

基于动态关键路径的仿真网格资源调度算法^{*})

王振明 都志辉

(清华大学计算机科学与技术系 北京 100084)

摘要 从仿真系统工作流的关键路径分析,确定关键路径上的联邦成员节点,使网格资源调度方面优先获得保证。鉴于仿真网格系统运行的复杂性和不确定性,引入了随机规划理论,提出了仿真网格的动态关键路径概念以及基于动态关键路径的资源调度算法,共同解决当资源有限以及任务相关情况下仿真网格的资源调度问题。具体实现,主要使用遗传算法以及 MCP 算法。仿真实验结果表明,基于动态关键路径的资源调度算法能够优化仿真网格资源的调度。
关键词 仿真网格,动态关键路径,资源调度,随机规划,遗传算法

Resource Scheduling Algorithm for Simulation Grid Based on Dynamic Critical Path

WANG Zhen-Ming DU Zhi-Hui

(Department of Computer Science and Technology, Tsinghua University, Beijing, 100084)

Abstract From the critical path analysis of simulation system work flow, federation members nodes of the critical path are confirmed. They are assigned to grid resource priority. In view of the complexity and uncertainty of simulation grid, stochastic programming is introduced. The dynamic critical path concepts for simulation grid and resource scheduling algorithm based on dynamic critical path are brought up to solve the problem of resource scheduling for simulation grid when simulation grid resources are limited and its tasks are dependent. The concrete implementation on the part of resource scheduling is made by mainly using the genetic algorithm and MCP algorithm. Simulation experiment results indicate that resource scheduling algorithm based on dynamic critical path can optimize the scheduling of simulation grid resources.

Keywords Simulation grid, Dynamic critical path, Resource scheduling, Stochastic programming, Genetic algorithms

1 引言

建模、仿真技术与高性能计算一起,正成为继理论研究和实验研究之后第三种认识、改造客观世界的重要手段。基于 HLA(High Level Architecture)^[1] 等技术的复杂协同仿真系统实现了仿真系统的互操作,并具备一定的资源重用能力,但还存在以下不足之处:

- 1) 仿真资源静态绑定,不能在仿真运行过程中动态调度,容易造成某些仿真资源成为影响整个仿真系统运行效率的瓶颈。
- 2) 资源静态分配,导致仿真系统的容错性较差。
- 3) 仿真应用同仿真模型紧紧地耦合在一起,不利于各个仿真单位之间的协调、高效开发。
- 4) HLA 规范中没有考虑分布仿真的安全问题与服务质量(QoS)。

网格(Grid)^[2] 可以被看作是一个集成的计算与资源环境,或者说是一个计算资源池^[3],而将地理上分布的计算资源(包括数据库、贵重仪器等其他各种资源)充分利用起来,协同解决复杂的大规模问题,这就是网格计算^[4-6]。按照最新文献^[7],将网格定义成一种系统,该系统通过标准、开放的通用协议和接口来协调分布式的资源,以提供最好的服务质量。开放网格服务结构 OGSA(Open Grid Services Architec-

ture)^[8,9] 是 Globus 项目组和 IBM 共同倡议的网格标准,它把 Globus 标准与以商用为主的 Web 服务的标准结合起来,网格服务统一以服务的方式对外界提供,是目前最新的一种网格体系结构。OGSA 是一个以服务为中心的模型,其中心概念就是网格服务,采用统一的 Web Service^[10] 框架,具备了 Web Service 的所有有利因素。

HLA 仿真系统和网格系统具有很多相似性,它们都是分布式系统,都提供系统之间的互操作性和可重用性。网格技术对于解决大规模的资源共享、动态调度等方面有很大的优势,而且网格技术中有完善的安全机制,这些正是 HLA 中所缺少的。所以,将网格计算与 HLA 规范相结合,建立仿真网格,将能够解决上述 HLA 实现中遇到的种种问题,同时仿真网格自然具备网格资源共享与协作工作的本质。

