# 一种基于会话的 Web QoS 控制策略*)

杨 武[1,2] 李 波[2] 李双庆[1]
(重庆大学计算机科学与工程学院 重庆 400044)[1]
(重庆工学院计算机系 重庆 400050)[2]

**摘 要** 通过改进SBAC控制策略来实现会话级的Web QoS控制,采用对不用级别的用户实施不同的基于会话访问控制策略,使级别较高的用户获得较多的系统资源,从而获得更好的服务和更快的响应。

**关键词** 电子商务,Session,Web QoS,分类服务

## A Session-Based Web QoS Control Strategy

YANG Wu[1,2] LI Bo[2] LI Shuang-Qing[1]
(School of Computer Science and Engineering, Chongqing University, Chongqing 400044)[1]
(Department of Computer Science, Chongqing Institute of Technology, Chongqing 400050)[2]

**Abstract** In this paper, a session-based Web QoS control strategy is presented through mending the SBAC. Different access control strategy can be imposed to the different classes customers. Higher-class customers can get more system resource, so that they can receive better service and faster HTTP response.

**Keywords** E-commerce, Session, Web QoS, Classified service

## 1 引言

随着 Web 应用的迅猛发展,迫切需要高性能、高性价比和高可用性的 Web 服务器系统。近年来,网络 QoS 技术研究已经取得重要进展,包括通过建立 IntServ 和 DiffServ 体系结构来提供性能保证和服务区分。然而,如果 Web 服务器不支持 QoS 控制,那么在服务器过载的情况下,具备端到端网络 QoS 保证的高级数据流仍有可能遭受服务拒绝,或者 Web 服务的平均响应时间比用户的期望值高出多个数量级,从而导致事实上的拒绝服务效果。因此,Web 服务器已经在某种程度上成为实现端到端 QoS 的瓶颈。显然,真正保障用户 Web 应用的服务质量,还需要在服务器系统级提供相应的 QoS 支持。

## 2 基于会话的访问控制

通常的 Web 应用需要动态内容、后台数据库及安全性保证为支撑,而完成一个业务处理通常需要若干个步骤才能够实现,也就是需要一组 HTTP 请求来实现,这些完成一个业务处理所需要的一组 HTTP 请求的集合就是会话(Session)。

在 Web 服务器处理单个、相互之间没关系的请求(即不是事务性请求)的情况下,服务器的吞吐率由它的最大能力来衡量,如服务器能支持的最大连接数等。在服务器达到最大连接数的情况下,由于负载能力的限制,任何额外的请求都会被服务器丢弃。因此服务器在其最大负载的情况下,其吞吐率将大致会维持在一个固定值,即是最大连接数。然而,如果在过载的服务器上运行一个基于 Session 的负载时就会丢弃一个请求,则该请求就可能出现在 Session 的任何环节上,导致整个 Session 的中断和无法完成,使得以前处理过同一业务流程中的其他请求无效,其后果是浪费了服务器和网络资源。

在系统负载过重时,给不能提供服务的客户,一个明确的拒绝服务信息是很重要的。这样可以避免用户进行没有必要的多次重复尝试,这种重试会加大网络和服务器的负载,使得服务器的服务性能变得更差。因此,在不能为某个用户提供服务的情况下,给该用户一个明确的提示,例如 5 分钟后才有可能提供足够的资源保证为其服务,就可以避开负载的高峰时用户频繁重试带来的额外负担。

然而,发出一个明确的拒绝信息,也会增加了系统的额外负担。而且,当系统的负载越高,那么被拒绝的人越多,同时要发出的拒绝信息也越多,这就形成了一个恶性的循环。文[1]中对拒绝服务所带来的额外的服务器开销进行研究,得出了下面的结论:拒绝服务带来的额外的开销和 Session 的平均长度和承受的负载有关,系统的负载越大而 Session 越短,那么拒绝服务带来的额外开销越大,反之越小。

## 3 SBAC 策略

SBAC (Session Based Admission Control) 是 Hewlett-Packard Labs 提出的一种基于 Session 的访问控制策略[1]。SBAC 的主要思想是,防止服务器的超载。以前的统计表明,在服务器超载的情况下,能够顺利完成的 Session 服务中大多数是比较短的那部分,而请求数目较多的那些长的 Session 受到比较大的影响,就是被中断,从被中断的角度来看,对较长的 Session 存在着歧视的情况。然而,真正产生经济效益的 Session 中所包含的 HTTP 请求的数量通常是没有产生效益的 Session 的 2～3 倍,所以说长的 Session 有更大的可能性为商家带来更大的利润。Hp 实验室提出的 SBAC 策略就是针对这种结论提出的,它确保在服务器超载的情况下保证较长 Session 和较短的有同等的完成的机会。

