

# 大规模文件上传接收服务的负载均衡引擎研究

刘必雄<sup>1,2</sup>, 许榕生<sup>2,3</sup>

(1. 福建农林大学 计算机与信息学院, 福建 福州 350002;

2. 福州大学 数学与计算机学院, 福建 福州 350002;

3. 中国科学院 高能物理研究所 计算中心, 北京 100049)

**摘要:**针对大规模网络环境下的文件上传接收服务系统的需求,设计了一个集群架构的文件接收服务系统,采用多台接收服务器来实现文件接收服务,提高了系统的稳定性和可用性。根据文件传输的特点,综合考虑文件流量负载情况和服务器当前负载情况,提出了一个综合负载统计的负载均衡算法,测试结果表明基于该算法的文件接收集群系统具有较好的负载均衡效果。设计并实现了文件上传接收服务系统的负载均衡引擎,解决了系统的负载失衡问题,提高了系统的运行效率。

**关键词:**负载均衡;集群;文件接收服务

**中图分类号:**TP393

**文献标识码:**A

**文章编号:**1673-629X(2008)06-0016-04

## Research of Load - Balancing Engine for Large - Scale Up - Transfer Files Receiving Service

LIU Bi-xiong<sup>1,2</sup>, XU Rong-sheng<sup>2,3</sup>

(1. College of Computer and Information, Fujian Agriculture and Forestry University, Fuzhou 350002, China;

2. College of Mathematics and Computer, Fuzhou University, Fuzhou 350002, China;

3. Computing Center, Institute of High Energy Physics, Chinese Academy of Sciences, Beijing 100049, China)

**Abstract:** According to the demand of up - transfer file receiving system in large - scale network circumstance, a file receiving system based on cluster is designed, so that the availability and stability of the system are improved by constructing receiving server cluster system from multi computers. A load - balancing algorithm for synthetic loading statistic is presented according to the character of the file transfer, taking the file transfer loading and the loading of the server into account. Test result shows that file receiving system based on this algorithm is proved to get better effect. Finally, the load - balancing engine for the system is designed and implemented to solve the out - of - balancing of the system, so that the efficiency of the system is improved.

**Key words:** load - balancing; cluster; file receiving service

## 0 引言

在大规模网络环境下,文件上传接收服务系统需要满足成千上万的用户同时上传文件的请求,单一接收服务器受到处理能力的限制,很容易成为整个系统的瓶颈。在这种情况下,通常采用多台文件接收服务器构成集群系统<sup>[1,2]</sup>,提供文件接收服务,从而提高了系统的稳定性和可用性。然而,大量的用户上传请求很容易导致负载失衡,即经常出现某些服务节点过于

繁忙,而某些服务节点却处于空闲的情况,甚至出现服务失败的情况。为了解决这个问题,必须将用户上传请求合理地分配到集群的各个服务节点上,使各个服务节点均衡地负载,以实现整个系统负载均衡,保证系统的处理能力和服务质量。

集群服务器的负载均衡算法主要分为静态和动态两种算法<sup>[3]</sup>。静态负载均衡算法不考虑服务节点的实际负载情况,而动态负载均衡算法<sup>[4]</sup>则要考虑服务节点的当前实际负载。由于在实际应用中,各个服务节点的瞬间负载差异比较大,静态算法均衡效果比较差,因此,动态负载均衡算法是当前研究的热点,基于DNS动态负载均衡算法就是其中一种<sup>[5]</sup>。虽然DNS技术已经很成熟了,而且在许多系统中得到广泛的应

收稿日期:2007-09-05

基金项目:国家自然科学基金资助项目(90412017)

作者简介:刘必雄(1979-),男,福建平潭人,讲师,硕士研究生,研究方向为网络安全、网格计算;许榕生,研究员,博士生导师,研究方向为网络安全。

用,然而它没有考虑到用户的实际请求所需的服务量,因此不能直接应用到文件上传接收服务系统中。文中根据文件上传接收服务的特点,综合考虑文件流量负载情况和服务器当前负载情况,提出了一个综合负载统计的动态负载均衡算法。该算法充分考虑到文件上传的数量、文件的总字节数以及服务节点的负载情况,在很大程度上达到文件接收负载均衡,提高了系统的运行效率。

