第 34 卷 第 3 期 计 算 机 工 程 2008 年 2 月
Vol.34 No.3 Computer Engineering February 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)03—0274—03 文献标识码：A 中图分类号：TP301

基于 OPNET 的局部网格任务调度平台设计

薛桂香[1]，赵 政[1]，史 伟[1]，孟 和[1]，宋建材[2]
(1. 天津大学计算机科学与技术学院，天津 300072；2. 天津航海仪器研究所，天津 300131)

摘 要：在充分考虑网格动态性和异构性的前提下，采用模块化设计方法，在 OPNET 环境下构建了一个局部网格任务调度仿真平台。在该平台上，比较了 SF, LF, FCFS, EDF 等网格任务调度算法。仿真实验结果表明调度算法运行良好，该网格仿真平台提供了一个通用的、模块化、可扩展的网格任务调度模拟环境，能够较好地满足网格任务调度要求。

关键词：网格计算；任务调度；建模；仿真

Design of Local Grid Task Scheduling Platform in OPNET

XUE Gui-xiang[1], ZHAO Zheng[1], SHI Wei[1], MENG He[1], SONG Jian-cai[2]
(1. Department of Computer Science, Tianjin University, Tianjin 300072; 2. Tianjin Navigation Instrument Research Institution, Tianjin 300131)

【Abstract】Adequately considering the dynamic attribute and heterogeneous attribute of grid, the simulator platform of local grid task scheduling is constructed in OPNET with modeling design method. On the basis of this simulation platform, the performance of some grid task scheduling algorithms such as SF, LF, FCFS, EDF is tested. The simulation result shows that it provides a universal, modularizing and extensible simulation platform of grid task scheduling.

【Key words】 grid computing; task scheduling; modeling; simulation

网格系统的设计是一个非常复杂的系统工程，需要考虑如资源的异构性、动态性以及性能优化等诸多问题。网格模拟器为帮助网格系统的设计者验证设计方案、测试设计性能提供了极大的便利。目前有代表性的网格模拟器主要有：Bricks[1], MicroGrid[2], SimGrid[3]等。MicroGrid需要用Globus构造实际的网格任务，而且受主机性能的影响，适用于实际系统开发过程中对调度算法的仿真评测。SimGrid模拟过程比较繁琐复杂。GridSim采用面向对象技术构建，整个框架比较清晰，开发者容易理解，但运行过程中不能体现资源动态变化。本文基于OPNET Modeler搭建了一个网格任务调度仿真平台，实现了长任务优先、短任务优先、先进先出、早截止时间优先等调度算法并进行了性能比较与分析。

1 网格任务调度机制

一个调度系统由程序任务、分布环境和调度程序 3 部分组成，在这个系统中，调度的性能得到优化。调度模型定义[4]如下：网格环境中的应用程序可以由多个任务构成的集合来描述，一个网格应用程序可以表示为 1 个DAG $G=(T, E)$，其中，T是节点集；E是有向边集。1 个在有m个处理单元和任务图$G=(T, E)$之上的调度是 1 个函数f，f将每个任务以某个特定的开始时间映射到某个处理单元。从形式上，1 个调度可描述为f：$T\rightarrow\{1, 2, \cdots, m\}\times[0, \infty]$，如果存在$v\in T, f(v)=(i, t)$，则表示任务$v$被调度到处理单元$P_i$上，且从时间$t$开始执行。函数$f$可以表示成Gantt图，直观地表示所有任务的开始时间和完成时间。在Gantt图中，有一个分布系统的处理单元表，表中每个处理单元都有一个任务分配表，即将分配到这个处理单元的所有任务按执行时间排列成表，包括开始时间和结束时间，分别用$ST(t_i, P)$和$FT(t_i, P)$表示。1 个任务调度系统的最大调度长度SL(Schedule Length)定义为所有处理机P_i中的

$$SL=\max_t\left\{\sum_j FT(t_j, P_i)\right\} \quad (1)$$

调度的目标是将任务适当分配到处理机并协调任务之间的执行顺序，使并行执行时满足并行任务之间的优先约束关系，而且 SL 最小。这里将 SL 命名为 MakeSpan。

