第34卷 第18期 计 算 机 工 程 2008年9月
Vol.34 No.18 Computer Engineering September 2008
·网络与通信· 文章编号：1000—3428(2008)18—0134—03 文献标识码：A 中图分类号：TP393

基于 JXTA 的 FTP 服务器集群负载平衡策略

梁双建[1,2]，舒 坚[1,2]，刘琳岚[1,2]，陈斌全[1,2]
(1. 南昌航空大学计算机学院，南昌 330063；2. 南昌航空大学软件学院，南昌 330063)

摘 要：针对 FTP 服务器集群全局平衡性差的问题，该文提出一种负载平衡方案。该方案包括分布式统一平台的构建和 FTP 负载平衡策略的选择。基于 JXTA 技术构建分布式统一平台，设计全局-局部-再全局的自适应负载平衡策略对 FTP 请求进行动态调度，将集群划分成多个互不相交的子集，每个子集有一个超级节点，子集间的协商通过超级节点间的相互通信完成。实验证明了该方案的有效性。
关键词：负载平衡；服务器集群；JXTA 技术

Load Balance Strategy of FTP Server-cluster Based on JXTA

LIANG Shuang-jian[1,2], SHU Jian[1,2], LIU Lin-lan[1,2], CHEN Bin-quan[1,2]
(1. Faculty of Computer, Nanchang Hangkong University, Nanchang 330063;
2. Faculty of Software, Nanchang Hangkong University, Nanchang 330063)

【Abstract】To solve problem of the FTP server-cluster's global balance, this paper proposes a novel load balancing scheme, including the choice of FTP load balancing strategies and the construction of the uniform distributed platform implemented by JXTA. It adopts the adaptive FTP load balancing strategy, called global-local-secondary global, to schedule the download requests of clients dynamically. The cluster is divided into several subsets disjointed mutually, and each subset has only one super node. The negotiation among subsets is implemented by communication of super nodes. The validity of the scheme is testified by experiments.
【Key words】load balance; server-cluster; JXTA technology

1 概述

随着FTP访问数量的激增和动态性能的提高，单个FTP服务器已不堪重负，更无法满足服务运行的高效性和处理海量请求的伸缩性[1]。目前，一般采用集群或分布式技术来解决这一问题，采用集群技术实现简单，能收到较好的效果，但存在以下问题：

(1)集群大多采用集中控制原理，请求任务由负载平衡器(Load Balancing, LB)统一分配给提供下载服务的节点(Real Server, RS)后，服务执行结果需再次经过 LB 返回到客户端，LB 处极易形成性能瓶颈。

(2)RS 的负载指标往往只考虑当前连接的任务数量而很少考虑这些连接处理的强度和 RS 本身的处理能力。

(3)在大多数服务器集群中，请求任务由 LB 分配后不再进行相应处理，很少考虑在任务执行期间集群可能出现的非平衡状态。

(4)服务器集群的 RS 数目一定，RS 间的物理连接比较固定，对于大型的服务器集群来说不够方便和灵活。

针对上述问题，本文提出一种负载平衡方案，包括分布式统一平台的构建和 FTP 负载平衡策略的选择。其中，分布式统一平台的构建通过 JXTA(Juxtapose，并列项目)技术来实现，采用全局-局部-再全局的自适应负载平衡策略对 FTP 请求进行动态调度。

2 基于 JXTA 的 FTP 服务器集群负载平衡方案

2.1 分布式统一平台的构建

采用JXTA技术构建分布式环境的统一平台。JXTA致力于为分布式计算提供一个通用的、统一的、可互操作的平台来容纳任何种类的网络服务[1]。在同等条件下，其往返时间(Round-Trip Time, RTT)明显高于传统的TCP传输[2]。

