第 36 卷 第 6 期 计 算 机 工 程 2010 年 3 月
Vol.36 No.6 Computer Engineering March 2010
·多媒体技术及应用· 文章编号：1000—3428(2010)06—0224—03 文献标识码：A 中图分类号：TP37

基于 Agent 的 MCU 负载均衡方法

邱 华[1]，聂明军[2]
(1. 海军潜艇学院作战指挥系，青岛 266071；2. 北京航空航天大学，北京 100083)

摘 要：针对视频会议系统中多点控制单元(MCU)负载均衡问题，提出一种基于 Agent 的 MCU 负载均衡方法和一种具有较低 MCU 级联的组优先负载分配算法。该方法通过负载 Agent 和负载均衡 Agent 收集区域内 MCU 的动态负载信息并执行负载均衡操作，使 MCU 的负载得到平衡。实验结果证明该方法与 FRFA 算法相比，传输时延约减少 8%。

关键词：负载均衡；多点控制单元；代理

Multiple Control Unit Load Balance Method Based on Agent

QIU Hua[1], NIE Ming-jun[2]
(1. Department of Campaign Command, Navy Submarine Academy, Qingdao 266071; 2. Beihang University, Beijing 100083)

【Abstract】 Aiming at the problem of Multiple Control Unit(MCU) load balance in video conference system, this paper proposes a MCU load balance method based on Agent and a group priority algorithm with lower workload. MCU Load Gathering Agent(LGA) and Load Balancing Agent (LBA) are proposed to gather MCU workload and execute load balance, therefore workload is balanced. Compared with First Request First Allocation(FRFA) algorithm, the method can reduce video transmission delay by 8%.

【Key words】 load balance; Multiple Control Unit(MCU); Agent

1 概述

多点控制单元(Multiple Control Unit, MCU)负责视频会议系统终端连接、音视频的转发和其他会议业务逻辑执行等工作。为解决 MCU 扩容问题，文献[1]提出了分布式多 MCU 的扩容方案。如何在分布式状态下有效地平衡多个 MCU 的负载、保障各个 MCU 稳定高效地运行成为关键问题。

负载均衡的主要工作是合理地分配负载，使每个节点能够均衡地承担任务，从而实现整个系统的负载平衡性，提高系统的处理能力和服务质量。国内外对负载均衡已有较多的研究[2-3]，目前主要使用 2 类负载均衡算法：(1)MCU 资源预留静态算法。该算法不考虑服务节点运行时刻的负载情况，根据“尽量将同一个会议分配到最少的 MCU 上”的原则，在会议申请时为每个客户预留 MCU 资源。该算法实现简单，减少了 MCU 级联，并具有较小的音、视频转发延时。但是它存在 2 个问题：1)以连接数表示负载进行资源预留，不能真正体现负载情况。2)系统长时间运行后，节点负载量没有修正，必然背离实际负载情况，导致负载不均衡。(2)FRFA (First Request First Allocation)动态平衡算法[4]。该算法动态收集每个 MCU 服务器节点的当前工作负载，将服务终端的服务请求连接到负载最小的 MCU 服务器上。相比资源预留静态算法，此算法能更有效地平衡各个 MCU 服务的负载，充分利用各个 MCU 服务器的 CPU 资源。但是此算法容易将一个会议中的成员分配到多个 MCU 上，形成较多的 MCU 级联，带来较大的媒体数据传输延时。因此，本文提出一种基于代理(Agent)技术的 MCU 负载均衡方法，实验证明此方法能够在平衡负载的基础上有效减少视频时延。

2 基于 Agent 的 MCU 负载均衡方法

Agent 技术具有自主性、协作性等突出特点[5]，可以在现有 H.323 系统中采用 Agent 技术，在每台 MCU 服务器上部署 Agent 并收集系统负载，然后通过与会议控制器 GK 交互协作实现 MCU 负载均衡。本文提出使用 Agent 管理 MCU 的负载均衡方法。该方法设计了 2 种 Agent：MCU 负载 Agent (Load Gathering Agent, LGA) 和负载均衡 Agent(Load Balancing Agent, LBA)。图 1 描述了基于 Agent 的 MCU 负载均衡方法的系统结构。

