文章编号:1001 - 9081(2006)02 - 0307 - 03

基于双层分配器的 Web 服务集群负载平衡解决方案

王志晓,牛 强
(中国矿业大学 计算机科学与技术学院,江苏 徐州 221008)
(zhxwang@cumt.edu.cn)

摘 要:为了解决目前 Web 服务集群存在的问题,提出了一个双层分配器负载平衡模型。在确定了负载指标的基础上,给出了该模型的负载平衡算法。并就服务器迅速过载问题和容错性问题提出了初步的解决方案。最后,对模型的性能进行了测试,结果表明模型达到了较好的负载平衡。

关键词:Web 服务集群;双层分配器;负载平衡
中图分类号: TP393.02 **文献标识码**:A

Double-dispatcher-based load balancing solution on Web service cluster

WANG Zhi-xiao, NIU Qiang
(*College of Computer Science and Technology, China University of Mining and Technology, Xuzhou Jiangsu* 221008, *China*)

Abstract: A double-dispatcher-based load balancing model of Web service cluster was given. The purpose was to solve the problems existing in the current web service cluster. One was that the load index was not desirable enough; the other was that documents distribution was not accurate enough. After making sure of the load index, the load balancing algorithm for the model was put forward. In addition, the preliminary scheme on the servers' instant over loading and fault tolerance was presented. At last, a test on the performance of the model shows that the model has better load balancing.

Key words: Web service cluster; double-dispatcher; load balancing

0 引言

随着 Internet 的迅速发展,网络服务器需要为越来越多的用户提供服务。在这种背景下,Web 服务集群应运而生。目前有很多 Web 服务集群解决方案,这些方案都在一定程度上收到了较好的效果。但是也存在一些问题:

1) 在进行负载分配的时候只考虑服务器当前连接的数量和服务器本身的处理能力而没有考虑这些连接的处理强度的不同,有些解决方案甚至连服务器本身处理能力的因素都忽略了。这导致负载指标不能真实地反映服务器的负载,系统不能达到良好的负载平衡,从而不能实现"能者多劳"的目的。

2) 关于 Web 文档分配的问题。Web 服务集群目前普遍的做法是每个 Web 服务器都包含全部的文档。文献[1]的研究成果表明,如果文档位置分配不当,将会降低整个系统的性能。所以需要寻找一个合适方法来进行文档的分配。

为此本文构建一个 Web 服务集群负载平衡模型来解决上述问题。

1 Web 服务集群负载平衡模型

整个模型的结构如图 1 所示,分为三个层次:客户层、请求分配器层和 Web 服务器层。该模型所要达到的总体目标是:"文档分布到位,负载分配合理"。

1.1 请求分配器层

请求分配器层是整个系统对外的唯一接口,负责分配来自客户端的所有请求,是实现负载平衡的关键所在。

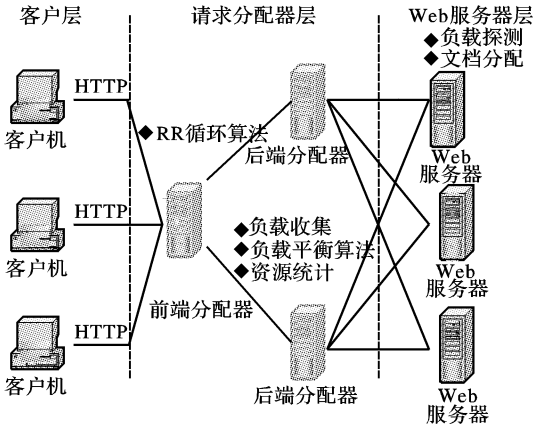

图 1 负载平衡模型

对一些访问量较大的 Web 站点来说,如果短时间内有大量的用户请求同时到达,请求分配器的存在将成为整个系统的瓶颈。为了解决这个问题,本文将请求分配器分成两层:前端分配器和后端分配器,如图 1 所示。两者有不同的侧重点,通过相互配合进行负载分配。

1) 前端分配器的主要功能

前端分配器运行的是非常简单的轮循均衡(Round Robin)算法,在进行请求分配的时候不考虑服务器的负载问题,只是简单的快速转发。目的是尽快地将用户请求分配到后端分配器,避免请求分配器层成为系统的瓶颈。

2) 后端分配器的主要功能

请求分配

系统负载平衡的真正实现是由后端分配器完成的。后端

收稿日期:2005 - 08 - 30;**修订日期**:2005 - 10 - 31 **基金项目**:中国矿业大学青年科研基金资助项目(004490)

作者简介:王志晓(1979 -),男,河南叶县人,硕士研究生,主要研究方向:集群技术、分布式计算; 牛强(1974 -),男,辽宁沈阳人,博士研究生,主要研究方向:数据挖掘.

