摘　要

随着网格技术的高速发展，网格资源管理已成为实现高性能计算的关键。如何高效、准确、科学地发现网格资源是网格资源管理的一个重要问题。因为整个网格的计算资源、连同网格本身都是动态的，应用开发者更加迫切需要移动计算技术的支持。移动Agent可以携带可执行代码、数据和运行状态在网格各主机之间自主移动，能够较好的适用于网格的动态环境。

在对国内外已有的网格资源发现机制研究的基础上，发现目前的分布式与集中式发现相结合的技术还不成熟、存在网格资源发现效率不高、不能适应网格动态性等问题。本文基于 Agent 技术将分布式与集中式网格资源发现机制相结合，建立了一种新的网格资源发现模型。在虚拟组织内提出了从本地存储到虚拟组织管理节点的快速资源发现方法，并改进了模型中移动 Agent 在动态网格环境下的路径优化算法。

针对移动 Agent 在网格环境中的路径优化问题，研究了蚁群算法及其已有技术在网格环境中发现资源效率和动态适应性的不足，提出了相应的可行性的改进算法，其核心思想是用遗传算法对蚁群算法初始化规则进行改进，提高算法收敛速度，并采用节点更新规则来反映网格中不同节点符合用户的满意程度和它们的变化，将算法更好的应用于网格资源发现问题中。

为了验证改进的蚁群算法在移动 Agent 网格路径优化方面的性能，本文设计并实现了几组实验，编程模拟了网格环境，从算法有效性、网格动态适应性以及算法的普适性方面对改进的蚁群算法与基本蚁群算法、遗传蚁群算法进行比较。实验结果证明，本文改进的算法是有效的，该算法解决了移动 Agent 在网格环境中的动态路径优化问题，从而，提高了网格资源发现的效率。

关键词： 网格，虚拟组织，资源发现，移动Agent，蚁群算法，路径优化

Research of Grid Resources Discovery Mechanism Based on Agent

Feng Xueli(Computer Application Technology)
Directed by Associate-Professor Liu Suqin

Abstract

With the rapid development of grid, grid resources management is the key to achieving high-performance computing. How to discovery grid resources efficiently, accurately and scientifically is an important issue to grid resources management.Because the whole grid computing resources, together with the grid itself is dynamic, so application developers more urgent need the support of mobile computing technology. Mobile agent technology used in grid can move automatically between the hosts of the grid carrying executable code, data and the running state, so it can be better used in dynamic grid environment.

On the basis of in-depth study on the domestic and abroad grid resources discovery mechanism, the current technology of distributed finding-mechanism integrating with centralized finding-mechanism was discovered still not be mature.There are still some problems existing,such as low-efficiency finding grid resource, not adjustting to dynamic grid resources.This paper based on agent technology,combined with distributed and centralized grid resources discovery mechanism, and established a new grid resources discovery model. Designed rapid resources discovery method from the local storage node to the virtual organization manage node in the virtual organization, and improved model of mobile agent path optimization algorithm in the dynamic grid environment.

Against to the path optimization problem of mobile agent in the grid environment, studied ant colony algorithm in the grid environment, found the ant conlony algorithm inadequate of efficiency and not adjustting to dynamic of the grid environment, the core idea of improving algorithm is to use genetic algorithms to improve ant algorithm initialization rules ,enhance algorithm convergence speed. And , use the rules of updating nodes to reflect the user's satisfaction to different grid nodes and their changes,the algorithm will be applied to better grid resource discovery issues and make the algorithm applied to grid resources discovery issues better.

In order to validate the performance of improved ant colony algorithm in grid path optimization of mobile agent, this paper designed a few experiments, simulate grid environment by programming, from algorithm effectiveness, grid dynamic environment as well as the universal application, compared the basic ant colony algorithm, genetic ant colony algorithm with improving ant colony algorithm. The experimental results show that this improved algorithm is effective, the algorithm solved the dynamic path optimization problem of mobile agent in grid environment, thus, improved the grid resources discovery efficiency.

