一种基于 WEB 集群的负载均衡算法①

Load Balancing Algorithm Based on WEB Cluster

郑　祺　(浙江科技学院 信息学院 浙江 杭州 310023)

摘　要： 集群技术为 Web 服务带来了新的解决方案。针对传统的动态负反馈算法的一些不足，提出了一种算法相对简单的临界加速递减的动态请求负载均衡算法，通过负载权值的等效变换和进入临界状态后的动态权值调整，有效的防止服务器负载倾斜，达到较好的负载均衡效果。

关键词： 集群 负载均衡 动态反馈 权值 临界加速递减

1　引言

近年来，随着 Internet 的飞速发展以及电子商务、多媒体技术等的广泛应用，网上信息流量正在呈指数增长，人们对服务质量的要求也日益提高，这对 Web 服务器的处理能力提出了严峻的挑战。如果数据量超出了服务器能承受的范围，必然会导致服务质量下降，这将直接造成用户的流失，提高服务器性能及数据的处理能力已经成为一个急需解决的问题。由于单台服务器的性能总是有限的，仅仅通过提高服务器的硬件性能并不能从根本上解决问题。集群服务器以其高可扩展性、高可靠性和高性价比，为 Web 服务器系统带来了新的解决方案[1]。

集群的负载均衡策略是提高集群整体性能的关键，其目的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。它将用户的请求按照一定的算法分发到集群的某个节点服务器上以实现对用户请求的并行处理，最大限度的利用各节点的处理能力，实现集群系统的负载均衡，提高集群系统的整体性能。

2　传统负载均衡算法

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相应时间指数分布的假设上，常用的如轮转法(Round－Robin)[2]、加权轮转法(Weighted Round-Robin)[2]、最小连接数法(Least-Connection)[2]及加权最小连接数法(Weighted Least-Connection)[2]等。由于 WEB 网站中既有大量静态浏览页面，又大量使用了动态嵌入对象技术和数据库操作任务，这使得不同请求任务的工作负载有很大差异，如静态网页和动态网页的负载可能相差 10 倍甚至 100 倍[3]。同时，由于集群内各节点服务器间可能存在性能差异，因而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，如何准确的评估服务器上的工作负载成为了实现集群负载均衡的关键。以上这些传统算法都没有考虑到用户请求任务间的差异以及集群内各节点服务器的性能差异，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。这些算法都只能做到负载的大致“均衡”分配，并不能有效地解决集群内各服务器间的负载均衡问题，也无法充分利用各节点服务器的处理能力。

当用户访问集群时，由于每个用户的具体需求和使用方式不同，其所要求的处理时间和所消耗的系统资源是不同的，它依赖于很多因素：用户请求的服务类型、当前网络带宽的情况、当前服务器的资源使用情况等。对用户请求处理时间的不同可能会导致节点服务器利用率的倾斜，即节点负载的不均衡，有可能存在这样的情况：有些节点已经超负荷运行，而其它节点则相对空闲，这会导致集群性能的急剧下降，严重影响对外的服务质量。

采用基于负反馈机制的动态负载均衡算法能够通过周期性的反馈信息不断调整用户请求分布的比例，

① 收稿时间:2008-11-18

充分考虑到每个节点的负载水平和响应能力，尽可能避免因节点负载不均衡而引起的部分节点过载的情况，从而有效提高集群的整体性能。

对一台服务器处理能力的计算一般都要考虑多种因素：如 CPU 资源、内存资源、系统当前进程数、磁盘 I/O 频度和响应时间等，任何一个因素都有可能成为服务器的处理瓶颈。大多数基于参数的加权负载分布策略和选择加权比例算法通常是根据 CPU 利用率、内存利用率、存活的请求连接数、磁盘 I/O 频度以及相应时间等参数来计算服务器的负载权值，然后根据负载权值来决定请求分配方案。这种算法的最大难度在于需要为每个因素选择一个合适的计算系数，如果系数选取不当，很可能适得其反[4]。此外对于一些节点数量较多的集群来说，参数的动态采集和权值计算也会大量消耗系统及网络资源，从而影响集群的整体性能。

