第 36 卷 第 23 期 计 算 机 工 程 2010 年 12 月
Vol. 36 No. 23 Computer Engineering December 2010

·软件技术与数据库· 文章编号: 1000—3428(2010)23—0047—03 文献标识码: A 中图分类号: TP311.52

海量图片的分布式存储及负载均衡研究

朱晓辉, 王杰华, 石振国, 陈苏蓉
(南通大学计算机科学与技术学院, 江苏 南通 226019)

摘 要: 针对海量图片给网站带来的访问速度下降、性能压力增大和 I/O 瓶颈等问题, 提出一种海量图片的分布式存储及负载均衡技术。通过把图片数据和网站内容分开部署、在数据库中记录和维护图片服务器状态信息等方法实现图片和页面数据的分离。实验结果表明, 该技术能提高网站的访问速度和运行效率, 并可动态增加图片服务器的数量满足日益增加的性能需求。

关键词: 海量图片; 分布式存储; 负载均衡

Research on Distributed Store and Load Balance of Mass Images

ZHU Xiao-hui, WANG Jie-hua, SHI Zhen-guo, CHEN Su-rong
(College of Computer Science and Technology, Nantong University, Nantong 226019, China)

【Abstract】 Aiming at the problems of the mass images can cause to Web site such as lower access speed, more performance pressure, I/O performance bottle-neck, etc., a technology of distributed store and load balance for mass images is proposed. By the means of deploying Web site pages and images separately and recording status of image servers in database, solves the problem of separation for image data and page data. Experimental result shows that the solution can improve the access speeds and running efficiency for Web site, and can add additional image servers to meet the increasing performance demands.

【Key words】 mass images; distributed store; load balance

1 概述

随着计算机网络技术的发展和普及, 出现了越来越多像“新浪”、“淘宝”大型门户站点及电子商务网站[1]。这类网站都保存有大量图片资源。用户在访问这些站点网页时, 网页中图片信息占到页面数据流量的大部分。由于受客户端浏览器限制, 无法从一台服务器上同时下载页面中所有图片信息, 因此即使服务器有很高带宽, 用户的访问速度还是会受到很大影响。由于图片保存在物理硬盘上, 访问图片需要频繁进行 I/O 操作, 因此当并发用户数越来越多时, I/O 操作就会成为整个系统的性能瓶颈[2]。同时, 由于受操作系统的限制, 一个目录中能存放的图片文件数量是有限的, 因此随着图片资源的不断增加, 如何合理有效地对图片进行管理和维护也是一个难题。

对于少数大型网站系统, 由于自身具有雄厚的资金和人力资源, 可采用 NFS[3]、CDN[4]、Lighttpd、反向代理、负载均衡等技术提高用户访问速度。但这些技术需要庞大的资金支持, 对于处于创业初期中等规模的商务网站, 由于缺少必要的资金支持, 因此无法采用这些技术提升网站的访问速度。

对此, 笔者提出了适用于中等规模商务网站的海量图片数据分布式动态存储及负载均衡的解决方案。该方案只需增加很少的硬件成本, 即可提升网站的访问速度, 并且可以根据需要动态调整图片服务器的数量及图片的存储目录, 确保系统具有可扩展性和伸缩性[5]。

2 系统架构设计

对于 Web 服务器而言, 用户对图片信息的访问是很消耗服务器资源的。当一个网页被浏览时, Web 服务器与浏览器建立连接, 每个连接表示一个并发。当页面包含多个图片时, Web 服务器与浏览器会产生多个连接, 同时发送文字和图片以提高浏览速度。因此, 页面中图片越多 Web 服务器受到的压力也就越大。同时由于受到浏览器本身的并发连接数限制(2 个~6 个并发), 意味着页面上有多于并发连接数限制的图片时, 也不能并行地把所有图片同时下载和显示。对于小型网站, 由于数据规模小, 可以把网站所有页面和图片统一存放在一个主目录下, 这样的网站对系统架构、性能要求都很简单。但大中型网站都保存有海量级的图片文件, 所采用的技术更是涉及广泛, 从硬件到软件、编程语言、数据库、Web 服务器、防火墙等各个领域都有较高要求。因此, 有必要设立单独的图片服务器来专门存放图片, 把图片数据的流量从 Web 服务器上分离开, 这样的架构可以有效缓解 Web 服务器的 I/O 性能瓶颈, 提升用户的访问速度。

