

一种支持 Web QoS 的前置服务器解决方案

陈红红

(西北师范大学 网络教育学院,甘肃 兰州 730070)

摘要:服务质量(QoS)描述了一个产品或服务满足消费者需求的能力。如何为不同类型的用户提供区分服务正在引起研究领域的关注。通过改进 Web 服务器应用软件来提供 Web QoS 支持是一种非常直观的 Web QoS 控制思路,但在实际应用中却存在着通用性和可扩展性等方面的局限。以此思路为出发点,将 QoS 控制部分功能提取出来,在一个前置服务器中去实现,该服务器置于客户端与 Web 服务器中间,主要负责完成 Web 请求的截取、QoS 控制及请求转发。实验证明以引入前置服务器的方式实现 Web 请求的 QoS 控制,能够保证高级别用户请求的较高完成率以及较低响应延时,以低优先级请求的尽早拒绝防止了服务器过载,有效地实现 Web 服务区分。

关键词:Web QoS; 区分服务;前置服务器

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2009)07-0227-03

A New Prepositive Server Plan That Support Web QoS

CHEN Hong-hong

(College of Distance Learning, Northwest Normal University, Lanzhou 730070, China)

Abstract:Quality of Service(QoS) describes the capabilities of a product or service to meet the requirements of consumers. The provision of guaranteed QoS is very critical to the success of Web. How to provide differentiated services for different types of users is an on going research direction in Web service community. By improving the Web server application software to provide support for the Web QoS is a very intuitive Web QoS control ideas. In practice, but there are limitations in versatility and scalability. As a starting point, QoS control functions are extracted and achieved in a prepositive server in this paper. Prepositive server is in the middle of client and Web server and primarily responsible for Web requests interception, QoS control and requests transmit. Experiment proved that this method can ensure the high-level user requests a higher completion rate and a low response delay, refuse low-priority request as soon as possible to prevent the overload of Web server. It can effectively provide differentiated services for different types of users.

Key words:Web QoS; Web differentiated service; prepositive server

1 问题提出

Web 服务质量主要由两个因素决定:网络传输的质量和 Web 服务器的处理性能^[1]。在网络传输质量能够得到很好保证的情况下,如果应用层的 Web 服务器服务质量得不到保证,HTTP 请求仍然有可能被拒绝服务,或者响应时间大大延长而导致“拒绝服务”的效果,Web 服务器在某种程度上成为实现端到端服务质量控制的瓶颈。因此,如何在 Web 服务器系统中引入和实现 QoS 控制的机制与策略,为不同类型的用户或请求提供服务区分和性能保证成为目前 Web 发展中所迫切需要解决的问题^[2]。

当前 Web QoS 控制技术的研究有两个主要切入点^[3]:

1)区分的 Web 服务机制与策略。主要方法是定义用户或服务请求的类别,确定优先级,利用基于优先级的请求分配策略和资源监控与调度机制来保证不同的服务水平协议。

2)Web 服务器系统体系结构设计。目标是确定能够向所有的 Web 用户或请求提供服务水平协议保证的正确的体系结构。主要方法是通过增加内存和 CPU 来扩大单个服务器的处理能力;在局域网范围内通过复制服务器内容建立 Web 服务器集群来增强本地处理能力;在地理上广域分布的范围内通过复制服务器集群来扩大全局处理能力。

以上两点可以概括为以下几个主要研究方向:

- (1) 基于 Web 服务器应用软件的 QoS 控制技术;
- (2) 基于操作系统的 Web QoS 控制技术;

收稿日期:2008-10-10;修回日期:2008-12-04

基金项目:甘肃省教育科研项目(0801B-05);西北师范大学青年教师基金(NWNU-QN-05-17)

作者简介:陈红红(1976-),女,甘肃会宁人,讲师,硕士,研究方向为计算机网络。

(3) 基于中间件的 Web QoS 控制技术;

