文章编号:1671-8860(2013)12-1504-05　　文献标志码:A

云存储中面向访问任务的小文件合并与预取策略

王　涛[1]　姚世红[1]　徐正全[1]　熊　炼[1]

(1　武汉大学测绘遥感信息工程国家重点实验室,武汉市珞喻路 129 号,430079)

摘　要:针对云存储中通用分布式文件系统的小文件问题,改进概率潜语义分析(PLSA)模型,提出了一种面向用户访问任务的小文件合并与预取策略。该策略分析用户的访问任务、系统应用和访问文件之间的关系,根据任务合并小文件,并基于任务的转移概率预取文件。对建立的效率模型的分析和基于 HDFS 的数字城市原型系统实验结果都表明,此策略有较高的预取命中率,可以有效减少元数据服务器的负载和用户请求响应时延。

关键词:分布式文件系统;概率潜语义分析;小文件;访问任务;合并与预取

中图法分类号:P208

云存储是云计算发展起来的一种新兴服务模式[1],通过集群、分布式文件系统,将网络中海量的异构存储设备集合起来协同对外提供数据存储和访问服务。目前,主流的分布式文件系统 GFS[2]、HDFS、PVFS 及 Lustre,都是基于优化大文件流数据访问模式设计的,忽略了小文件的存储和访问。研究发现[3],对小文件的请求占所有请求的 90% 以上,却只请求不到 10% 的 I/O 数据。小文件问题已经成为制约分布式文件系统性能的一个重要因素。目前的研究主要集中在两个方面:① 改进分布式文件系统的架构设计[4-7]。但是,改进架构非常复杂,成本高而且较难实现。② 合并小文件并采用缓存与预取机制[8-11]。但是,现有的合并策略大多在存储文件时没有考虑文件之间的相关性。文献[12]将小文件分成结构相关、逻辑相关和独立文件,针对具体应用分别对结构相关和逻辑相关小文件采用合并预取策略,但是对独立文件则没有采取任何措施。

目前,多数云公司主要面向公众提供云服务,公众的访问行为必然对云系统产生重要的影响。现有对小文件问题的研究,都只是面向文件以及文件系统本身,忽视了用户的访问行为。因此,本文从用户访问行为的角度出发,面向所有文件(包括独立文件),通过改进概率潜语义分析(probability latent semantic analysis,PLSA)模型[13]挖掘云系统中用户访问行为的规律,分析文件与用户访问任务之间的相关性,提出基于用户访问任务的合并与预取模型,实现对小文件的合并与预取,以改善云系统的小文件问题。

1　基于用户访问任务的合并与预取模型

在公共云系统包括数字城市等系统中,用户请求常具有一定倾向性和目的性,为完成某一目的而访问一系列文件,称之为一个访问任务。系统所提供的应用和服务是相对有限和相对固定的,用户为完成某一任务而访问应用,应用程序将底层对应的文件反馈给用户,使用户对文件的访问存在一定的特定模式。系统中存储的文件并非完全独立的,从用户的角度来看,访问的文件之间存在着某种统计相关。在数字城市系统中,用户访问了某地近一周的温度数据,将其视为一个任务,接着用户又访问了一系列的地图切片数据,也将其视为一个任务,用户的访问就从一个任务跳转到了另一个任务。将访问行为映射到文件的层面上,则视为访问的文件之间具有相关性。同一个任务下的文件的相关性较强,不同任务间访问的文件的相关性则弱很多,但任务之间存在着某种相关联系。如果仅从文件的层面考虑,强相关和弱相关的文件混在一起,很难统计出用户的访问规律。而且,用户常常访问一系列似乎不相关

收稿日期:2013-10-25。

项目来源:国家 973 计划资助项目(2011CB302306);国家自然科学基金资助项目(41271398)。

或者弱相关的文件,但是这些文件也是一个访问任务。此外,直接考虑对文件的访问往往还具有不确定性。因此,很有必要从访问任务的角度来改善小文件问题。本文从用户的访问任务出发,尽可能挖掘出所有基于用户访问任务的访问模式,将研究目标从文件层面映射到任务层面,为文件合并提供理论依据,同时,通过统计预测用户下一步的访问行为,引导后面的预取工作。