系统架构设计需要满足以下 4 点要求: (1) 图片能进行分布式存储; (2) 能实现负载均衡; (3) 能根据用户访问量及网站图片数据量的增加能动态添加图片服务器节点; (4) 图片服务

基金项目: 江苏省高校自然科学基金资助项目(07KJB520096); 南通市重大科技专项基金资助项目(XA2008004); 上海市信息安全综合管理技术研究重点实验室开放课题基金资助项目(AGK2009006)

作者简介: 朱晓辉(1976-), 男, 讲师、硕士, 主研方向: 分布式存储技术, 数据库技术; 王杰华, 副教授、硕士; 石振国, 副教授、博士; 陈苏蓉, 讲师、硕士

收稿日期: 2010-03-30 E-mail: wang.jh@ntu.edu.cn

器节点的动态调整对网站用户而言是透明的,并且不会中断系统的正常运行。系统整体架构如图 1 所示,包括客户端、Web 服务器、数据库服务器、图片服务器集群 4 个部分。

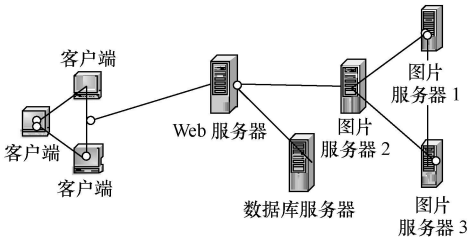

图 **1** 系统架构

客户端是指 IE、Firefox 等常用的客户端浏览器,用户可以通过客户端来浏览网站的图片信息,也可以通过客户端上传图片信息。

Web 服务器部署网站的 Web 页面,用于响应客户端用户的请求。当用户浏览网页时,Web 服务器响应请求并访问数据库服务器,获得网页中所有图片的 URL 路径,然后生成页面并返回给客户端,客户端接收该页面并根据页面中的图片 URL 路径自动从不同的图片服务器下载并显示相应图片。当用户上传图片时,Web 服务器首先从数据库服务器中获取所有图片服务器的当前状态,并根据相关算法选择一个图片服务器及保存的目录,再调用该图片服务器的 Web Service 方法把图片保存到该服务器,最后在数据库服务器中纪录该图片的编号及 URL 路径等信息。

数据库服务器用于记录所有图片的编号以及图片的存放位置等信息,同时需要记录所有图片服务器的配置及当前状态信息。

图片服务器集群用于存放网站的所有图片信息,该集群的服务器数量可以根据需要动态增加。

3 系统实现及关键技术

增加了图片服务器后,对于客户端而言,整个网站系统执行过程应该仍然是透明的,不会给用户带来任何影响。但后台系统需要解决以下 4 个问题:(1) 如何实现图片的分布式部署,图片上传时如何动态确定保存到哪台图片服务器;(2) 如何做到图片服务器的负载均衡,既要保证所有图片服务器都有均等的机会来保存图片,又要考虑到不同服务器的硬件配置和性能差异来区别对待;(3) 如何把一台图片服务器上图片均衡保存到多个子目录中以便突破操作系统在同一个目录中保存文件数的限制,对图片进行更好的管理和维护;(4) 如何能根据性能需要和图片数量的增加实现图片服务器的动态扩充。

