一种异构 Web 服务器集群动态负载均衡算法

郭成城 晏蒲柳
(武汉大学电子信息学院 武昌 430079)

摘 要 针对 Web 服务器集群系统中负载动态变化特性,提出了一种临界加速递减动态请求负载分配算法.通过负载权值的等效变换更准确地反映集群中单台服务器的当前负载状态;通过临界递减因子来有效抑制服务器可能出现的"拒绝访问"现象;通过随机概率分配方式替代固定转发分配方式,使访问负载的分布更均匀;通过实际测试获取算法中所需的计算参数,使配置操作更为简单.实验结果表明,该算法对较大负载的文件集的大密度访问情况效果明显.

关键词 Web 服务器集群;负载均衡;临界加速递减
中图法分类号 TP301

A Dynamic Load-Balancing Algorithm for Heterogeneous Web Server Cluster

GUO Cheng-Cheng YAN Pu-Liu
(*School of Electronic Information, Wuhan University, Wuchang* 430079)

Abstract As increasing the embedded objects and the database searching tasks in Web Pages, there is larger difference among the loads of different server in a cluster system, which becomes more difficult for a heterogeneous Web server cluster to achieve high performance. In this paper, the authors present a dynamic load balancing algorithm MDC(Multiplicative Decrease in Critical area). For each of the servers in the cluster, the algorithm can more accurately evaluate the current load state by using the Equivalent Load-Alternant and can more efficiently restrain the occurring of the reject service phenomenon by using a special MDC operator. Besides, the authors apply a method of random distributing base-probability to assign each request to an appropriate server in terms of their weight. All the parameters that will be used in the algorithm can be acquired by simulated test. The authors also provide improved approximation results of above algorithm for the case where documents consist of relatively many embedded objects or database searches and lots of requests arrived the dispatcher synchronously.

Keywords Web server cluster; load-balancing; multiplicative decrease in critical area

1 引 言

随着 Internet 和 WWW(World Wide Web)应用的迅速普及,Web 服务器的性能成为影响网络服务质量的关键因素之一.网络集群服务器以其可扩展性、高可靠性和低性价比,为 Web 服务器系统带来了新的解决方案[1].Web 服务器集群是通过高速局域网互连的多台 Web 服务器,用户的 HTTP 请求被均衡地、透明地分配到集群中具体的服务器上,由其完成请求响应过程,并将响应结果返回给用户.由于多台服务器同时并行工作,因此,可以具有更高

收稿日期:2004-04-04;修改稿收到日期:2004-11-20.**郭成城**,男,1961 年生,博士研究生,副教授,研究方向为计算机通信网.E-mail:netccg@whu.edu.cn.**晏蒲柳**,女,1962 年生,博士,教授,博士生导师,研究方向为计算机通信网.

的响应效率以及更高的可靠性.越来越多的 Internet 信息服务系统采用这种解决方法,如 Yahoo,Netscape 等.著名的网络查询系统 Google 使用了几千台服务器组成的超级集群对外提供服务[2].

集群系统中的一个重要问题就是负载均衡问题.集群系统负载均衡的目标是根据处理机的性能来分配与其相称的任务,以最小化应用程序的执行时间[3].对 Web 服务器集群来说,就是要根据负载均衡原则,为每台服务器分配与之处理能力相匹配的请求任务量.通过合适的请求负载调度算法,可以避免集群中某些服务器工作负荷重,某些服务器工作负荷轻,达到平均响应时间短的目标.在基于 TCP 连接的第四层交换模式中,主要是依据服务器的负载状态来分配新到达的请求;而在基于请求内容的应用层交换模式中,则需要依据请求内容的位置特性和服务器的负载状态来分配新到达的请求到后端[4].

传统的 Web 请求分配算法主要是建立在访问请求泊松到达和响应时间指数分布的假设上,如轮转法(RR)、加权轮转法(WRR)、最小连接数法(LC)、加权最小连接数法(WLC)等.这类算法仅使用少量静态特征信息,适用于小规模的、单一配置的静态网页信息服务系统[5].由于网页中越来越多地使用动态嵌入对象(Scripts 程序)和数据库查询任务,使得不同请求任务的工作负荷有很大差异,静态网页和动态网页的负载可以相差 10 倍或 100 倍[6],Web 负载在到达时间间隔和响应时间上都明显呈现重尾特点[7];同时,同一个请求任务分配到不同服务器所产生的影响也不同.因此,如何准确地评估服务器上的工作负载并且分布新的请求任务到每台服务器成为 Web 集群系统取得负载均衡的关键.一些可以动态计算每个服务器负载权值(负载状态)并且可以动态调整请求分配方案的算法被相继提出.动态策略需要监测和确定每台服务器的当前负载状态,进行信息汇集和综合分析,并且,实时地做出分配决策.动态负载均衡算法适用于大规模、多配置的动态网页信息服务系统.