概率潜语义分析(PLSA)已成功应用于文本学习和挖掘[13],Web 挖掘[14-15]和图像分类[16]以及相关主题的挖掘研究。用户访问任务直观上难以描述,本文利用 PLSA 模型,将用户的请求视为文档,挖掘其中的主题(即用户的访问任务),也就是说,用户的请求就是访问任务的集合。因此,PLSA 可以根据任务对用户请求中访问的文件进行聚类合并。

假设现有的观测样本足够分析出所有任务访问模式,假定一组用户请求 $Q=\{q_1,q_2,\cdots,q_m\}$,系统满足用户请求而访问的一系列文件 $F=\{f_1,f_2,\cdots,f_n\}$,则对文件的访问表示为一个 $m\times n$ 的矩阵 $\boldsymbol{\omega}=[\omega_{ij}]_{m\times n}$,$\omega_{ij}$ 表示文件 f_j 出现在请求 q_i 中的权值,其值为 1 或 0,表示是否存在。采用隐变量空间 $Z=\{z_1,z_2,\cdots,z_l\}$ 表示用户任务,将用户对文件的访问映射到任务空间,就可以获取用户的访问倾向,则用户请求文件的联合概率分布为:

$$P(f_j|q_i)=\sum_{k=1}^{l}\cdot P(f_j|z_k)P(z_k|q_i) \quad (1)$$

$P(f_j|z_k)$ 是任务 z_k 的分布,每个用户请求由任务 z_k 的集合 $P(z_k|q_i)$ 表示,可以定量表示用户请求、访问任务和文件之间的关系。要得到这个分布,需根据现有观测样本发现所有可能的 z_k。

在公共云系统中,文件数量过于庞大,直接估计每个文件概率是不现实的。用户访问文件,系统是以具体应用反馈给用户,且应用的访问具有时序性。采用 $A=\{a_1,a_2,\cdots,a_\theta\}$ 表示系统中的应用,则请求表示为一个应用序列 $q_i=\{a_1^1,a_3^2,\cdots,a_\theta^t\}$,其中,任一应用由文件序列表示,确定 $P(f_j|z_k)$ 的问题就变成了确定 $P(a_u|z_k)$ 的问题,$u\in[1,\theta]$,系统的应用是相对固定和相对有限的,简化了模型,使计算量大大降低。利用 EM 算法[13]估计出 $P(a_u|z_k)$ 和 $P(z_k|q_i)$ 及 $P(z_k)$,任务 z_k 通过训练观测样本得到。设定阈值 μ,当 $P(a_u|z_k)\geqslant\mu$ 时,则认为 a_u 是完成任务 z_k 的主要应用,可以得到完成每个任务 z_k 相应的应用集,因而可以得到完成每个任务 z_k 的文件集。

根据贝叶斯准则,用户访问一个应用,则用户在完成某一任务的概率为:

$$P(z_k|a_u)=\frac{P(a_u|z_k)P(z_k)}{\sum_{z_k\in Z}P(a_u|z_k)P(z_k)} \quad (2)$$

在某一特定任务下,计算一个请求中每个任务的先验概率,这些概率反应用户完成任务的倾向。假设用户请求中的应用序列包含 τ 个应用,则有:

$$P(z|a_u,u\in\{1,\cdots,\tau\})=\frac{\prod_{u=1}^{\tau}P(a_u|z)P(z)}{\prod_{u=1}^{\tau}P(a_u)} \quad (3)$$

得到访问这 τ 个应用时可能包含的每个任务的概率。设定阈值 η,当 $P(z_k|a_u)\geqslant\eta$ 时,则确定为请求的主要任务。请求用任务序列来表示,并且可以统计得到热任务和热应用。