3.1 状态信息表

Web 服务器需要及时掌握所有图片服务器的状态和信息才能动态决定把图片保存到哪一台图片服务器,因此,需要把所有的图片服务器的状态信息全部纪录到数据库服务器中,记录图片服务器信息和状态的表格式如表 1 所示。状态信息表中的 ServerId 字段为主键自增列,唯一代表一条图片服务器纪录。ServerName 字段记录服务器的名称,方便管理员识别该记录代表哪台服务器。ServerUrl 字段标识了图片服务器上图片主目录的 URL 根路径。PicRootPath 字段标识了保存图片的物理主目录。MaxPicAmount 字段表示图片服务器能保存的最大图片数,该数可以根据图片服务器的硬件配置和性能以及用户实际需要而进行动态调整。CurPicAmount 字段表示当前已保存的图片数,当 CurPicAmount ≥MaxPicAmount 时系统将不再把图片上传到该服务器。SubFoldAmount 字段描述了在 PicRootPath 中指定的图片主目录下的子目录数。这样可以把图片均匀分布到不同子目录下,避免在同一个目录下保存过多图片,从而方便对图片进行维护和管理。FlgUsable 字段表示图片服务器是否可用。

表 **1** 图片服务器状态信息表

字段名	类型	备注
ServerId	Int	自增字段,主键
ServerName	Nvarchar(50)	图片服务器名称
ServerUrl	Nvarchar(100)	图片服务器 URL 根路径
PicRootPath	Nvarchar(100)	图片服务器物理根路径
MaxPicAmount	Int	可存放最多图片数
CurPicAmount	Int	当前已保存图片数
SubFoldAmount	Int	可创建的图片子目录数
FlgUsable	Boolean	图片服务器是否可用

3.2 图片浏览

客户端用户通过浏览器向 Web 服务器发出浏览某页面的请求,Web 服务器从数据库服务器中获取该页面的所有图片 URL 信息,并根据 URL 信息去搜索表 1 所列的状态信息表,判断该 URL 所指向的图片服务器的状态字段 FlgUsable,若 FlgUsable = = false 表示该图片服务器当前因某种原因处于不可用状态,则把该图片的 URL 替换成 Web 服务器上保存的一个默认图片的 URL,否则把该 URL 直接返回给客户端。客户端再根据图片的 URL 路径自动从不同的图片服务器上下载并显示相应的图片。由于图片 URL 路径直接指向具体的图片服务器,因此需要在每个图片服务器的保存图片的主目录上建立一个 Web 站点。由于客户端浏览器所需要的图片是从多个图片服务器上直接下载,因此浏览器可以并发地从多台服务器上同时下载图片,这样就缩短了图片下载时间,同时也减轻了 Web 服务器的 I/O 请求及性能压力,因此,提高了网站的访问速度。浏览图片算法如图 2 所示。

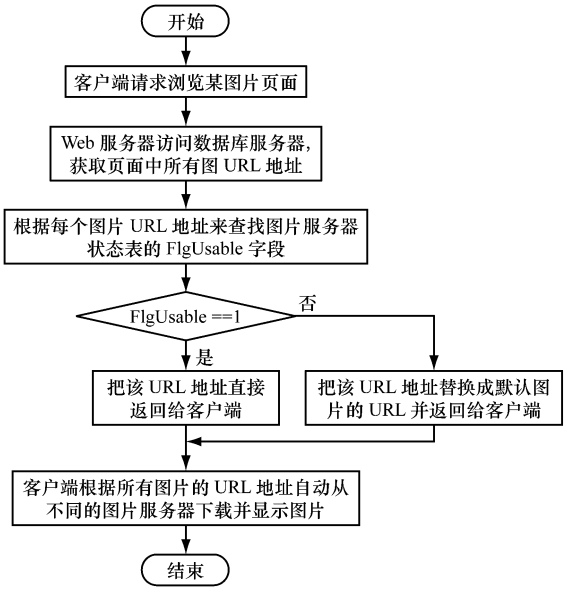

图 **2** 浏览图片算法

3.3 图片上传

由于 B/S 架构本身技术限制,图片无法通过 Web 服务器

直接上传到不同的图片服务器,因此需要在所有图片服务器上部署一个 Web Service[6] 以便 Web 服务器可通过调用不同图片服务器上的 Web Service 执行保存或删除图片的操作。

