

# 基于二维协调空间的网格工作流调度算法研究

郝丽波, 朱安新, 许靖祺

(湖南机电职业技术学院 信息工程系, 湖南 长沙 410151)

**摘要:**传统的网格工作流模型中分布式工作流管理器之间没有合作,因此可能发生源调度冲突问题,另外,在现有的工作流调度算法中,参与工作流调度的工作流管理器依托于集中或半集中的层次式的资源信息服务体系,导致系统的扩展性差。为了解决这些问题,在文中,提出了一个分布式的协同工作流调度算法。该算法基于二维协调空间来管理网格中的工作流管理器。二维协调空间负责资源发现和协调调度等功能。该算法不仅可以避免性能瓶颈,而且可以增强系统的可扩展性和自主性。

**关键词:** 网格;工作流;调度算法;二维协调空间

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2012)10-0157-04

## Research on Grid Workflow Scheduling Algorithm Based on 2-Dimensional Coordination Space

HAO Li-bo, ZHU An-xin, XU Jing-qi

(Department of Information Engineering, Hunan Mechanical and Electrical Polytechnic College, Changsha 410151, China)

**Abstract:** In the current approaches to workflow scheduling, there is no cooperation between the distributed workflow managers and as a result, the problem of conflicting schedules occur, on the other hand, the existing workflow managers rely on the centralized or semi-centralized hierarchical resource information services, so the extendibility is also bad. To overcome these problem, in this paper, propose a decentralized and cooperative workflow scheduling algorithm. The proposed approach utilizes a 2-dimensional coordination space with respect to coordinating the application schedules among the grid wide distributed workflow managers. The responsibility of the key functionalities such as resource discovery and scheduling coordination are delegated to the 2-dimensional coordination space. With the implementation of the approach, not only the performance bottlenecks are likely to be eliminated but also efficient scheduling with enhanced scalability and better autonomy for the users are likely to be achieved.

**Key words:** grid; workflow; scheduling algorithm; 2-dimensional coordination space

## 1 概述

网格计算为解决复杂计算问题提供了一种新型计算模式。网格中的许多大型应用通常被分解成若干个彼此联系的活动(任务),任务及任务间的复杂约束关系可看作是一个工作流<sup>[1-4]</sup>。有向无环图 DAG (Directed Acyclic Graph) 是工作流的一种常用描述方式,广泛存在 e-science、e-business 等网格应用领域中。工作流调度是一个有效的将工作流任务与适合的资源映射的过程,以达到用户所指定的执行时间最小化等调度目标。因此,工作流调度算法的效率直接影响到

服务质量和系统的性能<sup>[5-9]</sup>。

在当前的工作流调度算法中,分布式工作流管理器之间没有合作,因此可能发生时间表冲突问题。例如,网格环境中(如图1所示)由  $n$  个资源和  $m$  个工作流管理器组成。用户向工作流管理器提交他们的应用。这些管理器从网格信息服务(GIS)中获取资源信息产生时间表<sup>[10]</sup>。一个有效的时间表将工作流程中的任务集映射到资源集。然而,如果工作流管理器1和管理器2在同一时间通过查询GIS,他们将获得同样的资源可用性信息。基于这一信息,工作流管理器1和工作流管理器2有可能将工作流中的任务映射到相同的资源上,这将导致时间表冲突,系统和应用程序都不能正常执行。

在现有的工作流调度算法中另外一个主要缺点是参与工作流调度的工作流管理器依托于集中(参阅图1)或半集中的层次式的资源信息服务体系GIS,如

收稿日期:2012-02-13;修回日期:2012-05-17

基金项目:湖南省高等学校科学研究计划项目(10C0152)

作者简介:郝丽波(1980-),女,吉林梅河口人,硕士,研究方向为工作流;朱安新,副教授,研究方向为分布式计算和多媒体技术、软件工程。

MDS-2,3,4<sup>[11]</sup>等。研究表明,现有集中信息服务模型随着系统中的用户、工作流管理器的增加不具有良好的扩展性。因此,通过集中的方式链接到这些服务必将失败,系统由于缺乏最新的资源信息没有工作流管理器可以进行相关活动的调度。

