基于 GridSim ToolKits 的网格仿真环境设计与实现*)

刘宴兵　杨茜慧　王文斌
（重庆邮电大学计算机学院　重庆 400065）

摘　要　本文在研究 GridSim 的基础上，设计并实现一种基于 GridSim ToolKits 的网格仿真环境 MendSim，该网格仿真环境可以对各种高级调度算法进行模拟并实现对各种网格发布规则和调度算法的研究。

关键词　网格仿真环境，发布规则，禁忌搜索算法，SPRs

Design and Implement for Grid Simulation Environment Based on GridSim ToolKits

LIU Yan-bing　YANG Qian-hui　WANG Wen-bin
(College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China)

Abstract　In this paper, a novel grid simulation environment called MendSim is designed and implemented based on the GridSim ToolKits. Multifarious grid dispatching rules and scheduling algorithms can be realized in the new grid simulation environment.

Keywords　Grid simulation environment, Dispatching rules, Tabu search algorithm, SPRs

1　引言

网格的目标[1]是将地理上分布的、系统上异构的多种计算资源通过高速网络连接起来，协同解决大型应用问题，进行广域信息资源的共享。网格作为一种新出现的重要基础设施，和其它系统相比，存在分布性、自相似性、动态多样性以及管理的多重性等多方面的特点。因此，网格资源的管理和调度已经成为网格研究领域中的关键技术，并且作为网格系统的研究重点之一[2]。目前，网格系统的设计者通常使用网格仿真工具来验证设计方案及测试网格系统性能，使得模拟资源发布规则和调度算法的仿真工具成为研究的热点。GridSim ToolKit[3]是一个开放式的网格开放工具包，网格开发者可以利用它丰富的函数库来创建网格仿真环境。但 GridSim ToolKit 没有明确定义应用程序模型，对于各种类型的任务和应用程序，不同的网格拓扑，以及对调度算法的设计和评估等问题，都是由用户自行解决的。因此，GridSim ToolKits 存在二次开发的问题。本文给出了解决方案并对 GridSim 进行扩展，实现了对各种网格调度算法和任务发布规则可重复、可控制的研究实验。

2　网格仿真开发工具 GridSim ToolKits

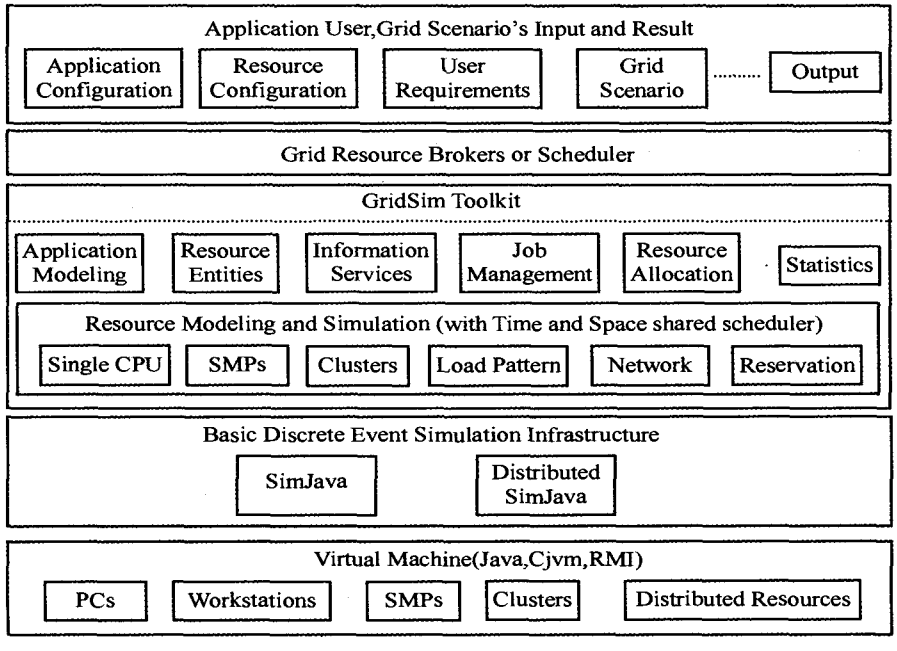

图 1　GridSim 平台的模块化结构

*)重庆市教委项目(KJ050507)；重庆市科委攻关和自然科学基金(CSTC,2005BB2060)。**刘宴兵**　博士生，主要从事计算机网络技术研究；**王文斌**　硕士研究生，主要研究宽带无线网络技术。

