

基于层次化的网格资源三层调度模型

杨 炼, 杨长兴

(中南大学 信息科学与工程学院, 湖南 长沙 410083)

摘 要:在分析现有的资源调度方案及模型的基础上,提出了基于层次化的网格资源三层调度模型,它由主调度器、次级调度器和计算节点组成。主调度器根据任务的性质和需求,并参考下层次级调度器的执行情况,将部分任务分发到各次级调度器上,实现了主调度器与次级调度器之间的并行工作。基于该模型提出轮循任务分发策略。通过分析和模拟,该资源调度模型及任务分发策略在调度性能上明显优于集中式调度方案。

关键词:网格计算;三层调度模型;任务分发

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2008)09-0043-03

Hierarchy - Based Grid Resource Scheduling Model with Three - Level

YANG Lian, YANG Chang-xing

(School of Information Science and Engineering, Central South University, Changsha 410083, China)

Abstract: Analyses existing resource scheduling strategy. A resource scheduling model based on hierarchy with three-level for grid computing is proposed, which consists of main scheduler, low-level scheduler and computing node. The main scheduler dispatches a portion of tasks to the low-level schedulers according to the requests of tasks and the execution performance of low-level schedulers, which actualizes the parallelism of task scheduling. A rotational task distributing strategy based on the model is proposed. By doing some analysis and simulation, the proposed resource scheduling model and the task distributing strategy is better than centralizing strategy on scheduling performance.

Key words: grid computing; three-level scheduling model; task dispatch

0 引 言

计算网格^[1,2]被定义为一个通用的、大规模的计算处理虚拟系统,它由若干个分布式资源所组成。它能够由分布式资源所提供的不同服务,各种资源独立于其所在的地理位置被相关应用透明地使用,就像使用一台独立的超级计算机一样,这些资源可以是CPU、存储系统或互联网络^[3]。用户通过任务管理系统向网格提交任务,为任务指定所需资源,删除任务并监测任务的运行状态^[4]。由于网格资源具有广域分布、异构、动态等特性,所以,好的资源调度策略显得尤其重要。在网格计算^[2]环境下,有效的资源调度对优化资源的使用也起着非常重要的作用。

1 现有的网格资源调度策略

目前,网格计算环境下资源调度的方案主要有三:集中式、层次式与分布式调度方案^[5]。在集中式调度方案中,只有一个网格调度器,网格调度器知道所有站点的信息,并且负责调度网格中的所有资源,所有任务都提交给网格调度器。该调度方案扩展性较差,当网络比较大时,调度系统很难掌握所有的资源,调度系统便会成为瓶颈。

在分布式调度方案中,网格调度系统有多个调度器,各调度器之间是平等的,可扩展性较好,可靠性比较高。但各调度器之间的通信量很大,也不能掌握网格中所有资源,因而很难找到全局最优的资源分配方式。

在层次式调度方案中,设置一个任务主分发器,它负责将任务提交给下级调度器进行调度。同时在任务主分发器下再设置一个任务分发器受其控制,该任务分发器负责从主分发器接受任务,再将任务提交给它管辖的下级调度器。

收稿日期:2007-12-29

基金项目:湖南省自然科学基金资助项目(06JJ5131)

作者简介:杨 炼(1980-),男,湖南邵阳人,硕士研究生,讲师,研究方向为网格计算、分布式计算技术;杨长兴,教授,研究方向为网格技术、医学信息表达与处理。

层次式调度方案可扩展性较好,在任务分发器下还可再设置第三级任务分发器。由于在任务主分发器下又设置了任务分发器,因此该方案适合于包含大量任务的用户应用,使系统能响应更多的请求,一定程度上缓解了一些任务长时间等待而得不到系统响应的矛盾。虽然该方案纵向扩展性较好,但由于涉及到的层次越多,各层次间通信量越大,而且调度系统很难监控底层的调度情况,也不利于出现各种差错的及时反馈。

2 三层调度模型

综合上述三种方案的优点与不足,文中提出了基于层次式的三层调度模型,该模型如图 1 所示。其中,第一层由一个主调度器组成,第二层由若干个次级调度器组成,第三层由底层的计算结点组成。次级调度器分别分布在与主调度器不同的机器上。在该模型中,为了避免排在任务队列后面的任务长时间等待,可以将部分任务分发给次级调度器进行调度执行,以减轻主调度器的负载。