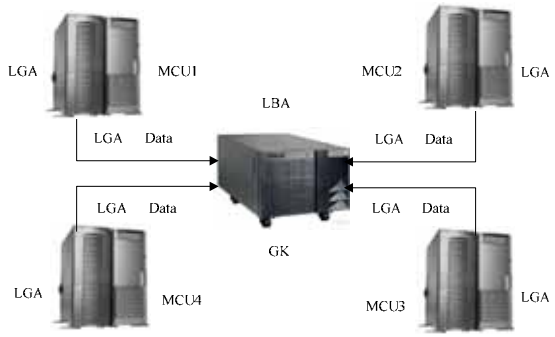

图 1 基于 Agent 的 MCU 负载均衡框架

2.1 MCU 负载 Agent

LGA 分布在每台 MCU 服务器上，负责收集当前 MCU 服务器的本地负载信息，并根据预先设计的周期时间 T_0 周期性地将本地负载服务信息传递给 GK。同时 LGA 从 GK 获取一份区域 MCU 负载信息表(MCU Load Table, MLT)，如表 1 所示。其中，MCU ID 是 MCU 的唯一标志；容量表示 MCU 支持的最大终端连接数，由 MCU 容量测试获得，固定不变，

基金项目：2005 CNGI 视讯会议系统应用示范基金资助项目(CNGI-04-15-1A)

作者简介：邱 华(1975－)，女，讲师，主研方向：多媒体应用技术；聂明军，硕士研究生

收稿日期：2009-08-11 **E-mail：**qddawn@gmail.com

它表征了一个 MCU 服务器的处理性能(CPU 和内存)；连接数表示连接到该 MCU 的终端数；CPU 使用率表示当前 MCU 服务器的 CPU 使用率。视频终端连接数和 CPU 使用率是衡量 MCU 负载的 2 个主要参数。

表 1　MCU 基本负载信息表

MCU ID	容量	连接数	CPU 使用率/(%)
MCU1	82	50	95
MCU2	78	40	43
MCU3	91	50	52

2.2　负载均衡 Agent

LBA 位于 GK 服务器上，用于动态分配客户端连接和执行负载重分配任务。LBA 由负载监控器、平衡触发器、MCU 选择器、终端重分配器 4 个部分组成。各部分功能如下：

负载监控器：负责更新 MCU 负载信息和监控 MCU 中各个终端的连接情况。负载监控器根据预设 T_0 周期性地从 LGA 中读取 MLT 信息，监控 MCU 中的各个终端连接情况，通过监控 RAS 消息收集每个 MCU 上视频终端的会议情况。H.323 中的 RAS 消息中有呼叫接纳系列消息和连接断开系列消息。呼叫接纳系列消息负责请求地址翻译，当视频终端和 MCU 建立连接时，向 GK 发送 ARQ 消息，其中包括主叫视频终端 ID、MCU ID 和请求会议 ID。同一个呼叫主叫端和被叫端都会向 GK 发送 ARQ 消息，因此，只须选择其一进行统计，本文选择视频终端的 ARQ 信息进行统计。

平衡触发器：用于决定负载平衡的条件和是否执行负载平衡。根据预先设置的 MCU 阈值，平衡触发器通过检查 MLT 判断 MCU 的负载是否超出其处理能力范围。一旦超出阈值，LBA 就向分配选择器发送信号，阈值信号表示为：<MCU 容量 c_0, CPU 使用率 U_0>，当出现新的终端请求或 MCU 超负荷时(MCU 终端连接数超过 MCU 容量 c_0 或 CPU 使用率超过预先设置的使用率 U_0)，都将触发 MCU 选择器进行选择。

