网格服务调度算法分布式部署和QoS性能分析

刘宏岚 郝卫东 高庆狮
(北京科技大学信息工程学院 北京 100083)

摘 要 网格是一种复杂的分布式计算系统,研究其网格服务对网格作业的调度算法的分布式部署和性能分析问题具有重要的意义。网格服务调度系统的状态空间模型考虑了具有不同的输入速率和输出速率的作业队列,提出了清空型调度策略和服务调度算法,并在此基础上分析了其分布式部署问题,计算了系统QoS性能指标,指出了稳态吞吐量、稳态响应时间与负载系数的关系。

关键词 网格,服务调度算法,分布式部署,性能分析
中图法分类号 TP393.04 **文献标识码** A

Distributed Deployment and QoS Performance Analysis of Grid Services Scheduling Algorithm

LIU Hong-lan HAO Wei-dong GAO Qing-shi
(Information Engineering School, University of Science and Technology Beijing, Beijing 100083, China)

Abstract Grid is a complex distributed computing system, distributed deployment and QoS performance analysis of grid services scheduling algorithm were studied. State space model was given to describe the grid dynamic scheduling system which processes a number of heterogeneous jobs concurrently. Clearing policy as a scheduling policy and services scheduling algorithm were given. Distributed deployment problem was analyzed. System QoS performance parameters were calculated, stable throughput and stable response time and their relationship with load factor were pointed.

Keywords Grid, Services scheduling algorithm, Distributed deployment, Performance analysis

随着面向服务的网格技术的不断发展,网格服务对网格作业的调度策略成为网格计算研究的趋势。文献[1]研究了网格服务调度系统的动态稳定性问题,给出和证明了使事件按期望的时序发生的稳定性条件和调度策略。文献[2]针对目前服务网格资源管理中存在的信任机制与调度机制分离的缺陷,基于网格信任模型与信任效益函数,讨论了信任QoS(服务质量)增强的计算服务调度问题。文献[3]针对计算网格资源的特点以及运用经济机制进行网格资源管理所具有的灵活性及有效性,提出了一种改进的基于双向拍卖机制的网格资源分配方法。其不足是没有把网格资源分配与网格作业的运行完成时间相结合。基于不确定性推理理论,文献[4]建立了适合网格环境的信任模型,提出了可信动态级调度算法,从而减少应用任务被欺骗的概率和执行失败的概率,使任务分配到值得信赖的环境下执行,其缺点是难以涵盖各种不同的不确定性现象。文献[5]提出了在复杂工作流应用中研究高级调度策略的网格仿真框架,其不足之处是没有研究清空策略。文献[6]通过改进传统的Min-Min作业调度启发式,提出了一个自适应、性能QoS驱动的Min-Min启发式算法。文献[11]提出的选择性网格资源配置策略解决了如何以低资源消耗保证用户需求的问题。作为支撑Web服务应用的主流平台的Web应用服务器对请求的调度仍然是传统的先来先服务策略(FCFS),这种策略无法区分请求的重要性,降低了关键请求的性能。对此,文献[12]提出了应用敏感的Web服务请求调度策略,该策略以应用获益来评估服务器为应用提供的性能保障效果,其缺点是难以对应用获益进行精确的量化处理。另外,前面提到的这些文献都没有从网格系统的分布式部署和算法的稳态性能指标方面加以分析。

本文采用连续流逼近的离散事件动态系统(DEDS)的状态空间模型分析网格服务调度方案中离散事件交互影响导致的系统状态的演化过程和相应的调度策略,给出了在分布式部署条件下分析网络服务调度系统的多种服务质量指标(如稳态吞吐量、稳态响应时间及其与负载系数的关系)的算法,并与理论计算结果进行了比较,说明了系统模型和调度算法的可行性。

1 网格服务调度的状态空间模型

对所描述的网格服务调度系统涉及的变量命名做如下假设:

系统中有n个异构的网格作业队列共享单一的网格服务资源,该网格服务系统具有有限容量的输入缓冲区;同一个网格作业队列中的网格作业是同种类型的,而且每个网格作业以平均速率λ_i进入网格作业队列i($i=1,2,3,\cdots,n$)。网格作业队列i中的作业请求的平均完成速率用μ_i($i=1,2,3,\cdots,n$)表示。