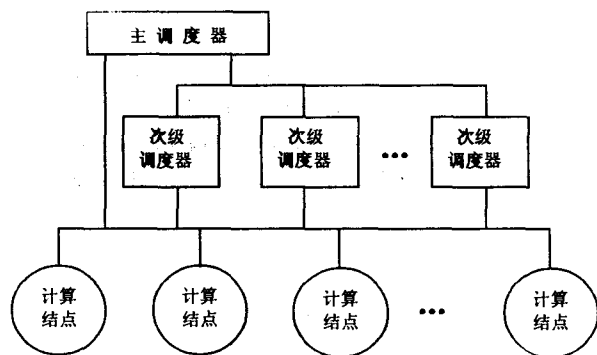


图 1 三层调度模型

当一部分任务在各次级调度器上调度时即实现了主调度器与次级调度器的并行工作,有效地平衡各个调度器的任务调度负载,避免某些任务的长时间等待^[6]。从整个执行过程可以看出,下层多个次级调度器的并行执行,减少了任务的等待时间,加快了任务的调度速度,可以有效地缩短任务的响应时间。由于任务从主调度器分发到次级调度器的时间比任务直接从主调度器上进行调度的时间要小得多,因此该模型能保证任务的平均响应时间比集中式调度要小。

下面具体说明主调度器与次级调度器的主要功能。主调度器的主要功能是:a. 接受用户应用请求,分析各个任务的资源请求类型及任务之间的耦合关系,把所有的任务划分为一到多组任务;b. 监控次级调度器当前的状况,执行任务分发策略,将部分任务(组)分发给适当的次级调度器;c. 监控执行情况,接收执行结果,对于未能完成的任务依据策略考虑是否

重新分发和调度;d. 向用户提交执行结果。次级调度器的主要功能有:a. 接收主调度器分发的任务,分析任务请求,若不符合要求则退回主调度器;b. 请求系统信息服务,匹配资源请求,定位资源并进行调度;c. 监控任务执行状况,接收执行结果;d. 向主调度器提交执行结果。

3 轮循任务分发策略

采用轮循任务分发策略对该调度模型进行定量分析,轮循任务分发策略就是先从主调度器的任务队列中取出第一个任务由主调度器直接提交给合适的计算结点进行调度,调度完之后再取第二个任务由主调度器分发给第一个次级调度器,第一个次级调度器接收到任务后,便选择适当的计算结点调度该任务。主调度器分发完第二个任务后便分发第三个任务给第二个次级调度器,依此类推直到主调度器将第 $k+1$ 个任务提交给第 k 个次级调度器,第一轮完毕后第二轮接着开始。

从上分析可明显看到,该分发策略的优点有:a. 简单,易于实现;b. 较大程度上平衡了主调度器与次级调度器之间的负载;c. 由于下层的次级调度器与主调度器的并行执行,也即各次级调度器只需等待少量的时间便可接受下一个任务进行调度,所以次级调度器上无需设置任务队列,这样既可以简化调度系统的设计,又避免任务被置入另一个任务队列再次等待调度,从而加快了任务的调度速度,这同时也体现了该模型的优越性。

4 性能分析与比较

对于提交的一组任务,由 m 个相互独立的任务组成, T_{ir} 为第 i 个任务的响应时间, T_{iw} 为第 i 个任务在被提交执行前的等待时间, T_{is} 为第 i 个任务在计算节点上的执行时间,那么 $T_{ir} = T_{iw} + T_{is}$ 。显然影响任务的响应时间的最大因素为任务执行前的等待时间,下面对该模型中的任务等待时间进行详细分析。

根据模型,当第 i 个任务直接在主调度器上调度时, $T_{iw} = T_{is}$, 否则在次级调度器上调度时, $T_{iw} = T_{id} + T_{is}$, 其中 T_{is} 为任务在主调度器或次级调度器上的调度时间,不妨假设各任务在各调度器上的调度时间相同,均为 T_s , T_{id} 为该任务在主调度器上的分发时间,为简化式子表达,设 T_{iw} 为 T_i , 则在第一轮调度中:

$$T_1 = T_s$$

$$T_2 = T_s + T_d + T_s$$

...

$$T_{k+1} = T_s + kT_d + T_s = 2T_s + kT_d, k \geq 1$$

$$\text{第一轮总的等待时间为} \sum_{i=1}^{k+1} T_i = (2k+1)T_s + \frac{k(k+1)}{2}T_d$$

m 个任务中共要调度 $\left\lceil \frac{m}{k+1} \right\rceil$ 轮, 设 $\left\lceil \frac{m}{k+1} \right\rceil = n$, 则

$$\begin{aligned} \sum_{i=1}^m T_i &\approx \frac{n(n+1)}{2} \sum_{i=1}^{k+1} T_i, \text{即} \\ \sum_{i=1}^m T_i &\approx \frac{n(n+1)}{2} \left[(2k+1)T_s + \frac{k(k+1)}{2}T_d \right] \end{aligned} \quad (1)$$