GridSim 最初是由澳大利亚墨尔本大学的 Rajkumar Buyya 于 2001 年发布的一个基于 Java 语言开发的网格仿真工具，它可用于对集群、对等计算和网格等分布式计算环境[4]中的调度算法进行建模和仿真。GridSim 是在 SimJava 的基础上开发的，它提供丰富的函数库以支持模拟网格环境中的异构资源、用户、应用程序、用户代理和调度器等实体。网格资源、用户和用户代理被视为不同的实体，它们通过消息事件（输入和输出）来进行通信。仿真结束后，用户可以调用 GridSim 中的称为 GridStatistics 的库函数来收集各种模拟的统计数据。

本文基于 GridSim 体系结构，如图 1 [5]，扩展了 gridsim 开发工具包，设计并实现了一种新的网格仿真环境 MendSim。在此环境中，通过模拟仿真可以研究各种网格调度算法和任务发布规则。

3 网格模拟器 MendSim 的建模与实现

在基于 GridSim 4.0 的基础上，进一步提出 MendSim 网格模拟器概念。它是基于 Gridsim4.0 和 Simjava 的一种网格模拟器。MendSim 采用模块化设计，通过多种不同的网格实体以及实体间相互通信来对网格环境进行模拟。实体之间既相互独立又相互联系，它们通过消息机制进行通信。这样，既可以减少网络流量，同时又可以增加模块间的独立性，从而增加了该模拟环境的可扩展性。

3.1 MendSim 建模与实现

网格模拟器 MendSim 的建模实体部分主要由以下几个模块[5]构成：

(1) Grid Resource entity(网格资源实体)

(2) Job Submission entity(任务提交实体)

(3) Job Scheduler entity(任务调度器实体)

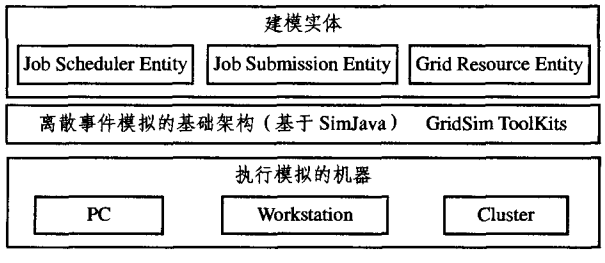

图 2 MendSim 模拟环境体系结构

从图 2 中可以看出，MendSim 可以运行在 PC、工作站和集群之上，通过调用 gridsim 开发工具包中各种类来构建 MendSim 网格仿真系统。从体系结构的建模实体部分可以看出，构建的仿真环境是模块化的，它们由任务调度器实体、任务提交实体和网格资源实体组成。这些实体通过发送消息进行通信，它们既相互独立又相互联系。图 3 是任务提交实体、任务调度器实体以及网格资源实体的通信流程图。下面具体描述其通信过程。

任务提交实体在任务执行之前和存储任务之后，通过它和调度器实体的通信来得到调度策略，调度策略进一步选择一个资源来执行一个任务。所以，MendSim 网格模拟器实现的网格任务调度机制是被动获得的，即网格资源接收网格任务调度器指派给自己的网格任务。

任务生成器附着于任务提交实体上，用它来模拟任务的到达情况。任务生成器可以在模拟过程中生成新的任务，任务到达时间间隔也可选用不同的统计分布规律。本文中的 MendSim 模拟器支持均匀分布(uniform distribution)和正态分布(normal distribution)。由于 MendSim 网格模拟器在实现时是按照模块化设计的，因此它具有很好的可扩展性，并且很容易在任务调度器实体中添加新的统计分布或者添加真实的工作量数据。

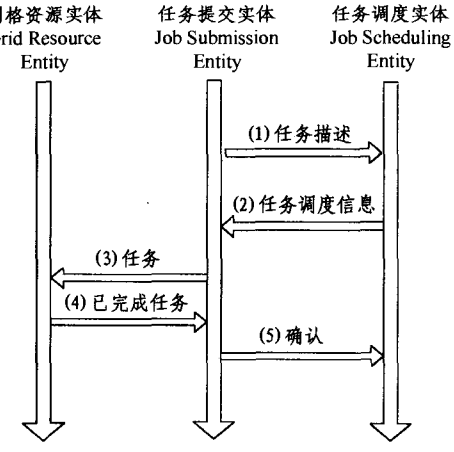

图 3 网格实体的通信流程图

任务调度器实体负责生成调度策略，并进一步对调度策略进行优化。在动态环境中，随着一些旧任务的完成和新的任务到来，任务调度器实体会及时地更新来适应新的网格环境。任务调度器实体是一个独立实体，所以它必须和其它实体进行通信(主要是和任务提交实体通信)。