到稿日期:2010-07-13 返修日期:2010-10-28 本文受国家自然科学基金(60873192,60673160,60873002)资助。
刘宏岚 女,博士,主要研究方向为离散数学、计算机网络,E-mail:Wed@ustb.edu.cn;郝卫东 男,博士,主要研究方向为网格计算;高庆狮 男,院士,主要研究方向为并行计算、计算机网络。

网格服务一旦选择执行某个网格作业队列，就会将网格服务能力切换到适当的作业队列。因此，通常用离散的逻辑变量表达相应的调度策略，记为 $s_i(i=1,2,3,\cdots,n)$。其中，当网格服务决定将服务能力切换到作业队列 i 时，$s_i=1$，否则 $s_i=0$，考虑可以同时被处理的网格作业队列的个数为 1 个，即 $\sum_{i=1}^{n} s_i=1, s_i\in\{0,1\}$。

在网格服务的缓冲区中，等待被处理的网格作业个数用连续取值的状态变量表达，记为 $x_i, i=1,2,3,\cdots n$，分别表示各个网格作业队列中同种类型作业的数量。其中，$x_i\geqslant 0, i=1,2,3,\cdots n$。$y_i$ 表示各个作业队列中单位时间内离开网格服务的作业个数。在某段时间内，离开网格服务的作业个数的总数等于进入的作业个数的总和减去在队列中等候被处理的作业的个数的总和。

由此，所描述的网格服务调度系统动态过程的状态方程为：

$$\begin{cases}\dfrac{\mathrm{d}x_i(t)}{\mathrm{d}t}=\lambda_i-s_i\mu_i\\ \int y_i(t)\mathrm{d}t=\int\lambda_i t\mathrm{d}t-\int x_i(t)\mathrm{d}t\end{cases}\tag{1}$$

上述状态方程模型中 x_i 是用连续流逼近的离散变量，s_i 是离散的逻辑变量，λ_i 和 μ_i 是连续变量，因此整个系统构成一个混合类型变量系统。

2 清空型调度策略

调度策略的本质可以理解为从多个对象中选择一个符合某个条件的对象进行服务[7,8]。

定义 1(清空型调度策略) 对一个网格服务系统，网格服务进行切换选择的决策时刻被定为“要么是开始时刻，要么是与该网格服务连接的缓冲队列有一个被清空的时刻”，在决策时刻 T_k 选择切换到某个队列的准则是所切换到的队列的长度不低于各个队列长度总和的 ε 倍，即在清空事件 k 对应的决策时刻 T_k 被选择清空的队列 i^* 的队列长度 $x_{i^*}(T_k)$ 满足如下关系：

$$x_{i^*}(T_k)\geqslant\varepsilon\sum_{i=1}^{n}x_i(T_k)\tag{2}$$

式中，ε 为 0 到 1 之间的固定常数，$k=0,1,\cdots$，初始时刻取为 $T_0=0$，n 为进入网格服务系统的网格作业种类数。

3 网格动态服务调度的算法

考虑用式(1)表达网格服务调度问题，与输入流 λ_i 对应的调度变量采用清空型调度策略，按如下公式计算清空时间间隔：

$$\Delta T_k=T_{k+1}-T_k=\frac{x_{i^*}(T_k)}{\mu_{i^*}-\lambda_{i^*}}\tag{3}$$

决策时刻 T_k 被选择清空的队列 i^* 的队列长度 $x_{i^*}(T_k)$ 满足如下关系：

$$x_{i^*}(T_k)=\max x_i(T_k)\tag{4}$$

考虑到式(4)蕴含如下不等式关系：

$$x_{i^*}(T_k)\geqslant\frac{1}{N}\sum_{i=1}^{N}x_i(T_k)\tag{5}$$

由定义 1 可知，式(5)实际上就是取 $\varepsilon=1/N$ 情况下的一种特殊的清空型调度策略。

给出离散化后队列长度状态变量 x_i 的变化规律如下：

$$x_i(T_{k+1})=\begin{cases}(\lambda_i-\mu_i)\Delta T_k+x_i(T_k), & s_i=1\\ \lambda_i\Delta T_k+x_i(T_k), & s_i=0\end{cases}\tag{6}$$