这就是 m 个任务采用三层调度模型和轮循任务分发策略的总的等待时间。单从这一式子看不出任务调度的效率究竟如何, 为此可与集中式方案作比较, 若采用集中式调度方案, 所有的任务在任务队列中顺序被主调度器直接进行调度, 易得其总的等待时间为:

$$\begin{aligned} \sum_{i=1}^m T_i &= \frac{m(m+1)}{2} T_s \\ \text{由于 } m &\approx n(k+1), \text{将其代入可得} \\ \sum_{i=1}^m T_i &= \frac{n(k+1)[n(k+1)+1]}{2} T_s \end{aligned} \quad (2)$$

(2)、(1) 两式相减, 可得

$$\frac{1}{2}nk(nk-1)T_s - \frac{nk(n+1)(k+1)}{4}T_d \quad (3)$$

由于 $T_d < T_s$, 所以(3)式结果总是为正值, 而且随着 n 与 k 的积的增大而增大。这说明采用三层调度模型的任务总的等待时间明显会比集中式调度的总的等待时间要少, 也即该调度模型在调度性能上明显优于集中式调度方案, 而且, (3) 式的值越大, 越能体现该模型及分发策略在任务调度上的优越性。

进一步分析可发现, 当保持次级调度器个数 k 不变时, 总任务数 m 越大(也即 n 越大), n 与 k 的积越大, (3) 式的值也就越大。而保持总任务数 m 不变, 仅增加次级调度器个数 k 时, n 与 k 的积基本保持不变, (3) 式的值也将变化不大。

5 模拟实验

目前, 已有一些较为完善的网格模拟器可供使用^[7]。根据对模型的分析并结合其特点, 文中使用 GridSim 模拟工具进行模拟实验^[8]。实验中, 取 $T_d = 0.1s$, $T_s = 1.2s$, 计算节点 150 个。为对比任务数对调度性能的影响, 分别在任务数为 1000, 2000 和 4000 的情况下进行模拟, 图 2 所示为模拟结果。

从上面的模拟结果, 可以看出采用三层调度模型和轮循任务分发策略时, 任务平均等待时间比集中式

调度明显要短。当增加次级调度器个数 k 时, 任务平均响应时间逐渐减小, 但减小的幅度逐渐减少。因此在实际应用中, 次级调度器的个数不宜设置得太多, 应在满足调度性能的同时尽量减少设备投入。当提交给计算节点的任务数增加时, 任务平均等待时间自然也随着增加。

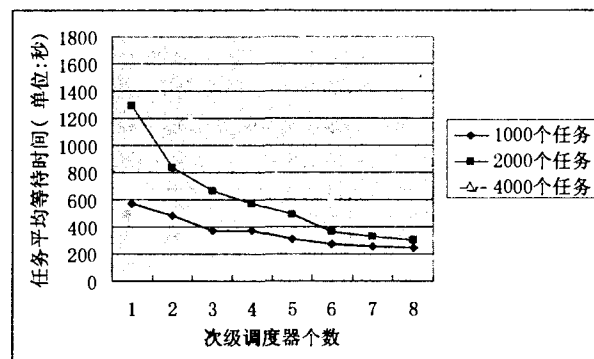


图 2 模拟实验结果

6 结 语

在分析现有的资源调度方案及模型的基础上, 提出了基于层次式的网格资源三层调度模型, 并基于该模型提出轮循任务分发策略。通过对该模型和任务分发策略的性能分析与模拟实验, 证实采用该模型及任务分发策略时任务的平均等待时间随着次级调度器数的增加而逐渐减少, 在调度性能上明显优于集中式调度方案。由于次级调度器与主调度器并行执行时, 各次级调度器需等待少量的时间才能接收下一个任务进行调度, 次级调度器个数越多, 等待的时间越长, 这说明轮循任务分发策略尽管简单, 但并没有将该模型的调度性能发挥到极致。进一步的工作是探讨基于该模型的更高效的任务分发策略。

参考文献:

- [1] Baker M, Buyya R, Laforenza D. The Grid: International Efforts in Global Computing[C] // In: Intl. Conf on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet(SSGRR 2000). I'Aquil, Rome, Italy:[s. n.], 2000.
- [2] Foster I, Kesselman C. The Grid: Blueprint for a New Computing Infrastructure[M]. [s. l.]:Morgan Kaufman Publishers, 1999.
- [3] Nakada H, Sato M, Sekiguchi S. Design and implementations of ninf: towards a global computing infrastructure[J]. Future Generation Computing Systems, 1999, 15(5-6):649-658.
- [4] Subramani V, Kettimuthu R, Srinivasan S, et al. Distributed