本文的后续部分将提出一种算法相对简单、考虑服务器处理能力和输入参数的动态负载均衡算法。

3　动态负载均衡算法

算法设计思想：为了保证控制器(Director)的转发效率，控制器不检查每个请求任务的类型，只周期性地监测各节点服务器(Node Server)的负载状况，通过反馈节点服务器的响应时间，控制器等效计算出每个节点服务器当前的负载权值，并根据实际负载状况进行相应调整，然后再采用加权轮转法对用户请求进行调度，此外为尽量避免或减少突发请求情况对集群性能的影响，考虑进一步引入输入参数来动态调整节点的负载权值。

3.1 权值计算

节点的初始权值计算主要考虑以下参数：CPU 处理速度、内存容量、系统 I/O 速率和网络带宽等，节点初始权值的计算公式如下：

$$Load(N_i)=R_1K_i\frac{L_{CPU}(N_i)}{BASE_{CPU}}+R_2\frac{L_M(N_i)}{BASE_M}+R_3\frac{L_{I/O}(N_i)}{BASE_{I/O}}+R_4\frac{L_{NET}(N_i)}{BASE_{NET}} \quad (1)$$

其中，$L_f(N_i)$表示节点 N_i 当前某一参数的值，以上公式中依次表示为：CPU 处理速度、内存容量、系统 I/O 速率和节点的网络带宽。K_i为处理器系数(根据节点的处理器数量确定)。$BASE_f$ 是某一参数的基准值，依次为基准 CPU 处理速度、基准内存容量、基准 I/O 速率和基准网络带宽，这些基准值可根据集群系统的实际情况统计确定。R_i 是为每个参数设定的一个可调常量系数，用以区分各个参数的重要程度，可以根据系统实际测试进行设置，以期达到最佳状态，其中 $\Sigma R_i=1$。一组典型设置可采用：(0.3、0.2、0、0.5)。

考虑到采用动态反馈算法确定权值需要周期性的对节点进行参数采集、计算，这需要消耗一定的系统及网络资源且存在计算系数选择困难的问题，因此考虑采用一种相对简单的方法来进行节点的权值等效变换，以达到提高集群整体效率的目标。

通过对单台服务器的响应状况(如图 1)和响应时间(如图 2)进行观察[4]：

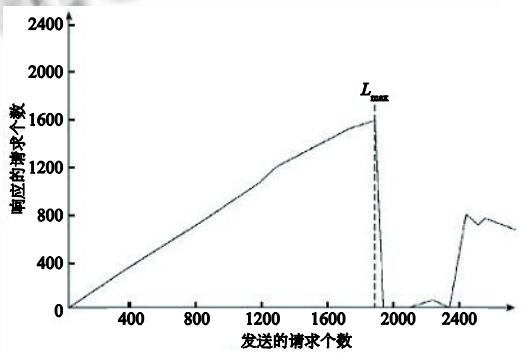

图 1　发送的请求个数与响应的请求个数关系图

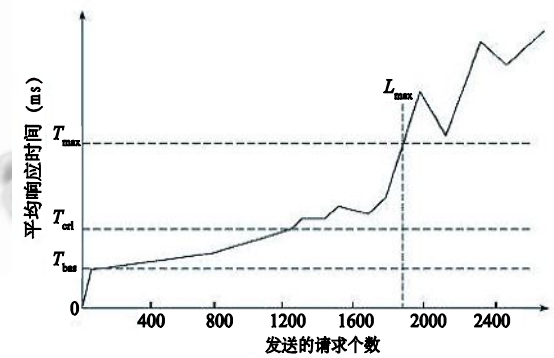

图 2　发送的请求个数与平均响应时间关系图

我们可以发现当发送的请求数量达到一定值 Lmax 时，服务器响应请求的数量突然下降为零并会持续一段时间，这是系统软件为防止死机而采取的一种措施，也叫“假死”现象，它会对服务器的性能产生严重影响，因此必须避免这种情况发生。当服务器出现“假死”现象时，其响应时间会急剧增加，会超过 Tmax。这时可以认为一旦请求响应时间超过 Tmax，则服务器的可用资源为零。从图 2 还可看出，