于是,根据不同应用将对应的文件组成文件集,再根据完成任务 z_k 所需的应用集合并对应文件集。分布式文件系统中文件的存储位置是随机的,用户访问的文件可能存储在不同数据块,甚至不同输入输出服务器(IOS)中。将属于同一个任务的文件尽可能合并成一个数据块,若超过默认的数据块大小,合并在同一 IOS 相邻几个数据块中。而且,用户请求往往包含多个访问任务,用户请求可以表示为任务序列,统计出系统中用户完成一个访问任务、执行下一个任务的概率,即任务的转移概率 $P(z_k|z_v)$。若当前访问任务为 z_v,则将转移概率 $P(z_k|z_v)$ 最高的任务作为预取目标,将其对应数据块预取至缓存,降低请求响应时延,改善系统存储效率和访问效率。

2　系统效率模型

本文分析几种常用分布式文件系统的工作原理,建立了元数据服务器(MDS)负载模型和请求时间响应模型。目前,大多数分布式文件系统不提供预取功能,本文将用户当前的访问任务下一步访问概率最高的任务作为预取目标,将其对应的数据块预取到缓存中,改善系统的访问效率。因为元数据非常小,所以不考虑元数据预取。

2.1　MDS 负载

MDS 负载过重是导致分布式文件系统性能下降的重要原因之一。在 MDS 中,文件的元数据和数据块的元数据都会进行备份,所以,设备份一个文件的元数据消耗的内存为 m_d,备份一个数

据块的元数据消耗的内存为 m_b，当没有文件存在时，MDS 本身消耗的内存为 m_0，一个数据块的块映射消耗的内存为 m_{map}。假设分布式文件系统中存储 N 个文件，存储在 R 个数据块中（最坏的情况 $R=N$），则 MDS 消耗的内存为：

$$M_{\text{MDS}} = m_d \cdot N + (m_b + m_{\text{map}}) \cdot R + m_0 \quad (4)$$

用户的访问包含 Y 个主要任务，理想情况是文件合并成 Y 个大文件，则 MDS 负载变为：

$$M_{\text{MDS}} = (m_d + m_b + m_{\text{map}}) \cdot Y + m_0 \quad (5)$$

则 MDS 消耗的内存仅与合并的文件数 Y 有关，Y 远小于 N，MDS 消耗的内存明显减少。

2.2 预取命中率

预取命中率是影响整体 I/O 性能的一个关键因素，严重影响着请求响应时间和访问效率。预取的文件越多，命中用户未来请求的文件的概率就越大，但过度预取会造成无效预取，浪费系统资源。假设预取的文件数为 E，预取的文件中用户访问的文件数为 H，预取命中率 r_{hit} 为 $r_{\text{hit}}=H/E$。在尽可能更多地命中用户请求的情况下，使预取的文件数 E 不至于过大。

2.3 请求响应时延

从分布式文件系统中读取一个文件所需要的时间主要包括：客户端向 MDS 发送一个请求的时间，记为 T_{CM}；MDS 在内存中查询请求文件元数据的时间，记为 T_{meta}；MDS 将元数据返回给客户端的时间，记为 T_{MC}；客户端发送读取请求到相关 IOS 服务器的时间，记为 T_{CIO}；IOS 服务器读取请求的数据块的时间，记为 T_{read}；数据块到达客户端的时间，记为 T_{Block}。T_{CM} 和 T_{CIO} 是发送请求消耗的时间，一般认为是常数。元数据非常小，所以 T_{MC} 也是常数。T_{Block} 主要与网络传输速度有关。请求 N 个文件的总响应时延为：

$$T = (T_{\text{CM}} + T_{\text{MC}} + T_{\text{CIO}}) \cdot N + \sum_{i=1}^{N} T_{\text{meta}_i} + \sum_{i=1}^{R} T_{\text{read}_i} + R \cdot T_{\text{Block}} \quad (6)$$

R 是读取的数据块的数量（最坏的情况 $R=N$）。用户访问具有一定时序性，用户请求表示为一个文件序列，因此，合并文件的第一个小文件被访问时，随即查询合并文件元数据，减少与 MDS 交互次数。假设预取 E 个文件，命中 H，直接从缓存获取，则请求响应时延为：

$$T = (T_{\text{CM}} + T_{\text{MC}}) \cdot Y + (N - H) T_{\text{CIO}} + \sum_{i=1}^{Y} T_{\text{meta}_i} + \sum_{i=1}^{Y'} T_{\text{read}_i} + (Y - Y') \cdot T_{\text{Block}} \quad (7)$$