Key words: grid, virtual organizations, resources discovery, mobile agent, ant algorithm, path optimization

图表清单

关于学位论文的独创性声明

本人郑重声明：所呈交的论文是本人在指导教师指导下独立进行研究工作所取得的成果，论文中有关资料和数据是实事求是的。尽我所知，除文中已经加以标注和致谢外，本论文不包含其他人已经发表或撰写的研究成果，也不包含本人或他人为获得中国石油大学（华东）或其它教育机构的学位或学历证书而使用过的材料。与我一同工作的同志对研究所做的任何贡献均已在论文中作出了明确的说明。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：冯雪刚　　日期：　2008 年 5 月 26 日

学位论文使用授权书

本人完全同意中国石油大学（华东）有权使用本学位论文（包括但不限于其印刷版和电子版），使用方式包括但不限于：保留学位论文，按规定向国家有关部门（机构）送交学位论文，以学术交流为目的赠送和交换学位论文，允许学位论文被查阅、借阅和复印，将学位论文的全部或部分内容编入有关数据库进行检索，采用影印、缩印或其他复制手段保存学位论文。

保密学位论文在解密后的使用授权同上。

学位论文作者签名：冯雪刚　　日期：2008 年 5 月 26 日

指导教师签名：刘志芹　　日期：2008 年 5 月 26 日

第一章 绪论

1.1 研究背景和意义

网格是信息社会的网络基础设施，它是把整个因特网整合成一台巨大的超级虚拟计算机，实现互联网上所有资源的互联互通，完成各种资源智能共享的一种新兴的技术[1]。在网格计算中，首先要找到网格内所有可以被使用的资源，这就需要获取其运行状态下的相关性能数据，通过这些性能数据分析可以提高下次程序运行的效率，这样可以提高网格环境中不同应用或者系统的性能。从本质上说，资源发现就是资源请求者给定一个预想的资源描述，一个资源发现机制将返回一组与描述相匹配的资源[2]。然而，目前网格中的资源发现面临许多问题:网格技术允许大范围的资源共享，这种共享关系可以是静态的、长期的，也可以是高度动态的，比如组织成员的加入和离开、成员可享用资源的改变。现有的网格资源发现机制，通常是采用集中式信息管理机制，对于大规模的网格环境，则容易造成性能瓶颈，而且可扩展性较差，不适应网格动态变化的特性。网格中数量庞大的实体之间巨大的差异性、行为的动态性和地理上的分布性，使得及时、准确地掌握资源信息和迅速地找到所需资源，变得相当有难度，使现有的网格资源发现机制捉襟见肘。

当前比较成熟的资源发现机制典型的有织女星网格的基于资源路由器[3]的资源发现机制和Globus的MDS(Monitoring and Discovery Service)[4]等。路由器的资源发现机制具有灵活性不高，性能不够好等缺点。MDS实现了基于LDAP(Lightweight Directory Access Protocol)的树状元数据目录服务，通过构建层次状目录服务器来完成资源的组织，但仍缺乏有效、易扩展的分布式目录服务器的构建、维护方法，而且目录服务器层次的搭建是通过配置文件来完成的，不易动态地调整结构。

为了解决复杂、动态、分布式智能应用，移动Agent[5]技术作为一种全新的计算手段被提出，它可以在异构的软、硬件网络环境中自由移动。把这种技术引入网格资源发现必将能有效地降低网络负载、提高通信效率、动态适应变化了的网格环境。

1.2 研究现状

如何有效地进行网格资源发现是当前研究的一个热点。在引入Agent技术的资源发现领域主要有以下几种模型：

在第一种模型中借鉴了Globus中网格资源发现和监控的一些思想[6]。在虚拟组织内，每个节点都部署基于移动Agent的网格发现和监视系统Grid Ferret，及本地的LDAP目录服务器。利用Grid Ferret系统，通过资源注册、注销目录进行被动资源发现。该模型包括两种目录信息树的结构:全局的目录信息和本地的目录信息，对资源信息采用了多层次的树型组织方式，是采用集中式资源发现与监控。但是本模型的缺点是：容易产生单点失效和系统瓶颈，系统可扩展性不好。