每一个网格资源实体负责网格任务的执行。资源实体是由任务调度器实体来选择的，适当的资源被任务调度器实体选择，从而在之上完成网格任务的执行。

4 MendSim 性能测试

硬件配置：Intel Pentium(R) D CPU 2.8GHz，512 MB 的内存。

软件配置：OS，Windows xp sp2；软件开发包，JDK1.50，Gridsim.jar，Simjava.jar；集成开发环境，Jcreator 3.0。在 MendSim 网格仿真环境中测试高级调度算法的性能，其指标为，总延迟(total tardiness)，理论计算公式为[6]

$$T=\sum_{i=1}^{n}T_i=\sum_{i=1}^{n}\max(0,C_i-d_i) \quad (1)$$

其中 C_i 和 d_i 分别表示任务 T_i 的实际完成时间和应完成时间。

研究 MendSim 网格模拟器实现的网格任务调度算法，首先需要考虑到网格环境问题[7]。网格环境问题一般是由一组资源(典型的包括机器、硬盘存储、内存和网络)、一组任务、最优化标准、环境规范和其它限制来定义的。网格是一种典型的动态环境，它的问题复杂性很大程度上依赖于机器、任务和环境特征。本文把机器理解为具有一两个 CPUs 的可计算资源。机器环境可能由单一机器、同一频率的并行机器、不同频率的机器组成，机器也可能变为不可获得的情形(比如宕机)。同时，任务参数也显得很重要。不同的任务有不同的计算时间(处理时间)、开始时间、到期时间和优先权，并且不同任务需要的 CPU 数量也可能不相同。有的任务在执行期间可能会由一个资源迁移到另外一个资源，而也有任务仅仅需要特定机器，不发生迁移(机器和任务的相配)。调度的目的就是满足用户的和系统管理者的需求，比如使得总延迟最小化。

本文实现了不同的任务调度算法来最小化总延迟。实现的调度算法是 SPR[8](Simple Priority Rules),它包括 ERD(Earliest Release Date)和 EDD(Earliest Due Date)等发布规则。SPRs 将任务分配到特定的资源队列中,然后使用 Tabu search[9]算法进行优化,使得总延迟最小。

假设网格环境中的系统无差错,比如没有资源宕机或者任务丢失;系统是由并行多处理器的机器组成;不同的机器可能有不同的 CPU 频率。而且在任务还未执行时,假定我们已知道任务的计算时间长度,并且每个任务运行仅需要一个 CPU。在满足以上情况下,本文分别模拟了在静态情况和动态情况下任务调度情况。在静态情况下,所有的任务都能够预先知道。而在动态情况下,随时有新任务的加入和旧任务的退出,因而产生的调度算法会随着时间的变化而变化。在这种情况下,任务调度器实体必须不断生成调度算法,此任务调度器实体不支持任务抢占或任务迁移。

本测试在网格中两到八个机器上进行,这些机器具有不同的 CUP 主频,每个机器有两个 CPU。此仿真不能在单个机器上进行,因为 Tabu 搜索算法需要在不同的调度器中移动任务。该仿真运行了 2000 个任务,这些任务均具有固定开始时间和到期时间。开始时间在某种意义上被认为是硬限制,即任务不能早于开始时间被执行。而到期时间是个软限制,即可以不严格遵守它的限制。同步测试数据集得到了 20 个不同数据集的平均值。调度器实体优化了所有任务的总延迟。下面的比较是在 MTEDD 发布规则(dispatching rule)、MTERD 发布规则和它们各自的 Tabu 搜索优化之间进行的。

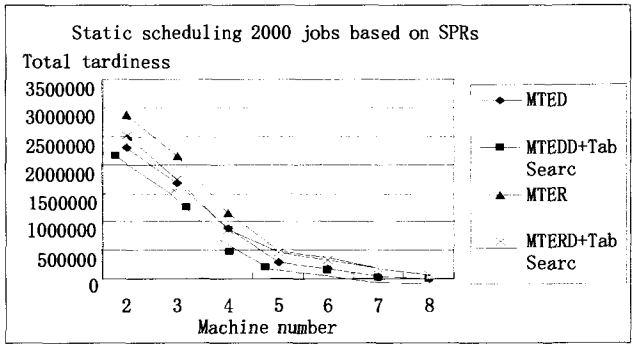

图 4 基于 SPRs 调度 2000 任务的总延迟性能图

图 4 表明使用 MTEDD,MTERD 和它们的 Tabu 搜索算法时所需要的总延迟情况。经过 Tabu search 算法优化的任务调度在总延迟方面明显优于单独使用 MTEDD 和 MTERD 的任务调度。