WRR_num 和 WRR_time[5]是基于轮转法的动态加权算法,前者依据活跃进程个数而后者依据请求响应时间计算每台服务器当前的负载权值.Round-trip 和 Xmitbyte[8]是基于最小连接数的动态加权算法,前者依据请求/响应间隔时间而后者依据平均输出流量计算每台服务器当前的负载权值.虽然,上述 4 个算法考虑了不同请求任务负载之间的差异,但没考虑每台服务器处理能力之间的差异.所以,主要适用于相同配置服务器组成的 Web 集群系统.对一台服务器处理能力的评估需要考虑多种因素,如 CPU、内存、总线、硬盘、网卡和操作系统等,任何一个部件发生拥塞都可能成为服务器的处理瓶颈.基于参数的加权负载分布策略[9]和选择加权百分比算法[10,11]是根据 CPU 利用率、内存利用率、存活的请求连接数、硬盘转速等参数计算服务器的负载权值,将服务器自身的处理能力与当前承受的工作负载联系在一起决定请求分配方案.这种算法的最大难度在于为每个被考虑的因素选择一个合适的计算系数,如果系数选取得不适当,很可能适得其反.

本文的后续部分,我们将提出一种算法配置相对简单、考虑服务器能力和请求负载差异影响的动态加权请求负载分配方法.

2 临界加速递减请求分配策略

算法设计思想:为了保证请求任务分配器具有较高的转发效率,分配器不对每个请求任务本身的负荷进行分析,而是动态监测请求任务被分配后对各个服务器工作负荷造成的影响;通过反馈服务器实际的工作负载状态,再结合设定的服务器固有处理能力,分配器计算出每台服务器当前的负载权值以及平均负载权值.根据服务器应分配与之能力相匹配的负载的原则,在后续的分配中,对高于平均负载权值的服务器减少分配给它的请求任务数量,对低于平均负载权值的服务器增加分配给它的请求任务数量,以达到负载均衡的目标.

2.1 负载权值

根据能力与负载相匹配原则,设集群服务器系统由 m 台服务器组成,其中服务器 $S_i(i=1,2,\cdots,m)$ 的固有能力为 ω_i,其当前的负载为 L_i,在集群服务器系统达到负载平衡时有

$$\frac{L_i}{\omega_i}=\frac{L_j}{\omega_j},\quad i\neq j;\ i,j=1,2,\cdots,m.$$

我们将某台服务器当前的负载与其固有能力之比定义为该服务器当前的负载权值,记为

$$W_L^i=\frac{L_i}{\omega_i} \tag{1}$$

集群服务器当前的平均负载权值:

$$\overline{W}_L=\frac{1}{m}\sum_{i=1}^{m}W_L^i \tag{2}$$

在集群服务器负载平衡时,有

$$W_L^1=W_L^2=\cdots=W_L^m=\overline{W}_L \tag{3}$$

为使集群服务器系统负载均衡,各服务器应以当前平均负载权值为基准调整各自的负载,因为,服务器的固有能力是常量.

2.2　负载权值的等效变换

要利用上述负载均衡原理,确定每台服务器的固有能力和当前负载是关键.首先,观察利用 HTTPF 软件对单台服务器的实际测试结果(如图 1).

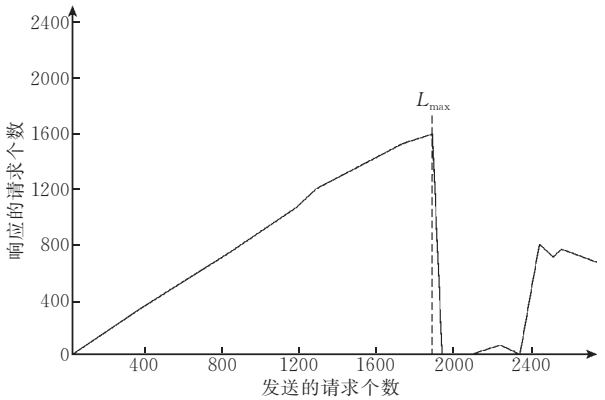

图 1　发送的请求个数与响应的请求个数关系图

从图 1 可以发现,当发送的请求数量达到一定值 L_{max} 时,服务器响应请求的数量突然下降为零,并持续一段时间.这是系统软件为防止死机而采取的一种措施,也称"拒绝服务"或"假死"现象."假死"会对服务器的性能产生严重影响,因此,必须避免这种现象发生[12].我们认为:服务器出现"假死"是因其服务能力已达到饱和状态,必须等待处理完一些已接收的任务后,才有可用的资源对外提供服务.