2.2 FTP 服务器集群模型

FTP 服务器集群模型由 LB 和服务执行层两部分组成，如图 1 所示。

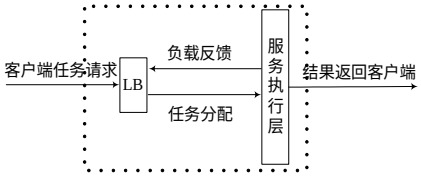

图 1 FTP 服务器集群模型

LB 是集群的对外唯一接口，负责对客户端的下载请求进行 Hash 处理，并根据服务执行层定期反馈的负载信息为其分配 RS。

服务执行层的主要功能是为任务提供服务，保证集群在运行过程中始终保持平衡和稳定，并将完成的服务结果直接返回客户端。

2.3 全局-局部-再全局负载平衡策略

全局-局部-再全局负载平衡策略包括全局平衡、局部协商平衡和全局协调平衡 3 个子策略，它对提高集群效率和吞吐率、缩短客户端响应时间起着至关重要的作用。

基金项目：江西省自然科学基金资助项目(0511066)；江西省测控中心开放基金资助项目(2003-014)；南昌航空大学科研基金资助项目(EC200606064)
作者简介：梁双建(1982－)，男，硕士研究生，主研方向：分布式系统；舒 坚、刘琳岚，教授；陈斌全，讲师、硕士
收稿日期：2007-10-20 **E-mail：**doublesword@163.com

如图 2 所示，将服务执行层划分为 N 个子集，每个子集由 $M(M>0)$个 RS 和一个超级 RS 组成，超级 RS 是子集的汇聚点并提供下载服务。

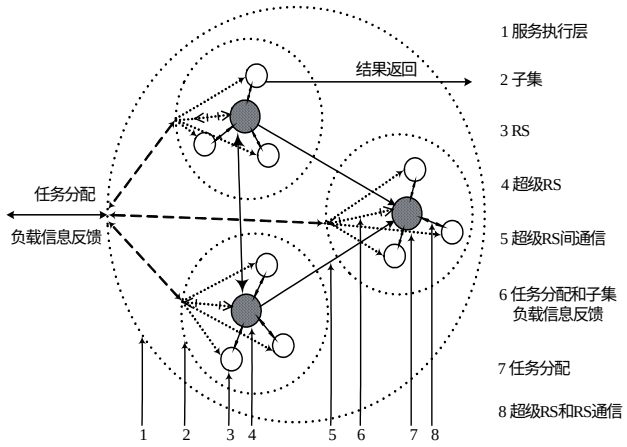

图 2　服务执行层的内部结构

全局平衡由 LB 完成，保证了系统的最空闲资源得到有效利用；局部协商平衡在子集内部 RS 间进行，是对全局平衡的一种补充和调整；全局协调平衡通过超级 RS 间的交流来完成，是对局部协商平衡后各个子集内部仍然处于非平衡状态的 RS 资源的整合，以提高集群效率和系统吞吐率。

3　全局-局部-再全局负载平衡策略的实现

3.1　RS 内容分布

将集群提供的下载内容划分成 6 块，每块包含若干文档。同一子集的 RS 存储相同内容，子集间的内容交叉存储。

同一子集的 RS 存储相同内容，那么进行局部协商平衡时只需考虑 RS 的负载状态，任务转移更高效、快捷，集群的通信代价更小、延迟更短，方便 RS 自由加入或退出集群。

子集间的内容交叉存储，可以避免在短时间内因热点内容访问量过大而导致单个子集局部过热，甚至因负载过重使整个子集失效，还可以提高单个 RS 的 Cache 利用率，同等条件下，存储相同内容的 RS 越多，命中同一 RS 的概率越小。