只有一个最基本的请求任务加载到空载服务器时，服务器的请求响应时间为 Tbas，如果忽略此请求的影响，则 Tbas 就是服务器维持运转的基本负载[4]。可将(Tmax-Tbas)定义为服务器对外提供的服务能力，设 Tnow 为某台服务器当前的响应时间，则可用(Tmax－Tnow)来表示服务器当前的剩余服务能力。由此我们可以得出节点当前权值的等效变换：

$$W'(N_i)=Load(N_i)\times\frac{T^i_{\max}-T^i_{now}}{T^i_{\max}-T^i_{bas}},\begin{cases}T^i_{now}\le T^i_{bas},T^i_{now}=T^i_{bas}\\T^i_{now}\ge T^i_{\max},T^i_{now}=T^i_{\max}\end{cases}\quad(2)$$

3.2 控制器调度算法

控制器根据各节点的反馈信息可计算出各节点的当前权值，然后再采用加权轮转法对用户请求进行调度，即根据节点权值的大小按照轮转的方式将用户请求分配到各节点去，权值越高的节点分配到的用户请求就越多，每个节点所分配到的用户请求数是按其权值占权值总和的比例来确定的。

3.3 权值的动态调整

从图 2 可看出当请求负载到达一定量时，对应的响应时间的变化会出现抖动现象，这表明系统将进入饱和状态，我们将此区域定义为系统进入饱和状态前的临界区，从响应时间上看为 Tcri～Tmax 区间[4]。为达到抑制服务器进入饱和状态的目的，我们可通过适当降低进入临界区的服务器权值来减轻下阶段的工作负载。具体调整公式如下：

$$W(N_i)=W'(N_i)\times K,K=\begin{cases}1,T^i_{now}\le T^i_{cri}\\\alpha(1-\frac{T^i_{now}}{T^i_{\max}}),T^i_{now}>T^i_{cri}\end{cases}\quad(3)$$

其中，K 是调节系数，只有当服务器负载进入临界区后才产生作用，系统负载越重则 K 值越小，这样可以使临界深度大的服务器加快递减速度，以避免其进入饱和状态。值可用来调节递减速度，通常取 2。

此外，由于网络数据具有间歇性、突发性特点，网络流量是呈波浪型发生的，即不定期的会出现一段突发的大流量[5]，这种突发的用户请求很可能会引起集群中的某些节点过载，从而导致整个集群的吞吐率急剧下降。图 3 指出了突发请求情况(Burst Condition)对服务器性能的影响[6]。这里使用了两个参数来表示突发状态：一是最大请求速率与平均请求速率的比值；二是请求速率超过平均请求速率的时间百分比。例如 Burst(6,5)表示有 5%的时间请求速率是平均请求速率的 6 倍。由图可见，少量的高突发请求就会严重降低服务器的吞吐率，因此有必要将这种情况考虑进集群的调度算法中，以避免因高突发请求而导致的集群性能下降的情况[7]。

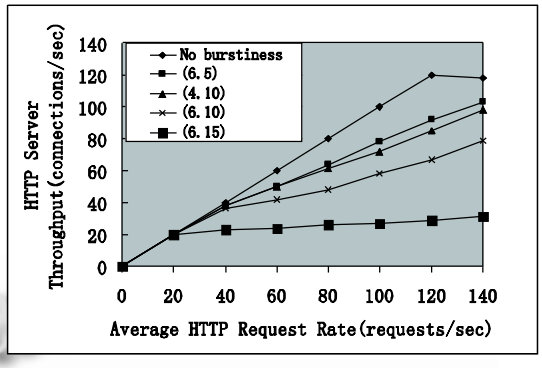

图 3　突发状态下请求速率与吞吐率的关系

我们再定义一个动态修正权值，用以在两次参数采集的间隔时间内动态调整节点的权值：

$$Value(N_i)=\beta\times\frac{T^i_{\max}-T^i_{now}}{T^i_{\max}-T^i_{bas}}\times Burst(N_i)\quad(4)$$