SBAC 策略所实现的 Web 访问控制的目标是,保证所有的 Session 访问不受自身长度的影响,有同等的完成的机会;较长的 Session 和较短的具有同等的完成的机会;为了最大

*)项目资助:重庆市科技攻关项目,编号:2001.6715。杨 武 副教授,博士研究生,主要研究方向:分布式处理,计算机网络。

化地把系统资源用在有效请求的服务上，而使系统作的无效的服务所消耗的资源达到最小化，就是要保证已经接受了的Session能够被处理完，不会中途被中断。

## 4 改进的SBAC

SBAC虽然实现了基于Session级的访问的控制，但它只是把Session按长度来进行控制访问，所以说还没实现真正意义上的对Web服务器的QoS的支持。为了实现Session级的Web QoS控制，我们对SBAC控制策略进行一些了改进。

### 4.1 改进思路

为了实现Web QoS的区分服务，要对不同级别的用户提供不同的服务质量保证。而SBAC仅仅实现的是系统的访问控制，没分类服务的概念，对所有用户的请求都采用相同的访问控制策略，分类服务的支持很弱。因此我们可以通过调节不同级别的用户的访问控制策略的控制参数，来达到对不同等级的用户提供不同服务质量保证的目的。在服务器的负载到达一定的负载程度就对较低级别的用户采取较严格的访问控制策略，控制其能够获得的系统资源。而对较高级别则不采用或较弱的控制策略，仍然接受新的Session请求，当负载依次加重时，就依次对级别从低到高的用户进行访问控制。这样就达到分类服务的目的。

具体的改进思想是，对不同级别的用户采用不同的$U_{ac}$，对级别较低的用户采用较低的$U_{ac}$，那么当服务器的负载在较低的水平时候就会拒绝本级别的用户的新的Session请求，节省出系统资源；而由于未到达高级别的$U_{ac}$，因此还会接受高级别用户发出的Session服务请求，从而达到分类服务的目的。当系统超载很多时，就会对所有的用户的请求进行访问控制，以保证现有的Session的完成和系统资源的有效利用。

### 4.2 算法设计算法策略的提出

**定义1** 系统将用户的级别分为$n$等；级别为$i$的用户的请求$r_i$，$1\leqslant i\leqslant n$。

**定义2** 级别为$i$的用户负载控制的阈值：$U_i^{ac}$；级别为$i$的用户的稳定系数为$k_i$。

改进的SBAC伪代码描述如下：

```
f_ac(1)=U_ac
for(i =1 to n) begin
u_i^ac ←(initial value of class i)
end
i=0;
while(true) begin
if (ac_interval over) begin
        f_ac(i +1)=(1-k) * f_ac(i)+k * U_i^measured;
        U_{i+1}^observed = f_ac(i +1);
        U_i^measured = U_{i+1}^observed;
        i ++;
  end
Fetch next request r
If U_{i+1}^observed < u_i^ac
     receive r_i
  else
     reject r_i
```

改进的SBAC算法首先对负载函数和各个用户级别的负载阈值进行初始化；当服务器接收到请求，就判断现在的负载情况，如果系统的负载低于该级别的请求的阈值就接受该请求；如果高于该请求级别阈值，就拒绝该请求。

## 5 仿真实验

Web服务器QoS控制建模是一个非常复杂的问题，因为不同的策略需要不同的评价模型。目前还没有商用的软件来进行类似的评价、分析。我们采用了常规的Web负载模型和Web服务器模型。

### 5.1 负载模型

为了说明基于HTTP请求和基于Session的访问负载的不同，定义了一个负载模型。SpecWeb96[2]测试平台是测试Web服务器性能的一个工业标准，主要是产生静态的HTTP请求，该访问按照特定分布规律大小的文件。

服务器的性能就是用在访问用上述四类文件混合的文件时其每秒最大支持的连接数目来衡量。这些请求之间是没有任何关系的，是完全相互独立的。

商务网站应用则与上述访问有很大的不同，一个商务流程是由一系列步骤来组成的。只有当这一组的请求全部成功之后才能完成一个商务活动。因此，引入一个Session负载作为组成一个Session一组HTTP请求所产生的负载的总和。而Session都是由一组客户端的单个独立的HTTP请求所组成，从所有的请求的整体分布来看还是服从SpecWeb96的请求分布规律。在客户端，同一个Session中只有当收到前一个请求的响应之后才能发出下一个请求。同时下一个请求发出之前用户要花一定时间来浏览页面，再决定发出下一个请求，这段时间称作思考时间(thinking time)。当用户的请求没收到服务器响应时，总会等待一段时间，再次尝试发送该请求，这段时间叫做等待时间(timeout)。用户等待这段时间后，如果还没有响应，用户总会再次试图再次发出该请求。同时，用户通常尝试若干次之后没有收到请求才会放弃该请求。每个用户都有不同的级别，这就包含在整个Session的处理过程中。

