被现有用户访问到的数据块. 传统连续媒体替换算法如 Interval、basic 并未很好的解决以上两点. 提出了同时考虑现有用户和即将接入用户请求的 EA 缓存替换算法.

### 2.1 系统模型

通过对现实统计中点播率的分析,表明用户请求在多个 CM 文件之间的分布一般是服从 Zipf 分布的.

使用 Poisson 分布来模拟用户请求这一过程[5],于是用户的平均到达时间为 $1/\lambda$. 则对第 $i$ 个 CM 文件的访问请求的平均到达时间为 $1/\lambda f_i$.

用户请求是以恒定速率读取数据(CBR),令 $d$ 代表每一个数据块的大小, $s$ 代表用户读取数据块的速率.

### 2.2 算法描述

2.2.1 **对数据块的描述** ——NEXT 与 PIT 首先,我们为数据块定义一个数据量 *NEXT*. 设 $A$、$B$、$C$ 用户请求正在读取的数据块分别为 x、y、z. 定义 $NEXT(x)$ 值为数据请求 $A$ 所读取的数据块 $x$ 与读取同一 CM 文件,时间上紧接 $A$ 的数据请求 $B$ 读取的数据块 $y$ 的距离(以数据块为单位). 如果 $C$ 其后没有读取同一 CM 文件的数据请求时,则数据块 $z$ 的 $NEXT(z)$ 值为正无穷大. 如图 1, $A$ 与 $B$ 距离为 3 个数据单元,那么我们就定义 $NEXT(x) = 3$,也就是说,数据请

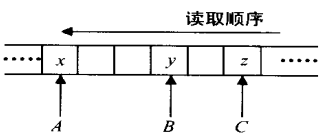

图 1

收稿日期:2002-07-31; 修回日期:2003-01-02
基金项目:国家重点基础研究发展规划(973 计划) (No. 2002CB312002);国家 863 项目:应用服务器的运行,管理与调度技术(No. 2001AA113050)

求 $A$ 所读取的数据块 $x$ 的一个测度为 3.

现在定义另外一个描述数据块状态的数据量 $PIT$. 它表示 CM 文件数据块的预测用户访问请求平均到达时间,那么 CM 文件 $i$ 的第 $n$ 块数据块的 $PIT$ 值为 $1/\lambda f_i + nd/s$.

综合以上两个描述数据块的属性 $PIT$ 和 $NEXT$,得到数据块 $B$ 的平均访问到达时间为: $IT(B) = \min\{PIT(B), NEXT(B) * d/s\}$.

2.2.2 **算法操作过程** 系统采用 Server-Pushing 的方式. 在一个服务周期结束后若干个数据块将会被读取完毕,设这些数据块的个数为 $n$. 将这些数据块按其 $IT$ 值升序压入栈中(以栈的方式将这些数据块管理起来). 我们称这个栈为 RBS,它负责存放那些可能将会被替换的数据块.

在需要发生数据块替换的时候,在缓存中被替换的数据块就是位于 RBS 栈顶的数据块. 也就是说最后进入 RBS 的数据块最先被替换出去. 换个说法,拥有大的 $IT$ 的数据块比拥有小 $IT$ 的数据块更容易被替换.

2.3 **对算法的讨论**

该算法利用了 MRU 的思想,不同的是对最近使用的若干块按照连续媒体的特点进行了排序,而这种排序是有利于提高系统的命中率的.

设 $N$ 为现有用户请求数目,计算数据块的 $PIT$ 可以通过预计算完成,计算每一个用户请求的数据块的 $NEXT$ 至少需要 $N\log N$ 次比较,生成 RBS 的运算复杂度至少是 $N\log N$.

## 3 模拟实验及结果分析

我们模拟测试了系统采用 LRU,MRU,未带预测算法(采用传统流媒体算法思想),带预测算法 EA 四种不同缓存替换算法情况下产生的命中率变化状况,并讨论相应参数对算法的影响.

在实验中,用户请求随机到达. 同时,如前所述,用户请求的平均到达的间隔时间是服从指数分布的. CM 文件的选择是服从参数为 $\theta$ 的 Zipf 分布($f_i = c/i^{1-\theta}$)的($\theta = 0.271$, $c$ 为常数),以 MPEG-1 标准压缩,播放速率为 1.5Mbps. 在所有实验中的命中率是在多次试验后得到的平均结果,每次实验持续 90000 秒,采用冷启动方式. 我们把用户要访问的连续媒体数据的集合称为访问媒体集.