3.2　RS 性能采集

本文选取 CPU 利用率和内存利用率作为评价负载的参数，因为这 2 个参数比较容易获取，系统额外开销小，负载信息能够及时准确地反映当前 RS 的负载状态，为异构集群取得一个统一的标准。RS 的负载值 $load$ 定义如下：

$$load = a \times CPU\% + b \times MEM\%$$

$$a + b = 1(\text{权值} a,b > 0)$$

RS的负载状态划分为 3 种：轻负载，正常负载和重负载。文献[3]指出，在负载达到峰值时，如果系统资源利用率不足 50%，说明系统资源没有得到有效利用。文献[4]指出，如内存、CPU利用率长时间大于 80%，则系统资源的有效利用率会降低，当内存利用率和CPU利用率的阈值是 70%, 80%时为最优。综合考虑FTP服务器集群的各种因素(权值a, b根据实验的具体情况进行调整，本文设定a=0.4, b=0.6)，设定 2 个理想的门限值M_1, M_2，分别以内存利用率 50%, 80%、CPU利用率 50%, 70%为界划分RS状态。$Load[i] \le M_1$，则RS为轻负载状态，Load[i]$\ge M_2$，则为重负载状态，其他情况为正常状态。

3.3　FTP 负载分配及任务转移

进行 FTP 负载分配时，依据全局平衡子策略判断任务请求所需的下载文档的存放位置，选择为其提供服务的 RS；为使集群处于平衡状态，需要在服务执行层内部进行任务迁移，依据局部协商平衡和全局协调平衡子策略对任务进行迁移。具体策略如下：

(1)全局平衡子策略

LB 根据超级 RS 定期反馈的负载信息为请求任务选择并分配一个 RS。超级 RS 按全局平衡的固定周期向 LB 反馈负载信息，LB 汇总各个超级 RS 提供的其所属子集中处于轻负载状态的 RS 相关信息，并依据内容分块号进行分类管理，对含有相同内容块的 RS 按照负载大小进行归并排序。当请求任务到达时，LB 查找其对应队列，对队列的第 1 个 RS 自动进行负载估算，以有效避免因热点内容访问量过大而导致单个 RS 失效。估算后新的负载值 P 的计算公式如下：

$$P = Q + [(M_2 - M_1)/T_1] \times \Delta t$$

其中，Q为RS所属子集的超级RS向LB反馈的负载值；T_1为全局平衡周期；△t为从全局平衡周期开始到RS即将被分配所经历的时间。

如果当前新负载$P \ge M_1$，说明当前队列中无空闲RS可提供服务，此时停止分配，销毁整个队列；如果第 2 个全局平衡周期没有到来，那么分配完毕后，将当前RS插入队尾，等待下次分配，此时RS负载值仍为Q。

(2)局部协商平衡子策略

全局平衡之后，随着任务的继续执行，集群可能又会处于不平衡状态，因此，有必要在各个子集内部进行负载平衡，以避免 RS 在整个集群范围内因过多信息查找和通信而导致的线路阻塞和信息延迟。单个子集内的超级 RS 负责搜集子集内符合要求的 RS 负载信息，并对其进行集中处理。RS 采用状态变化驱动和周期相结合的策略向超级 RS 反馈信息，即只有在局部协商平衡周期到来时，子集内非正常负载状态的 RS 向超级 RS 反馈其负载信息。

局部协商平衡子策略的任务选择策略是选择那些执行时间长且占用 CPU 资源最多的任务，其任务迁移模式是一对一模式，即一个负载最轻的 RS 接受最重负载 RS 的任务转移。在进行任务转移时，任务当前的执行状况、执行百分比和执行环境被同时转移。任务迁移虽增加了系统的额外开销和节点间的通信次数，但解决了传统负载平衡方法中重负载 RS 只能消极地依靠自身资源减轻负载的问题。

任务迁移直接在 2 个 RS 间进行，无须再次通过超级 RS 进行中转。

(3)全局协调平衡子策略

局部协商平衡后，子集资源的有限性仍会导致个别 RS 较长时间地处于过载状态。为解决这一问题，局部协商平衡时，超级 RS 对其子集内过载的 RS 予以标记。当全局协调平衡周期到来时，各个超级 RS 主动探测其所属子集内被标记的 RS，如仍处于过载状态，则以最高优先级进行全局协调平衡。如第 2 个局部协商平衡周期到来时，全局协调平衡周期仍未到达，则以第 2 个局部协商平衡周期所做标记为准。