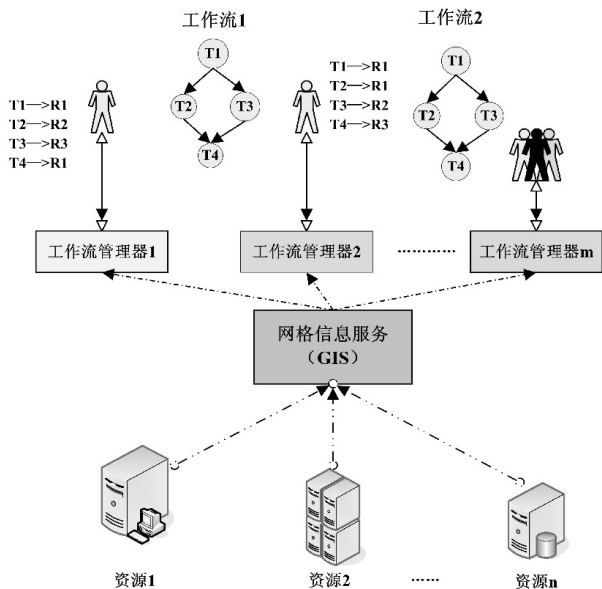


图1 当前的网格工作流调度

文献[12]中提出了一种完全的分布式的网格工作流调度算法,该算法提出了P2P的协调空间概念,并在算法中应用了DHT路由算法,保证系统完全是在分布式的环境下为各个任务进行调度。但是在该文中对于多个任务调用同一资源的情况按照先进先出的原则进行调度,这样对于执行时间要求高的任务就会因为没有及时的执行而影响整个应用程序的运行。

针对现有算法的缺点,提出一种基于二维协调空间的完全分布式工作流调度算法。算法中通过引入二维协调空间的概念为系统提供一个全局虚拟共享空间,该空间可以被系统中所有的参与者同时访问。网格中的每一个节点可以通过Chord算法映射到Chord环中,再将二维协调空间中的控制节点映射到Chord环中,从而可以完成对所有的资源申请和资源票的管理。

## 2 系统模型

### 2.1 网格模型

文中提出的工作流调度算法基于下面的网格模

型,网格 $G_F = \{R_1, R_2, \dots, R_n\}$ 由 $n$ 个站点组成,每个站点发布其资源,网格中的每一个网站有自己的资源描述信息 $R_i$ ,里面包含它所提供的资源的定义, $R_i$ 包含运算单位时间及单位成本, $R_i = (t_i, c_i)$ ,单位是秒和元。

通过一个资源管理系统来完成资源的代理、索引和分配。图2举例说明由互联网分布式并行资源组成的网格资源共享模型。每一个站点维护自身的GFA (Grid Federation Agent)服务。GFA服务是由2个软件实体组成:网格资源管理器(GRM)和分布式信息管理器(DIM)。

GRM负责调用Chord的定位算法将站点加入到网格中,同时负责用户提交的工作流程的调度,将待执行的工作流任务的信息传递给DIM,由DIM负责发布资源申请信息,GRM收到可以执行任务的网格节点信息之后,将任务交付给目标节点执行。另外GRM还管理远程节点提交的工作流任务的执行。

分布式信息管理器(DIM)提供发布/订阅索引资源功能,可以完成任务所需资源的查找和更新。一个分布式信息管理器(DIM)为了协调资源的映射会产生两种基本对象类型:

- (一)资源申请,是一个对象,发送到二维协调空间,用于资源定位,配合用户的应用需求;
- (二)资源票,是一个更新的对象,由网格的站点发出,描述基本资源条件。

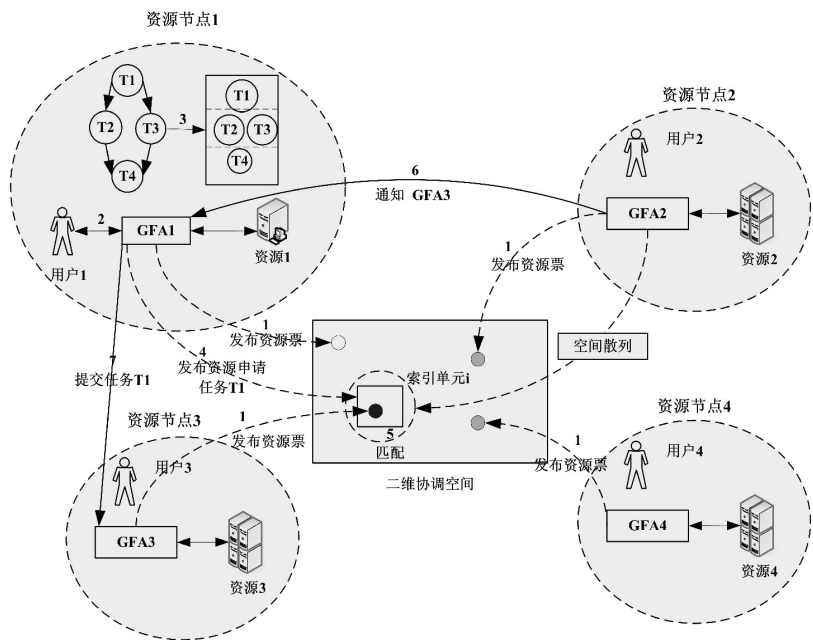


图2 网格工作流调度方法

申请和票都是二维的,分别表示处理器的数量及平均速度。

### 2.2 应用模型

网格工作流的任务结构可分为有向无环图(Di-

irected Acyclic Graph, DAG) 和非有向无环图 (Non-DAG)。文中针对 DAG 结构类型的网格 workflow 讨论其调度算法, workflow 中的任务描述为图形中的一个节点, 任务之间的关系描述为节点之间的边。workflow 应用可以描述为  $W_{i,j,k} = \{V_{i,j,k}, E_{i,j,k}\}$ , 其中  $V_{i,j,k}$  是第  $j$  个用户向第  $k$  个工作流管理器提交的第  $i$  个工作流的有限任务集  $\{T_1, T_2, \dots, T_x, \dots, T_y, T_m\}$ ,  $E_{i,j,k}$  是任务集  $\{T_{i,j,k}, Ty_{i,j,k}\}$  的依赖集, 其中  $T_{i,j,k}$  是  $Ty_{i,j,k}$  的前驱任务。

在 workflow 中, 称那些没有前驱任务的为初始任务, 没有后续任务的为结束任务。同时一个后续任务在前驱任务没有完成前不可以执行。在调度过程中, 称一个所有前驱任务都已完成的任务为待执行任务。

## 2.3 协调空间模型

在这一部分首先描述二维协调空间的通信、协调和索引对象模型, 然后提出了 GFA 之间进行协调应用的对象访问模型。

### (1) 协调对象。

这一部分给出网格系统中各 GFA 之间进行协调的基本组成元素资源申请和资源票对象的细节描述。

**定义 1** 每一个 GFA 通过本地的分布式信息管理器 DIM 提交资源票, 一个资源票包含对资源  $i$  的基本描述信息, 如:  $\text{processors} = 100 \&\& \text{processor-speed} = 2\text{GHz}$ 。

**定义 2** 一个资源申请封装了用户提交的工作流任务执行所需的资源要求。用户向本地的 GFA 提交工作流执行所需的资源信息。相应的 GFA 分布式信息管理器负责查找所需资源。如:  $100 \leq \text{processors} \leq 200 \&\& 2\text{GHz} \leq \text{processor-speed} \leq 5\text{GHz}$ 。

### (2) 二维协调空间。

在文中提出了二维协调空间的概念, 在二维协调空间中, 划分了多个空间区域, 即索引单元, 如图 3 所示, 每个索引单元的物理中心位置即为中心控制节点, 将中心控制节点通过散列算法映射到 Chord 环中, 则根据 Chord 算法, 控制节点下方的网格节点就负责该索引单元内所有的资源票和资源申请的管理。