仿真网格的关键优势之一是实现资源的动态调度,以完成时间(Complete Time)为优化目标的任务-资源映射是 NP 完全问题^[11],所以需要辅助的启发算法,包括模拟退火、遗传算法等等。本文从优化仿真用户对执行时间等性能要求的指标出发,考虑到仿真网格应用层工作流可能存在拓扑关系即任务相关这一特性,分析了仿真网格应用层工作流关键路径的动态性和不确定性,提出了仿真网格动态关键路径概念,并在此基础上给出基于动态关键路径的资源调度算法。鉴于仿真网格工作流具有不确定性因素,引入随机规划^[12] 理论进

^{*}) 本论文受“北京市自然科学基金(项目编号:4042018)、863 仿真应用网格项目“仿真网格的应用研究与开发”(项目编号:2004AA104330)和 973 子课题“大型互联网分布式计算理论和方法研究”(项目编号:2004CB217903)的资助”。王振明 硕士研究生,主要研究方向为网格计算;都志辉 副教授,主要研究方向为网格计算、集群式计算。

行研究,并采用遗传算法^[13]作为主要实现算法,同时使用了MCP算法^[14]。文中分类讨论了基于动态关键路径的仿真资源的预留与分配策略,最后进行仿真实验,验证处于不同关键路径的任务节点同时请求资源的分配算法,并使用Grid-Sim^[15]网格模拟器与先来先服务(Fist Come First Served, FCFS)、轮转法(Round Robin, RR)以及Min-min等算法在仿真网格有限资源调度方面进行了对比实验。

2 仿真网格体系结构及关键技术

2.1 基于 OGSA 的仿真网格体系结构

仿真网格综合应用复杂系统模型技术、先进分布仿真技术、VR技术、网络技术、管理技术、系统工程技术及其应用领域有关的专业技术,实现仿真网格联邦中模型资源、计算资源、存储资源、数据资源、信息资源、知识资源、与应用相关的物理效应设备及仿真器等各类资源的共享与动态调度以及协同工作。参照航天二院以及清华大学提出的仿真网格体系结构^[16],给出基于OGSA的仿真网格体系结构(如图1所示),包括应用层、仿真网格服务层、网格核心服务层和资源层。

1)应用层。支持分布、虚拟组织中的各类仿真网格应用人员安全、协同进行仿真系统项目的开发、运行与应用。

2)仿真网格服务层。提供协同建模、协同管理、协同仿真、协同可视化以及服务联邦管理、声明管理、对象管理、所有权管理、时间管理、数据分发管理、仿真网格应用门户等面向仿真系统的各类核心服务。

3)网格核心服务层。基于OGSA平台,提供网格中间件运行所需的核心服务。

4)资源层。提供仿真网格使用的各类资源,包括CPU、存储设备、网络资源、VR设备、各类模型、模拟器、仿真器、可视化工具等。

应用层	仿真应用, SBA应用, 虚拟样机等
仿真网格服务层	HLA (RTI) 服务: 联邦管理, 声明管理, 对象管理, 所有权管理, 时间管理, 数据分发管理, 协同建模, 协同仿真, 协同可视化, 仿真网格应用门户
网格核心服务层 (OGSA 平台)	GRAM, MDS, GSI, Gridftp, Qos, 工作流, 监控, 调度代理, 数据管理, 作业管理等
资源层	CPU, 存储设备, 网络资源, VR 设备, 仿真模型, 仿真工具, 可视化设备等

图1 仿真网格体系结构

2.2 仿真网格关键技术

仿真网格的关键技术很多^[16],其中最关键的是实现资源的动态合理分配。实现资源的动态分配,其首要条件就是把仿真客户端和仿真服务端分离开。仿真服务端提供各种仿真服务,例如仿真计算服务、各种仿真模型库等。仿真客户端包括用户使用界面、仿真对象状态维护,以及同RTI支撑环境的接口等。两者分离以后,仿真模型、仿真建模资源库、仿真实体、计算机生成兵力服务器、计算服务器、存储器等各种仿真资源通过封装注册为网格服务对外发布,仿真客户端可以查询、调用网格服务,这样就很自然地把网格技术融入到仿真中。仿真网格完全具备网格系统资源共享和协同工作的优势。要实现仿真系统的负载平衡,提高容错能力,需要进行资源的动态合理调度与分配,可见资源的动态合理调度与分配是至关重要的。本文提出了动态关键路径概念,从仿真网