## 1 文件上传接收服务系统

文件上传接收服务系统主要负责接收用户上传的文件,其任务由均衡分配器和接收服务器集群完成。该系统是一个基于动态负载均衡技术的文件接收系统,它能将前端节点上传请求均衡地分配给接收服务器集群中各个服务节点处理,其体系结构如图1所示。用户通过文件上传软件来实现文件上传任务,上传节点首先将要上传的文件的相关信息发送给均衡分配器,分配器根据文件流量负载情况和接收服务器的负载情况,找到一台最合适的服务器,并将该服务器的IP地址和接收服务端口Port发给上传节点。然后,上传节点根据新IP地址和端口号连接接收服务器,上传文件。

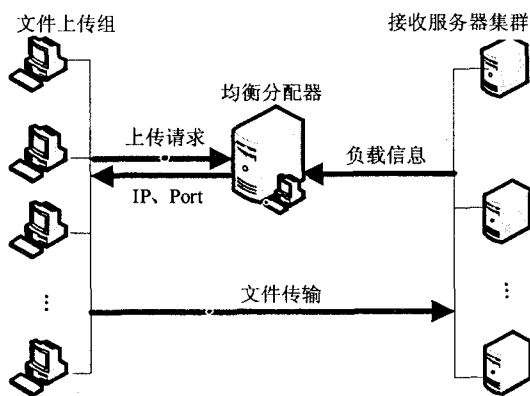


图1 文件上传接收服务系统的体系结构

## 2 基于综合负载统计的负载均衡算法

文件上传请求均衡分配中,由于每个请求所要上传的文件数量以及文件总字节数都是不断变化的,如果没有综合考虑接收服务器的当前实际负载情况以及文件流量情况,可能会出现服务失败情况。因此,文中根据文件传输流量的实际情况和接收服务器的实时负载情况,对上传请求进行分配,从而充分利用了系统的计算资源,提高了系统的运行效率。

### 2.1 服务器综合负载参数及负载量公式

综合负载统计根据文件流量负载统计和服务器负载反馈来统计接收服务器的综合负载,主要由五个参

数组成:单位时间内连接数的比值 $C$ 、单位时间内接收文件数的比值 $F$ 、单位时间内接收文件总字节数的比值 $B$ 、CPU使用率 $U$ 、内存使用率 $M$ 。

单位时间内连接数的比值是指在单位时间内服务器收到新的上传请求数与平均上传请求数的比例<sup>[6]</sup>。假设服务器集群内有 $n$ 台服务器,对于某个服务器 $i$ , $CN_i$ 表示在一个时间段 $T_2 - T_1$ 内,服务器 $i$ 收到的新上传请求数,这样就可以得到一组服务器在一个时间段 $T_2 - T_1$ 内收到新的上传请求数 $\{CN_1, CN_2, \dots, CN_m, 1 \leq m \leq n\}$ ,服务器 $i$ 的连接数比值 $C_i$ 为时间段 $T_2 - T_1$ 内其新上传请求数与 $n$ 台服务器收到的平均上传请求数的比值,其公式为:

$$C_i = \frac{CN_i}{\sum_{m=1}^n CN_m} \times n$$

同样,服务器 $i$ 单位时间内接收文件数的比值 $F_i$ 和单位时间内接收文件总字节数的比值 $B_i$ ,分别为:

$$F_i = \frac{FN_i}{\sum_{m=1}^n FN_m} \times n \quad \text{与} \quad B_i = \frac{BN_i}{\sum_{m=1}^n BN_m} \times n'$$

其中 $FN_i$ 和 $BN_i$ 分别表示在一个时间段 $T_2 - T_1$ 内服务器 $i$ 接收到新的文件数和文件总字节数。服务器当前CPU使用率为 $U_i$ ,内存当前使用率为 $M_i$ 。