图片上传的过程比较复杂,首先 Web 服务器接收客户端的访问请求并访问数据库,通过执行"select * from tb_ServerStatus where FlgUsalbe = 1 and MaxPicAmount > CurPicAmount"语句(tb_ServerStatus 为表 1 所列的图片服务器状态信息表),从状态表筛选出可用的图片服务器集合记作 C,并获取集合的总记录数 N。然后用随机函数产生一个随机数 R1 并用 R1 与 N 进行取余运算记作 I= R1%N。则C[I]即为要保存图片的图片服务器。获取 C[I]记录中的 SubFoldAmount 的值记作 K,K 即为 C[I]图片服务器中的图片子目录的个数。为了简化算法,规定所有的子目录名从"0"开始编号,直到"K-1"。例如: SubFoldAmount 值为 1 000,则图片服务器上图片子目录名分别为"0"、"1"、"2"、…、"999"。再用随机函数生成随机数 R2,使得 S= R2%K,则 S 即为要保存的图片的子文件夹名称。为了确保上传的图片名称不重复,以当前时间+随机数的形式组成图片名称。

综上所述,通过利用随机函数取值的随机性和取余运算,使每台图片服务器及同一台服务器上的所有图片子目录都有均等的机会保存图片。因此,所有图片都是被随机保存到不同图片服务器的不同子目录中,实现了图片的分布式部署和负载均衡。同时网站管理员也可通过设定服务器状态信息表中的"MaxPicAmount"和"SubFoldAmount"2 个字段的值来限定所能保存图片的最大数和子目录数,从而能够根据服务器的硬件配置和性能差异等因素来决定该服务器能保存图片的最大数和子目录数,因此,进一步提升了整个图片服务器集群的负载均衡能力。当需要增加图片服务器时,也只需在状态信息表中增加一条新的图片服务器纪录,添加新图片服务器的过程对整个网站系统的运行没有任何影响,从而实现了图片服务器的动态增加。用户上传图片的算法如图 3 所示。

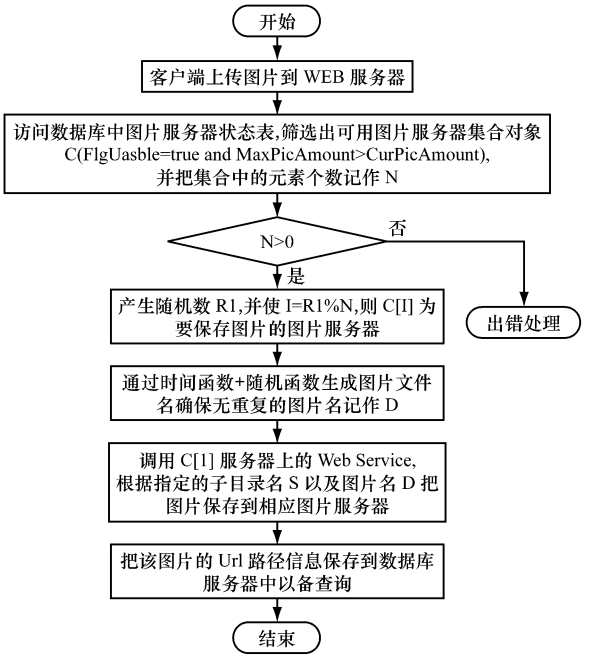

图 3　上传图片算法

3.4　图片删除

客户端向 Web 服务器发出删除某个图片的请求,Web 服务器接收请求并搜索图片数据库获取待删除图片的 URL 信息。把该 URL 信息通过字符串运算分隔为图片服务器的 URL 根路径 R、图片所存放的子目录 D 以及图片名称 N。再查找图片数据库状态信息表,获取与 R 匹配的记录记作 C,C 即为要删除图片的图片服务器。然后调用 C 图片服务器上的 WebService[7] 方法,并以图片名称 N 和图片所存放的子目录 D 为参数通知该方法删除该图片,最后把该图片纪录从数据库服务器中删除。用户删除图片信息的算法如图 4 所示。