格应用层出发,计算仿真工作流的动态关键路径,在此基础上来解决仿真资源的合理分配问题。

3 仿真网格的动态关键路径

3.1 仿真网格应用层的关键路径

仿真网格运行系统由1到 $N(N \geq 1)$ 条仿真工作流组成,工作流是由一组任务组成,任务之间有前后依赖关系,每条工作流可以由AOE(activity on edge)^[17,18]网来进行描述。由于每条工作流都具有相似的特性,本文选取其中一条典型工作流进行研究。

基于HLA技术的仿真系统工作流存在各种控制型的联邦成员,比如时间、事件控制等。当控制型的联邦成员资源受限或者运行受阻,将使受限成员都不能允许推进,大大影响整个联邦运行的推进。据此可以通过采用AOE网确定各条工作流的关键路径,使关键路径上的各个控制型的联邦成员优先获得网格资源的分配。通过对整个仿真工程全生命周期和并发度影响最大的关键路径的分析,可以在仿真应用层上保证资源合理分配的实现。

3.2 AOE网的关键路径

AOE网指边表示活动的网,是一个带权的有向无环图,顶点表示事件,弧表示工序的花费时间。设 $\Phi=(V,A)$ 为一个有向无环图, $V=\{1,2,\dots,n\}$ 是节点的集合, A 表示弧的集合,而 $(i,j) \in A$ 则表示在图 Φ 中从节点 i 到 j 的一条弧。对关键路径建立数学模型:

有向无环图中 $\chi=\{x_{ij} \mid (i,j) \in A\}$ 是一条从节点1到 n 的路,当且仅当

$$\sum_{(i,j) \in A} x_{ij} - \sum_{(j,i) \in A} x_{ji} = \begin{cases} 1, i=1 \\ 0, i=2,3,\dots,n-1 \\ -1, i=n \end{cases} \quad (1)$$

$x_{ij}=1$ 或 0 分别表示弧 (i,j) 在或不在这条路中。

令 ξ_{ij} 表示弧 $(i,j) \in A$ 的长度,记 $\xi=\{\xi_{ij} \mid (i,j) \in A\}$,那么路 x 的长度就可以写成:

$$T(\chi, \xi) = \sum_{(i,j) \in A} \xi_{ij} x_{ij}$$

如果对于图 Φ 中任何一条从节点1到 n 的路 χ' 都满足 $T(\chi, \xi) = T(\chi', \xi)$,称为 χ 为图 Φ 中的一条关键路径。

众所周知,确定工作流的AOE网并求其关键路径的算法可以参照相关文^[17,18]。

关键路径在具体的活动中有着重要的作用,当一个AOE网络中的关键路径只有一条时,加速关键路径上的任一关键活动,能够加速整个工程的完成。

3.3 仿真网格动态关键路径概念

针对关键路径在仿真网格工作流方面的具体应用,考虑以下三种情况:

(1)当一个AOE网络中的关键路径不止一条时,缩短某条关键路径对缩短整个工期的作用是有限度的,因为缩短后可能成为非关键路径,工期就由新的关键路径(比如原来第二长的路径)来决定了。这样需要考虑多条关键路径。

(2)关键路径在调度过程中不是静态不变的,每一个调度步骤后,由于通信量和任务执行值改变,任务图发生了改变,当前调度步骤后需要重新计算关键路径。

(3)实际应用中,弧的长度常常是随机变量,对于这种情况,引入 α 关键路径的概念。 α 关键路径定义^[19]:如果任何一条从节点1到 n 的路 χ' 均满足:

$$\max\{T \mid \Pr\{T(\chi, \xi) \geq T\} \geq \alpha\} \geq T$$

$$\max\{T|\Pr\{T(\chi',\xi)\geq T\}\geq\alpha\}$$

那么称 χ 是一条从节点 1 到 n 的 α 关键路径,其中 α 是预先指定的置信水平。

为了找出一条从节点 1 到 n 的 α 关键路径,Zhong 和 Liu^[20] 给出了 α 关键路径模型。

$$\begin{cases} \max T \\ \text{s. t.} \\ \Pr\{\sum_{(i,j)\in A} \xi_{ij}\chi_{ij} \geq T\} \geq \alpha \\ \chi_{ij} \in \{0,1\}, \forall (i,j) \in A \end{cases} \quad (2)$$