第二种在组织内部的资源采取集中式的管理模式，在组织间采用基于移动代理的分布式资源发现模式，各组织间的资源管理采用P2P(Peer-to-Peer)的模式。当组织内部的某资源请求需要在更广泛的领域进行转发时，由该节点向邻居节点派发一组移动代理，其邻居节点由语义覆盖网络SON (Semantic Overlay Network)决定，SON中的邻居定义为具有相似主题的两个节点，节点之间的距离越近，语义相似度越高。每个资源请求也被映射到语义空间的某个点，将请求的转发锁定在该点周围一个小范围内。每个代理将资源请求和其他节点本体进行比较，实现资源的语义匹配。完成匹配任务后，移动代理将结果返回给发出请求的节点。

第三种动态资源发现Agent是根据资源特性创建相应的移动Agent，MA(Mobile Agent)，并为之指定一个特定的约束条件和路由规则，用户派遣移动Agent，并处理由移动Agent带回的数据。这种按照资源发现Agent预先规定的路线和策略在各个资源节点之间迁移并与资源节点上的系统Agent交互的路由策略是不符合网格的动态特征的。

如何用最快的速度找到自己所需要的解决问题的资源，并且能够充分适应网格的动态性，从而显著提高系统执行效率和适应性是我们所面对的首要问题。

1.3 论文研究的内容与目标

本文在对已有的网格资源发现机制、移动 Agent 技术和蚁群算法作了较为深入研究的基础上，在网格资源发现中移动 Agent 路径优化算法的研究领域主要作了以下工作：

(1) 深入分析了已有的网格资源发现机制的特点，研究了集中式和分布式网格资源发现的关键技术，从原理上分析了已有的网格资源发现模型的不足，为提出新的网格资源发现模型奠定了基础。

(2) 仔细研究了移动Agent及其关键技术，包括移动Agent的概念、特点以及体系结构等方面内容，分析移动Agent系统结构模型以及其用在网格资源发现中的优势。

(3) 在分层思想的基础上建立一种新的基于Agent的网格资源发现模型，在虚拟组

织VO(Virtual Organizations)内提出了从本地存储到虚拟组织管理节点的快速网格资源发现方法。详细研究了网格系统的建立和网格服务的生成方法，对于网格服务接口的类型进行了定义，并对不同类型的Agent行为进行定义。

(4) 对网格环境中移动Agent迁移策略及其路由规则方面，在对传统蚁群算法进行深入研究的基础上，提出改进的蚁群算法，用于求解动态网格环境中进行资源发现时移动Agent的路径优化问题。同时，本文对蚁群算法结合遗传算法等内容进行研究，给出定量表达式和算法描述。

(5) 模拟网格实验环境，对网格资源节点不同类型的资源分布率进行初始值设定，将改进后的算法予以实现并和其他算法进行比较，得出结论。

通过以上几部分，对提出的新模型主要部分的功能进行模拟实验，通过在模拟网格环境中对改进算法的性能进行测试，得出实验结果证明本模型中路径优化策略的可行性，以解决网格资源发现机制中，移动Agent在动态网格环境下的路径优化问题，从而解决网格环境中资源发现的效率问题。以下将对本文的组织给出说明。

1.4 论文的组织与安排

第一章：绪论。介绍本论文的研究背景、研究意义、课题相关领域研究现状、课题的研究内容以及论文的组织结构。

第二章：网格及移动Agent相关技术。本章阐述了网格的基本概念、体系结构以及网格发展情况，并介绍了移动Agent的体系结构、特点及其相关技术，移动Agent在动态网格环境中的应用情况。重点分析了网格资源发现的需求以及目前已有的各种网格资源发现模型。

第三章：一种新的网格资源发现机制。在分层的思想基础上提出一种新的网格资源发现模型，在模型的虚拟组织内提出了从本地存储到虚拟组织管理节点的快速资源发现方法，并且对该模型的组织结构、工作原理、环境配置进行了阐述。