Y' 是未命中文件的块数。响应时延大大减小，有效改善了访问效率。

一个需要注意的问题就是预取数据块的放置位置，一般有两种：客户端缓存和 IOS 端缓存。第一种无需从 IOS 端发送给用户请求的文件，可以减小网络时延，但可能存在很多无效预取，增加网络传输负担。第二种即使预取命中仍需 IOS 端给用户发送，不会增加网络传输负担，也不会减小网络时延。本文选择第一种方式来进一步减少响应时延。

3 实验评估

HDFS 是目前最受欢迎的分布式文件系统，由一个名节点和若干数据节点构成。名节点也就是 MDS 元数据服务器，数据节点就是 IOS 服务器。本文搭建了一个基于 HDFS 面向“数字城市”[17]的存储访问平台。

3.1 实验环境

实验平台由 5 节点服务器组成，1 台 Dell PowerEdge T410 服务器作为名节点（即 MDS 元数据服务器），配置为 2A* Intel Xeon E5606 CPU(2.13 GHz)，内存 16 GB DDR3，硬盘 SATA 1TB；4 台 Dell OptiPlex 380 MT 服务器作为数据节点（即 IOS 服务器），每台配置均为 Intel 奔腾双核 E5800 CPU(3.2 GHz)，内存 8 GB DDR3，硬盘 SATA 2 TB。网络环境为千兆以太网。每个节点的操作系统均为 Ubuntu Server 9.04，Hadoop 版本为 0.20.2，JDK 版本 1.6.0。

实验数据主要是面向“数字城市”采集的数据文件，主要包括视频、环境感知、地图切片、水文及图片文本等几十至上百种数据文件，且大多数都是小文件（多数不超过 4 MB）。例如，环境传感器一天采集的数据生成一个文件，大小一般为5.2～14.8 kB，10 000 个传感器在 2012-01～2012-07 采集的数据文件大约为 14 650 000 个，但总大小只有约 99.6 GB。又如地图切片文件，一幅武汉市的栅格地图经过切片之后有 530 810 个切片文件，但总大小仅仅为 1.15 GB，每个切片文件的大小为 2～9 kB。图 1 为不同文件大小的分布图。

3.2 结果与分析

实验项目包括验证预取命中率、MDS 服务器负载和请求响应时延实验。每个实验重复 10 次，排除最大值和最小值后取平均值，尽量得到接近实际的结果。从图 2 中可知，本文策略经过训练预取命中率升高并稳定在 75%以上。分别利用本文策略与原始 HDFS、HAR[13]，比较相应的 MDS 服务

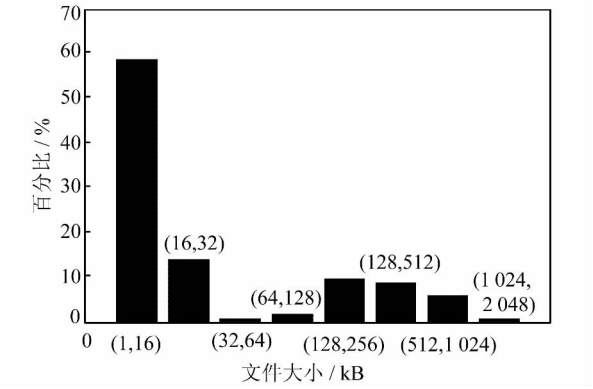

图 1　实验中文件大小的分布

Fig. 1　Distribution of File Sizes

器负载和请求响应时延。当系统分别存储 50 000、100 000、150 000、200 000、250 000、300 000 **个小文件时,采用原始** HDFS、HAR **和本文方法的** MDS **负载情况如图** 3 **所示。可以看出,**HAR **和本文方法在降低** MDS **负载方面都要明显优于原始的** HDFS。**本文方法根据用户访问规律合并文件,比** HAR **的合并策略更加优越。分别读取** 10 000、15 000、20 000、25 000、30 000 **个小文件,原始** HDFS、HAR **和本文方法的平均响应时延如图** 4 **所示。可以看出,本文方法的平均请求响应时延最少,随着请求文件数量的增加,时延增势最为缓慢。**