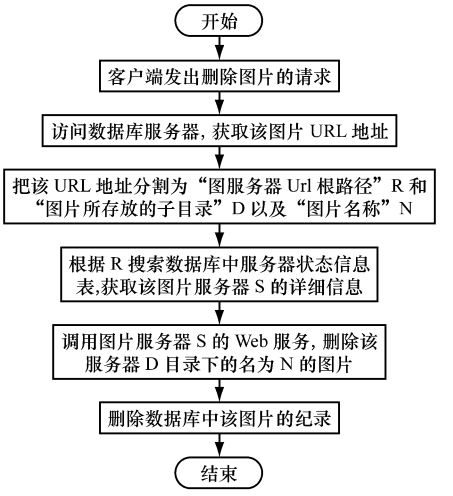

图 4　删除图片算法

3.5　图片修改

修改图片的算法是删除图片和上传图片 2 个功能的叠加。客户端发出修改图片的请求并把新的图片上传到 Web 服务器,Web 服务器访问数据库获取旧图片的 URL 地址,调用删除图片的功能把旧图片删除,最后调用上传图片的功能完成新图片的上传。最后修改图片数据库,纪录新图片的 URL 路径。其算法流程如图 5 所示。

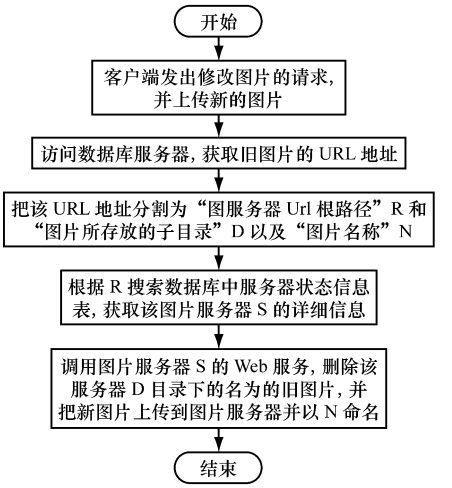

图 5　修改图片算法

4　系统性能分析

在局域网环境中,对采用图片服务器和不采用图片服务器 2 种情况进行了性能测试。硬件配置如下: Web 服务器、数据库服务器各一台,配置为 CPU: Intel Xeon 四核 2.2 GHz,内存 4 GB,网络带宽 100 Mb/s。客户端机器 5 台为 CPU: Pentium 3.0 GHz,内存 2 GB,网络带宽 100 Mb/s。图片服务器 3 台,为普通的 PC 机: CPU: Intel 双核 P2.0 GHz,内存 1 GB,网络带宽100 Mb/s。测试数据中共有

(下转第 52 页)

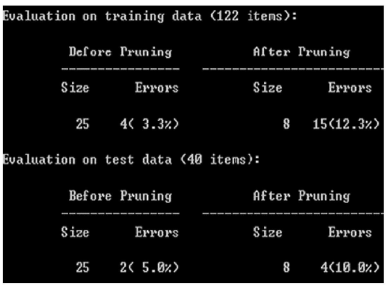

图 1 Monk's problem EBP 剪枝

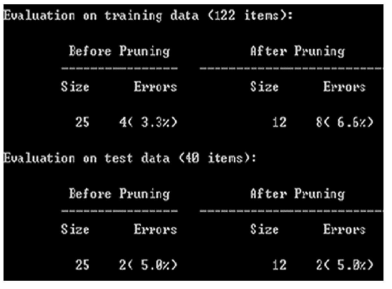

图 2 Monk's problem 条件误差剪枝

表 2 数据集实验结果

数据集	EBP 剪枝方法			条件误分类剪枝方法		
	Size	Errors	错误率/(%)	Size	Errors	错误率/(%)
Monks	8	4	10.0	12	2	5.0
Heart-statlog	11	12	13.3	7	11	12.2
Sonar	15	5	7.1	13	4	5.7
Vote	4	4	2.8	4	4	2.8

分析实验结果表明: 对于 monks 数据集, 从节点数量上看, 条件误差剪枝得到的决策树比 EBP 得到的决策树规模稍大, 但正确率有所提高, 一定程度上克服了过剪枝的情况; 其他数据集构建的决策树节点数减少或者保持不变, 改善了规模, 准确率得到提高。由此可以验证条件误差剪枝方法能够在一定程度上解决过剪枝或者剪枝不充分的情况, 提高决策树的预测准确度。

