

# 一种支持服务网格的动态负载平衡系统

申德荣, 陈翔宇, 吕立昂, 邵一川, 于戈

(东北大学信息科学与工程学院, 沈阳 110004)

**摘 要:** 为了实现服务网格系统内负载的均衡分布, 提高资源利用率和系统的吞吐率, 设计并实现了一种基于服务网格环境的动态负载平衡系统。提出了层次式负载平衡调度模式, 给出了本系统结构形式, 设计并实现了一种综合考虑各局部代理作业数和各个局部代理性能以及当前的负载情况的动态双阈值作业分配算法。实验结果表明, 此算法能有效地基于负载分派作业, 达到了提高网格内分布资源的利用率和减少作业调度时间的目的。

**关键词:** 服务网格; 负载平衡; 负载; 作业分配

## A Dynamic Load Balance System for Supporting Service Grid

SHEN Derong, CHEN Xiangyu, LV Li'ang, SHAO Yichuan, YU Ge

(College of Information Science and Engineering, Northeastern University, Shenyang 110004)

**【Abstract】** In order to realize load balance distribution and improve resource availability and throughput rate of a service grid system, a dynamic load balance system is designed and implemented. A hierarchical load balance scheduling mode is proposed and the system structure is presented. A dynamic double threshold algorithm for job assigning is designed and implemented by considering the number of jobs and the current load in local agents as well as the performance of these local agents together. The test results show that the algorithm can assign jobs effectively based on load to acquire the improvement of the availability of the distributed resource in the grid system and the decrease of the overall runtime of job scheduling.

**【Key words】** Service grid; Load balance; Load; Job assignment

网格把整个 Internet 整合成一台巨大的超级计算机, 实现计算、存储、数据、信息、知识、专家等资源的全面共享和协同。目前针对面向企业资源集成的服务网格的研究并不多。服务网格 (Service Grid) 推动了 Web Service 技术的应用, 为企业最大范围地提供了资源共享, 它将是提高企业总体水平和能力的最有效途径。

目前在网格<sup>[1]</sup>的各个方面都有了大量的研究, 比较著名的系统如: Globus, Legion, Condor等, 其开发目标是有效地使用地理上分布的资源, 这些系统主要处理的问题有资源发现、定位、安全等, 在调度问题上的研究很少。然而, 如何协调网格内分布的异构资源, 提高资源利用率和系统的吞吐率是很重要的, 也是网格系统中的核心研究问题。

本文针对服务网格系统负载变化的特点, 分析了已有负载平衡调度模式, 提出了一种层次式的负载平衡调度模式, 设计并实现了一种综合考虑各局部代理作业数和各个局部代理性能以及当前的负载情况的动态双阈值作业分配算法, 最后通过实验, 同已有的作业分配算法进行了对比, 验证了该动态负载平衡系统的有效性。

### 1 动态负载平衡调度模式

服务网格系统中, 作业是动态产生的, 每个局部代理的负载大小是动态变化的, 因而不考虑静态负载平衡调度方法,

只考虑动态负载平衡调度策略。一般说来, 动态负载平衡调度<sup>[2-4]</sup>可分为集中式调度、分布式调度两大类。前者所有的作业都提交给全局代理, 由全局代理负责搜集各局部代理负载