根据上述公式，给出队列长度、队列中作业的总个数、系统的吞吐量、作业请求的响应时间等性能指标的计算方法和步骤：

1)设队列长度初值为 $X(T_0)$ 以及设置清空事件编号 $k=0$，它对应的时刻 $T_k=0$，不同种类的作业队列有 N 个，队列中作业的总个数 $N_q(0)=\sum_{i=1}^{N}x_i(T_0)$，累计的各类作业到达个数 $N_\lambda(0)=N_q(0)$，累计的各类作业完成的个数 $N_c(0)=N_\lambda(0)-N_q(0)=0$。

2)令 $k=k+1$，根据清空最长队列的调度策略 $x_{i^*}(T_k)=\max x_i(T_k)$，选择 k 事件时适当的被清空队列 i^*。

3)计算清空时间间隔 $\Delta T_k=\dfrac{x_{i^*}(T_{k-1})}{\mu_{i^*}-\lambda_{i^*}}$ 以及对应 k 事件的时间 $T_k=T_{k-1}+\Delta T_k$，进一步计算累计的各类作业到达个数 $N_\lambda(k)=N_\lambda(k-1)+\Delta T_k\sum_{i=1}^{n}\lambda_i$。

4)计算各个作业队列的等待时间或处理时间。若 $i\neq i^*$，则清空时间间隔 ΔT_k 为 i 队列的等待时间，记为 $T_w(i)$。显然，等待时间可能累加，所以 $T_w(i)=T_w(i)+\Delta T_k$；若 $i=i^*$，则清空时间间隔 ΔT_k 为 i 队列的处理时间，记为 $T_f(i)$。由于清空处理是一次性的，因此 $T_f(i)=\Delta T_k$，无累加特性。

与 k 事件的发生相对应，i^* 队列被清空，因此 i^* 队列中的各个作业此时都被应答，这样就可以计算出 k 事件发生时 i^* 队列中的各个作业的最大响应时间 $T_{n^*}(k)=T_w(i^*)+T_f(i^*)$。

计算出 i^* 队列的最大响应时间 $T_{n^*}(k)$ 后，令 $T_w(i^*)=0, T_f(i^*)=0$。

5)根据式(6)计算 T_k 时刻的各个队列的长度 $X(T_k)=\begin{pmatrix}x_1(T_k)\\ x_2(T_k)\\ x_3(T_k)\end{pmatrix}$，进一步计算队列中驻留的作业的总个数 $N_q(k)=\sum_{i=1}^{N}x_i(T_k)$，累计的各类作业完成的个数 $N_c(k)=N_\lambda(k)-N_q(k)$。

6)计算对应 k 事件的 T_k 时刻系统的平均吞吐量 $H=\dfrac{N_c(k)}{T_k}$。

7)若各个队列的长度为 0 或者队列长度始终小于某个固定常数而不再变化，则退出程序，否则转第 2)步。

4 服务调度算法的分布式部署

利用建立的网格平台，根据第 3 节的服务调度 QoS 性能分析算法，通过实验分析算法在不同负载系数下的过渡过程时间、稳态响应时间、稳态吞吐量、稳态时的 K 值、稳态时的队列长度和，并与根据 Little 公式计算的稳态响应时间相比较。

实验内容主要包括两项：第一是获取服务调度算法在不同的网格节点上运行的相关参数，比如作业运行时间、作业分配结果、作业运行结果；第二是利用第一步的实验数据进行结果分析，包括对算法本身的分析(含与 Little 公式的比较)和对算法运行情况的分析。

这里需要获得 10 组数据，即负载系数 $\rho=\sum_{i=1}^{3}\dfrac{\lambda_i}{\mu_{\min}}=0.1\sim$

1.0，显然它们都满足稳定性条件[1]。

为了体现网格计算的分布式特点，把10组数据的计算任务分解成10个独立的计算作业，用服务指派算法自动匹配到8台相应的服务资源节点上运行，服务调度QoS性能分析算法在节点上本地化运行，其运行结果以格式化的.txt文件的形式通过网格集群管理服务器返回给用户。