因此，一个基于Session的客户端模型由以下参数来定义：

客户端地址：就是用户的浏览器的地址，包括ip地址和端口号等。

思考时间：两次请求之间，用户浏览网页，在决定下次的请求的时间。

延迟时间：是指从用户发出请求到由于没响应用户再次发出请求的时间间隔。

尝试次数：在Session中断以前，用户由于没收到响应重复发出请求的次数。

用户级别：系统分配给用的级别，决定用户所能得到的服务质量。

一个Session完成标准是它包含的所有请求都完成，所以我们用成功完成的Session的数目来评价Web服务器的能力。

### 5.2 服务器模型

基于服务器端的建立的模型，主要包括以下内容：一个Session负载的产生器；若干个客户端；一个Web服务器。Session负载产生器主要产生的负载由两个参数来描述：Session产生的负载；Session长度的分布规律。

每一个服务器都有一个监听队列(listen queue)，每收到一个客户端的请求，系统都会将该请求先放入监听队列中，监听的队列有个长度的限制。当监听队列满了时，就会拒绝新的请求。本模型中监听采用典型服务器的缺省值1024，当该队列满了时，就不能再接受新的请求。

通过这种方式，建立了一个Session产生器的模型，同一个Session中的所有HTTP请求都由同一个客户端产生和处理。客户端的请求等行为都限制在同一个环形回路中，客户端只有在收到下一个请求的响应之后才能发送下一个请求，这就模拟了Session的行为。

有两个原因可以导致一个请求和该请求的所属那个 Session 被中断：监听队列已满，服务器就拒绝在接受请求连接，所以导致请求被拒绝和该 Session 的中断。客户端发出请求，等待一定时间，经过等待时间时间之后，就会重新发出该请求，若这样重复若干次，当次数超过尝试次数的限制，还没有接收到请求的响应结果，那么就使得请求和 Session 中断。

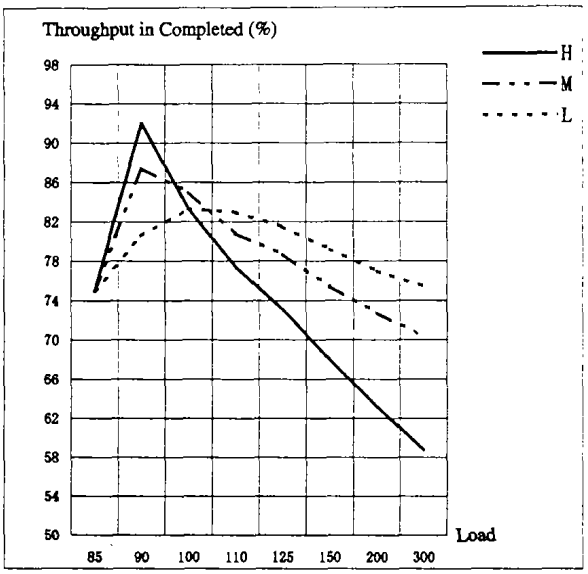


图 1　未采用 QoS 控制的 Session 吞吐率变化情况

### 5.3　仿真实验

下面的仿真实验就是依据 5.2 节的模型来进行的。在仿真实验中，假定服务器的处理能力是每秒钟 1000 个连接。在文[1]中，试验表明思考时间的长短对试验结果的影响很小而且它与 Session 的长度无关，因此我们可以假定一个固定的平均思考时间。在不影响结果的情况下，对服务器和客户端的参数进行了简化以提高效率，为了不失一般性，本文假定的客户端的参数如下：

同一个 Session 中两个请求之间的思考时间呈指数分布，平均长度是 5 秒；稳定系数 $k=09$；客户的发出请求没响应到再次发出相同请求的时间，等待时间是 1 秒；重复发送次数是 1 次；服务器上消耗的时间和请求的文件的大小呈线性正比的关系。