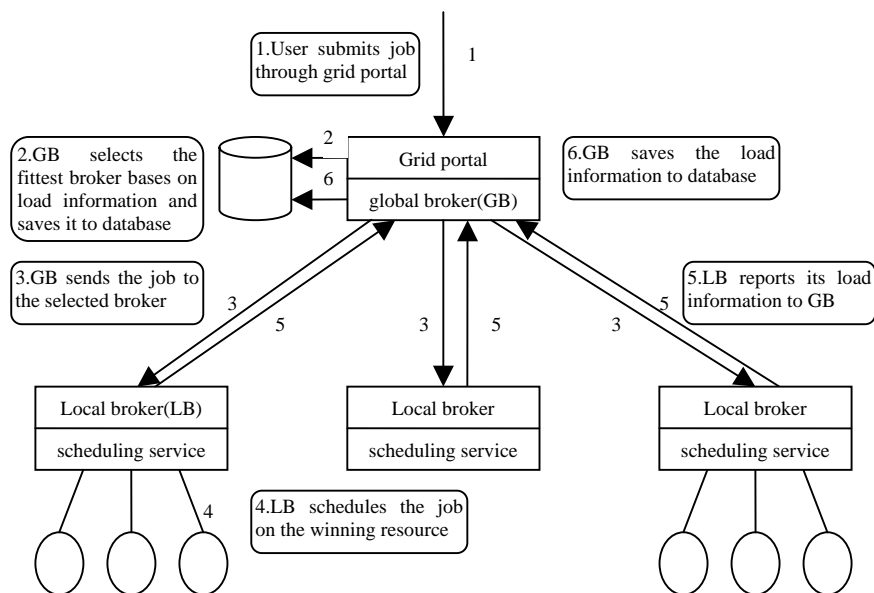


图1 层次式负载平衡调度模式

**基金项目:** 国家“863”计划 CIMS 主题基金资助项目(2003AA 414210); 国家自然科学基金资助项目(60173051)

**作者简介:** 申德荣(1964 - ), 女, 教授, 主研方向: 分布式数据处理, 服务网格, Web 服务; 陈翔宇, 工程师、硕士; 吕立昂, 硕士生; 邵一川, 硕士生、助教; 于戈, 教授、博士

**收稿日期:** 2005-11-21 **E-mail:** shendr@mail.neu.edu.cn

信息,并由它来决定负载平衡调度方案,在这种模式下局部代理不执行调度,只是负责对全局代理分配给其作业的分派并将作业结束信息提交给全局代理,它的主要优点在于实现比较简单,缺点是在结点数较多的大规模服务网格环境中全局代理会成为系统的瓶颈,所以调度开销比较大;后者每一个局部代理都可以接收作业并进行调度,它们根据局部范围内的一些负载信息来进行负载平衡操作,每台计算机定期把它的负载信息广播给其它计算机,去更新那些局部维护的负载向量,它的最大优点在于具有良好的可扩放性。缺点在于各节点间通信量较大,因此作业等待时间较长。

针对这两种调度模式的优缺点,我们提出了一种层次式调度模式,调度模式如图1所示。同样由全局代理负责收集各局部代理负载信息,所有的作业都提交给全局代理,但是与集中式调度不同,这些任务并不是全部保存在全局代理作业提交队列中等待调度,而是直接由全局代理根据负载平衡分配给局部代理,由局部代理进行调度,那么全局代理将不再干预作业的调度,这样全局代理的负载减小,避免了成为系统瓶颈,同时也减短了作业等待时间,而且实现起来也比分布式调度简单。

## 2 层次式负载平衡系统结构

该系统实现动态负载平衡。它主要由以下部分组成(如图2):

(1)作业分配模块(Job Distributing Model):它接收门户发出的作业派生请求,根据各局部代理的负载状况进行作业分配,达到动态负载平衡的目的。

(2)调度模块(Scheduling Model):每个局部代理有一个调度模块,负责进行作业调度及动态收集本地结点的负载信息。主要包括:CPU利用率、活动进程数以及内存使用情况和I/O使用情况等,并定期将信息通过通信模块传给作业分配模块。

(3)监视模块(Monitor Model):监视局部代理是否过于繁忙或清闲,启用或撤下局部代理。

(4)通信模块(Communication Model):用GT3包装好的Grid Service,通信模块负责监视模块与调度模块之间的通信。

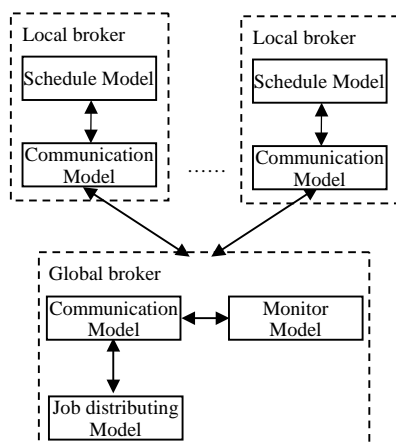


图2 层次式负载平衡系统结构

## 3 层次式负载平衡系统的算法与实现

### 3.1 层次式负载平衡调度原理

层次式负载平衡调度的主要原理是由全局代理集中接收所有的作业请求,然后依据一定的原则把它们分配到各局部代理上去进行调度执行。分配的主要目的是使各局部代理的

负载分布比较均衡,从而获得较高的整体吞吐能力和较快的反应速度。我们借鉴并行WWW服务器集群中采用的请求分配技术,目前常用的请求分配方法<sup>[2~4]</sup>主要有转轮法、最少连接法和最快连接法3种。第1种方法实现简单,但从本质上讲没有考虑负载均衡问题。第2种方法对各个服务器的性能不加区分。第3种方法虽然考虑了服务器的性能,但没有考虑服务器的当前负载情况。这些局限性使这些算法无法真正做到负载的“均衡”分配。