MCU 选择器：决定视频终端将被分配的目标 MCU 服务器。负载监控器保存了 MLT 和 MCU 上各个终端的连接情况，为了防止 MCU 过载和减少将同一会议成员分配到不同 MCU 带来的 MCU 级联，位于 GK 上的 LBA 根据 MLT 和视频终端连接情况采用组优先、最少级联最轻负载分配策略进行负载分配。对于区域内的所有 MCU，首先计算每个 MCU 的连接负载因子。假设区域中有 MCU_1, MCU_2,…, MCU_n, i=1, 2, …,n，定义：

$$P_{ci}=\frac{c_i}{c_{0i}}\ (P_{ci}\geq 0) \tag{1}$$

$$P'_{ci}=\frac{c_{i+1}}{c_{0i}} \tag{2}$$

$$P_{ri}=U_i\ (P_{ri}\geq 0) \tag{3}$$

$$P'_{ri}=\frac{c_{i+1}}{c_{0i}}\times U_i \tag{4}$$

$$P_{cmin}=\min_i\{P_{ci}\} \tag{5}$$

$$P_{rmin}=\min_i\{P_{ri}\} \tag{6}$$

其中，P_{ci} 是 MCU_i 的连接负载因子，数值越大说明 MCU_i 的连接负载越重；c_i 是 MCU_i 连接的终端数目；c_{0i} 是 MCU_i 的容量，容量越大，说明 MCU 的总体处理能力越大；P_{ri} 即 U_i，是 MCU_i 的 CPU 利用率，P_{ri} 越大说明 MCU_i 的运行负载越重；P'_{ci} 是再接入一个终端后的连接负载因子；P'_{ri} 是再接入一终端后的估计 CPU 利用率。负载监控器定时更新区域内所有 MCU 的负载基本信息和每个 MCU 上的会议终端连接情况，此外还记录了 MCU 服务器负载阈值 c_0 和 U_0。

本文提出的组优先、最少级联最轻负载 LBA 平衡分配方法流程如下：

(1)网守收到终端连接进入会议请求。

(2)搜索网守中 MCU 终端连接情况表，找到已经接入该会议成员的 MCU，将 MCU 加入列表 $MCUList_0$。若不能找到符合条件的 MCU，执行(5)。

(3)对于 $MCUList_0$ 中的每个 MCU，调用式(2)、式(4)。

(4)在 $MCUList_0$ 中搜索满足 $P'_{ci}\leq 1$ 且 $P'_{ri}\leq U_0$ 的 MCU，并返回满足条件的 MCU 中具有最小 P'_{ri} 值的 MCU_i，执行(6)；如搜索不到，执行(5)。

(5)在全部 MCU 中搜索满足 $P'_{ci}\leq 1$ 且 $P'_{ri}\leq U_0$ 的 MCU，并返回满足条件的 MCU 中具有最小 P'_{ri} 值的 MCU_i；如搜索不到，向终端发送拒绝连接，执行(8)。

(6)令 $P_{ci}=P'_{ci}$，更新 MCU_i 负载信息。

(7)选中 MCU_i 为目标 MCU，调度终端分配器将视频终端分配到该 MCU。

(8)结束。

本方法首先搜索会议中已有的成员所在 MCU 的列表，优先考虑将同会议成员分配到相同 MCU 服务器中，大型会议中单 MCU 不能容纳所有会议成员，因此，在域内所有 MCU 中搜索负载最小的服务器进行分配。式(4)和式(6)实现了在 2 个异质 MCU 具有相同 CPU 使用率 P_{ri} 的情况下，优先将新会议的成员分配到处理能力比较强的 MCU 服务器上(具有更大的容量 c_0)。

终端重分配器：作用是将视频终端分配到目标 MCU 服务器上。一旦 MCU 选择器选择了目标 MCU，终端分配器就执行分配操作：将选择的目标 MCU 地址发送给网守，同时 LBA 更新相关负载统计信息，与视频终端建立连接。图 2 展示了负载均衡 Agent 的工作流程。