另外,文中引入一组可以动态调整的系数 $R_i$ 来表示各个负载参数的重要程度,其中 $\sum R_i = 1$ 。根据这五个参数计算每台服务器的综合负载量 $L$ ,如式(1):  

$$L_i = R_1 \times C_i + R_2 \times F_i + R_3 \times B_i + R_4 \times U_i + R_5 \times M_i$$
 其中 $\sum R_i = 1$  (1)

如果系数 $R_i$ 不能很好地反映当前的负载,系统管理员可以对系数进行不断修正调整,直到找到贴近当前情况的一组系数。

### 2.2 上传请求优先级队列

由于上传请求的文件数和文件总字节数都存在一定的差异,如果按照先来先服务的原则,可能存在小工作量的请求分配给负载最轻的服务器,而大工作量的请求分配给负载最重的服务器,从而又引起负载失衡。因此,上传请求队列应当用优先级队列来实现,各上传请求的优先级由两个参数决定:上传文件数 $SF$ 和上传文件总字节数 $SB$ 。由于文件总字节数的单位是字节,所以文中还引入系数 $K$ ,用来调节文件总字节数和文件数关系。文中系数 $K$ 是指队列中文件平均字节数,可以表示为:

$$K = \frac{\sum SB_j}{\sum SF_j}$$

根据这两个参数和系数 $K$ ,可以计算每个上传请求的

优先级  $P$ , 如式(2):

$$P_j = K \times SF_j + SB_j \quad (2)$$

其中,  $P_j$ ,  $SF_j$ ,  $SB_j$  分别表示发送请求  $j$  的优先级、上传的文件数以及上传的文件总字节数。

### 2.3 上传请求分配策略

根据公式(1)计算各个服务器的综合负载量  $L_i$ , 并对  $L_i$  进行排序, 对所有的前端上传请求根据公式(2)计算每个请求的优先级  $P_j$ , 并按  $P_j$  进入队列。然后将上传请求队列中各个请求依次按循环方式分配给已按综合负载量从小到大排序好的服务器列表。这样就可以将上传工作量大的发送请求安排给综合负载量轻的服务器, 上传工作量小的发送请求安排给综合负载量较重的服务器, 从而实现负载均衡。

## 3 负载均衡引擎的设计与实现

### 3.1 负载均衡引擎总体结构

负载均衡引擎主要由综合负载统计模块和上传请求分配决策模块来实现文件接收的负载均衡, 其总体结构如图 2 所示。对所有的前端上传请求, 按优先级顺序进入队列, 然后综合负载统计模块根据服务器负载反馈得到服务器负载信息: CPU 使用率  $U$ 、内存使用率  $M$ ; 再通过文件流量负载统计取得各服务器文件流量负载信息: 单位时间内连接数的比值  $C$ 、单位时间内接收文件数的比值  $F$ 、单位时间内接收文件总字节数的比值  $B$ 。接着根据公式(1)计算各个服务器综合负载情况, 提交给请求分配决策模块。分配决策模块根据服务器综合负载量和上传请求队列, 给每个上传请求分配一个合适的服务器。

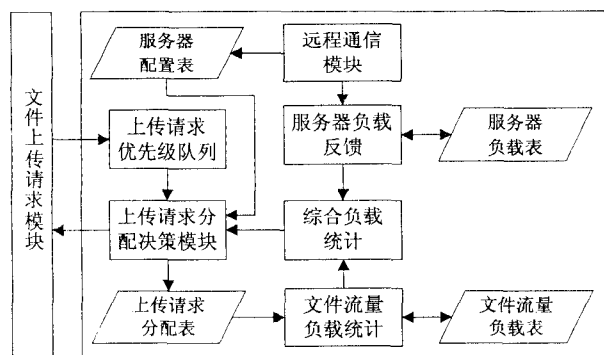


图 2 负载均衡引擎总体结构

### 3.2 信息表维护

负载均衡引擎中涉及到服务器配置表、服务器负载表、上传请求分配表以及文件流量负载表。服务器配置表是存储接收服务器的配置信息, 主要字段有: {服务器编号, 服务器 IP 地址, 服务器接收端口号}, 它由远程通信模块通过 SNMP 协议获取; 服务器负载表