为了有效地提高作业分配算法的效率,并使算法能够适应服务网格系统,应使全局代理能够知道每一台局部代理的处理能力,同时应能比较准确地跟踪各个局部代理的负载情况。局部代理的处理能力可以在调度过程中通过实测获得,而负载跟踪则可以通过在请求分配器上记录各服务器的应答进度来实现。基于以上考虑,本文设计了一种综合考虑各局部代理作业数和各个局部代理性能以及当前的负载情况的动态双阈值作业分配算法。

### 3.2 负载的定义

常用的负载指标<sup>[5]</sup>(也称负载分量)包括CPU利用率、CPU就绪队列长度、内存使用情况和磁盘访问频度,以及I/O速度等。同时需要考虑到异构结点的处理能力的差异。

测试表明,计算密集型任务的运行会迅速提高CPU利用率,在单CPU工作站上,投入运行一个这样的任务,就能够使CPU利用率达到85%~99%的程度。也就是说,对于科学计算应用程序,CPU利用率只能明显地区分空闲状态和非空闲状态,而不能进一步反映忙的程度,于是也就不足以用来准确地进行负载决策和预测。于是,本系统将综合使用上述几个负载指标来综合考察结点的负载信息。

#### 3.2.1 分类算法

可以根据作业的性质(计算密集型CPU类、I/O类等),确定主负载指标,例如:

CPU类:以CPU利用率、CPU进程队列长度为重权负载分量;

I/O类:以磁盘读写速率、磁盘访问频率以及磁盘可利用空间为重权负载分量;

Mem类:以FreeMem和MemorySwap频度为主要指标。

#### 3.2.2 权重向量法

结点负载用下面的公式来表示:

$$L = \sqrt{\sum_{i=1}^n (k_i \cdot a_i^2)}$$

其中:  $L$  表示本地结点的负载值;  $a_1, a_2, \dots, a_n$  分别是选定的负载分量(负载指标),  $k_1, k_2, \dots, k_n$  分别是权重。通常来说,对于用户事先指定的主负载指标,权重较大。

### 3.3 性能评价参数和影响性能的因素

评价局部代理的性能我们采用吞吐率和平均应答延迟两个参数。吞吐率RPS定义为平均每秒处理的作业个数,平均应答延迟MRD定义为从接收作业到作业结束之间的时间的平均值。所采用的测试集由若干大小不一的作业按照一定的分布规律组成。所谓作业的大小,是指作业执行在独占处理机时的执行时间。

### 3.4 动态双阈值作业分配算法的实现

动态双阈值作业分配算法主要由3部分组成:一部分用于得到各局部代理的负载信息,另一部分用于全局代理根据负载平衡选择合适的局部代理进行作业分配,再一部分用于监视各局部代理的当前负载是否过重或过轻并采取相应的措施。

### (1) 负载跟踪

每隔一个固定的小时间段 (此时间间隔由局部代理自身决定), 局部代理向全局代理汇报局部代理负载信息, 以及局部代理上运行的所有作业状态或作业结束信息, 全局代理进行如下操作:

- 1) 根据局部代理负载信息计算各个局部代理的负载;
- 2) 更新各局部代理服务器的当前作业数;
- 3) 计算各局部代理的吞吐率和平均应答延迟。

### (2) 动态作业分配

服务网格系统中, 作业是动态产生的, 作业分配算法是系统最重要的算法。我们考虑的负载指标主要包括各局部代理当前负载、性能、任务数等。可以静态或动态地确定负载指标的权重, 按权重链入负载队列向量, 结点的负载根据权重计算记录其负载轻重情况。

通过计算下列公式, 函数  $H(A)$  给出对应于局部代理  $A$  的处理能力, 得到作业分配的目标结点:

$$T_z = H \left\{ \max \sqrt{\sum_{i=1}^n w_i L_i^2} \right\} \quad (1)$$

其中:  $z$  为待分配的作业,  $L$  为各项负载指标,  $w_i$  为相应权重。一般规定:

$$w_i \geq 0, \text{ 且 } \sum w_i = 1$$

如果有作业请求到达, 对每个局部代理, 采用式(1)的方法计算得到作业分配的目标局部代理。

### (3) 双阈值负载监视

局部代理负载大小超过预定设置阈值时, 该算法被触发。当监视模块检测到各局部代理负载过大或过小时, 通过启用或撤下局部代理, 从而达到系统内的负载平衡。具体如下:

- 1) 当最低负载代理超过一定的阈值时, 说明各代理的负载过重, 负载监视模块会自动启用备用代理。
- 2) 如果最高负载低于某值时, 说明各代理的负载过轻, 负载监视模块会撤下部分局部代理。