(4) 基于 Web 服务器集群的 QoS 控制技术。

文中的研究主要以第一个研究方向做为出发点。

2 方案提出

目前很多研究都试图通过改进 Web 服务器应用软件来提供 Web QoS 支持,主要方法是将客户 HTTP 请求进行分类,并且实现优先化调度、接纳控制、资源分配^[4]等机制。文献[5]中实现了在用户级和内核级分别通过基于优先级的请求调度来提供区分的 Web QoS,文献[6]中提出了一个基于 QoS 的 Web 服务器的模型。另外有效的接纳控制机制,Web 内容自适应,基于控制理论的方法等也属于被关注的范围,商业软件如 HP 公司的“Web QoS”服务器软件^[7]和 IBM 公司的 WebSphere 软件平台^[8]中包含了以上范围中的部分功能。

通过改进 Web 服务器应用软件来提供 Web QoS 支持,是一种非常直观的 Web QoS 控制思路,但是需要对 Web 服务器软件进行一定程度的修改^[9],要求 Web 服务器软件开放源代码,在实际应用中存在着通用性和可扩展性等方面的局限。因此笔者以此思路为出发点,将 QoS 控制部分功能提取出来,在一个单独的服务器(QoS 服务器)中去实现,并将该服务器置于客户端与 Web 服务器中间,QoS 服务器即为 Web 服务器的前置服务器。

QoS 服务器各模块组成及工作示意如图 1 所示。

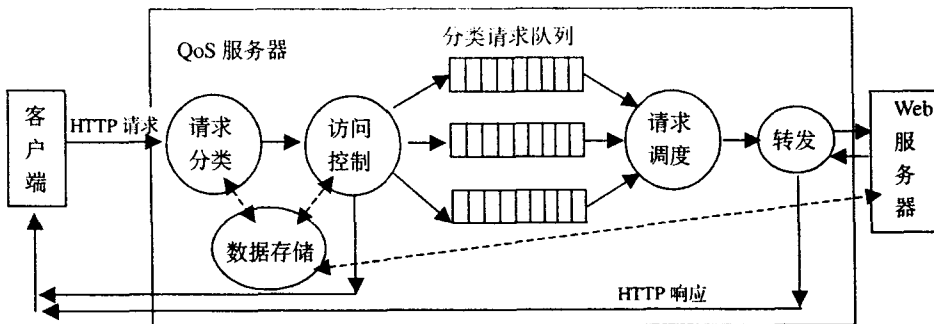


图 1 QoS 服务器工作示意图

QoS 服务器介入后 Web 请求的处理过程如下:

1) 客户端发出请求后 QoS 服务器对该请求进行截取。

2) QoS 服务器对请求进行分类,访问控制,并将请求放入不同等级的等待队列。请求分类和访问控制处理依据来源于数据存储模块的用户信息及 Web 服务器负载信息。

3) QoS 的请求调度根据一定策略调度各队列中的请求并将请求提交给 QoS 转发过程。

4) QoS 的转发过程将请求转发到 Web 服务器,并给客户端转回 Web 的处理结果。

5) 对每一个客户端请求重复 2)~5) 步骤。

3 方案实现

笔者以 C# (C sharp)^[10]做为开发语言对前置服务器予以实现,用到一个主进程和 3 个类,主进程是服务器应用程序的主入口点,它主要负责监听客户端请求,当监听到请求后便立即创建对应的应答线程。第一个类为客户端连接类,负责接受客户端 Socket 请求并经过一定的分类和接纳控制机制将请求放入优先级队列。第二个类根据一定的队列调度算法调度队列中的请求,并将调度的请求送往第三个类进行处理。第三个类负责将用户请求转发到 Web 服务器,并将 Web 服务器处理的请求结果反馈给客户端。