5 分布式服务调度算法的性能分析

考虑用矩阵表达网络服务调度问题。当$\rho=1.0$时，有

$$X=\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix},X(T_0)=\begin{bmatrix}x_1(T_0)\\x_2(T_0)\\x_3(T_0)\end{bmatrix}=\begin{bmatrix}0\\100\\0\end{bmatrix},$$

$$\lambda=\begin{bmatrix}\lambda_1\\\lambda_2\\\lambda_3\end{bmatrix}=\begin{bmatrix}50\\50\\60\end{bmatrix},\mu=\begin{bmatrix}\mu_1\\\mu_2\\\mu_3\end{bmatrix}=\begin{bmatrix}200\\160\\180\end{bmatrix}$$

由于结果中产生的数据比较多，因此把稳态时的运行结果按照负载系数ρ的大小依次排列。为了与理论值相比较，把根据Little公式计算的稳态响应时间也罗列了进去。

在计算中，系统的最小服务处理速率参数采用$\mu_{min}=160$个/s，根据不同的负载系数ρ，可计算得到不同的到达速率和$\lambda=\sum_{i=1}^{3}\lambda_i$，并分配到3个不同的输入队列中。系统达到稳态时，各队列的长度分别小于等于5，整个系统的队列长度总和小于等于7。

从排队系统性能的角度分析，要考虑稳态吞吐量（见图1）和稳态响应时间（见图2）。

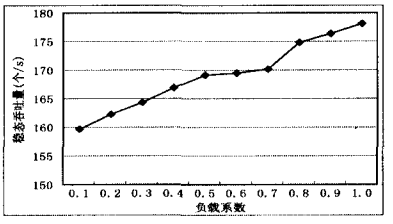


图1 稳态吞吐量与负载系数ρ的关系

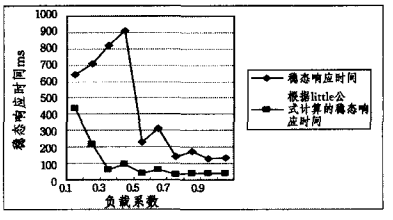


图2 理论与实际的稳态响应时间与负载系数ρ的关系

从图1可看出，稳态时的吞吐量随负载系数ρ的增加而单调增加。当负载系数$\rho\geqslant0.8$后，增加速度明显变慢，近似饱和延伸。这是合理的，因为ρ的增加将导致到达速率和λ的增加，而处理速率保持不变，因此调度系统需要在不改变自身输出处理能力的前提下处理更多的用户作业。当$\rho\geqslant0.8$后，系统处于高负载率情况下，吞吐量接近系统的最大负载，增加不再明显。有趣的是，可以认为此时网格作业在网格服务资源中的状态与TCP/IP网络的拥塞避免状况类似[9]。更重要的相同点是，一旦$\rho>1.0$，系统的吞吐量将突降到零，在TCP/IP网络中称为拥塞崩溃。同样地，这种情况在本文的服务调度系统中进入不稳定状态。在本例的数值分析中，$\rho=1.1$显示的结果是吞吐量为一个负数，数值分析算法将失效。此时，为了满足用户作业到达速率的要求，就需要提高网格处理速率，其方法是选择更好的机器或选择更多的机器实现分布式计算。

从图2可看出，稳态时的响应时间随负载系数ρ的增加而振荡减少。当负载系数$\rho\geqslant0.8$后，减少速度明显变慢，近似水平延伸。这是合理的，因为从图2可知道，ρ的增加一方面导致到达速率和λ的增加，另一方面导致系统吞吐量不断增加。当到达速率和λ的增加速率快于系统吞吐量的增加速率时，每个作业的稳态响应时间增加。当到达速率和λ的增加速率慢于系统吞吐量的增加速率时，每个作业的稳态响应时间减少。但对每个作业而言，由于系统吞吐量的不断增加，其响应时间的总趋势是下降的。当$\rho\geqslant0.8$后，系统吞吐量进入饱和状态，每个作业的稳态响应时间的变化趋势也趋于平稳。但同时整个系统中的作业数量随到达速率和λ的增加而继续增加，这就导致系统稳态时的决策时间（过渡过程时间）和稳态时的事件编号k值近似直线上升式地突然增加，从而使系统进入高负载率情况。