第四章：新模型中路径优化算法的改进。网格资源发现中的移动Agent路径优化算法的研究以及对蚁群算法的改进主要体现在本章。主要内容包括：基本蚁群算法的介绍、已有的蚁群算法的改进、蚁群算法的初始化问题的改进、节点更新算法的改进、网格动态性更新规则及路径更新规则的研究。

第五章：算法验证及分析。先从理论上分析了改进后的算法并设计实验，模拟网格环境，对算法进行测试与分析，根据实验结果对算法的各个方面做出了客观评价。

结论：主要总结了作者在完成论文期间所做的研究和创新工作，并对将来的工作提出一些设想。

第二章　网格及移动 Agent 相关技术

2.1　网格技术综述

网格是根据电力网的概念提出来的。我们在使用电力时，不需要知道电是从哪个地方的发电站输送出来的，也不需要知道是通过什么样的发电设备产生的。用户只是使用统一形式的电能。网格的最终目的是希望用户在使用网格时，如同现在使用电力一样方便。网格代表了一种先进的技术和基础设施，它把整个互联网集成为一台巨大的超级计算机，实现全球范围的计算资源、存储资源、数据资源、信息资源、知识资源、专家资源、设备资源的全面共享。当然，网格的规模并不一定要很大，我们也可以构造地区性的网格、企事业内部网格、局域网网格、甚至家庭和个人网格。网格的根本特征是资源共享，消除资源孤岛[7]。

本质上说，网格计算需要解决的问题是如何在动态的、异构的虚拟组织间实现资源共享以及协同的解决某一问题。

2.1.1　网格的概念及现状

网格计算最初也被称为元计算，它的概念最早出现于1995年的I-WAY项目中，到目前为止，学术界对于网格和网格计算的定义一直存在分歧，没有一个能够被普遍接受的定义。Ian Foster[8][9]等人在网格和网格计算理论方面做出了巨大贡献，他们认为:网格就是在缺少中央控制、全局信息以及严格信任关系的情况下能够协同使用地理分布的各种资源。网格技术为人们提供更强大、更方便、更高级的问题求解手段[10]。

(1) 国内研究现状

我国的网格技术研究主要集中在一些大学和中科院的相关研究所。中国科学院计算技术研究所在 2000 年建立了“国家高性能计算环境”（NHPCE），通过 CASNet 将分别位于北京、上海、西安、长沙、成都、合肥的六个计算节点连接到一起，使得用户能够方便地使用这几个计算节点的计算能力，可以被视为一个初步的网格系统。清华大学计算机系承担建设的先进计算基础设施将位于北京清华大学的机群系统 THNPSC2 和位于上海大学的超级计算机自强 2000 为服务节点，建立了包括资源管理、用户管理、任务管理、数据库访问接口等部分的 ACI 管理系统，实现了网格计算环境。

2002 年 12 月，上海信息网格正式立项，目的是研究符合国际标准的信息网格体系结构和关键技术，开发具有自主知识产权和推广价值的信息网格系统软件、应用开发环

境和虚拟研究平台，通过对虚拟组织中的计算、数据、软件等各种信息的共享和协同，建立具有上海特色的信息网格，并初步实现交通信息网格典型应用。2003 年 10 月，教育部与 IBM 合作的中国教育科研网格项目正式启动，该项目计划通过教育科研网，将全国 100 所 211 建设重点大学联网，实现资源的广泛共享。“高性能计算机及其核心软件”是研究网格技术的一个专项项目，在建立大型计算环境的基础上，重点研究网格计算的相关技术问题，开发基础网格软件，实现多个大型超级计算机中心的计算资源共享，基于 Globus 开发了织女星网格（VEGA GOS 4.0）平台。该项目已于 2006 年年初通过验收，达到了预期目标。

(2) 国外研究现状

网格技术研究发源于美国，美国是目前网格研究走在世界最前列的国家。开发了 Globus、Condor、Legion 等比较有影响的软件和工具。

美国军方正规划实施一个宏大的网格计划，叫做“全球信息网格GIG(Global Information Grid)”，预计在2020年完成。作为这个计划的一部分,美国海军和海军陆战队已启动了一个耗资160 亿美元、历时8年的项目，包括系统的研制、建设、维护和升级。美国能源部的山地国家实验室的“先进战略计算创新计划网格(ASCI Grid)”主要用于核武器研究。美国国防部和欧洲能源机构等在两三年前先后采用了网格技术。