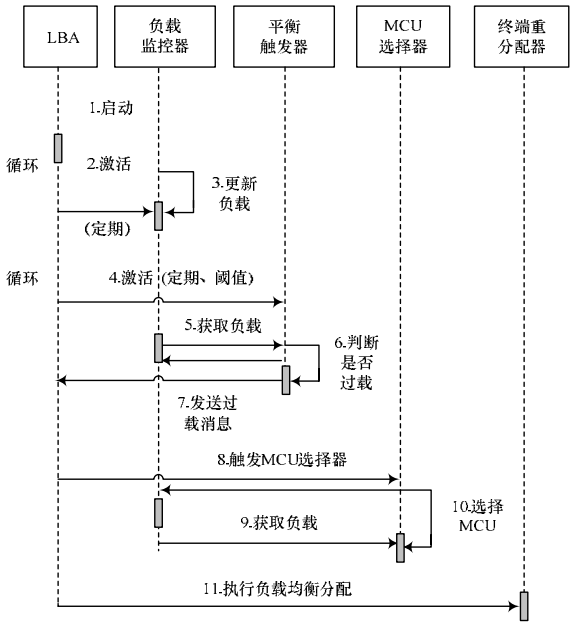

图 2　负载均衡流程

在程序运行时，LBA 先初始化 Agent 环境，定期调用负载监控器更新负载。同时平衡触发器周期性地检查获取负载监控器中的更新信息，并根据获取的 MCU 负载阈值判断 MCU 服务器是否过载。如果过载，平衡触发器发送过载消息

通知 LBA，LBA 将触发 MCU 选择器选择目标 MCU 服务器，然后通过终端重分配器将终端分配到目标 MCU 上。

3 实验

常用 MCU 利用率和 MCU 之间级联消耗作为衡量 MCU 分配算法的标准，这 2 个标准要求 MCU 分配算法将参与同一个会议的终端分配在尽可能少的 MCU 上(减少 MCU 间的级联)，并且 MCU 的利用率最高。

实验采用 CNGI 视讯会议系统终端的 MCU 软件和客户端的测试程序。实验硬件采用 2 台配备 Intel Xeon 3.2 GHz CPU 和 3.0 GB 内存的 IBM 刀片双 CPU 服务器作为 MCU，以 6 台奔 4 CPU(主频 2.4 GHz)、内存 512 MB 的 PC 运行基于客户端的测试程序。测得的 MCU 服务器的容量如表 2 所示。

表 2 MCU 容量

MCU ID	容量
MCU1	80
MCU2	75

实验设置 2 个会议：Ma 和 Mb。每个会议终端都随机取 4 路广播视频，为每个会议分别模拟 20 个、30 个、40 个、50 个终端连接 4 个测试序列。之后对静态资源预留算法、FRFA 算法和本文算法进行比较，测试结果如表 3~表 5 所示。从结果可以看出：在负载平衡方面，相比静态资源预留算法，FRFA 算法和本文算法能更有效地将终端连接负载分配到各个 MCU 上，各 MCU 负载相对均衡(2 台 MCU 的 CPU 使用率相差都在 5%左右)，而静态资源预留算法由于不能动态调节负载平衡，会出现负载失衡(负载全部在 MCU1 上，MCU2 则空载)。此外应用本文的方法比 FRFA 算法 CPU 使用率下降 10%左右。

表 3 静态资源预留算法测试结果

测试序列	测试会议用户数	MCU ID	终端分配情况	连接建立时间/ms	平均视频延时/ms	CPU使用率/(%)
1	Ma=20	MCU1	Ma=20,Mb=20	36	212	34
	Mb=20	MCU2	Ma=0,Mb=0	-	-	0
2	Ma=30	MCU1	Ma=30,Mb=30	107	288	52
	Mb=30	MCU2	Ma=0,Mb=0	-	-	0
3	Ma=40	MCU1	Ma=40,Mb=40	583	997	96
	Mb=40	MCU2	Ma=0,Mb=0	-	-	0
4	Ma=50	MCU1	Ma=50,Mb=0	92	545	64
	Mb=50	MCU2	Ma=0,Mb=50	104	582	73