再观察测试的响应时间变化曲线图(如图 2).

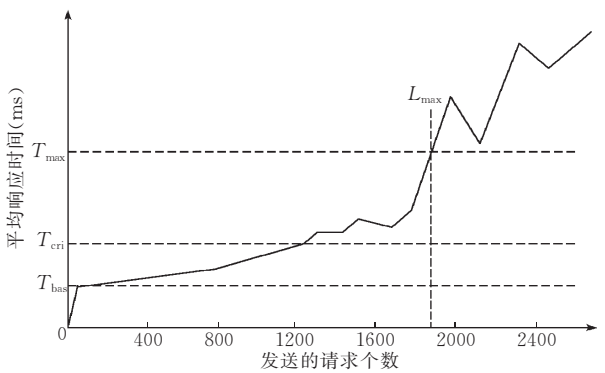

图 2　发送的请求个数与平均响应时间关系图

从图 2 可以看出,当服务器出现"假死"时,其响应时间急剧增加,超过 T_{max}.我们认为一旦请求响应时间超过 T_{max},则服务器的可用资源为零.从图 2 中还可以看出,只有一个最基本的请求任务加载到原本空载的服务器上时,请求的响应时间为 T_{bas},这一时间为系统自身的开销和响应开销.如果忽略这个小负荷的影响,T_{bas} 就是系统维持运转的基本负载.我们将($T_{max}-T_{bas}$)定义为服务器的固有能力,即服务器所有对外提供的能力,有 $\omega_i=(T^i_{max}-T^i_{bas})$.令对某台服务器实测的当前响应时间为 T_{now},我们将($T_{now}-T_{bas}$)定义为服务器的当前负载,有 $L_i=(T^i_{now}-T^i_{bas})$.将新定义的固有能力和当前负载代入公式(1),有

$$W_L^i=\frac{L_i}{\omega_i}=\frac{T^i_{now}-T^i_{bas}}{T^i_{max}-T^i_{bas}},\ \begin{cases}T^i_{now}\le T^i_{bas},\ T^i_{now}=T^i_{bas}\\ T^i_{now}\ge T^i_{max},\ T^i_{now}=T^i_{max}\end{cases} \tag{4}$$

公式(4)称为服务器 S_i 当前负载权值的等效变换.为了不受具体请求任务负载变化的干扰,我们由分配器统一向集群内所有服务器发送固定负载的测试请求,根据各台服务器对该请求的响应时间,分别计算它们的当前负载权值;而每台服务器的 T_{max} 和 T_{bas} 则预先测定(投入使用前)且作为参数录入.所以,公式(4)更准确地表现了服务器固有能力与其当前工作负载之间的关系.

2.3　临界递减因子

从图 2 中还可以明显看出,当请求负载增加到一定量时,响应时间的变化出现抖动,表明系统即将达到饱和状态.我们将这一负载区域定义为"假死"出现前的临界区,从响应时间上看为 $T_{cri}\sim T_{max}$ 区间.

通过增加进入临界状态服务器的负载权值(即减轻下阶段的工作负载),从而可达到抑制服务器进入饱和状态的目的.我们定义负载权值增加的部分为递减因子 δ.加入递减因子后的负载权值计算公式和递减因子的计算公式如下:

$$W_L^i=\frac{L_i}{\omega_i}+\delta=\frac{T^i_{now}-T^i_{bas}}{T^i_{max}-T^i_{bas}}+\delta \tag{5}$$

$$\delta=\begin{cases}0, & T^i_{now}<T^i_{cri}\\ \dfrac{1}{\sigma}\left(1-\dfrac{T^i_{cri}}{T^i_{max}}\right), & T^i_{now}\ge T^i_{cri}\end{cases} \tag{6}$$

上述公式中,递减因子只在服务器负载进入临界区后才产生作用,且 $1-\frac{T^i_{cri}}{T^i_{max}}$ 为服务器的临界深度.由于每台服务器的临界深度不一样,我们设置参数 σ 来调整其递减的速度(通常 σ 取值 2).