英国建立了全球最大的网格计算机系统。该网格系统由6000多台分散在78个国家的计算机组成，而英国就拥有其中的1000多台，它是“大型强子对撞机计算网格”(LCG)的一部分。LCG是全球目前最大的网格系统，而且也是第一个永久性网格系统。

总的来看，在涉及企业应用时，网格计算目前仍处于早期阶段。在公用事业计算中，网格应用刚刚起步，IBM、HP、Oracle 和Sun等厂商目前都还没有将网格技术视为公用事业计算的基础，而是把它当做公用事业计算的关键组成部分。

2.1.2 网格的体系结构

首先，是以协议为中心的“五层沙漏”网格体系结构。

“五层沙漏”结构模型(图2-1)是Globus项目在早期提出的具有一般性的网格体系结构，是以协议为中心的“协议结构”，强调协议的层次及其在网格的资源共享和互操作中的地位。该结构建立在互联网协议之上，以互联网协议中的通信、路由、名字解析等功能为基础，自下而上分为五层:构造层、连接层、资源层、汇集层和应用层。每层都有自己的服务、API(Application Programming Interface)和SDK(Software Development

Kit)，上层协议调用下层协议的服务。

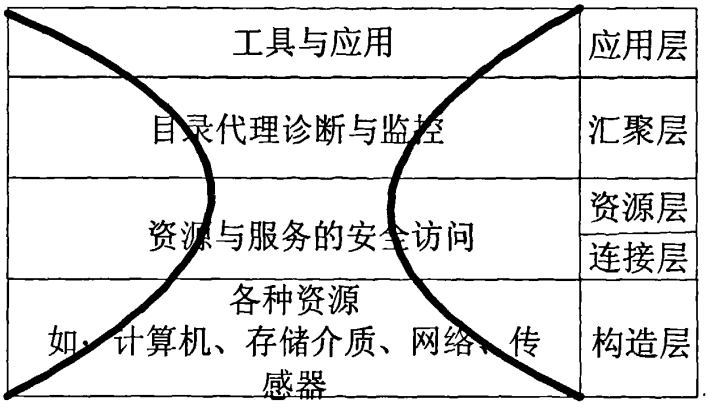

图2-1 五层沙漏结构

Fig2-1 Five hourglass structure

构造层的功能是向上提供网格中可供共享的资源，它们是物理或逻辑实体。其资源可以是计算资源、存储系统、目录、网格资源、分布式文件系统、分布式计算机池、计算机集群等。构造层提供的功能越丰富，就可以为上层支持更多的高级操作。

连接层定义了网格中网络事务处理通信与授权控制的核心协议。通信协议允许在构造层资源之间交换数据，包括传输、路由、命名等功能。建立在通信服务之上的认证协议提供加密的安全机制，用于识别用户和资源。

资源层的主要功能就是实现对单个资源的共享。建立在连接层的通信和认证协议之上，该层定义的协议包括安全初始化、监视、控制单个资源的共享操作，审计以及付费等。资源层考虑的是单个的局部资源，而对全局状态和跨越分布资源集合的原子操作则不予考虑，留给汇集层解决。

汇集层的作用是将资源层提交的受控资源汇集在一起，供虚拟组织的应用程序共享、调用。为了对来自应用的共享进行管理和控制，汇集层提供目录服务、资源分配、日程安排、资源代理、资源监测诊断、网格启动、负荷控制、账户管理等多种功能。

应用层是网格上用户的应用程序。应用程序通过各层的API调用相应的服务，再通过服务调用网格上的资源来完成任务。应用程序的开发涉及大量的库函数。为便于网格应用程序的开发，需要构建支持网格计算的库函数。