定义了三个级别的用户：$H$，$M$ 和 $L$，级别是从高到低。

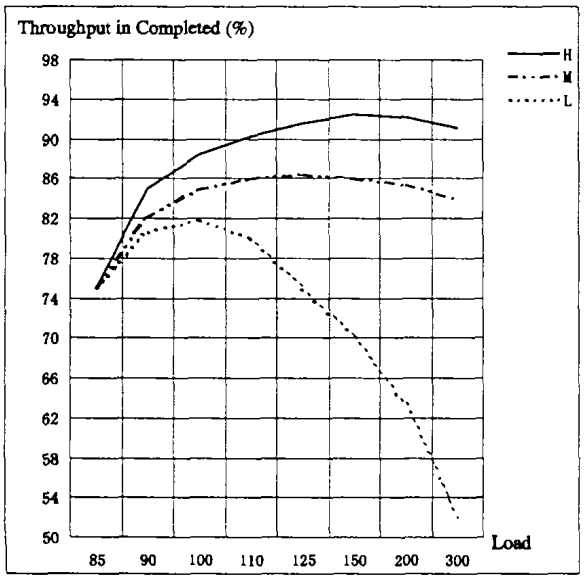


图 2　采用 QoS 控制的 Session 吞吐率变化情况

为了更清晰地了解 Web QoS 控制策略的效果，首先采用了无 Web QoS 控制的策略进行实验，获得的结果如图 1 所示。图 1 中的纵轴表示的是一 Session 来统计的吞吐率，也就是 Session 的完成率，横轴表示的服务器的负载。从图 1 中可以看出，由于没有对各种服务类加以区分，当服务器未达到其性能的极限时，各类用户的 Session 的吞吐率都上升。当系统负载量超过所能处理的上限时，各种服务类被平等处理，均出现了请求负载被丢弃的情况，使得三个级别的用户的 Session 吞吐率都相应的迅速下降。高级用户的下降速度最快，中级的次之，低级最慢。

采用了改进的 SBAC 控制策略的 Session 吞吐率变化情况如图 2 所示，明显看到，当服务器的超载加大时，级别高的 $H$ 用户的吞吐率变化不大，没有出现降低的情况，而中等级别的用户的 Session 吞吐率出现了幅度较小的降低；级别最低的 $L$ 降低的速度最快。说明了级别高的用户的吞吐率得到了保证，体现了较好的分类服务控制的效果。

**结束语**　本文首先分析了基于 Session 访问控制的特点，以及它与基于 HTTP 请求的访问控制的区别。介绍了一种基于 Session 的访问控制策略 SBAC，它的主要特点是在服务器超载的情况下，保证较长的 Session 能够完成。本文利用其控制策略的原理，考虑对用户进行分类控制，对不同级别的用户实施不同的访问控制策略，以实现 QoS 控制的目的。并且，建立了一个基于 Session 访问模拟的系统仿真模型，验证了策略的有效性。

## 参 考 文 献


1　Cherkasova L, Phaal P. Session Based Admission Control: a Mechanism for Improving the Performance of an Overhead Web Server: [HP Laboratories Report No. HPL-98-119]. June 1998. Http://www. hpl. hp. com/98/HPL-98-119. html 8/HPL-98-119. html

2　The WorkLoad for the SPECweb96 Benchmark. http://www. specbench. org/osg/web96/workload. html

3　Mohit A, Darren S, Peter D, Willy Z. Scalable Content-aware Request Distribution in Cluster-based Network Servers. In: Proc. of the 2000 Annual Usenix Technical Conference, San Diego, CA, June 2000

4　Andreolini M, Casalicchio E, Colajanni M, Mambelli M. QoS-aware switching policies for a locally distributed Web system. In: Proc. of the 11th Int'l World Wide Web Conf. Honolulu, Hawaii, May 2002

5　Valeria C, Michele C, Philip S Y. Dynamic Load Balancing on Web-Server Systems. IEEE Internet Computing, 1999(5-6): 28～39

6　Nochols K, et al. Definition of the Differentiated Service Field in the IPv4 and IPv6 Headers. RFC 2474, Network Working Group

7　Mohit A, Darren S, Peter D, Willy Z. Scalable Content-aware Request Distribution in Cluster-based Network Servers. In: Proc. of the 2000 Annual Usenix Technical Conf. San Diego, CA, June 2000

8　Vivek S, Mohit A. Locality-Aware Request Distribution in Cluster-based Network Servers. In: Proc. of ASPLOS-VIII, ACM SIG-PLAN, 1998. 205～216

9　Teo Y M, Ayani R. Comparison of Load Balancing Strategies on Cluster-based Web Servers. Transactions of the Society for Modeling and Simulation, 2001