2 网格任务调度建模与仿真

2.1 仿真平台整体设计

本文在局部网格结构仿真模型的设计和实现过程中，充分考虑实际情况，采用Agent技术[5]和模块化设计。主体部分由 2 个模块构成：Agent模块和Host模块。Agent模块扮演调度员的角色，用户主要和Agent交互，隐藏了复杂的网格计算结构。Agent的功能是发现用户可以访问的资源，把任务映射到资源上，准备处理的程序和数据，开始执行任务，最后收集结果，评估调度算法的性能。Agent同时负责监测和跟踪任务完成的进度，自适应于网格运行时环境的变化或资源缺失。Host模块模拟具有不同处理能力的计算节点，接收Agent模块分配给的任务后执行，同时定期向Agent模块汇报自己的运行状态。本文设计了具有 1 个Agent，12 个Host的局部网格模型。由于网格计算环境的动态性，每个Host的执行速度和运行状态都在不停地随机变化，它们只与Agent进行通信，接收并完成计算任务。Agent与Host之间的通信采用数据包，在OPNET

基金项目：国家部委基金资助项目；天津科委基金资助项目(90604013)

作者简介：薛桂香(1979－)，女，博士研究生，主研方向：网格任务调度，人工智能；赵 政，教授；史 伟，硕士研究生；孟 和，博士研究生；宋建材，硕士

收稿日期：2007-03-12 **E-mail：**xueguixiang@gmail.com

中用Packages描述。

2.2 Agent 模块设计

用户提交到 Agent 的任务具有动态性，根据统计规律，任务长度呈指数分布，任务到达时间按泊松分布，即任务到达时间间隔服从指数分布。Agent 的功能如图 1 所示，它根据任务长度和 Deadline 属性按照一定的调度算法将任务分配给网格中合适的 Host。每隔一段不同的时间，Host 执行速度会变化，随时可以加入或离开网格系统，因此，Host 需定时向 Agent 发送信息包汇报其运行状态和执行速度，Agent 据此实时更新计算资源数据库。Agent 从结构上主要包含任务产生(source)和控制(process)两个子模块和与 Host 收发包的接口。

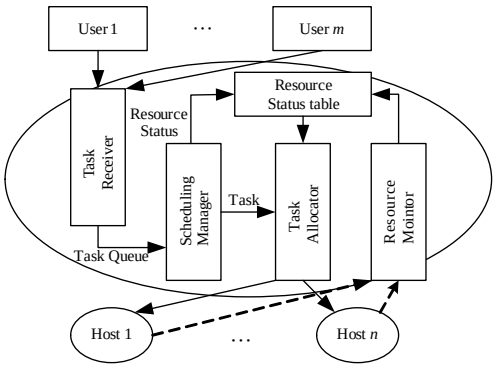

图 1 Agent 功能

Source 子模块生成任务包，给出了任务长度和截止时间，任务包生成以后，将包发给 process 子模块，其状态转移如图 2 所示。

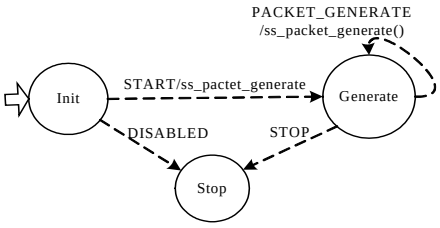

图 2 source 状态转移

Process 子模块是整个 Agent 的核心，负责完成任务调度。由于网格计算相关技术甚至模型本身仍处于快速发展之中，因而对网格任务调度算法运行效率的评价，国内外目前尚无成熟的模型与基准，难以进行横向对比。

本文主要分析与评价了网格任务调度算法的 2 个参数：任务执行平均延迟(average delay)和超时概率(tardy rate)。Average delay 是从任务产生到任务成功返回的平均时间延迟，包括任务的排队时间、传输时间和执行时间。Tardy rate 是任务超时(未能在截止时间内完成)的个数与任务成功完成总数的比值，即在完成的任务中超时任务所占比例，分别定义如下:

Tardy Rate：

$$\gamma = \frac{\sum_{j=1}^{N} \Delta(d_j - \eta_j)}{N}$$

其中，$\Delta(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$ (2)

