# 基于 CDN 和 P2P 的分布式网络存储系统

# 马 军 郑 烇 殷保群

(中国科学技术大学网络传播系统与控制联合实验室网络传播系统与控制安徽省重点实验室 安徽 合肥 230027)

摘要 把用户的文件分片后均衡存储在不同的分布式存储节点上,并利用虚拟目录服务器和基于 P2P-DHT 的目录服务器把文件元数据与文件数据片高效地对应起来,以提供高效目录服务,分布式存储节点以 P2P 方式工作以快速完成用户对文件数据的请求任务。分布式网络存储系统 DNSS 充分利用了 CDN 和 P2P 的技术优势,有较高的可用性、可靠性和可扩展性。 DNSS 已经在中国科学技术大学应用。

关键词 分布式存储 P2P 虚拟目录 DHT

#### CDN AND P2P-BASED DISTRIBUTED NETWORK STORAGE SYSTEM

Ma Jun Zheng Quan Yin Baoqun

(Joint Lab of NCSC, Key Lab of Anhui NCSC, University of Science and Technology of China, Hefei 230027, Anhui, China)

**Abstract** The user's file is balanced stored at different distributed storage nodes after being split. File metadata and file data slice are matched high-efficiently by virtual directory server and P2P-DHT-based directory server in order to provide high-efficient directory service. Distributed storage nodes complete quickly by means of P2P system the tasks of file data requested by users. Distributed network storage system (DNSS) takes full advantage of the technology superiority of CDN and P2P and is of high availability, reliability and scalability. DNSS has been used in University of Science and Technology of China.

Keywords Distributed storage P2P Virtual directory DHT

# 0 引 言

随着网络技术的不断发展、网络带宽的不断提高以及单机存储容量的不断增大,近年来,分布式网络存储系统的研究和应用已经成为一个热点<sup>[1]</sup>,如已经出现的分布式网络存储系统OceanStore<sup>[2]</sup>、CFS<sup>[3]</sup>、PAST<sup>[4,5]</sup>、Tangler<sup>[6]</sup>、Pangaea<sup>[7]</sup>、Granary<sup>[8]</sup>等。CDN作为一种提高网络内容、节省骨干网络带宽的技术,存在应用和扩展成本高的缺点<sup>[9]</sup>。P2P技术突破了传统的C/S架构的模式,具有非常好的扩展性,但存在安全性、可控性问题<sup>[10]</sup>。本文充分利用CDN的资源管理优势和P2P的高扩展性,构建了一个在全互联网范围内使用的可靠高效的海量分布式存储系统DNSS。

# 1 系统架构设计

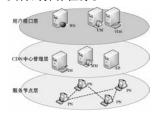
### 1.1 系统架构

基于 CDN 的资源管理优势和 P2P 的高扩展性为实现 DNSS 提供了良好的支持,本文按照分层原则设计了整个系统的架构。 DNSS 分成三层:用户接口层,CDN 中心管理层、以 P2P 方式工作的服务节点层。图 1 和图 2 分别表示 DNSS 的分层体系架构和系统架构。

• 用户接口层主要向用户提供操作接口,包括三个功能模块:Web 服务模块(WS),用户管理模块(UM),虚拟目录服务模块(VDS)。Web 服务模块负责为用户提供文件操作界面;用户

管理模块主要负责管理和维护用户的信息;虚拟目录服务模块则负责组织和维护文件系统的树形目录结构信息。

- CDN 中心管理层主要负责维护全局文件数据分布信息、管理服务节点和实现全局负载均衡,包括三大功能模块:节点管理模块(MM),基于 P2P-DHT 的目录服务模块(DS),负载均衡模块(LB)。节点管理模块主要负责管理服务节点的增加、退出、删除等;目录服务模块则管理和维护整个系统的每个文件在各个服务节点的存储信息;负载均衡模块则主要负责整个系统的负载均衡;
- 服务节点层主要负责整个系统的文件数据的存储以及 对用户提供服务。该层由很多服务节点(PN)组成,所有服务节 点间以 P2P 方式相互协作,完成 CDN 中心管理层下发的用户对 文件的操作任务。





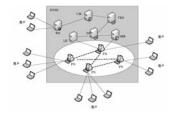


图 2 DNSS 系统架构图

收稿日期:2008-09-25。国家高技术研究发展计划项目(2008AA01Z147);国家自然科学基金(60574065)。**马军**,硕士生,主研领域:网络新媒体。