分配器运行负载平衡算法进行请求分配,该算法充分考虑了服务器负载和文档分配情况。

负载信息搜集

要分配就需要知道各个 Web 服务器当前的负载情况。Web 服务器进行负载信息探测,定期向后端分配器提供自己的最新负载信息。后端分配器上有记录各个 Web 服务器最新负载信息的负载统计表。该表的信息是负载分配的重要依据。

文档分配情况统计

本文采用了文献[1]中给出的文档分配算法,文档分配的结果是:每个 Web 服务器上只包含所有文档的一部分。一个文档可能只存在一个服务器上,也可能存在于多个服务器上。后端分配器对文档的分配结果进行统计,确认每个文档的存放位置。负载平衡算法在运行的过程中会用到这些统计信息。

1.2 Web 服务器层

如果说请求分配器层是模型"管理中心"的话,那么 Web 服务器层就是具体业务的"实现中心"。Web 服务器的具体功能描述如下:

1) 处理应用中的业务

Web 服务器是应用处理的核心,是具体业务的实现。它能够及时响应用户请求,并将结果返回给用户。这是 Web 服务器最基本的功能。

2) 负载信息探测

后端分配器在分配用户请求的时候要考虑各个 Web 服务器当前的负载情况。这些负载信息由 Web 服务器提供。要完成负载信息探测,需要做两个方面的工作:一是确定负载指标,明确到底需要采集哪些信息;二是通过什么样的方法探测出这些指标的具体数据,并定期将最新的负载信息向后端分配器报告。

3) 文档分配

本文采用文献[1]提供的文档分配算法对文档进行分配。

1.3 负载指标的确定

解决负载平衡问题的第一步就是确定负载指标。负载指标对目标服务器实际情况的反映决定了负载分配的有效性。理想的负载指标要遵从下面的标准[4]:

1) 测量开销低。这意味着可以频繁测量,以确保负载信息尽可能新;

2) 能体现所有竞争资源上的负载;

3) 各个负载指标在测量及控制上彼此独立。

比较常用的负载指标有:目标运行队列中的任务数、目标的响应时间、空闲内存的比例等。这些参数中,有些是瞬间值,可能是纳秒级的,而网络传输是毫秒级的。即使采集了这些数据,传输到目的地时也不能真实地反映服务器最新负载。实际应用中,目标运行队列中的任务数这个指标较为常用。

绝大多数集群系统在选择负载指标的时候都假设集群中所有服务器的配置完全相同。这样的假设过于简单化,在实际的应用中,配置不同的情况是很常见的。如果服务器配置不同,则处理用户请求的能力就有所不同。配置不同主要表现在两个方面:一是硬件。CPU 型号和工作频率的不同,内存大小的不同等等,服务器的性能就不同。二是软件。软件配置的不同对服务器性能的影响也是比较明显和常见的。

还有另外一个因素也是目前的一些系统所忽略的:请求对资源消耗的差别问题。请求的对象不同,对服务器系统资源的消耗就不同。一个浏览网页的请求和一个对多个数据表进行关联查询的请求对系统资源的消耗肯定是不同的。

基于上面的考虑,当前的实际连接数可能就不能很好地反映出服务器真实的负载。本文将 Web 服务器的当前连接数 CurC,处理完这些连接大致需要花费的时间 Time 以及服务器本身的处理能力 ProC 综合在一起作为负载指标。负载指标为:

$$Load = \frac{CurC * Time}{ProC^2} \quad (1)$$

对于服务器处理能力 ProC 的测定,现在有很多专门的软件。这些软件依据服务器的硬件(如 CPU 型号、数量和内存数量等)和软件(如操作系统类型等)的信息,按照一定的原则换算成处理能力。在 Web 应用中,服务的提供是通过一些固定的端口进行的。通过对特定的端口进行扫描后统计,就可以获取 CurC 的信息。精确地统计 Time 的值对系统的资源消耗非常的大,所以是不可取的,只能进行估测。