5 结束语

本文引入条件误差的概念, 针对决策树的模型进行优化, 提出条件误差剪枝方法, 并将其应用于 C4.5 算法中。条件误差充分考虑了父节点分裂对节点以及以该节点为父节点的子树的影响, 有效地权衡了过剪枝和剪枝不充分的情况, 提高了准确率。但目前的算法只局限于分类数是 2 的数据, 多分类的决策树剪枝是进一步研究的课题。

参考文献

[1] Zhong Mingyu, Georgiopoulos M. A K-norm Pruning Algorithm for Decision Tree Classifiers Based on Error Rate Estimation[J]. Machine Learning, 2008, 71(1): 55-88.

[2] Quinlan J R. Simplifying Decision trees[J]. Humar-Computer Studies, 1999, 51(2): 221-234.

[3] Wu Xindong, Kumar V, Quinlan J R, et al. Top 10 Algorithms in Data Mining[J]. Knowledge and Information System, 2008, 14(1): 1-37.

[4] Elomaa T, Kaariainen M. An Analysis of Reduced Error Pruning[J]. Journal of Artificial Intelligence research, 2001, 20(15): 163-187.

[5] Esposito F, Malerba D, Semeraro G. A Comparative Analysis of Methods for Pruning Decision Tree[J]. IEEE Transaction on Pattern Analysis and Machine Intelligence, 1997, 19(5): 476-491

[6] 鲁 为, 王 枞. 决策树的优化与比较[J]. 计算机工程, 2007, 33(16): 189-191.

编辑: 索书志

(上接第 49 页)

300 万张图片, 均匀分布在 3 台图片服务器上, 每台图片服务器建立1 000 个子目录。在 5 台客户端上同时运行压力测试软件, 分别模拟 200 个~ 1 000 个并发用户的请求, 测试结果如图 6 所示。

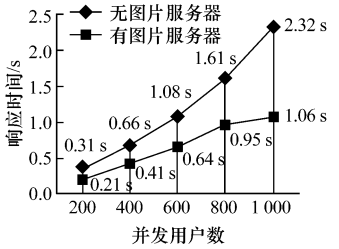

图 6 系统测试结果

从图 6 可以看出, 采用 3 台普通 PC 机作为图片服务器后, 整个系统的响应时间大大减少, 性能得到明显提升, 而且并发访问量越大, 性能的提升越明显, 而对于整个系统而言增加的硬件成本却很有限。

5 结束语

面对网站日益增长的图片数据, 本文设计并实现了一种适用于中等规模网站的图片分布式部署和负载均衡的解决方案。论述了对图片分布式存储、数据库结构设计及相关查询、修改、删除算法等关键技术。通过性能分析数据可知, 该解决方案只需增加很少的硬件成本即可大幅度提升网站的访问速度和运行效率。

参考文献

[1] 胡兴军. 内容分发网络(CDN)技术及市场应用[J]. 当代通信, 2005, 30(17): 23-26.

[2] 田 臣, 陈金华, 王 玮, 等. CDN 内容分发策略数学建模研究[J]. 计算机工程与科学, 2009, 31(5): 4-7.

[3] 郭 劲, 李 栋, 张继征, 等. iSCSI, CIFS, NFS 协议的性能评测[J]. 小型微型计算机系统, 2006, 27(5): 833-836.

[4] Radkov P, Li Yin, Goya P, et al. A Performance Comparison of NFS and iSCS I for P2networked Storage[EB/OL]. (2009-08-30). http://www1.cs.columbia.edu/~cs699810/nfs-iSCSI.pdf.

[5] Laoutaris N, Zissimopoulos V, Stavrakakis I. Joint Object Placement and Node Dimensioning for Internet Content Distribution[J]. Information Processing Letters, 2004, 89(6): 273-279.

[6] 毕 敬, 朱志良, 铁 鸣. 基于 Web services 的分布式企业信息整合模型[J]. 计算机工程, 2008, 34(12): 280-282.

编辑: 索书志