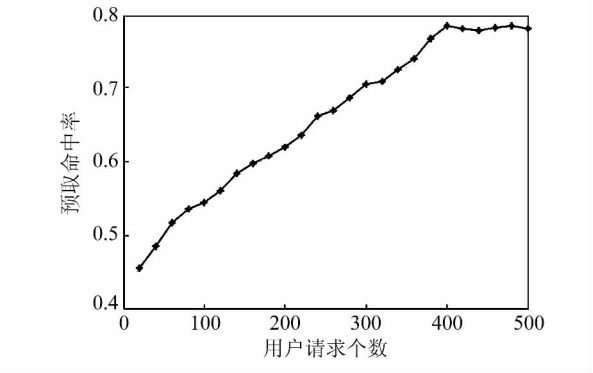

图 2　本文策略的预取命中率

Fig. 2　Prefetching Hit Rate of the Proposed Strategy

4　结　语

本文针对云存储中的小文件问题,首次从用户访问任务的角度出发,改进 PLSA **模型分析访问文件,系统应用和访问任务之间的相关性,提出了基于用户访问任务的合并与预取模型,尽可能地将属于同一任务的小文件合并,并统计用户的访问任务和任务间的转移概率,选择预取集来减轻** MDS **负载,减少访问的请求响应时延,改善云存储系统的访问效率。将此模型应用在基于**

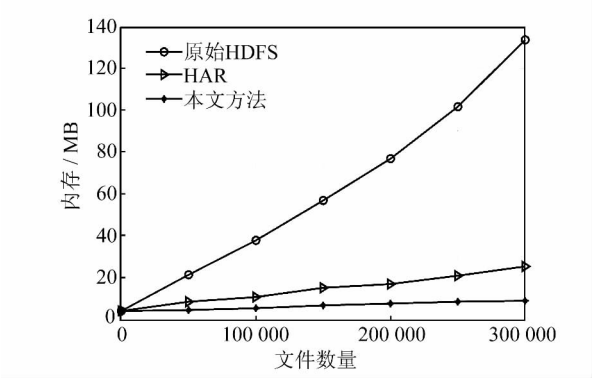

图 3　MDS 服务器的负载比较

Fig. 3　Comparison of Original HDFS, HAR and the Proposed Strategy in MDS Workload

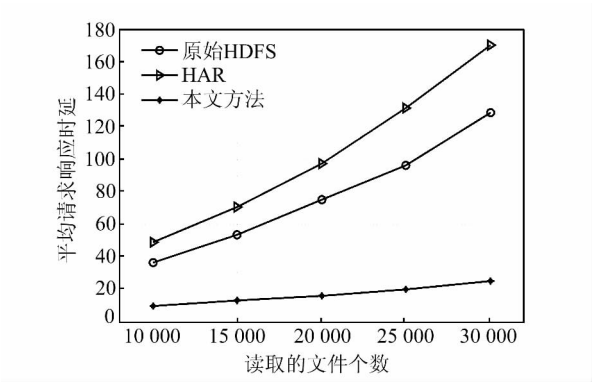

图 4　平均请求响应时延

Fig. 4　Comparison of Original HDFS, HAR and the Proposed Strategy in Request Response Time

HDFS 的面向"数字城市"的存储访问平台上,结果表明可以有效地减少 MDS **负载和请求响应时延。**

参　考　文　献

[1]　**李德毅. 云计算技术发展报告**[M]. **北京:科学出版社**, 2011

[2]　Sanjay G, Howard G, Shun-Tak L. The Google File System[C]. The 19th ACM Symposium on Operating Systems Principles, Lake George, N Y, USA, 2003

[3]　Adaptive Trade off in Metadata-Based Small File Optimizations for a Cluster Files System[J]. International Journal of Numerical Analysis and Modeling, 2012, 9(2): 289-303