是接收服务器当前负载状态, 主要字段有: {服务器编号, CPU 使用率, 内存使用率}; 上传请求分配表是文件上传请求的分配情况, 主要字段有: {服务器编号, 分配时间, 文件数量, 文件总字节数}, 由分配决策模块在分配时记录; 文件流量负载表是接收服务器文件流量负载情况, 主要字段有: {服务器编号, 单位时间内连接数的比值, 单位时间内接收文件数的比值, 单位时间内接收文件总字节数的比值}。

服务器负载表和文件流量负载表必须及时更新来反映服务器负载状态。负载信息更新策略主要有两种方式: 触发更新和定时更新<sup>[6]</sup>。

①触发更新是指一旦前端节点对分配器发出上传请求, 就触发服务器负载模块去查询各个服务器的负载状态, 得到当前负载最轻的服务器。

②定时更新是指服务器负载模块定期去查询服务器集群内各个节点负载状态, 得到一组负载最轻的服务器集。

触发更新方式的实时性强, 前端上传节点总可以得到任意时刻的最佳接收服务器。然而, 在大量前端节点的上传请求的前提下, 触发更新就会导致新的负载失衡。这是因为每次前端节点的上传请求都会触发服务器负载模块去收集各个服务器的负载状态, 各个服务器也要启动相应的服务来计算当前的负载情况, 这样就很容易导致分配器过载以及在各个服务器上产生新的负载失衡。利用定期更新方式, 只要查询时间间隔设定合理, 并在数据表中维护一组最佳服务节点, 就可以取得比较合理的效果。因此, 文中对服务器负载信息的更新采用定时更新策略来实现。

### 3.3 负载均衡统计

为了更准确地描述当前各个服务器的负载情况, 应当为公式(1)找到一组合适的系数, 为此, 根据上述的综合负载统计算法的基本思想, 测试了大量的配置相同的机器运行情况, 得到一组数据, 然后估算出一组较合适的参数的权重系数为:  $\{R_1 = 0.1, R_2 = 0.3, R_3 = 0.2, R_4 = 0.3, R_5 = 0.1\}$ , 于是, 服务器的负载量计算公式(3)为:

$$L_i = 0.1 \times C_i + 0.3 \times F_i + 0.2 \times B_i + 0.3 \times U_i + 0.1 \times M_i \quad (3)$$

这样就可以通过公式(2)和公式(3)以及各个信息表中的信息来进行上传请求分配。

## 4 实验测试结果

对文中提出的算法进行测试, 并与静态的循环负载均衡算法结果进行比较。循环算法<sup>[5]</sup>就是在分配器中维护一个所有服务节点的列表, 然后分配器以一种

循环的方式从列表中挑选一个服务节点分配给上传请求节点,从 1 至  $N$  分配,然后重新开始。实验由 5 台普通的 PC 机组成了服务器集群,其中一台 CUP 为 2.4G、内存为 512M 作为负载均衡分配器,其它 PC 的 CPU 为 2.0G、内存为 256M 的作为接收服务器节点。对上传请求的平均响应延迟进行测试,测试结果及分析如图 3 所示。