主进程 Main1 的部分源代码如下:

```
static void Main(string[] args)
{
    //建立各种等级的队列,这里假设三个等级
    // ClrQ1 队列最高优先级请求
    // ClrQ2 队列中等请求
    // ClrQ3 队列低级请求
    ClrQ1 = new WQueue();
    ClrQ2 = new WQueue();
    ClrQ3 = new WQueue();
    //创建 Scheduler 类,
    //该类根据队列调度算法调度队列中的请求,并将调度的请
    求送往 Sendback 类

    Scheduler scheduler = new
    Scheduler();
    Thread thread1 = new Thread
    (new ThreadStart(scheduler.
    Run));
    thread1.Start();
    //qos 服务器侦听端口
    const int port = serverport;
    //定义端口号
    TcpListener tcpListener = new
    TcpListener(port);
    Console.WriteLine("侦听端口号: " + port.ToString());
    tcpListener.Start();
    //侦听端口号
    while (true)
    {
        Socket socket = tcpListener.AcceptSocket();
        //创建一个新的线程,Class1 类实例化
        //负责接受客户端 Socket 请求并经过一定的分类和接纳控
        制机制后将请求放入优先级队列
```

```
Class1 proxy = new Class1(socket, Webserverip, webserver-  
port);  
Thread thread = new Thread(new ThreadStart(proxy.Run));  
//创建线程  
thread.Start();  
//启动线程  
}  
}
```

其他三个类的源代码这里不再赘述。

4 测试结果分析

笔者在局域网中搭建了一个测试平台对前置服务器的有效性进行测试,分采用和不采用前置服务器两种情况进行测试,测试用到2个客户端,客户端1代表高级类请求,客户端2代表低级类请求,2个客户端分别在相同的时间内发送相同数目的请求,请求数从10个/秒增至1000个/秒,以对服务器造成不同的负载。下面对服务器在不同负载时的请求完成率和平局响应时间进行分析比较。

从图2和图3的比较可以看出,当Web服务器负载增高至75%后,在不采用前置服务器的状况下两类请求的完成率开始迅速下降,而采用前置服务器时低优先级类请求虽也处于下降趋势,但客户端1(高优先级类)请求的完成率一直保持在85%以上。

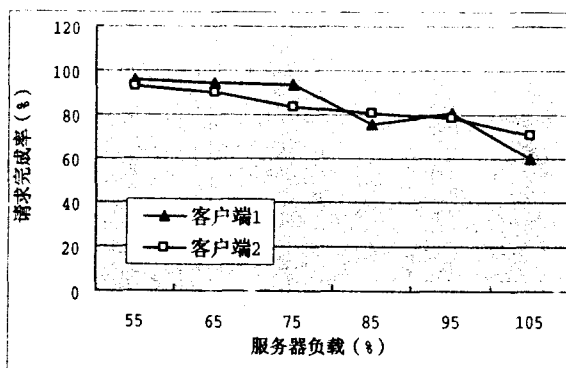


图2 不采用前置服务器时的请求完成率

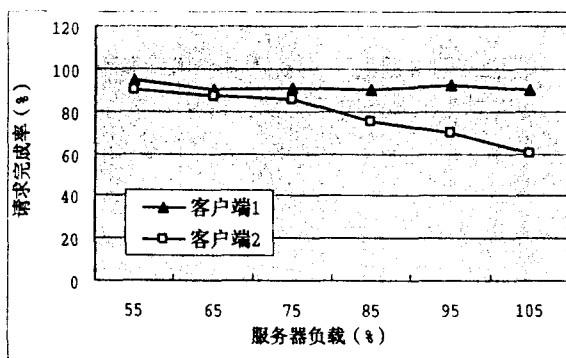


图3 采用前置服务器时的请求完成率

从图4和图5比较可以看出,当服务器负载增高

至85%后,2类请求的响应时间开始迅速上升,而在引入前置服务器的情况下,虽客户端2(低优先级类)的平均响应时间在服务器负载增至75%后开始快速上升,但客户端1的平均响应时间始终保持在3秒以内。