为尽可能缩短集群的延迟，依据先到先服务的原则进行全局协调平衡，当 RS 到达时有与其状态相反的 RS，就进行协调平衡。N 个超级 RS 轮流对全局协调平衡的负载信息进行汇聚，执行时间为 2 个全局协调平衡周期，以防止单个超级 RS 因总是忙碌而处于过载状态。

(4)周期设置问题

3 个子策略的周期设置依次为T_1, T_2, T_3，如图 3 所示。文献[5]指出，T_1一般以 5 s~10 s为好。T_2和T_3的选取要适度，周

期太短，RS间通信次数过多，易造成线路阻塞；周期过长，则达不到预期的负载平衡效果。

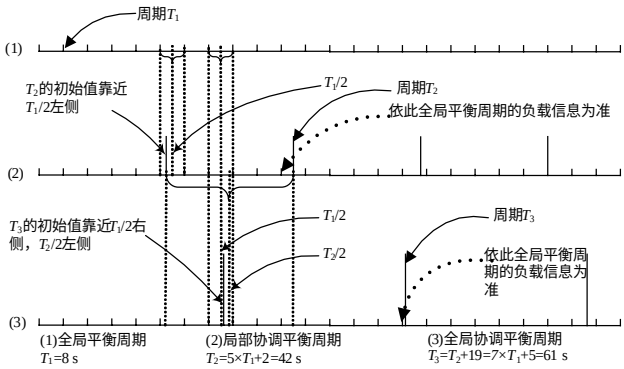

图 3　负载平衡子策略的周期设置

为防止因频繁搜集负载信息导致集群额外开销过大，在进行局部协商和全局协调平衡时，以最近一次全局平衡时的负载信息为准。任意 2 个策略的周期启动时刻越接近，越容易使集群产生抖动，这样集群效率不但不会提高，甚至还会降低。

为避免上述现象的出现，采取如下 2 种措施：

(1)设定$T_2=5\times T_1+2$, $T_3=T_2+19=7\times T_1+5$。其中，T_2的初始值从左侧靠近$T_1/2$，T_3的初始值从左侧靠近$T_2/2$，从右侧靠近$T_1/2$，按照子策略的启动顺序，后一个子策略在前面已启动子策略的 1/2 周期附近启动，可避免发生初始时刻抖动。

(2)如果$k\times T_1=m\times T_2$, $k\times T_1=n\times T_3$, $m\times T_2=n\times T_3$(k, m, n 均为≥0 的整数)中的任何一种情况成立，则只执行周期最短的子策略，可以避免集群运行过程中发生抖动。

3.4　容错问题

LB 作为集群对外的唯一接口，在实际应用系统中，应为其配置一个备用 LB，利用心跳机制连接起来，以便 LB 出现问题时能够及时地将任务转移到备用 LB。在本文的实验中，没有配置备用 LB。

4　仿真实验

4.1　实验环境

实验在校园网的同一子网内进行，所用设备配置如表 1 所示。

表 1　设备基本配置

实验设备	CPU/GHz	内存/MB	操作系统	台数
A 类服务器	Xeon 3.00	1 024	Linux2.4.20-8	2
B 类服务器	Celeron 2.00	512	Windows 2000	3
C 类服务器	Celeron 1.70	256	Windows XP	4
LB	P4 1.70G	512	Windows XP	1
客户端	Celeron 1.70	256	Windows XP	3
交换机	一台(D-Link DES-1024R+，24 个 10 Mb/s 或 100 Mb/s 端口)			

JVM采用jdk1.5 版本，JXTA采用 2.3.7 版本。为方便实验数据的统计、分析，采用Apache 组织的开放源代码Log4j项目，结合Apache Jakarta Project 中的 Commons Logging 组件来实现一种跟踪机制[6]，得到程序运行时间、当前状态等信息，并进行实验数据的存储。