图 2 说明 CM 文件长度对各算法产生的命中率的影响. 该实验的 CM 文件数为 5,数据块大小为 192K 字节,缓存大小为 192M 字节,即缓存一次能容纳 1024 个数据块,用户请求平均到达时间为 400 秒. 可以看到随着 CM 文件长度增大,用户访问的媒体集增大,四种算法的缓存命中率总体降低. 当 CM 文件为一部通常的影片,长度约为 108 分钟 1.19G 字节时,EA 的命中率为普通流媒体算法的 2.15 倍,MRU 的 6.41 倍和 LRU 的 40.6 倍. 在 CM 文件长度由 1500 向 500 变化的时候,缓存命中率有大幅度的提高,这是由于用户访问媒体集数据量急剧减小的缘故.

图 3 表明随着缓存的增大,在不同替换算法下缓存命中率的变化情况. 该实验的 CM 文件数为 5,用户请求平均到达时间为 400 秒,数据块大小为 192k 字节,CM 文件长度约为 108 分钟 1.19G 字节. 图中 EA 算法产生的命中率始终优于其他三种算法,它比未带预测普通流媒体替换算法命中率始终要高出一倍左右. 由图中 LRU 和 MRU 算法产生的命中率随着缓存增加并没有较大的提高,而针对流媒体提出的缓存替换算法表现出了较强的优势. 这正是与设计的初衷相吻合.

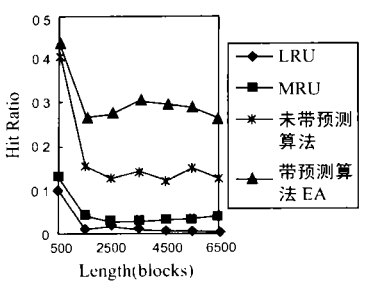


图 2 CM 文件长度的影响

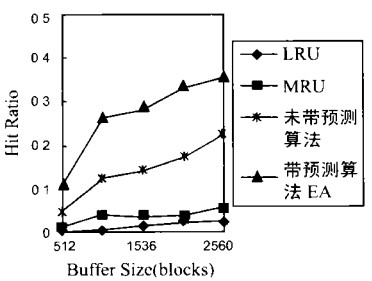


图 3 缓存大小的影响

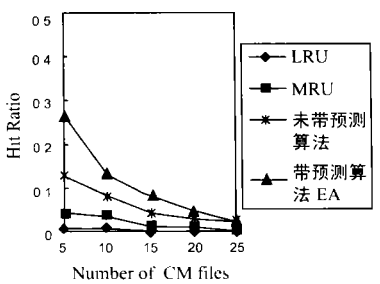


图 4 CM 文件数的影响

图 4 说明随着 CM 文件数的增加,在不同替换算法下缓存命中率的变化情况. 该实验 CM 文件长度约为 108 分钟 1.19G个字节,数据块为 192k 字节,缓存为 192M 字节,即缓存能容纳 1024 个数据块,用户请求平均到达时间为 400 秒. 从图中可以看到,四种算法的命中率都随访问媒体集的增大而总体减少. 在 CM 文件数从 5 个增加到 25 个的过程中,EA 算法产生的命中率始终比其他三种算法要高,而且在 CM 文件数相对较少时,EA 算法体现的优势较为明显.

图 5 表明随着用户请求平均到达时间增加,在不同替换算法下缓存命中率的变化情况. 该实验的 CM 文件数为 5,CM 文件长度约为 108 分钟 1.19G 字节,数据块为 192k 字节,缓存为192M字节. 可以看到图中EA算法的缓存命中率随用户平均到达时间的减小而显著增加,但是一般的未带预测的流媒体替换算法随着用户平均到达时间的减小(即系统所支持的并发流增加)并未有明显变化. 在用户平均到达时间为 200 秒的情况下,EA 算法产生的命中率分别是普通流媒体替换算法的 3.82 倍,MRU 算法

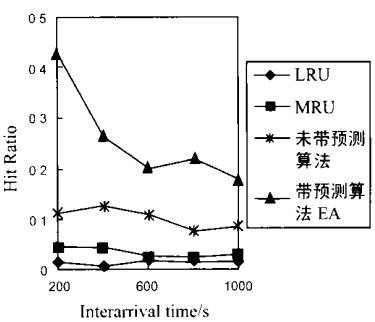


图 5 平均到达时间的影响

的 10.26 倍,LRU 算法的 26.9 倍. 在用户平均到达时间由 400 秒向 200 秒的跃变过程中,EA 算法的命中率提高了 62 %. 而普通流媒体算法的命中率却还略有下降. 这是由于 LRU,MRU 与普通流媒体算法没有考虑未接入用户的数据服务请求,仅仅对现有用户进行考虑,在数据读取速率一定的情况下,该三种算法对系统承受的并发流数量的变化不敏感,造成缓存命中率无明显变化.