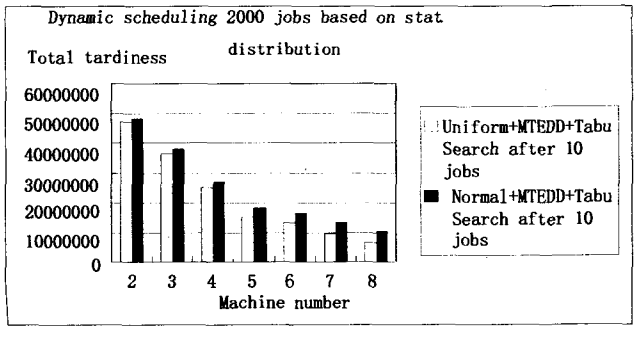

图 5 基于统计分布动态调度 2000 任务的总延迟

从图 5 中不难看出,通过 MTEDD 和 Tabu search 合成的 SPRs 进行资源管理和调度时,在机器数量不同的情况下,当任务加入时间间隔遵守均匀分布时产生的总时延比遵循正态分布时产生的总时延要小。

总体可以清楚看出总延迟的大小在很大程度上依赖于可获得的机器数量。随着机器数目的增加,总延迟明显呈现下降趋势。另外从上面两个图中可以看出,在测试环境中 Tabu search 算法使移动性得到了改进。在改善目标函数值方面,即公式(1),单独使用 MTEDD 和 MTERD 调度算法并不能取得很好的效果,而使用 Tabu 搜索算法却能够取得很大的改进。在所有的模拟测试中,任务的数量相同,当具有较多的机器时,执行的时间会减小。这是因为先前调度的任务已经完成了,并且在每次运行中调度算法只是对数目较少的任务进行调度。在这样情况下,期望完成但还未完成的任务会更快地被调度。因此,可以看出本文设计的 MendSim 网格模拟器可以有效实现对各种高级调度算法的模拟。

结束语 本文在 GridSim 的基础上提出一个新的网格模拟环境 MendSim,并用它实现了网格中各种调度算法的模拟。下一步工作集中处理 MendSim 网格模拟器模拟环境的扩展,例如增加网络拓扑,支持不同的任务类型,包括并行任务、工作流、抢占和具有优先级的任务等等,同时也能够对工作迁移进行建模。进一步扩展模拟不同的网格故障(资源或者任务被破坏、网络断链)来测试调度算法的健壮性。另一方面,可以进一步使用这个模拟环境实现和评估不同的调度算法和技术,例如强制编程和收敛调度等,可以进一步实现和评估不同的网格资源发布规则。还可以为 MendSim 设计一个可视化操作界面,在其中可以自由设置用户数量、任务数量、资源数量及其具体属性、各种自定义的任务调度算法,这样可以降低 MendSim 网格模拟器用户的使用难度。

参 考 文 献

[1] Joshy J, Craig F. Grid Computing. 战晓苏,张少华,译. 北京:清华大学出版社, 2005:1-17

[2] Luo Junzhou, Ji peng, Wang Xiaozhi, et al. Resource management and task scheduling in grid computing // Berlin, Proceedings of the 8th International Conference on Computer Supported Cooperative Work in Design [C]. New York: Springer, 2005: 431 - 436

[3] 刘祥瑞,朱建勇,樊孝忠. 基于 GridSim 的网格调度模拟[J]. 计算机工程, 2006,35(2): 42-44

[4] Granvill L Z, Da rose D M, Panisson A, et al. Managing computer networks using peer-to-peer technologies[J]. IEEE Communications Magazine, 2005,43: 62-68

[5] 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): 1175-1220

[6] Ho N B, Tay J C. Evolving dispatching rules for solving the flexible job-shop problem // Proceeding of the IEEE Congress on Evolutionary Computation [C]. Piscataway, NJ, USA: IEEE Press, 1998: 69-73

[7] Fosteri I. The Grid: A New Infrastructure for 21st Century Science [J]. Physics Today, 2002,55(2): 42-47

[8] Schmidt G. Performance guarantee of two simple priority rules for production scheduling [J]. International Journal of Production Economics, 2000,68(2): 151-159

[9] Huang Wenqi, Zhang Defu. An algorithm based on Tabu Search for satisfiability problem. Journal of Computer Science and Technology, 2002,17(3): 340-346