其中，β是一个动态调整系数，可根据集群的具体应用而定，如以 WEB 服务为主，则β可取一较小值；若以数据库查询或文件服务为主，则β的值应相应加大；Burst(Ni)是该节点在单位时间内收到的用户请求数与正常情况下的平均请求数之比。

结合各节点的采集权值和修正权值，可计算出最终的动态权值：

$$Weight(N_i)=W(N_i)-Value(N_i)\quad(5)$$

当 Value 值高时，说明该节点可能当前负载较重或者突发情况较严重，因此应相应降低该节点的权值，以减少分配到该节点的请求，避免该节点出现过载死锁情况，保证集群的整体性能。当 Weight 值小于等于零时，说明该节点可能已进入“假死”状态，此时将不再向该节点分配请求，并同时向管理员发送过载警告。

4　性能测试

我们使用了 NAT 方式构造了一个 LVS 集群[8]，集群内有 3 台节点服务器，配置分别为双 CPU 服务器、单 CPU 服务器和普通 PC，另使用 2 台 PC 作为客户端对集群进行模拟加压测试，使用的工具是 Linux 下

的 Httperf，其中一台用以模拟正常请求数据流，另一台则模拟突发请求数据流。选择的对比测试算法为加权轮转法(WRR)。

性能对比测试结果如图 4 所示，可以看出采用本算法(FB)可以较好的平衡负载，尤其是在突发较严重的情况下更能够充分的利用各节点的资源，从而有效提高集群的整体性能。

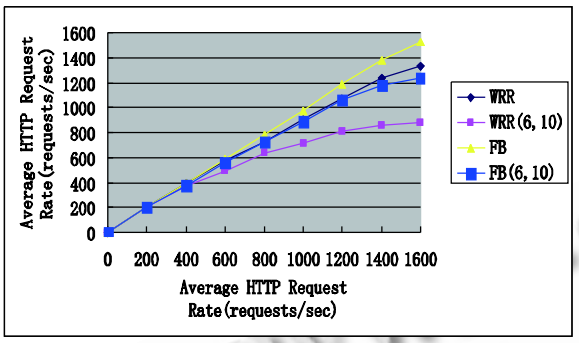

图 4 算法测试结果

5 结束语

集群负载均衡策略是提高集群整体性能的关键，考虑到用户请求间的差异性和集群内部节点服务器间性能的差异性，通过简单的等效计算节点权值和对用户请求速率变化的计算来合理分配用户请求，以平衡节点的负载，充分利用各节点的资源。通过测试表明本算法基本可行并明显优于常用的加权轮转法，可以有效提高集群的性能，尤其是在集群内部节点服务器性能不一的情况下，可以取得较为有效的负载均衡分配效果。

参考文献

1 Chen LC， Choi HA. Approximation algorithms for data distribution with load balance of Web servers. Proceedings of IEEE International Conference on Cluster Computing, 2001:274－281.

2 Colajanni M, et al. Dynamic Load Balancing in Geographically Distributed Heterogeneous Web Servers. Proc.of 18th IEEE Int'l Conf. On Distributed Computing System (ICDCS 1998), Amsterdam, The Netherlands, May 1998:295－303.

3 Arun I, Ed M, Thao N. An analusis of Web server performance. Proceedings of Global Telecommunications Conference, 1997,3:1943－1947.

4 郭成城,晏蒲柳.一种异构 Web 服务器集群动态负载均衡算法.计算机学报, 2005,28(2):179－184.

5 Stalling W. Self-similarity Upsets Data Traffic Assumptions. IEEE Spectrum, 1997,6:28－29.

6 Gaurav Banga，et al. Measuring the Capacity of a Web Server. Proceedings of the USENIX Symposium on Internet Technologies and Systems. Monterey, CA, USA, 1997:61－71.

7 吴松涛等. 请求速率对集群 Web 服务器调度的影响. 计算机应用研究, 2003,6:139－140.

8 马双良,张英敏,宋丽君.基于 LVS 和计算任务的实时集群负载均衡方法.计算机工程与设计, 2007,10(20):4934－4937.