根据文献[6]中提出的方法, 索引单元的个数等

于  $(f_{\min})^{\dim}$ , 其中  $f_{\min}$  是最小划分层次,  $\dim$  为欧几里德空间的维数。

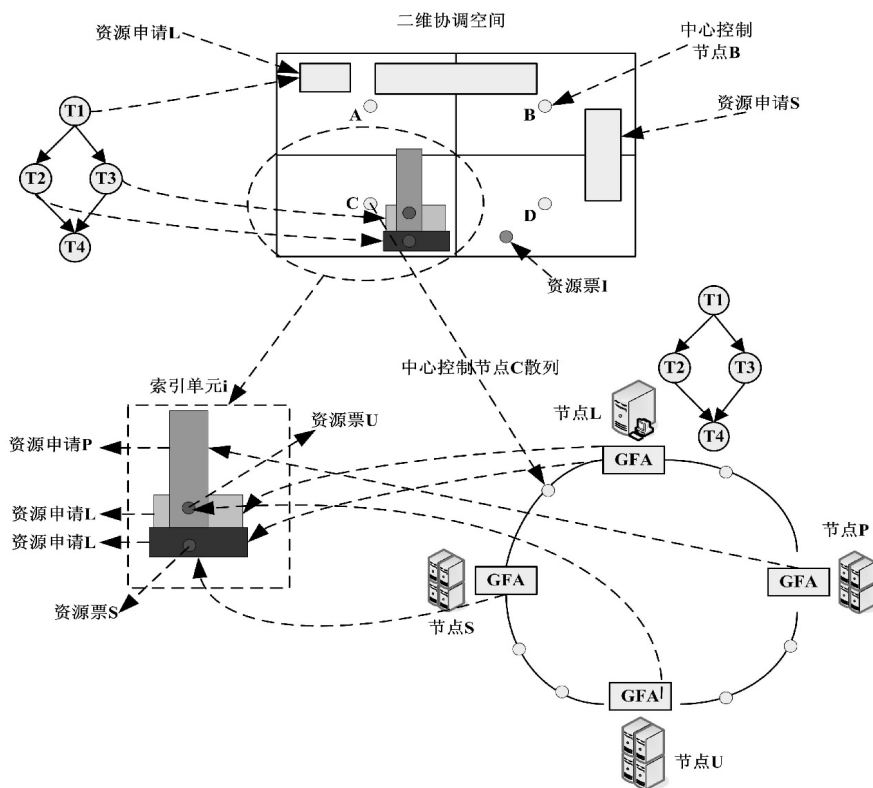


图 3 二维协调空间及中心控制节点的空间散列

## 3 算 法

整个网格系统首先根据文献[6]中提出的方法, 设定二维协调空间的最小层次数  $f_{\min}$ , 当有网格节点加入系统时, 首先调用 Chord 算法, 将自己的 IP 地址散列到 Chord 环中。

系统的调度算法如下:

① 当网格节点加入到系统时, 向二维空间中按照一定的时间间隔发布自己的资源票对象, 执行步骤 3。

② 当一个工作流应用  $W_{i,j,k}$  提交到 GFA 时, GFA 负责编译工作流中的初始任务或待执行任务的资源申请对象, 然后向二维协调空间中提交资源申请对象。根据文献[7]中提供的方法, 为工作流中的每一个任务计算其优先级。

③ 当资源申请或资源票对象发布到二维空间中时, 发布者的 GFA 计算该资源票或资源申请所属的索引单元, 通过散列算法将索引单元的中心控制节点映射到 Chord 环中, 并通过 Chord 算法找到负责管理该对象的网格节点, 将票或申请发送到该网格节点。如果该对象是资源票, 执行步骤 4, 如果是资源申请, 执行步骤 6。

④ 网格节点收到资源票, 从资源申请队列中查找与该资源票匹配的资源申请, 当匹配的资源申请数  $n$

> 1 时, 执行步骤 5, 当  $n = 1$  时, 执行步骤 7。

⑤ 根据优先级的大小, 选取优先级高的资源申请作为当前资源票匹配的对象, 当优先级相同时, 根据先进先出的原则选取资源申请。执行步骤 7。

⑥ 网格节点收到资源申请, 将资源申请放到申请队列中。

⑦ 网格节点 GFA 通知资源申请对象符合条件的资源票站点信息, 资源申请 GFA 向资源票 GFA 发出询问, 可否执行, 若资源申请 GFA 收到确认信息, 则将工作流任务传递给资源票 GFA 执行, 执行步骤 9, 否则资源申请 GFA 重新发出资源申请, 执行步骤 6。

⑧ 资源票节点 GFA 在执行完工作流任务后, 重新将资源票对象发送到二维协调空间中。执行步骤 3。

当有网格节点退出系统时, 根据 Chord 算法的思想, 将该节点负责管理的所有资源申请移交给 Chord 环上的下一个节点, 从而保证无资源申请丢失的情况发生。

## 4 实验结果与分析

### 4.1 实验环境设计

模拟的网格系统采用法国一个实际网格系统——Grid'5000[8]中的相关数据。设定实验中的工作流都是单入单出的, 网格节点的总数为 9, 用户提交的工作流的任务数在[20, 100]之间均匀分布, 每个任务的指令数在[50000, 500000]之间随机产生。在实验中, 资源票对象的输入时间间隔在[100, 300]之间, 并设置以 100 为一级, 设定二维协调空间的最小层次数  $f_{\min}$  为 2, 每个索引空间根据它的控制节点的值散列到 Chord 环中。根据文献[6]中的方法,  $\dim = 4$ , 则索引空间的总数为 16, 因此, 由 9 个 GFA 组成的网格系统中, 平均每个节点负责管理 1.8 个索引空间。