#### 1.2 主要工作流程

用户通过 Web 服务模块和用户管理模块登录到系统后,根据用户的操作类型,可以把用户的操作请求分为二级:第一级是用户对文件或者目录的元数据操作。例如,用户对文件的移动、改名和共享,以及对目录的创建、移动和共享操作,这些对于元数据的操作只需要虚拟目录服务模块对目录树的信息进行相应修改来完成,在用户接口层即可实现。第二级是用户对实际文件数据的操作,如用户对文件的上传、下载、删除操作,这些操作则需要系统所有三层协作来完成,特别是服务节点群成为用户的数据服务器,直接向用户提供数据服务。图 3 描述了用户登录系统并下载文件时的系统时序图。

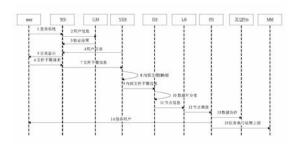


图 3 用户下载文件时的系统时序图

# 2 关键技术

DNSS 系统的设计与实现涉及以下关键技术问题:1)如何将分布在互联网范围内存储节点的存储资源和服务资源以 P2P方式高效整合起来,共同组成一个海量的分布式网络存储系统的架构问题;2)如何完成用户虚拟目录和系统内部分片内容的精确定位有效对应问题;3)如何将内容和服务任务均衡地分布在各个存储服务节点的部署及调度问题。其中的架构问题在第1节已经详细介绍了,下面主要解决其它关键技术问题。

#### 2.1 文件系统组织结构

为了构建成完整的树形结构的全局目录,全局目录树采用超级虚根技术。全局目录最高层为虚根,虚根下第一层每个结点对应一个用户名,为该每个用户的根目录,每个从用户名开始往下的结构为用户的目录树;用户的根目录下为用户的共享目录和用户的个人目录信息。全局目录树结构图如图 4 所示。目录树结构的组织、管理和维护全由虚拟目录服务模块完成,所有的目录都是虚拟的。



图 4 全局目录树结构图

虚拟目录服务模块可以看作是 DNSS 的文件系统的元数据服务器。由于所有文件都被分片存储在不同的服务节点(PN)上,所以服务节点是 DNSS 的文件数据服务器。由于服务节点只负责存放文件数据,而虚拟目录服务器只负责元数据,则需要一个模块把元数据和分片的文件数据高效地对应起来,基于P2P-DHT的目录服务器则起到了这个作用,它维护了元数据所指向文件的数据片在服务节点的存储分布。于是,服务节点的文件数据片与虚拟目录服务器的元数据通过基于 P2P-DHT 的目录服务器高效地对应起来便组成了 DNSS 的文件系统,如图 5

所示。

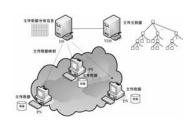


图 5 DNSS 的文件元数据和实际数据片对应关系图

文件系统操作包括元数据操作和文件数据操作,系统把元数据操作和文件数据操作分开处理,提供专一的元数据服务器和专一的文件数据服务器,有利于提高系统的性能和健壮性;并且,由于服务器的职能专一,这样负载均衡服务器能够更方便和准确地进行负载均衡。

### 2.2 部署算法

用户向系统上传文件后,会被分片存储在各个 PN 节点上,如何将数据片部署到各节点上会对整个系统的性能有重要影响。DNSS 系统中由 LB 模块来完成数据分片存储任务的部署管理,部署策略的主要目的是均衡各个 PN 节点的存储负担,同时在向用户提供文件内容服务时,能够有利于节点的调度。具体部署算法如下:

- 1)设定划分节点临近用户的参数为 L,最小分配节点数为 PMIN,节点已存储的文件数的权重影响因子为  $P(0 \le P \le 1$ ,已存储的文件块数权重为 1-P),文件分片大小为 kMB,服务 节点的最小剩余存储容量为 RMIN:
- 2) 把要部署的文件(假设大小为 nMB)按照 kMB 大小的片进行分割,会被分成  $M = \lceil n/k \rceil$  片;
- 3) 根据参数 L 获取临近用户的所有服务节点  $(PN_1, PN_2, \cdots, PN_s)$  ,这些节点对应的已存储的文件数为  $(F_1, F_2, \cdots, F_s)$  ,已存储的数据片数为  $(B_1, B_2, \cdots, B_s)$  ,剩余存储容量为  $(R_1, R_2, \cdots, R_s)$  ,服务能力为  $(SP_1, SP_2, \cdots, SP_s)$  ,带宽为  $(BW_1, BW_2, \cdots, BW_s)$  ;
- 4) 计算节点  $PN_j$ 的权重为 $(F_j \times P + B_j \times (1 P))/(R_j \times SP_i \times BW_i)$ ,  $1 \le j \le S$ ;
- 5) 把这些 PN 按照权重排序,并去掉  $R_j < RMIN$  的节点  $PN_j$ ,得到结果为  $(PN_{i_1},PN_{i_2},\cdots,PN_{i_V})$ ,  $0 \le V \le S$ ;
- 6) 如果 V < PMIN,则调整参数 L(以获取更多的临近用户的节点),转3);否则转7);
  - 7) 计算  $U = \lceil M/V \rceil$ ;
- 8) 把第  $B_{V*u+j}$  片数据部署到  $PN_{i_j}$  上,  $0 \le u \le U$  ,  $1 \le j \le V$  。

#### 2.3 调度策略

如何为用户分配节点服务器是 DNSS 至关重要的负载均衡问题。负载均衡策略主要指对节点服务器的调度,为了使系统中当前有数据请求的用户能够均衡地分布在不同的节点服务器上,并且分配的节点服务器能够快速高效地为用户服务,同时又节省主干网络带宽,我们采取最近最少原则,即选择当前距离用户最近、负载最轻的节点服务器。基于这种原则,对节点服务器的调度算法如下:

//分析用户登录到系统时的 IP,检测系统中靠近用户 IP 的所有服 //务节点

PN[] pns = getNearPNs(userIP);

//衡量这些服务节点的服务状态,找到这些节点中当前负载最轻的

//服务节点
Long pnLoad = MAXLOAD;
Long pnID = -1;
For (int i = 0; i < pns. length; i ++)

{ Pns[i]. load < pnLoad;
pnLoad = Pns[i]. load;
pnID = Pns[i]. pnID; }

//把任务分配到这些节点中当前负载最轻的服务节点上

### 2.4 数据备份

setServer(pnID);

随着广域网技术的不断发展,数据信息访问量越来越趋向海量化。在频繁地对数据进行访问的过程中,数据丢失和系统不稳定成为越来越突出的问题。从而,为了保障服务的可靠性,需要对数据备份存储。目前,高可用的数据备份存储有 RAID 和数据复制等。DNSS 采用双机备份,每个中心服务器与服务节点都有一个与其相邻的服务节点且含有与其完全一样的数据存储结构,可以认为是这两个服务节点互相成为对方的影像。服务器及其影像同时提供服务,这样虽然不能扩大系统的存储容量,但是提高了系统的可靠性和服务效率,并减少了用户的等待时延。

# 3 总 结

本文设计与实现的基于 CDN 和 P2P 相结合的 DNSS,能有效利用 CDN 架构的优点和 P2P 的技术优势,使系统有较高的可用性和扩展性。本文采取的文件结构能够有效地建立起目录树,并可以方便地实现文件或者目录的共享,元数据与文件数据的分离,以及文件数据的分布式存储;所采取的存储策略和调度算法能够有效地解决各个文件服务器的存储和服务负载均衡问题。

### 参考文献

- [1] 魏青松. 大规模分布式存储技术研究[D]. 电子科技大学,2004.
- [2] Kubiatowicz J, Bindel D, Chen Y, et al. OceanStore; An Architecture for Global-scale Persistent Storage [C]//9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS IX). November 2000.
- [ 3 ] Bek F, Kaashoek M F, Karger D, et al. Wide-area Cooperative Storage with CFS[ C]//18th ACM Symposium on Operating Systems Principles (SOSP01). October 2001.
- [4] Peter Druschel, Antony Rowstron. PAST: A Large-scale, Persistent Peer-to-Peer Storage Utility[C]//8th Workshop on Hot Topics in Operating Systems (HotOS VIII). May 2001.
- [5] Rowstron A, Druschel P. Storage Management and Caching in PAST, a Large-scale Persistent Peer-to-Peer Storage Utility [C]//18th ACM Symposium on Operating Systems Principles (SOSP01). October 2001.
- [ 6 ] Waldman M, Mazieres D. Tangler: A Censorship-Resistant Publishing System Based On Document Entanglements [C]//8th ACM Conference on Computer and Communication Security (CCS-8). November 2001.
- [7] Saito Y, Karamanolis C, Karlsson M, et al. Taming Aggressive Replication in the Pangaea Wide-Area File System [C]//5th Symposium Operating Systems Design and Implementation (OSDI02). December 2002.
- [ 8 ] Zheng W, Hu J, Li M. Granary: Architecture of Object Oriented Internet Storage Service [ C ]//IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-EAST 2004). September 2004.
- [9] Shoaib Khan, Rüdiger Schollmeier, Eckehard Steinbach. A Perform-

