一种统一的网格任务层动态映射策略

刘 颖, 夏靖波
(空军工程大学 电讯工程学院, 陕西 西安 710077)

摘 要:提出了一种统一的网格任务层动态映射策略,集成了基于MCT的立即模式和改进Sufferage的批模式,两种调度模式可以自适应转换。并通过引入效用函数的概念保证了用户的QoS需求。仿真实验表明该策略和基准算法相比,有较好的性能,能够更加适应动态变化的网格任务流。

关键词:网格;动态映射;立即模式;批模式;效用函数
中图分类号: TP393 **文献标识码**:A **文章编号**:1009-3516(2007)05-0056-04

网格[1-2]的核心思想是为终端用户提供网络资源共享与协同一体化的运行服务平台,使用户可以最大限度地共享资源,协同资源完成目标。达到这一目标的关键问题首先是资源匹配(Matching),即找到最适合任务运行的资源;其次是次序调度(Scheduling)。这2个步骤统一起来称为映射问题。网格中的映射问题可以分为2类:任务层的映射和子任务层的映射。在任务层的映射中,独立任务在工作组网络之间被调度,以期获得最优的系统性能。网格环境下的大部分映射问题是NP完全问题。因此,众多研究致力于寻找映射问题的最优解[3]。我们主要研究任务映射到一组资源上的动态映射问题。动态映射可分为立即模式(Immediate mode)和批模式(Batch mode)[4]。批模式调度能够收集更多的系统及资源信息,并对更多的任务进行资源选择,但是和立即模式相比实时性比较差。而立即模式又不适用于大量独立任务的调度。本文提出了一个统一的映射策略和效用函数的概念,集成了立即模式和批模式。基于MCT的立即模式和改进的Sufferage算法的批模式可以自适应转换。

1 映射策略

1.1 映射模型

网格任务层的动态映射问题是网格资源管理的重要组成部分。一般情况下,网格资源管理结构分为两层:全球网格资源管理系统(GGRMS)和本地资源管理系统(LRMS)[5]。

在映射模型中,网格的本地资源域被LRMS划分为3个部分,批模式和立即模式可以根据网格任务流进行交替转换。模型中有4个主要部件:映射器,分配器,管理器和缓冲队列。所有的网格任务根据执行期限进入缓冲队列。调度器的核心是映射器,它负责检查缓冲队列,选择调度模式。批模式和立即模式都被设定在此处。分配器通过将任务发送到每个节点来完成调度。任务执行的特征由管理器监控,同时还监控节点的工作负载和特征。任务映射器策略是:

1)如果一个调度周期开始执行时间已到,则继续,否则等待;
2)检查缓冲队列并选择映射模式;
3)执行任务调度;
4)将结果存放到队列管理器;

收稿日期:2007-03-26
基金项目:陕西省自然科学基金资助项目(2004F14)
作者简介:刘 颖(1980-),女,江苏淮安人,博士生,主要从事网格资源管理与调度研究;
夏靖波(1963-),男,河北秦皇岛人,教授,博士生导师,主要从事网络管理及网络安全技术研究.

5)计算下一个周期的开始时间,跳转至1)。

1.2　立即模式和批模式的转换策略

本文分别采用MCT和改进的Sufferage算法作为立即模式和批模式算法。为了描述2种模式之间的自适应转换问题,定义了如下的参数:

1)D_i为任务i的执行期限D_i由 用户的效用函数U给出,U将在2.1节给出定义;

2)U_i为任务执行所获得的效用,也是由用户的效用函数U给出;

3)T_i为用来判断网格节点i运行性能的标准任务;

4)C_i为标准任务Ti在一个节点上的标准完成时间;

5)S_i为执行一个标准任务,用户所获得的标准效用。

标准任务$T_0,T_1,\cdots,T_{n-1}$,调度开始时间T_0,可用节点$N_0,N_1,\cdots,N_{m-1}$,标准完成时间$C_0,C_1,\cdots,C_{m-1}$。

任务剩余可用时间:

$$E_n = D_n - T_0 \tag{1}$$

设定

$$E_{all} = \sum_{k=0}^{n-1}(D_k - T_0), C_{all} = \sum_{k=0}^{m-1} C_k \tag{2}$$

转换策略如下:

Compare Ek and Max($C_0,C_1,\cdots,C_{m-1}$);

if Ek > Max ($C_0,C_1,\cdots,C_{m-1}$);

Abandon T_k, …, T_n;

And return a error code to the tasks' submitters(the code contain Max($C_0,C_1,\cdots,C_{m-1}$));

Else

If n − k≤m Immediate mode

Else

If Eall < < Call Immediate mode

Else Batch mode

Computing Si of all tasks Mapping tasks according to the value of Si;

First mapping high;

And others are put into buffer queue;

Until all task finished

Then begin the next Mapping.