为了解决以上三方面的问题,给出一种动态关键路径(Dynamic Critical Path,DCP)概念,其含义一方面在于关键路径在调度过程处于动态变化之中,即新的关键路径(比如原来第二长的路径)可能取代旧的关键路径;另一方面在于工作流工序的花费时间本身是随机变量,从而导致关键路径长度动态变化。要解决动态关键路径问题,需要引入随机规划理论进行研究,同时在调度过程中的每一个调度步骤后重新计算关键路径。

4 基于动态关键路径的资源调度

在引入动态关键路概念以后,可以有效地确定某一时刻工作流的关键路径,这样在对任务节点进行资源调度时依照关键路径就可以很好地划分出优先级别,具体到仿真网格资源的调度,从两个方面考虑:带有优先级的资源预留和带有优先级的资源分配。

4.1 带有优先级的资源预留

首先设定在此讨论的资源预留是一种网格系统主动性的资源调度行为,而对于多个成员节点同时申请资源预留的情形,其调度策略类同于下面的带有优先级的资源分配。

仿真网格实时性要求较高,同时关键路径是动态确定的,因此引入带有优先级的资源预留策略,以关键路径上的节点成员作为资源预留的优先级对象,参照 MCP 算法,按照计算确定的工作流某一时刻的动态关键路径,将任务节点排成一个链表,排在前面的任务具有更早的最迟启动时间,在每一步调度中链表中的第一个任务节点被取出来作为被调度节点。链表中的任务节点相对于不在链表中的任务节点具有高的优先级。

对于资源预留需要考虑执行的时机选择,如果时间允许,基于关键路径的动态特性,在要实施预留的任务节点的前一个任务节点启动时进行比较合适。具体实施可以参照在 Globus^[21] 项目的 GARA(Globus Architecture for Reservation and Allocation)^[22] 来实现。

4.2 带有优先级的资源分配

仿真网格资源的管理除去资源预留以外,资源分配更加重要。对于网格资源的动态负载平衡,容错性至关重要。对于仿真网格资源的分配,从以下 4 种情况进行考虑:同一关键路径任务节点、不同关键路径任务节点、非关键路径节点、关键路径节点与非关键路径节点。

4.2.1 同一关键路径任务节点的资源分配

同一关键路径任务节点同样参照 MCP 算法,按照计算确定的动态关键路径,将任务结点根据最迟启动时间的先后排成一个链表,排在前面的任务具有更早的最迟启动时间,在每一步调度中,链表中的第一个任务结点被取出来作为被调度结点,它被放置到那个能够使其启动时间最小的处理结点,

MCP 算法的时间复杂度为 $O(n^2 \log n)$ 。

4.2.2 不同关键路径任务节点的资源分配

对于不同关键路径的任务节点同时请求资源的情形,引入随机规划理论中的相关机会规划^[22] 来解决。相关机会规划的主要思想是在不确定环境下通过极大化随机事件成立的机会从而给出最优决策。参照相关文献^[23],对 n 个网格资源分配给 m 个联邦成员的规划实例具体说明如下:

如图 2 所示,为了求解合适的资源组合,设定 $m \times n$ 个决策变量,即 $\chi_{11}, \chi_{21}, \dots, \chi_{m1}$ 表示某一决策方案, χ 是一个决策向量,其中 $\chi_{12}, \chi_{22}, \dots, \chi_{m1}$ 分别表示 m 个联邦成员从网格资源 1 需求数量,依次类推 $\chi_{12}, \chi_{22}, \dots, \chi_{m2}$ 分别表示 m 个联邦成员从网格资源 2 需求数量,实际应用中由于客观存在的因素,有些变量是不存在的。