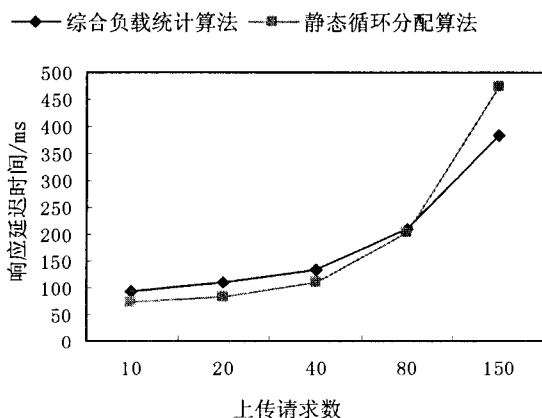


图 3 平均响应延迟对比

从图 3 可以看出,当上传请求数很少时,由于静态分配不需要分配器做复杂运算,响应速度快于综合负

载统计算法。当请求数达到 150 时可以看出综合负载统计算法明显优于静态算法。

#### 参考文献:

- [1] Trinitis C, Walter M, Leberecht M. Balanced High Availability in Layered Distributed Computing Systems[C]//Proceeding of the 14th International Workshop on Database and Expert Systems Applications (DESA03). Prague, Czech Republic: IEEE Computer Society, 2003: 713 - 717.
- [2] Hou Z, Huang Y, Zheng S, et al. Design and Implementation of Heartbeat in Multi-machine Environment[C]//Proceedings of the 17th International Conference on Advanced Information Networking and Applications 2003. Xi'an, China: ISTP, 2003: 583 - 586.
- [3] 薛军, 李增智, 王云岚. 负载均衡技术的发展[J]. 小型微型计算机系统, 2003, 24(12): 2100 - 2103.
- [4] Zaki M J, Li W, Parthasarathy S. Customized Dynamic Load Balancing for a Network of Workstations[J]. Journal of Parallel and Distributed Computing, 1997, 43(2): 156 - 162.
- [5] 田绍亮, 左明, 吴绍伟. 一种改进的基于动态反馈的负载均衡算法[J]. 计算机工程与设计, 2007, 28(3): 572 - 573.
- [6] 裴尔明. 网络环境下海量数据处理若干关键问题研究[D]. 北京: 中国科学院高能物理研究所, 2007: 72 - 75.

(上接第 15 页)

流配送路径的最优解或近似最优解。

表 1 配送中心与客户之间的距离/km 及各客户的需求

$i$	$j$								
	0	1	2	3	4	5	6	7	8
0	0	0.4	6.0	7.5	9.0	20.0	10.0	16.0	8.0
1	4.0	0	6.5	4.0	10.0	5.0	7.5	11.0	10.0
2	6.0	6.5	0	7.5	10.0	10.0	7.5	7.5	7.5
3	7.5	4.0	7.5	0	10.0	5.0	9.0	9.0	15.0
4	9.0	10.0	10.0	10.0	0	10.0	7.5	7.5	10.0
5	20.0	5.0	10.0	5.0	10.0	0	7.0	9.0	7.5
6	10.0	7.5	7.5	9.0	7.5	7.0	0	7.0	10.0
7	16.0	11.0	7.5	9.0	7.5	9.0	7.0	0	10.0
8	8.0	10.0	7.5	15.0	10.0	7.5	10.0	10.0	0
$q_j(t)$		1	2	1	2	1	4	2	2

表 2 物流配送路径优化问题的遗传算法计算结果计算次序

计算次序	1	2	3	4	5	6	7	8	9	10
配送总距离( $Z/km$ )	72.0	72.0	76.5	70.0	67.5	70.0	73.5	75.0	71.5	69.0

## 4 结束语

实验表明,先建立优化物流配送路径的数学模型,

然后用遗传算法优化求解,这是一种性能优良的启发式搜索方法。可以快速有效地求得优化物流配送路径的最优解。

文中所使用的编码方法、适应值的求法以及选择、交叉和变异算子,对求解类似的组合优化问题具有参考价值。

#### 参考文献:

- [1] 蔡希贤, 夏士智. 物流合理化的数量方法[M]. 武汉: 华中工学院出版社, 1985.
- [2] Holland J. 遗传算法的基本理论与应用[M]. 李敏强译. 北京: 科学出版社, 2003.
- [3] 李军, 郭耀煌. 物流配送车辆优化调度理论与方法[M]. 北京: 中国物资出版社, 2001.
- [4] 赵刚. 物流运筹[M]. 成都: 四川人民出版社, 2002.
- [5] 陈国良, 王煦法, 庄镇泉, 等. 遗传算法及其应用[M]. 北京: 人民邮电出版社, 1996.
- [6] 姜大立, 杨西龙, 杜文, 等. 车辆路径问题的遗传算法研究[J]. 系统工程理论与实践, 1999, 19(6): 40 - 44.
- [7] 米凯利维茨 Z. 演化程序——遗传算法和数据编码的结合[M]. 北京: 科学出版社, 2000.