“五层沙漏”结构是一个抽象层次结构，它的一个重要特点就是“沙漏”形状。这是因为各层协议的数量不同，对于最核心的部分——沙漏的瓶颈，定义核心抽象和协议的一个小集合(如在Internet中的TCP和HTTP)。许多不同的高层(沙漏的顶部)行为映射

到它们的上面，它们自身也能被映射到不同的基本技术之上(沙漏的底部)，所以核心协议的数量是较少的，这样有利于协议的移植，也容易实现和得到支持。在五层结构中，资源层和连接层共同组成了这一核心瓶颈部分。五层沙漏结构使得不同的应用可以在统一的网格体系结构框架下使用相同的低层协议。

另一种重要的网格体系结构是以服务为中心的开放网格服务结构OGSA[11]（Open Grid System Architecture)。

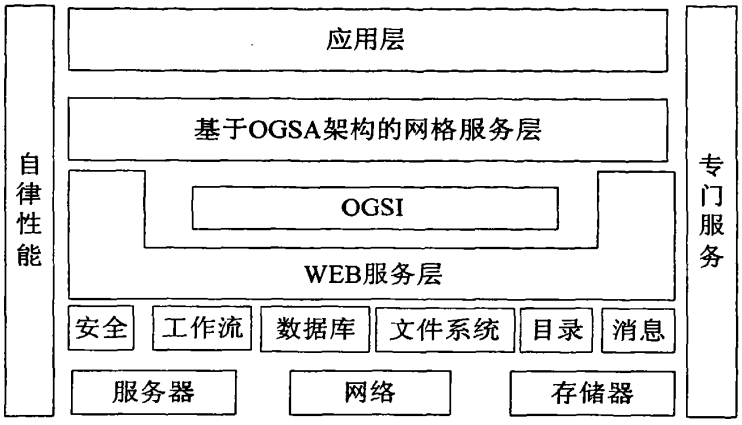

图2-2 OGSA的体系结构

Fig 2-2 OGSA architecture

OGSA的核心思想是结合Globus与目前热门的分布式技术Web Service，通过抽象、封装各种类型的资源、信息、数据，并按照统一规范定义一系列的接口用于服务发现、动态服务实例创建、生命期管理、通知等操作。OGSA对Web Service进行了扩展，提出了网格服务(Grid Service) [12]的概念。在一个OGSA架构的网格中，一切都是网格服务，因此网格就可以视为可扩展的网格服务的集合。

OGSA是一个层次结构体系，每层都有非常清晰的功能划分，核心层是OGSI(Open Grid System Infrastructure)，它提供了基础构件和OGSA的核心平台服务，是一个标准服务集。最高层的应用和服务使用低层核心平台组件和OGSI[13]。