从图2可看出，Little公式[10]计算的稳态响应时间比模型计算的值明显小，不过仍保持随负载系数ρ的增加而振荡减少的特征，在高负载系数下响应时间变化趋于平缓。这一方面说明模型计算的稳态响应时间是合理的，证明了模型的有效性；另一方面，也说明了状态空间模型描述的服务调度系统本身具有复杂性，Little公式不能完全适用于该系统。

结束语 建立了基于DEDS的分布式网格服务调度问题的状态方程模型，讨论了在适当调度策略下系统的多种服务质量指标（系统的吞吐量、作业请求的响应时间等）的计算。计算表明，本调度策略可以保证具有有限容量的输入缓冲区的网格服务系统实现平稳作业处理。计算还进一步表明了本网格服务调度系统与TCP/IP网络的拥塞避免状况类似。还讨论了Little公式的适用性。

参 考 文 献

[1] Yang Yang, Hao Wei-dong, Zeng Ming. Stability Analysis for Grid Services Dynamic Scheduling System[C]// IEEE International Conference on Computational Intelligence and Security (CIS-2007). Harbin, China, Dec. 2007: 350-353

[2] 张伟哲，方滨兴，胡铭曾，等. 基于信任QoS增强的网格服务调度算法[J]. 计算机学报，2006(7)

[3] 翁楚良，陆鑫达. 一种基于双向拍卖机制的计算网格资源分配方法[J]. 计算机学报，2006，29(6)：1004-1008

[4] 袁禄来，曾国荪，姜黎立，等. 网格环境下基于信任模型的动态级调度[J]. 计算机学报，2006，29(7)：1217-1224

[5] Hirales-Carbajal A, Tchernykh A, Roblitz T. A Grid simulation framework to study advance scheduling strategies for complex workflow applications [C]// 2010 IEEE International Symposium on IPDPSW. 2010: 1-8

[6] He X, Sun X, Gregor V L. QoS guided Min-min heuristic for grid task scheduling[J]. Journal of Computer Science and Technology, 2003, 18(4): 442-451

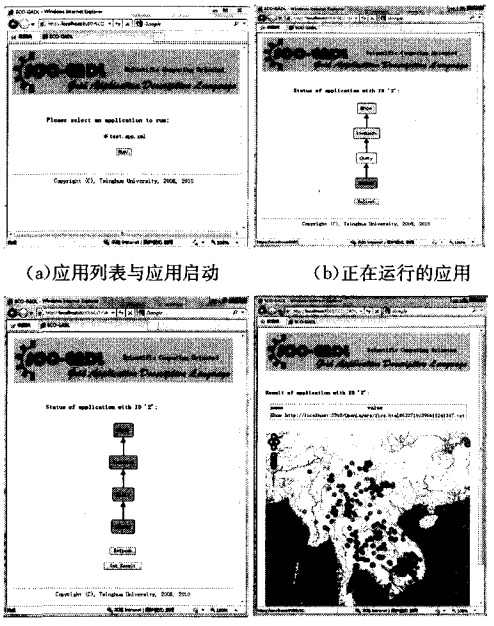
[7] Lin Chuang, Xu Ming-wei, Marinescu D C, et al. A sufficient condition for instablility of buffer priority policies in re-entrant lines[J]. IEEE Trans. on Automatic Control, 2003, 48(7): 1235-1238

[8] Burgess K, Passino K M. Stable Scheduling Policies for Flexible

Manufacturing Systems[J]. IEEE Trans. on Automatic Control, 1997,42(3):420-425
[9] 林闯,单志广,任丰原. 计算机网络的服务质量(QoS)[M]. 北京:清华大学出版社,2004
[10] 郑大钟,赵千川. 离散事件动态系统[M]. 北京:清华大学出版社,2001
[11] 李响,孙华志. 网格资源选择性配置研究[J]. 计算机科学,2010,37(4):114
[12] 官荷卿,张文博,魏峻,等. 一种应用敏感的 Web 服务请求调度策略[J]. 计算机学报,2006,29(7):1189-1198

(上接第 30 页)