## 4 实验测试

本文通过实验对提出的层次式负载平衡调度方法和已有的 3 种调度方法 (轮转法、最少连接法和最快连接法) 的性能进行了对比。

### (1) 实验环境

在校园网环境下, 采用 4 台微机 (2.4GHz CPU 256M MEM(local) WindowsXP, 1.8GHz CPU 256MB MEM(local) WindowsXP, 700MHz CPU 320M MEM(local) Windows2000, 400MHz CPU, 320M MEM(global) WindowsXP)。

### (2) 测试作业

本实验只设置了内存型和 CPU 型两种作业。因此作业大多是处理型作业, 实验中并没有考虑 I/O 型作业。具体作业如下:

内存型任务: 30 个作业, 每个作业包括 64 次 10 层的汉诺塔

CPU 型任务: 30 个作业, 每个作业包括 32 000 次求公因数 (214 748 3647, 3)。

在实验中, 将作业组成作业序列, 之后根据不同的作业分配方法进行作业分配和处理, 测出作业的总处理时间。我们对每种实验都进行了 5 组实验, 具体实验如下:

1) 采用内存型作业和层次式负载平衡调度方法时: 变化 CPU 和内存的权重, 测试作业完成的时间, 结果如图 3 所示。其中  $W_{cpu}$ 、 $W_{Mem}$  分别表示 CPU 和内存 (MEM) 因素所占的权重。

2) 采用 CPU 型作业和层次式负载平衡调度方法时: 变化 CPU 和内存的权重, 测试作业完成的时间, 结果如图 4 所示。

3) 采用 CPU 型作业: 令  $W_{cpu}=1$ ,  $W_{Mem}=0$  时, 测试分别采用层次式负载平衡调度方法、转轮法、最少连接法和最快连接法时的作业完成的时间, 结果如图 5 所示。

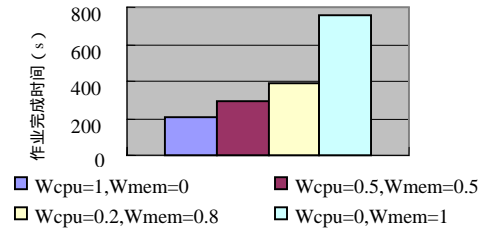


图3 不同权重对内存型作业运行时间的影响

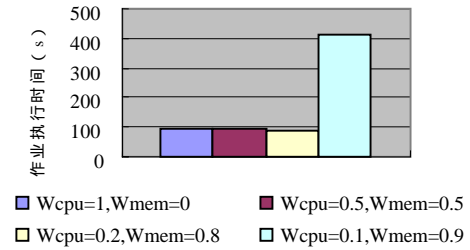


图4 不同权重对CPU型作业运行时间的影响

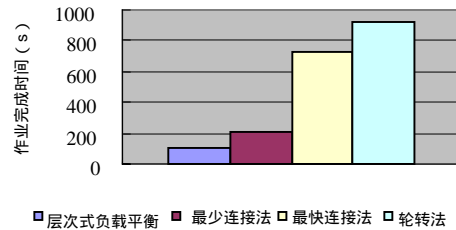


图5 采用不同算法的作业完成时间

从测试结果看, 对于内存类型的作业 (汉诺塔), 随着 MEM 权重逐渐变大, 性能逐渐变差。而对于 CPU 类型的作业, 随着 MEM 权重逐渐增大, 对性能影响不大, 但当 MEM 权重非常大时, 会造成性能急剧下降。内存权重加大而产生不良影响的原因在于: 内存变化率小 (内存型作业也是), 不能正确反映机器忙碌状态。可见, 判断局部代理忙碌与否, 应以 CPU 指标为主, 而对于内存等其他因素影响很小。

实验表明, 在负载平衡系统支持下, 可以改善动态任务的分配和调度, 使得任务的运行时间缩短, 提高了服务网格的并行运行性能。同时如果作业量增多, 本系统可以自动启用备用的局部代理, 保证运行时间没有增长很多。

## 5 结论

目前关于网格调度策略的研究还不是很多, 本文分析了服务网格中动态负载平衡几种调度模式, 提出了层次式调度模式, 给出了本系统负载平衡系统结构形式, 设计并实现了一种综合考虑各局部代理任务数和各个局部代理性能以及当前的负载情况的动态双阈值作业分配算法。

该算法具备如下特点:

- (1) 智能化处理: 监视模块智能化监控局部代理的负载进行启用或撤下代理, 无须人工控制;
- (2) 动态配置优先级: 用户可在运行前根据作业种类的不同配置各项负载指标权重;
- (3) 负载均衡: 综合考虑了作业数、局部代理负载以及局部代理性能进行作业分配, 真正做到了负载均衡。

(下转第 129 页)