最后，我们介绍一下基于移动Agent的网格体系结构：

(1) 通信基础设施。提供移动Agent平台的网络通信机制，这些机制大多是分布式计算技术。

(2) 移动Agent平台。提供移动Agent的运行环境。

(3) 注册服务。支持各种网格相关的组件注册到网格中，以声明这些组件的功能和通信方式，这些组件有一般组件和移动Agent组件两种类型。

(4) 目录及元数据服务。支持移动Agent的目录存储。

(5) 登录服务。为用户提供登录入口。

(6) 通讯服务。通讯服务主要针对组件间的交互。一方面是一般组件的通讯，另一方面是多Agent交互，可以使用一些Agent通讯语言来达到Agent交互的目的。

(7) 日志服务。日志记录的内容包括网格组件的行为、消息、故障等。

(8) 安全服务。支持网格相关安全措施的采用和配置管理。

(9) 发布和订阅服务。发布和订阅服务更新等消息，便于感兴趣的用户及时获得需要的信息。

(10) 可视化服务及工具。提供可视化服务的一些基础信息，如移动Agent的实时交互信息、系统响应信息，包括流量和性能的可视化信息等。

(11) 网格管理服务及工具。提供对网格服务的配置、检测、安全及故障管理功能。

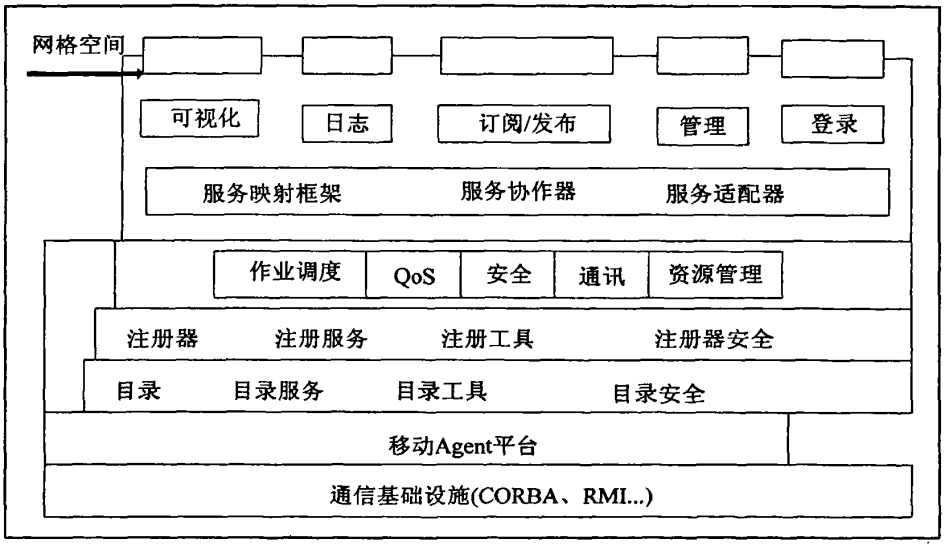

图2-3 基于移动Agent的网格体系结构

Fig2-3 Based on mobile agent grid architecture

2.1.3 网格服务及开发工具

目前的开放网格服务体系结构(OGSA)将Globus标准与商用的Web Services标准结合起来，让网格能够支持Web Services，在Web Services的基础上提出了“网格服务”概念。OGSA下，所有资源都抽象为服务，资源发现机制面对的是封装成服务的资源，而不是各种各样原始的物理资源，大大简化了资源发现问题的解决，从而使得资源发现问题归于服务发现问题。

依照OGSI的定义，每个网格服务都是一个Web Service，反之则不一定正确。网格服务在RPC协议、接口描述等方面分别沿用了Web Service中的SOAP(Simple Object Access Protocol)，WSDL(Web Service Description Language)等技术，只是需要对WSDL进行扩展，同时又在以下方面对Web Service进行了扩充：

(1) 提供有状态服务。这一点是相对Web Service所提供的无状态服务而言。在Web Service中，服务并不保存与用户相关的状态信息，一个Web Service在不同用户眼中都是同一个东西。而网格服务则可向用户提供有状态服务，可以为每个用户生成一个服务实例，同时提供了一套机制使用户可以对状态数据进行查询。

(2) 提供瞬时服务。在Web Service中，所有的服务都是永久服务，即服务实例的生命期贯穿其宿主环境生命期的始终。而网格服务除继续对Web Service中的永久服务提供支持外，还提供了瞬时服务模式，即用户需要服务时才创建服务实例，分配资源。一旦用户使用完毕后即可释放资源、销毁服务实例，从而提高资源的利用率。

下面介绍一下网格开发工具包 Globus：

Globus是一个网格技术的基础研究项目，由美国Argonne国家实验室及多所大学和研究机构联合开发，目前，Globus已成为网格业界公认的网格基础开发平台。此工具包能够用来帮助规划和组建大型的网格实验和应用平台，开发适合大型网格系统运行的应用程序。

Globus Toolkit是Globus项目最重要的实践成果，其第一版在1999年推出，其后的主要版本有1.1.3和1.1.4，2002年底推出2.2版。2003年Globus项目推出基于OGSA体系结构并融合Web Service技术的Globus Toolkit3.0版。2005年初发布了基于WSRF(Web Services Resource Framework)的Globus Toolkit4.0版。WSRF是建立在已存在的Web服务定义和技术基础上的，帮助实现了网格计算系统管理和Web服务的统一。GT4提供API(Application Programming Interfaces)来构建有状态的Web服务，其目标是建立分布式异构计算环境。从中间件的角度来看WSRF提供了应用间的简单的互联互通，成为网格中间件事实上的国际标准，使GT4简化了打造网格运算应用程序。