为了测试 SCO-GADL 及其应用引擎,图 2 所示的应用也被部署到了 SCO-GADL 应用引擎中,命名为 test. app. xml。这个应用的运行过程如图 5 所示。



(a)应用列表与应用启动　(b)正在运行的应用

(c)应用运行结束　(d)应用的结果

图 5　示例应用

图 5 显示了一个由 SCO-GADL 描述的森林火灾监测应用,该应用首先根据 source 数据项中描述的条件搜索合适的 MODIS 数据,然后对该 MODIS 数据进行反演处理以获得疑似火点的经纬度,最后再通过一个 Web 服务将这些疑似火点显示在地图上以供查找确认。在 SCO-GADL 应用引擎的支持下,该应用得到成功运行并得到了正确的结果,这表明 SCO-GADL 具有足够的能力来表达各种科学计算应用。

结束语　虽然已经有很多项目试图提供包括被广泛使用的网格工作流在内的各种工具和库来降低网格应用开发的难度,但是网格应用开发的门槛依旧是阻止网格成为科学计算基础设施的重要因素。本文首先研究了当前网格应用开发所使用的基于流程或基于 Petri 网的应用描述模型,并提出一种与之不同的基于数据依赖关系的应用描述方法,称为"基于数据依赖的应用描述模型"。在该模型中,应用可以以被称作"数据依赖图"的有向图表示,而且可解应用的数据依赖图一定是一个有向无环图。在这个模型基础上,本文提出了一种更加接近于科学工作者们思维方式的网格应用描述语言 SCO-GADL,并给出其 XML 描述。在 SCO-GADL 中,一个应用可以由一个根元素为 application 的 XML 文档所描述,application 元素有一系列的 data 子元素,用来描述应用数据集中的所有数据项。最后,本文设计和实现了一种采用核心-插件结构的应用引擎,用来支持 SCO-GADL 描述的应用的运行。在应用引擎上运行示例程序的实验表明,SCO-GADL 在保证描述科学计算应用的能力的同时,提供了更符合科学工作者思维方式的网格应用描述方法,大大降低了网格应用开发的难度,使得网格成为科学工作者进行科学研究的一种有力工具。

参 考 文 献

[1] Berman F, Chien A, Cooper K, et al. The GrADS Project: Software Support for High-Level Grid Application Development [J]. International Journal of High Performance Computing Applications, 2001, 15(4): 327-344
[2] Frey J, Tannenbaum T, Livny M, et al. Condor-G: A Computation Management Agent for Multi-Institutional Grids [C]// 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10 '01). 2001: 55-66
[3] Karonis N T, Toonen B, Foster I. MPICH-G2: A Grid-enabled implementation of the Message Passing Interface [J]. Journal of Parallel and Distributed Computing, 2003, 63(5): 551-563
[4] Fox G C, Gannon D. Workflow in Grid Systems [J]. Concurrency and Computation: Practice and Experience, 2006, 18(10): 1009-1019
[5] Majithia S, Shields M, Taylor I, et al. Triana: A Graphical Web Service Composition and Execution Toolkit [C]// IEEE International Conference on Web Services (ICWS'04). 2004: 514
[6] Altintas I, Berkley C, Jaeger E, et al. Kepler: An Extensible System for Design and Execution of Scientific Workflows [C]// 16th International Conference on Scientific and Statistical Database Management (SSDBM'04). 2004: 423
[7] Oinn T, Addis M, Ferris J. Taverna: a tool for the composition and enactment of bioinformatics workflows [J]. Bioinformatics, 2004, 20(17): 3045-3054
[8] OASIS Standard, Web Services Business Process Execution Language Version 2.0 [OL]. (11 April 2007). http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html, 2010-7-26
[9] Guan Z, Hernandez F, et al. Grid-Flow: a Grid-enabled scientific workflow system with a Petri-net-based interface [J]. Concurrency and Computation: Practice and Experience, 2005, 18(10): 1115-1140
[10] Huang Z C, Li G Q, Du B, et al. SIGRE-An Autonomic Spatial Information Grid Runtime Environment for Geo-computation [C]// The 7th International Symposium on Advanced Parallel Proceeding (APPT 2007). 2007: 319-326