Average Delay：

$$\overline{T} = \frac{\sum_{j=1}^{N} (\eta_j - g_j)}{N} \quad (3)$$

其中，d_j表示第j个任务的deadline；η_j 表示第j个任务的完成时间；g_j表示第j个任务的产生时间；N表示任务的总数。

Process 子模块状态转移如图 3 所示，初始化状态定义了一个任务队列和一个计算资源队列，包含 Host 的属性值：地址，可用性，状态和计算速度。初始化之后进入调度状态，当任务队列空的时候，系统空闲；在队列不为空的时候，尽量将任务调度到优先级较高的计算资源上去运行，而优先级主要由计算资源的能力和稳定性来决定，从而进一步提高了任务能够成功执行的概率。在调度阶段，每当有包到达触发中断，模块随时响应中断，包中断共 4 种：任务到达中断，任务完成结果包到达中断，Host 状态包到达中断，Host 速度包到达中断。

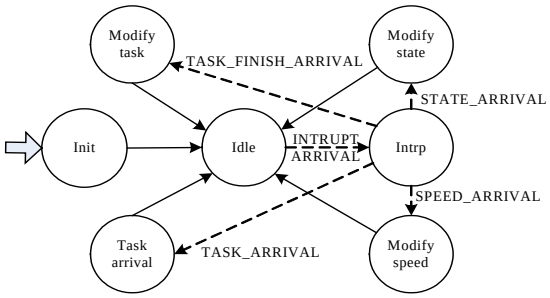

图 3 process 状态转移

Process 子模块中可观测统计量有任务执行平均延迟和超时概率，还有系统自带统计量任务队列长度及 Agent 与每个计算节点接收和发送的吞吐量。另外在 Agent 结构中，链路设置为双工传输模式，传输速率为 1 Mb/s，误码率为 0，对应 Host 也作相同设置。

2.3 Host 模型设计

Host 是具有计算能力的处理节点，其状态转移图如图 4 所示。初始化(init)阶段进入空闲(idle)状态，没有任务时系统处在空闲状态，每当有包到达则触发中断，模块随时响应中断。该模块可观测统计量有点到点延迟(ETE delay)和 Host 接收与发送的吞吐量。ETE delay 是任务在 Agent 队列内排队时间与其从 Agent 到 Host 的传输时间之和，不包括任务执行时间。包中断分为 4 种形式：任务包到达中断，速度改变中断，速度报告中断，状态改变中断。

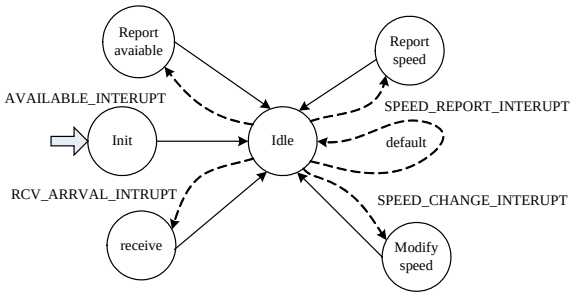

图 4 Host 的处理模块状态转移

3 仿真结果及分析

本文在上述所建仿真平台上设计实现了先来先服务算法(FCFS)、短任务优先算法(Short_First, SF)、长任务优先算法(Long_First, LF)、早截止时间任务优先算法(Early_Deadline_First, EDF)等网格任务调度算法。描述任务调度性能的指标很多，选取哪个主要取决于系统要求。本文主要采用 Average Delay 和 Tardy Rate 两个指标来刻画网格任务调度算法的性能。

由图 5、图 6 及表 1 可以看出，短任务优先算法的平均延迟时间和平均超时概率指标都最小，性能最优，而长任务优先算法的平均延迟时间和平均超时概率指标都最大，性能最差，早截止时间任务优先算法和先来先服务算法的性能接近。该实验结果满足任务调度算法理论的性能分析结果[4]，证明了该仿真平台合理可行。

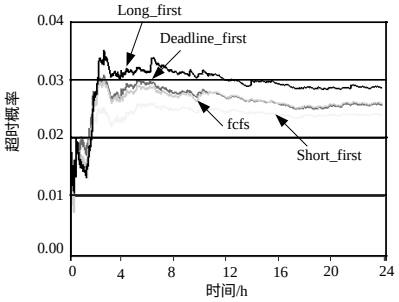

图 5　平均超时概率

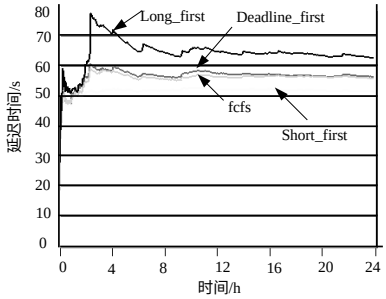

图 6　平均延迟时间

表 1　各算法统计量均值比较

Algorithm	AVERAGE Delay/s	ETE Delay/s	TARDY Rate
SF	50.66	1.17	0.023 380
LF	57.13	8.58	0.028 734
FCFS	54.50	5.34	0.025 974
EDF	53.61	4.90	0.025 708