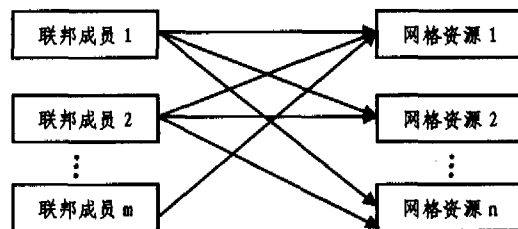


图 2 仿真网格资源分配与需求示意图

设 $\xi_1, \xi_2, \dots, \xi_n$ 分别表示 n 处网格资源所能提供的最大资源量,这些资源量动态变化,至少有一个是随机的, ξ 是随机向量,则可以获得以下随机约束:

$$\begin{cases} \chi_{11} + \chi_{21} + \dots + \chi_{m1} \leq \xi_1 \\ \chi_{12} + \chi_{22} + \dots + \chi_{m2} \leq \xi_2 \\ \dots\dots\dots \\ \chi_{1n} + \chi_{2n} + \dots + \chi_{mn} \leq \xi_n \\ \chi_i \geq 0, i=1,2,\dots,n \end{cases}$$

这类约束称作不确定环境。

设 $v_1, v_2, \dots, v_m$ 分别表示 m 个联邦成员的资源需求量,因此产生以下事件:

$$\begin{cases} \chi_{11} + \chi_{12} + \dots + \chi_{1n} = v_1 \\ \chi_{21} + \chi_{22} + \dots + \chi_{2n} = v_2 \\ \dots\dots\dots \\ \chi_{m1} + \chi_{m2} + \dots + \chi_{mn} = v_m \end{cases}$$

引入以下机会函数对以上事件进行评估:

$$\begin{cases} f_1(\chi) = \Pr\{\chi_{11} + \chi_{12} + \dots + \chi_{1n} = v_1\} \\ f_2(\chi) = \Pr\{\chi_{21} + \chi_{22} + \dots + \chi_{2n} = v_2\} \\ \dots\dots\dots \\ f_m(\chi) = \Pr\{\chi_{m1} + \chi_{m2} + \dots + \chi_{mn} = v_m\} \end{cases}$$

在不确定环境下希望极大化某些机会函数,例如对于处在关键路径链表中具有更早的最迟启动时间的联邦成员来说,希望尽可能提高其事件实现的可靠性水平,在此设定为

$$\max f_i(\chi), \dots, \max f_j(\chi) (0 \leq i < j \leq m)$$

其中 $\max f_i(\chi) = \Pr\{\chi_{i1} + \chi_{i2} + \dots + \chi_{in} = v_i\}$;

同时对于某些负载已经过重的网格服务器,希望尽可能少使用其资源,在此设定为

$$\min f_s(\chi), \dots, \min f_t(\chi) (0 \leq s < t \leq m)$$

其中 $\min f_s(\chi) = \chi_{s1} + \chi_{s2} + \dots + \chi_{sn}$ 。

这样可以在不确定环境下的仿真网格资源的分配与需求问题建立随机相关机会规划模型;

$$\begin{cases} \max f_i(\chi) = \Pr\{\chi_{i1} + \chi_{i2} + \dots + \chi_{in} = \nu_i\} \\ \dots\dots \\ \max f_j(\chi) = \Pr\{\chi_{j1} + \chi_{j2} + \dots + \chi_{jm} = \nu_j\} \\ \min f_s(\chi) = \chi_{1s} + \chi_{2s} + \dots + \chi_{ms} \\ \dots\dots \\ \min f_t(\chi) = \chi_{1t} + \chi_{2t} + \dots + \chi_{nt} \\ \text{s. t.} \\ \chi_{11} + \chi_{21} + \dots + \chi_{m1} \leq \xi_1 \\ \dots\dots \\ \chi_{1n} + \chi_{2n} + \dots + \chi_{mn} \leq \xi_n \\ \chi_i \geq 0, i=1, 2, \dots, n \end{cases} \quad (3)$$