表 4 FRFA 算法测试结果

测试序列	测试会议用户数	MCU ID	终端分配情况	连接建立时间/ms	平均视频延时/ms	CPU使用率/(%)
1	Ma=20	MCU1	Ma=11,Mb=9	24	114	24
	Mb=20	MCU2	Ma=9,Mb=11	29	123	27
2	Ma=30	MCU1	Ma=12,Mb=18	31	165	43
	Mb=30	MCU2	Ma=18,Mb=12	34	227	39
3	Ma=40	MCU1	Ma=20,Mb=20	34	238	64
	Mb=40	MCU2	Ma=20,Mb=20	36	271	68
4	Ma=50	MCU1	Ma=27,Mb=23	71	297	75
	Mb=50	MCU2	Ma=23,Mb=27	79	312	77

表 5 本文方法测试结果

测试序列	测试会议用户数	MCU ID	终端分配情况	连接建立时间/ms	平均视频延时/ms	CPU使用率/(%)
1	Ma=20	MCU1	Ma=20,Mb=0	38	108	15
	Mb=20	MCU2	Ma=0.Mb=20	35	113	18
2	Ma=30	MCU1	Ma=30,Mb=0	47	132	26
	Mb=30	MCU2	Ma=0,Mb=30	52	143	29
3	Ma=40	MCU1	Ma=20,Mb=20	67	210	54
	Mb=40	MCU2	Ma=20,Mb=20	74	216	63
4	Ma=50	MCU1	Ma=27,Mb=23	79	273	65
	Mb=50	MCU2	Ma=23,Mb=27	83	285	76

在系统处理性能方面，客户端视频数据往返延迟情况如图 3 和表 6 所示。可以看出，静态资源预留算法的时延最大。而本文算法相比 FRFA 算法在传输时延方面减少了 8%左右。

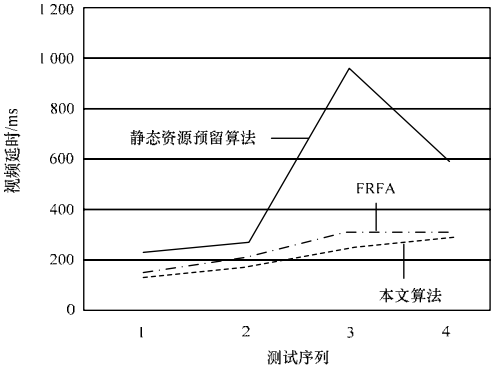

图 3 平均时延比较

表 6 3 种算法的平均时延比较 ms

测试序列	静态资源预留算法	FRFA	本文算法
1	212.0	118.5	110.5
2	228.0	196.0	137.5
3	997.0	254.5	213.0
4	582.0	304.5	279.0

4 结束语

本文提出的基于 Agent 的负载均衡方法由于采用了会议组优先的分配策略，兼顾 MCU 最少级联最少负载的分配原则，因此能平衡各个 MCU 的负载，并有效地降低 MCU 的级联，从而减少因级联带来的媒体数据传输时间。

参考文献

[1] 刘瑞芳, 曾晓轩, 孟庆昌, 等. H.323 视频会议中 MCU 的设计与实现[J]. 北京邮电大学学报, 2003, 26(2): 77-81.

[2] 汪建球, 张忠能. 分布式 Web 服务器中负载均衡的实现[J]. 计算机工程, 2003, 29(15): 125-127.

[3] 侯宗浩, 董小社, 郑守淇, 等. 多入口集群负载均衡问题研究[J]. 计算机工程, 2004, 30(18): 93-95, 126.

[4] 樊 华, 朱 莉, 王媛妮. 在基于 H.323 的视频会议中实现负载均衡[J]. 计算机技术与发展, 2006, 16(4): 122-124.

[5] Wooldridge M. Agent-based Software Engineering[J]. IEE Proceedings of Software Engineering, 1997, 144(1): 26-37.

编辑 张 帆