有了递减因子,就可以主动控制服务器不出现"假死"状态.在实现的程序中,我们禁止负载权值为 1 的服务器参加下一阶段的请求分配过程,这样就避免了服务器上原有的请求任务受系统能力饱和影响出现响应时间急剧增加的情况;并对负载权值为

1 的服务器要进行告警,如果多台服务器告警则启动冗余备份的服务器来分担请求负载. 如何激活备份服务器不在本算法讨论范围,在此不作论述.

由于对进入临界状态的服务器采用增加其负载权值、加倍减少后续阶段分配负载的策略,我们的算法也称为临界加速递减(Multiplicative Decrease in Critical area,MDC)负载分配策略.

2.4 分配权值

计算出了每台服务器的负载权值后,就可以根据公式(2)得到平均负载权值. 在此基础上,通过调整下一阶段分配到每台服务器的请求负载达到负载均衡目标. 由于无法确知后续请求数量和负载状况,我们依据当前的请求负载状况(负载权值、分配的请求个数等)来推算应做出的调整.

令达到负载均衡状态时服务器 S_i 分配的请求个数为 $\bar{n}_i$,对应的负载状况为 $\bar{L}_i$;而服务器当前分配的请求个数 n_i 和负载状况 L_i 都可以测得. 假设负载条件相同,那么,分配的请求数量的变化与负载的变化应是一致的,即

$$\frac{\bar{n}_i}{n_i}=\frac{\bar{L}_i}{L_i} \tag{7}$$

根据公式(1),可以得到

$$L_i=W_L^i\times\omega_i,\ \bar{L}_i=\bar{W}_L^i\times\omega_i=\bar{W}_L\times\omega_i,$$

代入公式(7),有

$$\bar{n}_i=\frac{\bar{L}_i}{L_i}\times n_i=\frac{\bar{W}_L}{W_L^i}\times n_i \tag{8}$$

公式(8)计算的就是下阶段服务器 S_i 应被分配的请求个数的推算值,由于请求不可能被分割,故实现应用中取整数值. 对于当前负载权值大于平均负载权值的服务器来说则会减少所分配的请求任务数量;反之,则会增加服务器所分配的请求任务数量.

我们将每台服务器应分配的请求个数在整个集群系统应达到的请求个数中所占的比例定义为该服务器的分配权值,记为 W_A^i,计算公式:

$$W_A^i=\frac{\tilde{n}_i}{\sum_{i=1}^{m}\tilde{n}_i} \tag{9}$$

分配器按照集群中每台服务器的分配权值为其分配一定的请求任务负载.

2.5 随机概率转发

目前的静态和动态加权分配算法中大都使用固定比例的分配模式,分配比例与权值对应. 由于无法控制请求任务具体到达的时间和数量,这种固定转发模式本身就可能造成请求负载分布的不均匀①. 为了避免两次分配周期间的关联影响,我们采用了随机概率转发模式.

首先,根据每台服务器计算出的分配权值确定其概率空间,整个集群服务器系统的空间为 1. 图 3 给出了 4 台分配权值分别为 0.1,0.15,0.25,0.5 的服务器的概率空间.

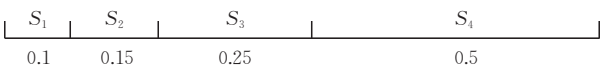

图 3 服务器的概率空间分布

当一个新的请求任务到达分配器时,临时计算出一个[0,1]间的随机数,根据该随机数在概率空间内的落点,确定本次转发的目标服务器.

对于具有较大分配权值的服务器,因占有较大的概率空间,自然具有较高的分配几率,即被分配更多的请求任务;同时,每次转发都是一个独立的事件,既不受上次转发的影响,也不影响下一次的转发操作. 从而,可以达到更为均匀的转发效果.

3 算法性能测试

我们根据实验室现有的条件,对所实现的请求分配算法进行了性能测试.

3.1 测试环境

后端采用四台不同配置的服务器,分别为:(1)双 P-Ⅳ 1.8G CPU、512MDDR 内存、SCSI 15.8G×2 硬盘(RAID 0);(2)单 P-Ⅳ 1.8G CPU、256MDDR 内存、IDE40G×2 硬盘(RAID 0);(3)单 P-Ⅳ 1.8G CPU、256MDDR 内存、IDE40G 硬盘. 前端采用一台专用的集群分配器 LAS100,其理论最大转发速率为每秒 20000 个 TCP 连接. 客户端采用十台普通 PC 机. 集群内部网络连接设备采用 100Mb/s 的 Cicso2924 交换机.