4　结束语

本文介绍了网格任务调度机制，采用模块化设计方法设计了负责任务调度功能的 Agent 和执行网格任务的 Host，从而实现了 SF, LF, FCFS, EDF 等网格任务调度算法。仿真结果表明调度算法运行良好。

参考文献

[1] Takefusa A. Bricks: A Performance Evaluation System for Scheduling Algorithms on the Grids[C]//Proc. of JSPS Workshop on Applied Info. Tech. for Science. [S. l.]: ACM Press, 2001.

[2] H J Song, Liu Xin, Jakobsen D, et al. The MicroGrid: a ScientificTool for Modeling Computational Grids[C]//Proc. of Supercomputing, ACM/IEEE 2000 Conference. [S. l.]: IEEE Press, 2000.

[3] Casanova H. SimGrid: A Toolkit for the Simulation of Application Scheduling[C]//Proceedings of the 1st IEEE/ACM International Symposium on Cluster Computing and the Grid. [S. l.]: ACM Press, 2001: 430-437.

[4] 朱福喜，何炎祥. 并行分布计算中的调度算法理论与设计[M]. 武汉: 武汉大学出版社, 2003.

[5] Cao Junwei, Spooner D P, Jarvis S A, et al. Grid Load Balancing Using Intelligent Agents[J]. Future Generation Computer Systems, 2005, 21(1): 135-149.

（上接第 268 页）

主程序流程图和中断程序流程如图 8~图 10 所示。

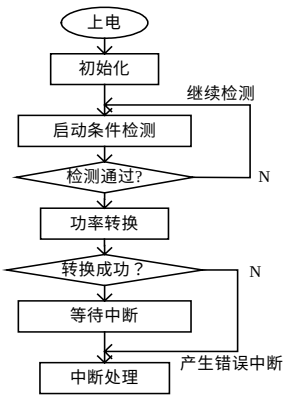

图 8　主程序流程

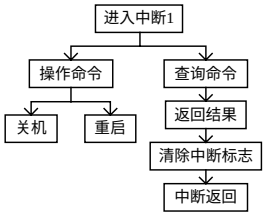

图 9　中断处理程序 1

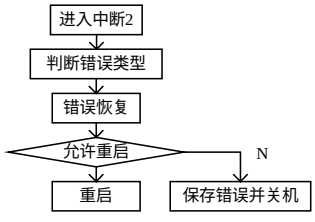

图 10　中断处理程序 2

2.3.3　管理界面

该部分软件的功能是为用户显示整个数字电源系统的各种运行参数值。如电池剩余工作时间、环境温度、输入/输出端电压和电流值等。当系统检测到异常情况时采取相应的应急措施并向用户发出警报信息提示并自动记录下故障种类和发生时间(图略)。

3　结束语

本文对数字电源系统的原理进行了简单的描述，并设计出了基于便携式设备数字电源系统的实现方案。实验证明，数字电源具有良好的人机交互功能和较高的技术指标，可以取代传统的模拟开关电源。本系统具有较强的可扩展性，预留了温度管理功能。随着便携式设备的广泛应用和发展，用户对产品的小型化和智能化的不断追求，必将进一步推动数字电源的发展。

参考文献

[1] PMBus™ Power System Management Protocol Specification(Version 1.0)[Z]. (2007-01-17). http://pmbus.org/specs.html.

[2] White R V. Using the PMBus™ Protocol[Z]. (2007-01-17). http://www.pmbus.info/Using_The_PMBus_20051012.pdf.

[3] SBS-IF. System Management Bus Specification(Version1.1)[Z]. (2007-01-17). http://www.smbus.org/specs/.

[4] SBS-IF. Smart Battery Charger Specification(Version1.1)[Z]. (2007-01-17). http://www.sbs-forum.org/specs/.

[5] SBS-IF. Smart Battery System Manager Specification(Version 1.0)[Z]. (2007-01-17). http://www.sbs-forum.org/specs/.