2　效用函数及映射算法

2.1　效用函数

传统情况下,一组没有数据依赖的独立任务的映射问题很难用最小化元任务的完成时间来表述。在实际的网格应用中,用户通常有很多细化的QoS需求,包括可靠性、安全性、数据精确性等[6]。因此,需要根据应用的不同的QoS需求来重新表述映射问题。需要设计相应的QoS驱动的映射算法。每个用户都有明确的QoS需求,因此用户可以定义不同的效用函数来表示自己的QoS需求。为了避免过载,设定标准效用函数Si,所有用户的效用函数值为U,U不能超过Si。几种典型的函数由图1给出。

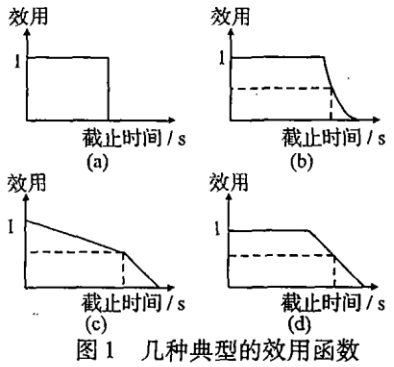

图1　几种典型的效用函数

本方案采用的是图1(d),α,β,γ是网格用户定义的常数,其效用函数为

$$U=\begin{cases}\alpha & t<\gamma D\\ \alpha-\beta(t-\gamma D) & t\geqslant\gamma D\end{cases} \tag{3}$$

2.2 映射算法

2.2.1 立即模式算法

我们使用 MCT 启发式作为立即模式。MCT 启发式将每一个任务映射到能够最早完成的节点。为了决定最早完成时间,任务的规模和节点的能力由相关数据由系统部件、用户或者预测技术提供[7]。这里我们假定所有这些数据是已知的,任务映射器和协同器能够访问它们。但实际上映射过程并非是一个纯粹的立即模式,批模式到立即模式的过渡几乎是一个在线的过程,因为有一些任务到达后并不会被立即调度,但这并不影响映射结果。

2.2.2 批模式算法

许多批模式启发式,例如 Min - min、GA 和 Sufferage 都能得到较好的执行性能。但是 GA 的执行周期较长,Min - min 负载均衡能力较差,相比之 Sufferage 算法的性能略胜一筹。因而,我们采用 Sufferage 作为批模式启发式算法。传统映射算法的目标一般都是得到最短的任务完成时间。然而在网格环境下,用户行为和系统资源状况都相当复杂,简单地追求最短完成时间并不合适,我们应对用户情况加以区分。该启发式在追求较短的任务完成时间的同时,兼顾到任务的服务质量(QoS)需求。算法通过定义任务的效用函数来评估任务的 QoS 需求,即用户在提交任务时根据其需求同时提交一个具体的效用函数,这个效用函数反映出任务经过一定的时间完成后,用户可以得到的效用。图 1 给出了 4 种简单的效用函数,用户提交任务时,可以根据不同 QoS 需求采用任何一种或者定义新的效用函数。

3 仿真研究

在本实验中,为了作对比,选择 Min - Min,Fast Greedy 和 GA 作为基准算法。实验用 PC (Core 2 Processor 1.86GHz, 1GB RAM),采用 GridSim ToolKit4.0[8],所有时间均为仿真时间。模拟的本地网格资源域,由 1 个调度器,3 个具有不同时间请求和任务产生率的用户,10 个具有不同计算能力的空间共享的节点。仿真实体通过一个虚拟的网络连接,每 2 个实体连接具有唯一的带宽。用户将任务抽出调度器。任务流的分布服从 Trace Downloaded[9]。对于整体动态计算能力,很难分析结果,所以我们设定节点的负载是稳定的。定义参数任务的到达率 ,并假设其是可变的和可控的。实验分 2 种情况进行。

第 1 种情况, 在一个较高数值和一个较低的数值间波动,但波动不是很频繁,数值也不是特别的低。如果 λ 太低,则所有算法之间没有明显的差异了。在这种情况下,缓冲队列中充满了任务。3 种基准算法的完成时间都长于本文的算法。5 000 个任务到达以后,本文的映射策略所花费的时间比 Min - min 算法低了 14.59%。因为本文的策略可以转换为 MCT,这就可以尽可能快的将任务调度到节点。在本文动态映射机制下,映射器能够既能够等待即将到达的任务,同时避免节点的空闲,提高了资源利用率。其它结果如图 2(a)所示。第 2 种情况是 λ 数值较高时,可以观察到缓冲队列中充满了任务。本文的映射策略总是选择批模式状态。这就意味着只是 Sufferage 和 3 种基准算法之间的相比较。当 5 000 任务完成时,本文算法所花的时间比 Min - min 算法低 20.31%。与基准算法的对比结果如图 2(b)所示。