前端分配器和后端服务器上运行我们设计的系统软件,分配算法包括了静态加权轮转法(WRR)、基于请求数量的加权最小连接数优先法(WLC)、基于响应时间的动态加权轮转法(DWRR)、临界加速递减法(MDC)等. 客户端运行 Web 请求仿真软件,请求事件流为符合负指数分布的 Poisson 事件流.

根据目前的 Web 访问负载特性,我们在服务器端设置了两组不同负载类型的文件集. 第 1 组由长度较短的静态网页文本、多媒体文件和小的程序脚

① Zhang Wen-Song. Linux Server Cluster System: The Project Introduction. http://www-900.ibm.com/developerWorks/cn/linux/theme/special/index.shtml#cluster

本组成;第 2 组由一些长静态网页文本、复杂计算和数据库查询程序组成.每组中不同负载文件的数量分布按照对数正态分布和 Pareto 分布的混合模型来设置[14],其特征满足重尾特性.

3.2　测试结果

分别使用上述四种请求分配算法测试对不同负载文件集的访问情况,观察请求任务平均响应(完成)时间的变化.对每种算法选择 8 个检测点,每个点采样 5 次,以平均值作为该检测点的测试结果.测试结果如图 4,图 5 所示.

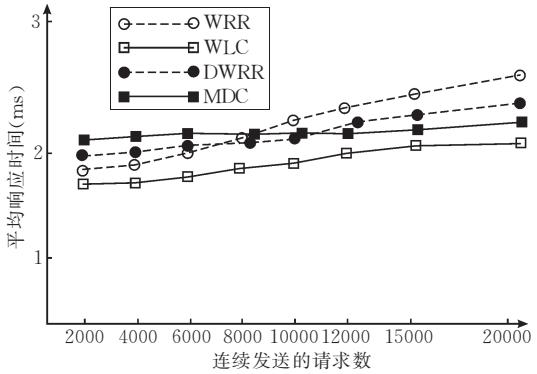

图 4　第 1 组访问负载条件下的平均请求响应时间

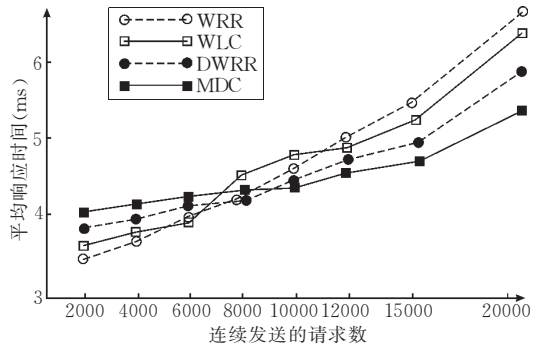

图 5　第 2 组访问负载条件下的平均请求响应时间

从图 4 可以看出,针对轻负载文件集的访问,四种分配算法的平均响应时间差不多,相比较而言 WLC 算法最好,这是因为每个请求都能够比较快地被响应完成,累计负载误差不大,所以连接个数可以比较准确地反映负载状态.由于 MDC 算法自身的开销要大于其他算法,所以在请求任务个数比较少时(小于 10000),其平均响应时间略大于其他算法,但随着请求个数的增加,这种差异逐渐消除.

从图 5 可以看出,针对重负载文件集的访问,当请求任务数比较多时(大于 6000～8000),MDC 算法的优势就很明显了.比较而言,WRR 的平均响应完成时间增涨最快,DWRR 和 WLC 好于 WRR,MDC 的平均响应完成时间增长最慢.在测试中 WLC 算法每次测试值的跳变比较大,这是因为随被访问文件的负载差异加大,通过连接个数产生的累计误差也较大所造成的.

从图 4 和图 5 可以看出,MDC 算法的平均响应时间变化曲线最为平滑,这说明 MDC 算法在对集群服务器当前负载的评估更准确,而随机概率分配策略使访问负载的分布更均匀.

4　总　结

针对 Web 服务器集群系统中请求负载分配问题,我们在本文中提出了一种临界加速递减动态请求负载分配算法.该算法的主要特点:通过负载权值的等效变换更准确地反映了 Web 集群中每台服务器的当前负载状态,为均衡地分配请求负载奠定了基础;通过临界递减因子的使用有效地抑制了服务器出现"拒绝访问"现象,避免了单台服务器性能严重下降对集群整体性能造成的负面影响;通过随机概率分配方法使访问负载的分布更均匀,消除了采用固定比例模式中分配周期间的关联效应;通过预先的测试可以获得算法中所需的配置参数,使得算法的配置操作比较简单.实验数据表明,该算法对大负载文件的大密度访问情况可以取得较为有效的负载均衡分配效果.