### 4.2 实验结果分析

图 4(a) 显示了在资源票输入时间间隔不同的情况下, 随着任务数的增加每个任务的协调响应时间情况, 结果表明, 随着资源票输入时间间隔的增大, 每个任务的平均协调延迟时间也在逐渐的增大。产生这一结果的主要原因是资源申请为了找到合适的资源票对象需要等待更长的时间, 从而协调的延迟时间增大。任务处理时间不受资源输入时间间隔的影响, 图 4(b) 中显示的任务的平均响应时间和协调延迟时间有相同的趋势, 随着工作流任务的增加, 系统中的资源票和资源申请的数量也在增加, 这将导致协调的延迟时间增大, 从而响应时间也会增大。

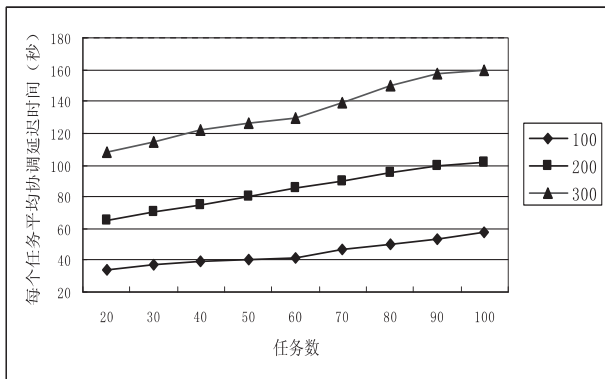
### 4.3 算法实例说明

下面应用图 3 示例讲解该调度算法的运行过程。

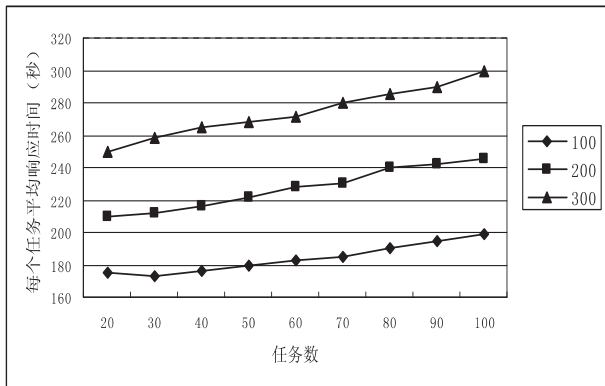
(1) 网格中的各个节点通过 Chord 算法, 以此将各自的 IP 地址映射到 Chord 环中;

(2) 加入 Chord 环中的节点 P、L、S、U 定期向二维协调空间发布自己的资源票信息;

(3) 用户向网格节点 L 提交工作流程, 节点 L 的 GFA 分析工作流, 并将任务 T1 的资源申请发布到二维协调空间中;



(a) 任务数—平均协调延迟时间



(b) 任务数—平均响应时间

图 4 工作流任务数对协调延迟时间和响应时间的影响

(4) 任务 1 执行成功后, 节点 L 的 GFA 将任务 T2 和任务 T3 的资源申请发布到二维协调空间, 同时网格节点 L 也向二维协调空间中发布资源申请;

(5) 网格节点 P 和 L 提出的资源申请对象映射到索引单元  $i$ , 索引单元  $i$  的中心控制节点 C 被散列到 Chord 环中, 根据 Chord 算法负责管理该索引单元的站点是 S, 这样站点 S 负责散列到单元  $i$  的所有资源申请对象的协调和资源共事;

(6) 站点 U 提交一个资源票正为 P 和 L 所需的资源, 这种情况下, 站点 S 的 GFA 根据 P 和 L 的优先级决定哪一个站点可以访问站点 U 的资源, 这种负载分配可以避免站点 U 的资源被重复装载。

## 5 结束语

文中提出了一种基于二维协调空间的网格工作流

(下转第 164 页)





# 基于二维协调空间的网格 workflow 调度算法研究

作者: [郝丽波](#), [朱安新](#), [许靖祺](#)  
作者单位: [湖南机电职业技术学院 信息工程系, 湖南 长沙 410151](#)  
刊名: [计算机技术与发展](#)  
英文刊名: [Computer Technology and Development](#)  
年, 卷(期): 2012(10)

本文链接: [http://d.g.wanfangdata.com.cn/Periodical\\_wjz201210041.aspx](http://d.g.wanfangdata.com.cn/Periodical_wjz201210041.aspx)