图 6 说明随着读取 CM 文件速度的增加,在不同替换算法下缓存命中率的变化情况. 该实验的 CM 文件数为 5,CM 文件长度约为 108 分钟 1.19G字节,数据块为 192k 字节,缓存为 192M 字节,用户请求平均到达时间为 400 秒. 和上一个减小用户平均到达时间的实验相比较,值得注意的是,该实验减小用户请求的读取速率同样意味着增加了系统支持的并发流的数量,但是 EA 算法的命中率增加并未有较为明显的变化,读取速度在 128Kbps 时的命中率比读取速度在 1.5Mbps 时的命中率仅仅增加了约 19 %. 而在上一个实验中,EA 算法在 200 秒的用户平均到达时间的命中率比在 1000 秒的用户平均到达时间的命中率提高了约 139.8 %. 我们认为产生这种现象的原因是:上一实验减小用户平均到达时间意味着增加了系统支持的并发流的数量,但是它主要是未来用户请求频率的增加导致的,而该实验中系统支持的并发流的数量的增加,主要是因为现有系统中服务用户的数据读取速度减慢造成的. 在本次实验中 EA 算法产生的命中率随着用户数据读取速率的减小增加并不是十分的显著. 普通的未带预测的流媒体替换算法命中率变化比较明显.

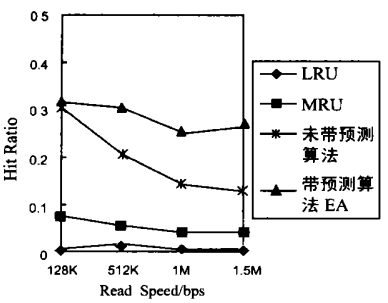


图 6　读取速度的影响

在各种参数变化的情况下,EA 算法产生的命中率最高,都优于其他几种策略,并且当可用的内存与访问媒体集成适当比例时,有很高的能耗比. 在用户请求达到一定规模的情况下,它能大大提高普通未带预测流媒体替换算法的缓存命中率. 从实验数据分析可以得到结论,EA 是一种适合大规模连续媒体服务系统的缓存策略.

## 4　结论

我们在项目中协作缓存管理部分实现了 EA 缓存替换算法,在实际运行环境中,该算法体现了上述的优势.

缓存设计是整个连续媒体服务系统的关键技术之一. 本文针对连续媒体服务的特点,提出了 EA 缓存替换算法. 这种算法比传统的缓存替换算法更加适合大规模连续媒体服务系统的服务要求,提高缓存的命中率,减少了硬盘的输入输出. 由于该算法是针对大规模的连续媒体服务这一特定情形,所以如何提高预测的精确程度以及算法的通用程度将是今后的进一步的工作重点;另外,如何减小算法的时间复杂度,也需要进一步的研究.

**参考文献**:

[1] A Dan, D Sitaram. Buffering and caching in large-scale video servers [A]. Proc. of IEEE CompCon [C]. March 1995. 217 - 224.

[2] D Lee, J Choi, J Kim, S H Noh, S L Min, Y Cho, C S Kim. LRFU Replacement Policy: A spectrum of block replacement policies [R]. Seoul National University Technical Report SNU-CE-AN-96-004, March 1996.

[3] B Ozden, R Rastogi, A Silberschats. Buffer replacement algorithm for multimedia storage systems [A]. Proc. International Conf. on Multimedia Computing and Systems [C]. Hiroshima, Japan, June 1996. 580 - 589.

[4] A Dan, D Sitaram. A generalized interval caching policy for mixed interactive and long video enviroments [A]. Proc. of Multimedia Computing and Networking Conference [C]. San Jose, CA, Jan. 1996.

[5] Derek Eager, Mary Vernon, John Zahorjan. Minimizing bandwidth requirements for on-demand data delivery [J]. IEEE Transactions on Knowledge and Data Engineering, 2001.

[6] P Venkat Rangan, Harrick M Vin. Efficient storage techniques for digital continuous multimedia [J]. IEEE Transaction on Knowledge and Data Engineering, 1993, 15(4).

[7] P Sumari, M Merabti, R Pereira. Video-on-demand server: Strategies for improving performance [A]. IEEE Proc-Softw [D]. Feb 1999, 146(1).

**作者简介**:

**张　潇**　男,1980 年 3 月生于四川安岳,硕士研究生,主要研究方向为分布式计算.

**吴敏强**　男,1977 年 8 月生于江苏吴县,硕士研究生,主要研究方向为分布式计算.