这个随机相关机会规划模型连同前面 α 关键路径模型, 即公式(2), 可以使用遗传算法求解, 具体算法下面讲述。

4.2.3 非关键路径任务节点的资源分配

由于非关键路径任务节点对于工作流的完成进度影响较小, 可以采用传统的资源分配算法, 比如先来先服务 (FCFS) 等, 具体参照相关文献。

4.2.4 关键路径与非关键路径任务节点的资源分配

关键路径与非关键路径任务节点同时请求资源, 这在仿真网格资源调度管理中是最常见的情形, 下面进行具体分析。首先对请求节点进行分类, 即分为关键路径节点和非关键路径任务节点两类, 分类方法是根据动态关键路径算法计算出各个工作流关键路径节点并分别组成相应链表, 通过将请求节点与各个关键路径链表节点进行比较实现。其次, 根据分类优先对关键路径节点进行资源分配。相关情况参照 4.2.1 ~ 4.2.3 节的情况分别进行处理。

5 仿真实验

为了评估算法的性能, 进行了一系列的模拟研究。实验内容主要分为以下两部分。由于实验用到遗传算法, 所以先对其进行说明, 可以用来求解相关机会规划模型, 其算法可以归纳如下如下:

遗传算法程序^[23]

输入参数 pop-size, P_c , P_m 个染色体;

通过初始化过程产生 pop-size 个染色体;

重复

对染色体进行交叉和变异操作;

计算所有染色体的评价函数;

根据某种抽样机制, 比如旋转赌轮, 选择染色体;

直到满足终止条件, 比如完成给定的循环次数。

最好的染色体不一定出现在最后一代, 所以在进化开始, 必须把最好的染色体保留下来记为 V_0 , 如果新的种群中又发现更好的染色体, 则替代原来的染色体 V_0 。在进化完成以后, 这个染色体就是优化问题的解。

5.1 基于动态关键路径任务节点的资源分配实验

本实验主要验证处于不同关键路径的任务节点同时请求资源的分配算法。

参照图 2 仿真网格资源分配与需求示意图, 设定联邦成员为 $m=4$, 网格资源 $n=3$ 。如图 3 所示。

χ_1, χ_2, χ_3 分别表示联邦成员 1, 2, 3 从网格资源 1 需求的数量, χ_4, χ_5, χ_6 分别表示联邦成员 2, 3, 4 从网格资源 2 需求的数量, χ_7, χ_8 分别表示联邦成员 3, 4 从网格资源 3 需求的数

量, 希望极大化 4 个机会函数, 即尽可能提高这四个事件实现的概率。根据上面 4.2.2 节中的公式给出一个相关机会多目标规划:

$$\begin{cases} \max f_1(\chi) = \Pr\{\chi_1 = \nu_1\} \\ \max f_2(\chi) = \Pr\{\chi_2 + \chi_4 = \nu_2\} \\ \max f_3(\chi) = \Pr\{\chi_3 + \chi_5 + \chi_7 = \nu_3\} \\ \max f_4(\chi) = \Pr\{\chi_6 + \chi_8 = \nu_4\} \\ \chi_1 + \chi_2 + \chi_3 \leq \xi_1 \\ \chi_4 + \chi_5 + \chi_6 \leq \xi_2 \\ \chi_7 + \chi_8 \leq \xi_3 \\ \chi_i \geq 0, i=1, 2, \dots, 8 \end{cases}$$

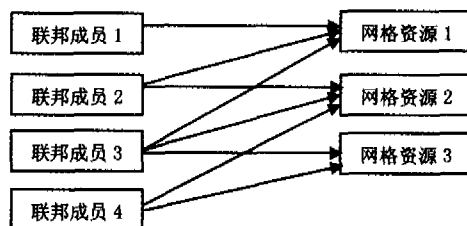


图 3 资源分配与需求示意图

设定随机参数 ξ_1, ξ_2, ξ_3 分别服从双参数对数正态分布 $\text{LOGN}(1.56, 0.56^2), \text{LOGN}(1.36, 0.45^2), \text{LOGN}(0.95, 0.38^2)$, 4 个联邦成员的需求量分别为 $(\nu_1, \nu_2, \nu_3, \nu_4) = (1, 2, 2, 3)$ 。

设定联邦成员 3, 4 分别属于工作流中第二长关键路径成员, 同时假设网格资源 3 负载已经过重, 给定优先结构和目标值:

- 1) 满足联邦成员 1 需求的机会尽可能达到 90%;
- 2) 满足联邦成员 1 需求的机会尽可能达到 85%;
- 3) 满足联邦成员 3, 4 需求的机会尽可能达到 70% 和 50%;

- 4) 从负载已经过重的网格资源 3 尽可能地少分配资源。

利用遗传算法给出最优解 $\chi = (1.00, 1.67, 1.96, 0.33, 0.03, 1.28, 0.01, 1.72)$ 。满足目标 1, 2, 3, 4 的机会分别为 99%, 85%, 47%, 85%, 其中满足目标 3 的机会小于希望的 70%, 从网格资源 3 分配资源的资源为 1.72。

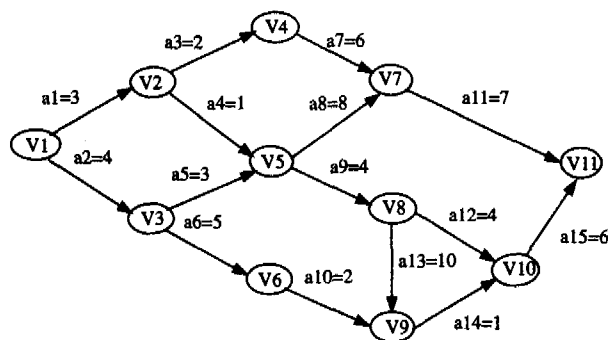


图 4 工作流网络图

5.2 GridSim 中几种资源调度算法的对比实验

使用 GridSim^[23] 网格模拟器来研究网格环境下的资源的管理和调度, GridSim 既支持时间共享资源也支持空间共享资源, 是基于 Java 的离散事件模拟工具, 具有跨平台的特点, 支持多用户环境。具体实施如下:

1)实验平台采用 GridSim, 安装配置 GridSim, J2sdk1. 4. 2, 使用 Eclipse3. 1 作为开发调试工具。

2)对 GrdSim 的几个主要类, 如 GridSim, NewPolicy, Gridlet, GridletList 进行改造, 加入基于动态关键路径的调度算法以及先来先服务(FCFS)、轮转法(RR)算法进行网格调度算法的模拟。

3)编译改造的 GridSim 的程序, 使用新的 API 库进行网格资源调度的模拟。

4)设置 11 个不同情况的节点(以运行时间约束作为区分关键路径与非关键路径节点的标准), 具体参照图 4(工作流网络图)。为了简化, 设定每个节点需要两类资源: CPU 与一种仿真模型。资源有限, CPU 可以同时满足 6 个节点的需求, 仿真模型只能同时分配至 2 个节点使用。由于工作流事件的最早发生时间有先后顺序, 当节点任务完成时立即释放资源给别的节点。当前执行节点违反拓扑有序序列时, 需要等待时间 2 再释放资源计算。其关键路径为 a2, a5, a9, a13, a14, a15, 关键路径上节点为: V1, V3, V5, V8, V9, V10, V11。

5)先来先服务(FCFS)算法中请求节点的到达顺序是随机的, 轮转法(RR)中请求节点的顺序按照 V1, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11 排列, 时间片长度的选取分别为 0.5。

6)由于存在拓扑关系, 先来先服务、轮转法、Min-min 算法在实验中都可能存在得到资源但不能执行任务的节点。为了使实验能够进行, 设定此类节点第 2 次请求资源时按照拓扑关系顺序。

表 1 不同资源调度策略工作流完成时间影响

实验编号	调度策略	先来先服务	轮转法	Min-min 算法	动态关键路径
1		38	68	38	28
2		36	68	38	28
3		54	68	38	28
4		48	68	38	28
5		40	68	38	28
6		64	68	38	28

表 1 给出了动态关键路径(DCP)算法同先来先服务(FCFS)、轮转法(RR)以及 Min-min 算法的对比实验结果。从实验结果可以看出, 基于动态关键路径的调度算法明显优于先来先服务(FCFS)、轮转法(RR)以及 Min-min 算法。