1.4 负载平衡算法设计

请求分配的整个过程如图 2 所示。按照"能者多劳,各尽所能"的原则,在进行负载分配的时候应尽量使服务器的负载和自身的处理能力成比例。本模型考虑了 Web 服务器的连接数量,连接持续的总的时间以及服务器自身能力三个因素。本模型中每个 Web 服务器只包含一部分的文档,所以在设计负载平衡算法的时候还要考虑到文档的分布情况。

具体的负载平衡算法描述如下:

1) 确定请求所要访问的文档;

2) 查询文档分配统计表,获取目标文档的分布情况;

3) 目标文档只存在于一个服务器上吗? 是,转去 4),否则转去 5);

4) 将请求分配到这个唯一的服务器上,本次分配结束;

5) 根据 3)的结果查询负载统计表,利用式(1)计算每个 Web 服务器的负载值 Load;

6) 从负载值中选择一个最小的,将请求分配给对应的 Web 服务器,本次分配结束。

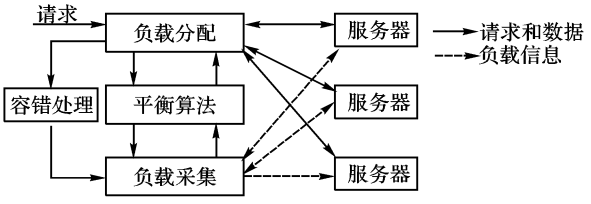

图 2 负载分配过程

1.5 服务器迅速过载问题

每个后端分配器上都有一个负载统计表。当 Web 服务器向后端分配器更新自己负载的时候,它更新的是所有后端分配器上的负载统计表。这样做保证了负载数据的一致性,但是也有一定的问题:负载统计表的信息是完全一致的,负载平衡算法又相同,那么通过负载平衡算法选择的 Web 服务器肯定也是同一个服务器。这样,多个后端分配器可能会在短时间内将多个请求分配给同一个 Web 服务器,导致该服务器迅速过载,甚至陷入瘫痪状态。

最容易的解决方法是提高负载信息采集的频率,让 Web 服务器的负载变化情况迅速及时地反馈给后端分配器。这样做虽然解决了服务器迅速过载的问题,但是对整个系统的性能有较大的影响。因为:1) 负载信息探测次数的增多,会增

大 Web 服务器的负担。2）负载信息更新次数的增多,会增加整个模型的通信负担。

为此,本文提出了另外一个方法。Web 服务器仍然按照原先的频率进行负载信息探测。不同的是,在前后两次更新负载的时间间隔内,负载统计表中的负载值不是不变的,而是在不断地更新。更新的原则是这样的:每次后端分配器将用户请求分配给某个 Web 服务器后,就会在负载统计表中将该服务器所对应的负载值进行一定程度的增加(这种增加不一定非常准确地反映了该服务器的负载值,只是反映了负载信息的变化趋势)。并将此次变化通知其他的后端分配器,以保证负载统计表信息的一致性。在没有收到 Web 服务器传送过来新的负载值之前,Web 服务器的负载值是在不断变化的。当再次收到新的负载值的时候,就以新的负载值为准进行更新。此后开始了新一轮的循环,具体过程如图 3 所示。

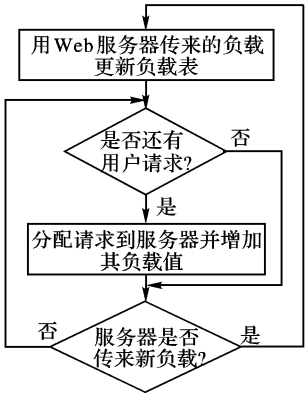

图 3　负载更新过程

这种处理方式能够及时反映 Web 服务器的变化趋势,既没有给整个系统增加负载信息探测和负载信息传递的负担,又避免了服务器迅速过载现象的发生。

1.6　模型的容错性问题

在本模型中,某些 Web 文档可能只存在于某一个服务器上。如果这唯一的服务器出现故障,将导致访问该 Web 文档的请求失效。为此,本模型配置了一台备份服务器,它包含了整个站点所需的全部文档。正常情况下,用户请求是不会分配到该服务器的,只有当某个服务器出现故障时才启用该备份服务器。