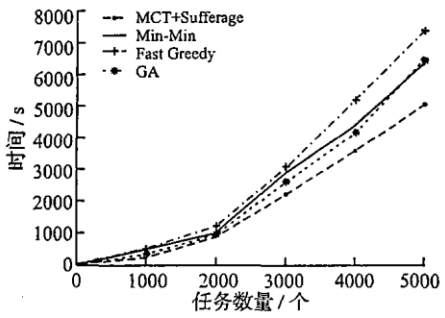

图 2(a) λ 变化时的 Makespan

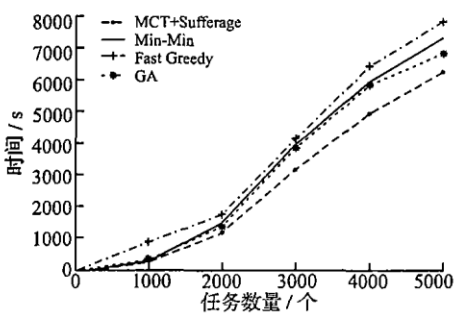

图 2(b) 高 λ 时 Makespan

4　结束语

网格任务层的动态映射问题和网格研究的重要组成部分。本文提出了一个统一的动态映射策略,同已有的研究不同,本方案将基于 MCT 和改进 Sufferage 算法的立即模式和批模式同时集成在映射策略中,并能够根据任务到达率和用户定义的效用函数自适应转换。利用仿真研究验证了本策略更加适应网格计算环境中变化的任务流,同时保证了用户的 QoS 需求。

参考文献:

[1] Foster I, Kesselman C. The Grid: Blueprint for a New Computing Infrastructure (2nd Edition) [M]. San Francisco: Morgan Kaufmann Publishers, 2004.

[2] 刘　颖,余侃民,汪胜荣. 支持资源协同分配的网格任务调度算法研究[J]. 空军工程大学学报:自然科学版,2006,7(1):80-83.

[3] Siegel H J, Ali S. Techniques for Mapping Tasks to Machines in Heterogeneous Computing Systems [J]. Special Issue on Heterogeneous Distributed and Parallel Architectures: Hardware, Software and Design Tools, 2000, 46 (8): 627-639.

[4] Mashewaran M, Ali S, Siegel H J, Hensgen D, et al. Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing System [J]. Journal of Parallel and Distributed Computing, 1999, 59(2): 107-131.

[5] 刘　颖. 一种新型的网格资源协同分配机制研究[J]. 空军工程大学学报:自然科学版,2006,7(5):81-84.

[6] Giuseppe Bianchi, Nicola Blefari-Melazzi, Pauline M Chan L, et al. Design and Validation of QoS Aware Mobile Internet Access Procedures for Heterogeneous Networks [J]. Mobile Networks and Applications, 2003, 8 (1): 11-25.

[7] Dinda P A. Online Prediction of the Running Time of Tashs[J]. Cluster Computing, 2002, 29(1): 225-236.

[8] Buyya R, Murshed M. Gridsim: A Tool kit for Modeling and Simulation of Distribted Resource Management and Scheduling for Grid Computing[J]. Concurrency and Computation: Practice and Experience, 2002, 14(13): 1175-1220.

[9] HPC Workload Resource Trace Respository [DB/OL]. [2006-12-15] http://www.supercluster.org/research/traces.

(编辑:田新华,徐楠楠)

A Unified Dynamic Mapping Strategy for Grid Tasks

LIU Ying, XIA Jing-bo

(The Telecommunication Engineering Institute, Air Force Engineering University, Xi'an 710077, China)

Abstract: A unified strategy for dynamic mapping in grid computing environments is brought forward. Immediate mode and batch mode are symbiotic and can switch adaptively in this strategy, and MCT for the immediate mode mapping and an improved Sufferage algorithm for the batch mode scheduling are utilized. And the utility functions meet the requirements of diverse QoS of tasks. The experimental results indicate that the improved scheme is superior to the benchmark algorithms in performance and adaptable to the varying task flow in grid computing environments.

Key words: grid; dynamic mapping; immediate mode; batch mode; utility functions