结束语 仿真网格系统的运行具有复杂性和不确定性, 本文主要贡献在于提出了仿真网格的动态关键路径概念以及基于动态关键路径的资源调度算法, 并引入随机规划理论就以上两方面进行研究。对于基于动态关键路径的仿真网格资源调度进行的仿真实验表明了算法的可行性, 同其他算法的对比实验也表明了仿真网格系统中存在动态性和不确定性因素。并且, 任务相关和资源有限的情况下, 基于动态关键路径的资源调度策略能够实现资源调度的最优化。

资源管理是仿真网格的核心问题, 有很多问题需要进一步研究, 比如仿真网格资源分配中的死锁与抢占问题、资源管

理对 QoS 的支持问题等。

参考文献

- Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society. IEEE standard for modeling and simulation (M&S) high level architecture (HLA) - IEEE std 1516-2000, 1516.1-2000, 1516.2-2000. New York: Institute of Electrical and Electronics Engineers, Inc, 2000
- 郝志辉, 陈渝, 刘鹏. 网格计算[M]. 北京: 清华大学出版社, 2002
- Foster I, Kesselman C. The Grid: Blueprint for a new computing Infrastructure. Morgan Kaufmann[M], San Francisco, CA, 1999
- Foster I. Internet Computing and the Emerging Grid. Nature Web Matters[EB/OL]. <http://www.nature.com/nature/web-matters/grid/grid.html>, 2000
- Foster I, Kesselman C. Computational Grids: The Future of High-Performance Distributed Computing[M]. Morgan Kaufmann Publishers, 1998
- Foster I, Kesselman C, Tuecke S. The anatomy of the grid: enabling scalable virtual organizations[J]. International J Supercomputer Applications, 2001, 15(3): 6~7
- Foster I, Kesselman C. The Grid 2: Blueprint for a New Computing Infrastructure[M]. Morgan Kaufmann[M], San Francisco, CA, 2003. 25~29
- Foster I, Kesselman C, Nick J M, et al. The Physiology of the Grid - An Open Grid Services Architecture for Distributed Systems Integration[EB/OL]. <http://www.globus.org/research/papers/ogsa.pdf>, 2002
- Tuecke S, Czajkowski K, Foster I, et al. Grid Service Specification [EB/OL]. <http://www.gridforum.org/ogsi-wg/drafts/GS-Spec-draft03-2002-07-17.pdf>, 2002
- Web Services Architecture[EB/OL]. W3C Notes. <http://www.w3.org/TR/ws-arch/>, 2004
- Ibarra O H, Kim C E. Heuristical gorithms for scheduling in dependent tasks on non-identical processors. Journal of the ACM, 1997, 24(2): 280~289
- 刘宝琰, 赵瑞清, 王纲. 不确定规划及应用[M]. 北京: 清华大学出版社, 2003
- Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs, 3rd ed. Springer-Verlag, 1996
- Kohler W H. A preliminary evaluation of the critical path method for scheduling tasks on multiprocessor systems. IEEE Trans Computer, 1975, 24(12): 1235~1238
- Murshed M, Buyya R, Abramson D. GridSim: A Toolkit for the Modeling and Simulation of Global Grids; [Technical Report]. Monash-CSSE 2001/102, Monash University, Australia, November 2001
- 卿社政, 熊新平, 李伯虎, 等. 仿真网格的初步实践[J]. 现代防御技术, 2004, 32(4)
- 严蔚敏, 吴伟民. 数据结构[M]. 北京: 清华大学出版社, 1992
- 杨美清. 软件复用及相关技术[J]. 计算机科学, 1999, 26(5), 40(33)
- Zhong J, Liu B. The critical path problem: Stochastic models and hybrid intelligent algorithm; [Technical Report]. 2003
- Soroush H M. The most critical path in a PERT network. Journal of the Operational Research Society, 1994, 45: 287~300
- Foster I, Globus C K. A Meta computing Infrastructure Toolkit. International Journal of Supercomputer Applications, 1997, 11(2): 115~128
- Foster I, Roy A, Sander V, et al. End-to-end Quality of Service for High-end Applications; [Technical Report]. Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, 1999
- 刘宝琰, 赵瑞清. 随机规划与模糊规划[M]. 北京: 清华大学出版社, 1998