Web 服务器需要定期向后端分配器报告自己的最新负载信息。如果后端分配器不能按时收到新的负载信息,且时间间隔超过了事先的设定值,就认为这台 Web 服务器出现故障。

在这种情况下,后端分配器将做两个方面的工作:一方面,及时向系统管理员发出警告信息,提示系统中服务器可能出现故障,并给出具体编号的,以便管理员有针对性地进行检修,排除故障;另外一方面,后端分配器会及时调整分配策略,不再将新的请求分配给这台服务器,而是将请求进行转移。转移的目标分两种不同的情况:如果后续请求访问的文档存在于一个或者多个其他正在运行的服务器上,就按照负载平衡算法从中选择一个最优的服务器分配该请求;如果后续请求访问的文档只存在于故障服务器上,那么所有访问该文档的请求都被分配到备份服务器上。

这种处理方式避免了用户请求分配之后未被响应的现象,整个模型具有一定的容错性。

2　性能测试与结论

在现有的实验环境中很难精确测试负载平衡的性能,所以本文采用多线程方法进行了模拟测试。整个模拟的过程如图 4 所示。用户线程和服务器线程各有三个。负载平衡线程有三个,一个充当前端分配器,两个充当后端分配器。

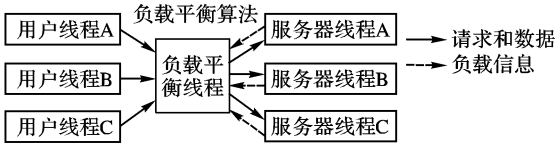

图 4　负载平衡测试

通过比较能直观地看出方案的优劣。故本文将新负载平衡算法与现存的最少连接法进行比较。首先比较响应时间。分别运行新负载平衡算法与最少连接法,在用户线程端测响应时间,测到的结果如图 5 所示。可以看出新算法所对应的响应时间相对比较小。负载平衡的目的就是为了缩短响应时间。

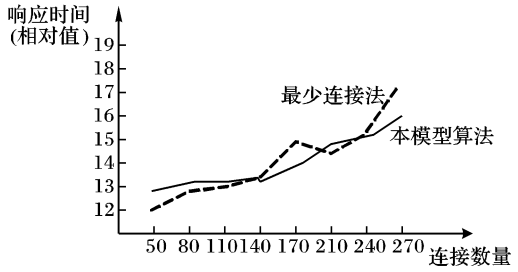

图 5　响应时间

然后测试服务器负载的平衡效果。客户线程发出不同数量的请求,在服务器端测得的三个服务器线程负载的动态变化情况如图 6 所示。可以看出三个服务器线程基本达到了负载平衡。

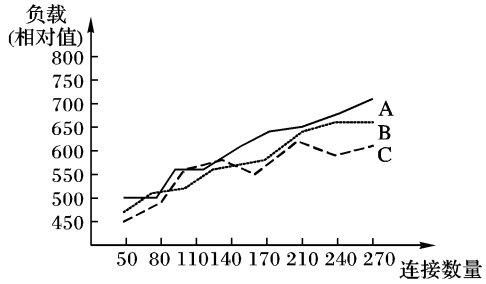

图 6　负载平衡效果

参考文献:

[1] LEE SL, HO H-J, LIEN F-W. A Load Balancing Algorithm on Heterogeneous Distributed WWW Servers[EB/OL]. http://www.csse.monash.edu.au/developerworks/load-balancing.html, 2005.

[2] CARDELLINI V, COLAJANNI M, YU PS. A dynamic load balancing on web-server systems[J]. IEEE Internet computing, 1999, (6):28-39.

[3] CARDELLINI V, COLAJANNI M. DNS Dispatching Algorithms with State Estimators for Scalable Web Server Cluster. Word Wide Web J[J]. Baltzer Science Bussum, Netherlands 1999, 2(2):45-56.

[4] 鞠九滨, 杨鲲, 徐高潮, 等. 使用资源利用率作为负载平衡系统的负载指标[J]. 软件学报, 1996, 7(4):238-243.

[5] 林涛, 唐宁久, 李天翼. 基于 TCP 连接与服务器负载的二级负载平衡策略设计[J]. 计算机应用研究, 2003, 20(5):153-156.

[6] 刘振英, 张毅, 胡铭曾, 等. 多机系统的动态负载平衡[J]. 计算机科学, 1999, 26(1):38-40.