为方便子集的划分，对RS进行标号，A类服务器为A_1, A_2，B类服务器为B_1, B_2和B_3，C类服务器为C_1, C_2, C_3和C_4。子集 1 包括A_1, B_1和C_1，子集 2 包括A_2, B_2和B_3，子集 3 包括C_2, C_3和C_4。集群的虚拟IP为LB的IP地址，假设有 3 个客户端进行任务请求模拟。

4.2　仿真及结果分析

集群下载内容划分为 6 块，每块包含 10 个文件，文件大小均为 2 MB。编程模拟客户端的任务请求，3 个用户线程并发执行，每个线程在 0 s~2 s 内随机选择一个文档进行下载，任务请求总数事先确定，记录从第 1 个任务请求起到最后一个任务完成的间隔时间。

(1)4 种策略的响应时间

对策略 A 策略~策略 D 分别进行测试，且每一个策略的同一任务请求总数均进行 5 次实验，取 5 次响应时间总和的平均值，结果如图 4 所示。

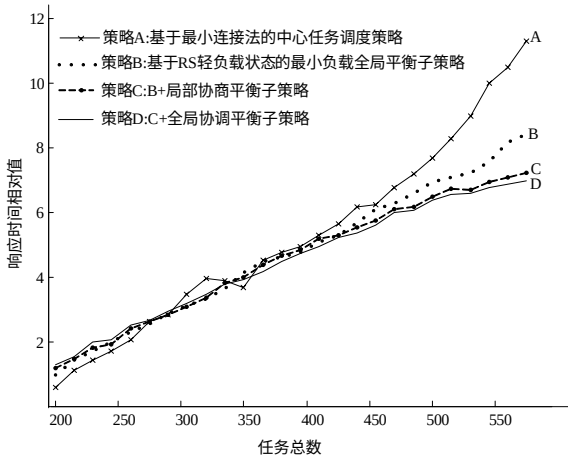

图 4　4 种策略响应时间对比曲线

由图 4 可知，与策略 A 相比，采用策略 B, C 或 D 时，客户端的响应时间明显缩短。策略 B, C 随着任务总数的增加，响应时间接近策略 D。可见，在 JXTA 集群上进行局部协商平衡和全局再次协调平衡可以大大缩短响应时间。

(2)子集内 RS 和子集随任务总数增加所呈现的负载状况

客户端任务请求得到及时响应，集群资源得到充分合理的利用，是负载平衡的最终目的。采用策略 D 时，各个子集内的 RS 和子集随任务总数增加所呈现的负载状况如图 5 所示。实验选取各个 RS 局部协商平衡周期时的负载数据，统计同一个 RS 的同一任务请求总数 5 次测试的所有周期负载总和，取其平均值。分析单个子集负载状况时也采用上述方法。

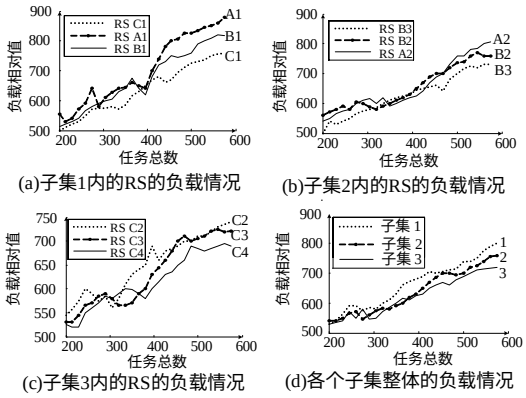

图 5　子集内 RS 和子集随任务总数增加所呈现的负载状况

由图 5 可知，随着任务请求总数的增加，每个子集内部 RS 的负载值和单个子集的负载值与服务器和单个子集自身的最大处理能力越来越接近，整个集群达到了负载平衡。

(下转第 139 页)