ance Comparison of Multiple Description Video Streaming in Peer-to-Peer and Content Delivery Networks [C]//2004 IEEE International Conference on Multimedia and Expo (ICME).

[10] Adarand E, Huberman B. Free riding on Gnutella. Technical Report, CSL-00-3, Palo Alto: Xerox PARC, 2000.

#### (上接第35页)

SPACE(4.0)下的运行时间情况,可以知道  $SPACE(\delta)$ 的  $\delta$  越大, 预处理时间越短。

表 1 3 种细化算法在不同粗糙空间下运行时间对比

	图像在 SPACE(1.0)下			图像在 SPACE(2.0)下			图像在 SPACE(4.0)下		
	FPA	文献[3]	文献[4]	FPA	文献[3]	文献[4]	FPA	文献[3]	文献[4]
图像1	5.4370	4.9060	4. 1720	3.0470	1.5310	1.2660	0.4220	0.4220	0.4220
图像2	6.9370	6.4060	3.3440	1.5000	1.4690	1.2350	0.3750	0.3600	0.3750
图像3	5.5780	5.3120	2.4370	1.0940	1.0940	1.0310	0.3750	0.3600	0.2970
图像4	3.8910	3.2970	2.5000	1.8910	1.1560	1.0620	0.7030	0.4220	0.4220
图像5	8.7650	4.8900	3.3120	1.4840	1.4680	1.0310	0.5620	0.3750	0.3750

注:时间单位为 ms

在不能破坏汉字的拓扑结构和本质特征前提下,增大  $\delta$ ,粗 糙化不同粗糙空间下汉字图像,再细化粗糙化图像。500 个样本实验结果的统计数据如表 2 所示, $\delta$  越大,去除的伪特征点越  $\delta$ ,但当  $\delta$  = 8.0 时,部分粗糙化图像拓扑结构被改变。而在 SPACE(4.0)下,拓扑结构和本质特征保持良好,同时可去除大量伪特征点,因此  $\delta$  取 4.0 较好。

表 2 不同粗糙空间下汉字图像的总体情况

	图像在 SPACE (2.0)下	图像在 SPACE (4.0)下	图像在 SPACE (8.0)下
拓扑结构和本质特征 保持情况(%)	99.9%	99.6%	86.7%
去除伪特征点数(个)	194	358	372

# 4 总 结

经过粗糙化后的手写汉字图像,图像的尺寸可以减少,对粗糙化后的汉字图像细化,可以防止一些断点、伪交叉点、伪线段的产生,提高了手写汉字的识别率和识别速度。

### 参考文献

- [1] 马垣. 粗糙几何学[C]//中国第6届 Rough 集及软计算学术会议 大会报告,2006.
- [ 2 ] Chen M S, Park J S, Yu P S. Efficient Data Mining for Path Travsersal Paterns [J]. IEEE Trans. Knowledge Data Engineer, 1998, 10(2):209
  –211.
- [3] 贺继刚,杨晓伟,吴广潮,等. 基于模板保留的快速并行细化算法 [J]. 计算机应用与软件,2007,24(12):26-28.
- [4] 廖开阳,张学东,章明珠. —种新的指纹图像快速细化算法[J]. 计算机工程与应用,2008,44(5);93-95.
- [5] 黄铁英,姜昱明. 一种快速手写汉字细化算法[J]. 计算机工程. 2004,30(19):121-128.
- [6] 吴选忠. Zhang 快速并行细化算法的扩展[J]. 福建工程学院学报, 2006,4(1):89-92.
- [7] 史绍强. 一种改进型的汉字字符图像细化算法[J]. 计算机技术与发展,2007,17(9):88-91.