[4]　Zhang Zhihui, Ghose K. hFS: A Hybrid File System Prototype for Improving Small File and Metadata Performance[C]. SIGOPS/EuroSys European Conference on Computer Systems, New York, USA, 2007

[5]　Ma Can, Meng Dan, Xiong Jin. Dawning Nebula Distributed File System HVFS: For Large Scale Small File Access[J]. Journal of Chinese Computer

Systems, 2012, 33(7): 1 481-1 488

[6] 赵跃龙,谢晓玲,蔡咏才,等. 一种性能优化的小文件存储访问策略的研究[J]. 计算机研究与发展, 2012, 49(7):1 579-1 586

[7] Zhang Qifei, Pan Xuezeng. A Novel Scalable Architecture of Cloud Storage System for Small Files based on P2P[C]. IEEE International Conference on Cluster Computing and Workshops, Hangzhou, China, 2012

[8] Mackey G, Sehrish S, Wang Jun. Improving Metadata Management for Small Files in HDFS[C]. IEEE International Conference on Cluster Computing and Workshops, Orlando, USA, 2009

[9] 余思,桂小林, 黄汝维. 一种提高云存储中小文件存储效率的方案[J]. 西安交通大学学报, 2011, 45(6): 59-63

[10] Liu Xuhui, Han Jizhong. Implementing WebGIS on Hadoop: A Case Study of Improving Small File I/O Performance on HDFS[C]. IEEE International Conference on Cluster Computing and Workshops, Beijing, China, 2009

[11] Dong Bo, Zhong Xiao, Zheng Qinghua, et al. A Novel Approach to Improving the Efficiency of Storing and Accessing Small Files on Hadoop: A Case Study by Power Point Files[C]. IEEE International Conference on Services Computing, Miami, USA, 2010

[12] Dong Bo, Zheng Qinghua, Tian Feng, et al. An Optimized Approach for Storing and Accessing Small Files on Cloud Storage[J]. Journal of Network and Computer Applications, 2012, 35(6): 1 847-1 862

[13] Hofmann T. Unsupervised Learning by Probabilistic Latent Semantic Analysis[J]. Machine Learning Journal, 2001, 42: 177-196

[14] Jin Xin, Zhou Yanzan. Web Usage Mining Based on Probabilistic Latent Semantic Analysis[C]. The 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA, 2004

[15] Xu Guandong, Zhang Yanchun, Zhou Xiaofang. Using Probabilistic Semantic Latent Analysis for Web Page Grouping[C]. Conference of the 15th International Workshop on Research Issues on Data Engineering: Stream Data Mining and Applications, Tokyo, Japan, 2005

[16] Bosch A, Zisserman A, Munoz X. Scene Classification Via PLSA[C]. The 9th European Conference on Computer Vision, Graz, Austria, 2006

[17] 李德仁,黄俊华,邵振峰. 面向服务的数字城市共享平台框架的设计与实现[J]. 武汉大学学报·信息科学版, 2008, 33(9): 881-885

第一作者简介:王涛,博士。主要从事云存储和大数据研究。
E-mail:wangtao.mac@gmail.com

A Small File Merging and Prefetching Strategy Based on Access Task in Cloud Storage

WANG Tao[1] *YAO Shihong*[1] *XU Zhengquan*[1] *XIONG Lian*[1]

(1 State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan 430079, China)

Abstract: Aiming to improve the small file problem of a general distributed file system in cloud storage, a file merging and prefetching strategy based on a users' access task is proposed that improves the PLSA model. Analyzing the relationships among the access tasks, applications, and access files the strategy merge small files on the basis of tasks and selects prefetching files based the transition probability of the tasks. Efficiency model analysis and experimental results of a digital city prototype system based on HDFS all show that the proposed strategy has a high prefetching hit ratio and can effectively reduce the metadata servers' load and the response delay for users' request.

Key words: distributed file system; probability latent semantic analysis; small files; access task; merging and prefetching

About the first author: WANG Tao, Ph D, majors in cloud storage and big data.
E-mail: wangtao.mac@gmail.com