参　考　文　献

1　Li Chuan Chen, Hyeon Ah Choi. Approximation algorithms for data distribution with load balancing of Web servers. In: Proceedings of IEEE International Conference on Cluster Computing, 2001, 274～281

2　Athanasion E. Papthanasion, Eric Van Hensbergen. KNITS: Switch-based connection Hand-off. In: Proceedings of INFOCOM 2002, Twenty-first Annual Joint Conference of the IEEE Computer and Communications Societies, 2002, 1: 332～341

3　Buyya Rajkumar. High Performance Cluster Computing Architectures and System. Prentice Hall, 2000

4　Pai Vivel S., Aron Mohit, Banga Gauray. Locality-aware request distribution in cluster-based network servers. In: Proceedings of the 8th ACM Conference on Architectural Support for Programming Languages and Operating System. San Jose, CA, 1998, 205～216

5　Casslicchio Emiliano, Tucci Salvatore. Static and Dynamic scheduling algorithm for scalable Web server farm. In: Proceedings of the IEEE 9th Euromicro Workshop on Parallel and Distributed Processing, 2001, 369～376

6　Iyengar Arun, MacNair Ed, Nguyen Thao. An analysis of

Web server performance. In: Proceedings of Global Telecommunications Conference, 1997, 3: 1943～1947

7 Hao Qin-Fen, Zhu Ming-Fa, Hao Ji-Sheng. WWW traffic access characteristic distribution research. Journal of Computer Research & Development, 2001, 38(10): 1172～1180(in Chinese)
(郝沁汾,祝明发,郝继生. WWW 业务访问特性分布研究. 计算机研究与发展, 2001, 38(10): 1172～1180)

8 Bryhni Haakan. A comparison of load balancing techniques for scalable Web servers. IEEE Network, 2001,(7/8): 58～64

9 Yu Lei, Lin Zong-Kai, Guo Yu-Chai, Lin Shuo-Xun. Load balancing and fault-tolerant services in multi-server system. Journal of System Simulation, 2001, 13(3): 325～328(in Chinese)
(于 磊,林宗楷,郭玉钗,林守勋. 多服务器系统中的负载平衡与容错. 系统仿真学报, 2001, 13(3): 325～328)

10 Shan Zhi-Guang, Dai Qiong-hai, Lin Chuang, Yang Yang. Integrated schemes of Web request dispatching and selecting and their performance analysis. Journal of Software, 2001, 12(3): 354～366(in Chinese)
(单志广,戴琼海,林 闯,杨 扬. Web 请求分配和选择的综合方案与性能分析. 软件学报, 2001, 12(3): 354～366)

11 Li Shuang-Qing, Gu Ping, Cheng Dai-Jie. Analysis and research on load balancing strategy in Web cluster system. Journal of Computer Engineering and Application, 2002, (19): 40～43(in Chinese)
(李双庆,古 平,程代杰. Web 系统负载均衡策略分析与研究. 计算机工程与应用, 2002, (19): 40～43)

12 Hwang Suntae, Jung Naksoo. Dynamic scheduling of Web server cluster. In: Proceedings of IEEE 9th International Conference Parallel and Distributed System, 2002, 563～568

13 Yi Qi-Na, Guo Cheng-Cheng, Yan Pu-Liu, Xiong Zhi. A realization of testing algorithm for Web server's performance. Journal of System Simulation, 2004, 16(1): 25～28(in Chinese)
(易琦娜,郭成城,晏蒲柳,熊 智. 一种服务器性能测试仿真算法的实现. 系统仿真学报, 2004, 16(1): 25～28)

GUO Cheng-Cheng, born in 1961, Ph. D. candidate, associate professor. His research interests include computer network and network communication.

YAN Pu-Liu, born in 1962, professor and Ph. D. supervisor. Her research interests include network communication and network management.

Background

This work is supported by the project named "The network server cluster with high performance"(No. 20001001004), which is the one of important projects of science research of Wuhan. For the project, we mainly study the technologies on Web cluster and implement a Web servers system. Our research include the mechanism of dispatcher, the algorithm of load balancing, Web caching policy in memory, the strategy of Web documents clustering and distributing. The algorithm of load balancing is a key by which the cluster system can achieve higher performance.