(下转第 49 页)

次幂为2、二次项系数不为零对题目的求解没有帮助可以约去。约简后的题目信息进行组织后可生成进入解题系统的问题模型。

3 数学领域自动建模的实现

在数学领域中,依据以上分析自动建模的具体流程如图3所示。

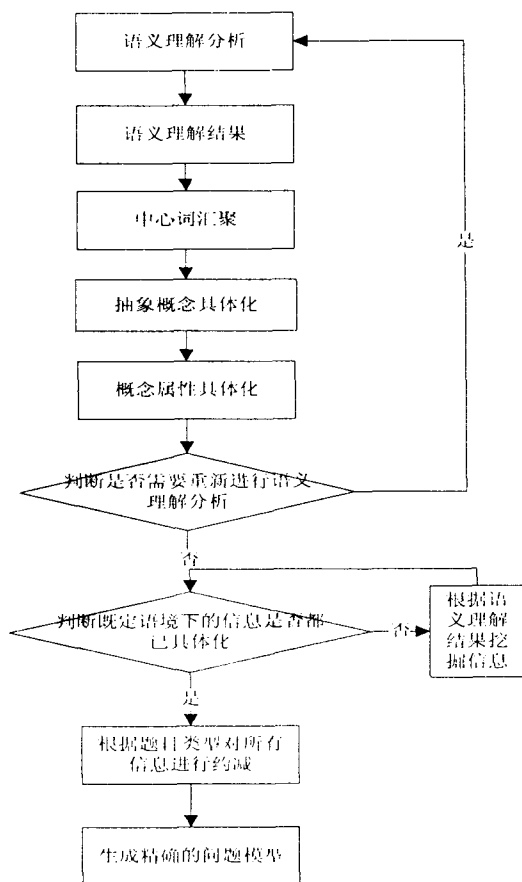


图3 数学领域建模流程图

经过以上处理后自然语言中的信息就可被计算机直接处理。

4 测试结果

本系统在设计时搜集了近300道的一元二次方程的题目,对其进行分类,设计处理流程。一元二次领域中的题目类型涉及面约为95%,其中包括以下几类;

(1)求未知量值(已知一根求另一根,求两根,已知两根关系求与其相关的第三未知量的值);

(2)求代数式的值(已知两根关系与方程系数求未知表达式的值);

(3)求范围求符号(仅由不等式推出,需要通过不等式与方程联合才可求解)。

测试后通过率可达90%以上。已可与自动解题系统相结合得到最终的结果,基本实现了初步的设想。

5 结束语

论述了一种智能辅导系统中模型建立的基本过程,并使用一元二次方程问题为例子,对其具体实现进行了详细的描述。为同类问题的解决和系统的进一步的壮大提供了借鉴的基础。但由于系统所掌握的知识程度有限对于需要深层理解的纯文字的应用题目目前还无法处理。

参考文献:

- [1] Allen J. 自然语言理解[M].第2版.北京:电子工业出版社,2005.
- [2] 俞士汶. 自然语言理解与语法研究[M]. 北京:商务印书馆,1999.
- [3] 周经野. 基于自然语言计算模型的汉语理解系统[J]. 软件学报,1993,4(6):41-46.
- [4] Jurafsky D, Martin J H. 自然语言处理综合[M].北京:电子工业出版社,2005.
- [5] 王继成,武港山,周源远,等. 一种篇章结构指导的中文Web文档自动摘要方法[J]. 计算机研究与发展,2003(3):27-31.

(上接第45页)

- Job Scheduling on Computational Grids Using Multiple Simultaneous Requests[C]//Proceedings of 11th IEEE International Symposium on High Performance Distributed Computing. Washington, DC, USA: IEEE Computer Society, 2002:359-366.
- [5] 罗红,慕德俊,邓智群,等. 网格计算中任务调度研究综述[J]. 计算机应用研究,2005(5):16-19.
- [6] Cao J, Kerbyson D J, Nudd G R. High performance service discovery in large-scale multi-agent and mobile agent systems[J]. International Journal of Software Engineering and

Knowledge Engineering, 2001,11(5):621-641.

- [7] Nudd G R, Kerbyson D J, Papaefstathiou E, et al. PACE a toolset for the performance prediction of parallel and distributed systems[J]. International Journal of High Performance Computing Applications, 2000,14(3):228-251.
- [8] Buyya R, Murshed M. GridSim: a Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing[J]. Concurrency and Computation: Practice and Experience,2002,14(13-15):1175-1220.