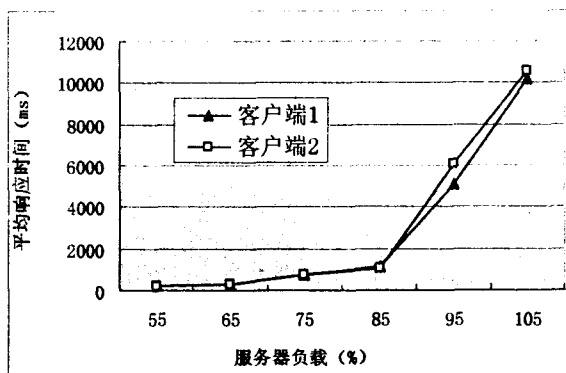


图4 不采用前置服务器时的请求完成时间

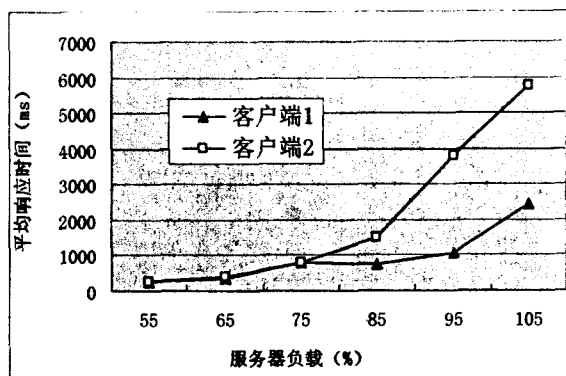


图5 采用前置服务器时的请求完成时间

从各图之间的比较可看出服务器在低负载时(即请求数较少时),采用前置服务器反而会增加用户时延,因此此方案主要适用于用户量大且对用户有分类请求的Web服务提供商,对于用户量小且没有用户分类需求的Web服务提供商来说,该方案反而会增加额外引入服务器的负担及用户响应时延。

5 结束语

提出了一种Web服务器的前置服务器解决方案并对该方案予以实现,结果证明与传统的Web服务相比,以引入前置服务器的方式实现Web请求的QoS控制,能够保证高级别用户请求的较高完成率以及较低响应时延,以低优先级请求的尽早拒绝防止了服务器过载,有效地实现Web服务区分。希望通过以上的工作能为今后的研究提供一定的借鉴和参考。

参考文献:

- [1] 单志广,林 闯,肖人毅,等. Web QoS 控制研究综述[J]. 计算机学报,2004,27(2):145-148.

(下转第233页)

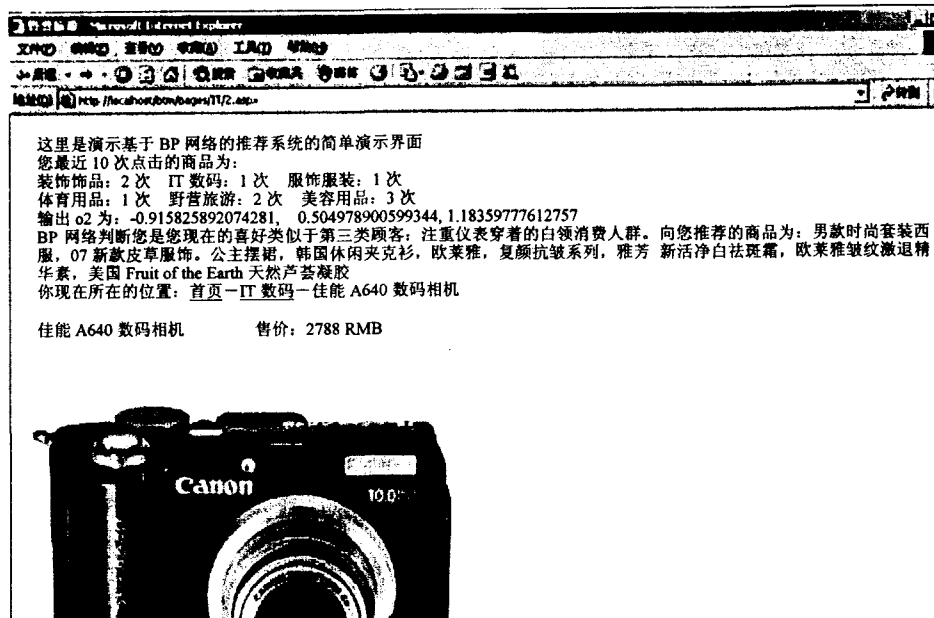


图3 演示平台:具体商品页面——IT 数码商品

的推荐商品数目和品种,使推荐系统的命中率进一步的提高。

参考文献:

- [1] Mobasher B, Dai H, Luo T. Discovery of Aggregate Usage Profiles for Web Personalization[C]//In Proceedings of the Web Mining for E-Commerce Workshop(WebKDD'2000). [s.l.]:[s.n.],2000:584-595.
- [2] 王方华. 市场营销学[M]. 上海:复旦大学出版社,2001.
- [3] Schafer J B, Konstan J, Riedl J. E-Commerce Recommendations Applications[J]. Data Mining and Knowledge Discovery,2001,5:115-153.
- [4] Schafer J B, Konstan J, Riedl J. Recommender System in E-Commerce[C]//Proceedings of the first ACM conference on Electronic Commerce[s.l.]:[s.n.],1999:158-166.
- [5] Resnick P, Varian H R. Recommendation Systems[J]. Communications of ACM,1997,40(3):56-58.
- [6] Nasraoui O, Krishnapuram R. A New Evolutionary Approach to Web Usage and Context Sensitive Associations Mining[J]. International Journal on Computational Intelligence and Applications-Special Issue on Internet Intelligent Systems,2002,2(3):339-348.
- [7] 张富国,徐升华. 推荐系统安全问题及技术研究综述[J]. 计算机应用研究,2008(3):656-659.
- [8] 林霜梅,汪更生,陈弈秋. 个性化推荐系统中的用户建模及特征选择[J]. 计算机工程,2007(17):196-198.
- [9] 高 隽. 人工神经网络原理及仿真实例[M]. 北京:机械工业出版社,2003.
- [10] 刘平峰,聂规划,陈冬林. 电子商务推荐系统中推荐策略的自适应性[J]. 计算机工程与应用,2007(4):23-25.

(上接第229页)

- [2] 杨胜文,史美林. 一种支持 QoS 约束的 Web 服务发现模型[J]. 计算机学报,2005,28(4):589-594.
- [3] 刘 渡,张立臣. Web QoS 及其保证方法[J]. 计算机技术与发展,2006,16(7):32-34.
- [4] 刘克非,王 红,许作萍. 一种基于服务质量预测的 Web 服务选择方法[J]. 计算机技术与发展,2007,17(8):103-109.
- [5] Alemeida J, Dabu M, Manikutty A, et al. Providing differentiated level of service in Web content hosting[C]// In: Proc 1998 SIGMETRICS Workshop on Internet Server Performance. Piscataway, NJ: IEEE Press,1988:243-254.
- [6] Shan Zhiguang, Lin Chuang, Wei Yaya. Prototype Implementation and Performance Evaluation of a QoS-based Web Server[C]//In Proceedings of the IEEE International Workshop on Service-oriented System Engineering 2005 (SOSE 2005). Beijing, China:[s.n.], 2005:191-196.
- [7] HP Labs. Web2K: Bringing QoS to Web Servers. HPL-2000-61, HP Labs Technical Reports[EB/OL]. 2000. <http://www.hpl.hp.com/techreports/2000/HPL-2000-61.pdf>.
- [8] WebSphere Service Registry and Repository Handbook. IBM Redbooks publication [EB/OL]. 2007. <http://www.redbooks.ibm.com/redbooks/pdfs/sg247386.pdf>.
- [9] 赵海萍,赵东风. 基于 Web2K 服务器的 QoS 调度策略分析[J]. 计算机工程与应用,2005(13):125-128.
- [10] Nagel C, Evjen B, Glynn J. C# 高级编程[M]. 李敏波译. 北京:清华大学出版社,2006.