2.2 网格资源发现的需求

资源加入网格时，向网格注册自己，把与自己有联系的信息报告给网格，以便自己能够被他人使用。网格的资源管理设施把资源报告信息存储下来，供日后使用资源的用户使用。一个资源一旦加入网格，就可以被请求该类资源的用户使用。

网格资源有以下特点:资源种类繁多，功能各异，资源的地理分布极广，资源之间、资源和客户以及客户之间往往通过广域网(如internet)连接。网格应该为用户提供一种功能，能够根据用户的请求从网格资源中找到满足用户请求的资源。不同于DNS (Domain Name Service)服务或Web搜索，网格中的资源发现将是更复杂的发现，是根据资源请求者的资源请求描述，从网格上为请求者找到满足请求描述要求的合适资源的过程，是把资源和请求者联系起来的重要环节，有了资源发现请求才能使用自己请求的资源。否则，大量资源放在网格上，请求者不知道自己能够使用哪个资源。

目前网格中的资源发现面临许多问题，作为一种广域的大规模分布式环境，现有internet存在的带宽和延迟限制以及网络的不可靠性，资源类型和数量巨大，而且要求一定程度的协同工作，并且资源是动态变化的，包括资源属性的变化，以及在网格内的复制和迁移等，使现有的网格资源发现机制不能满足用户的需求。

资源发现功能的强弱，直接决定了网格的使用效率和友好程度。因此，我们需要一种适合网格特征的、高效的资源发现机制。

2.3 已有的网格资源发现模型

目前对于在网格环境下资源发现机制的研究众多，这些研究为基于Agent的网格资源发现机制的研究打下了坚实的基础。这些资源发现机制从本质上来说，可分为集中式、分布式和层次式，主要的研究如下。

2.3.1 集中式网格资源发现模型

Globus MDS[14]是Globus Toolkit提供的信息服务组件，它提供网格资源的信息和状态。MDS包含网格资源信息服务GRIS(Grid Resource Information Service)和网格索引信息服务GIIS(Grid Index Information Service)。MDS实现了基于LDAP的树状元数据目录服务。每个组织内通常都有一个集中的索引服务，组织规模比较大时，需要形成树状层次结构。

网格资源发现模型中引入Agent技术，可以借鉴Globus中网格资源发现和监控的一些思想。对资源信息采用多层次的树型组织方式，模型使用两种目录信息树的结构:全局的目录信息和本地的目录信息，采用集中式资源发现与监控。

这种发现机制的缺点为：被动更新。如果查询近期没有执行，则GIIS服务器必须更新它的LDAP registry，因为GIIS不能主动得知机器以及服务的当前状态；非分布式管理。

GIIS服务器要联系所有注册的主机检查当前状态。显然，这种操作是不可伸缩的。在Condor项目中，根据属性匹配进行网格资源发现。

R.Raman等人1998年实现了资源共享系统Condor[15] 的Matchmaker也采用集中式结构来查找计算资源。集中式的体系结构对于局域网比较有效，但是它是以提供中心服务器为前提的资源发现的方法。

2.3.2 分布式网格资源发现模型

不同于传统的集中式资源发现机制，结合网格和P2P的特点可构造一个应用P2P技术的网格环境下的资源发现机制。采用分布式的P2P结构，该类方法支持基于属性的查找，具有良好的扩展性，支持资源的动态加入和离开。

分布式网格资源发现方法基本上有以下几种：

泛洪[16][17](flooding)：节点间形成无结构网络，或具有某种结构的网络，如树状层次结构或超立方结构。接收到资源发现请求的节点除了搜索本地注册资源信息外，还向所有邻节点转发请求(请求进入的邻节点除外)，直至满足结束条件。为了减少发送的冗余消息，有一些方法使用随机或基于启发式规则的请求转发策略，如Random walk，Best Neighbor等，我们将这些方法统一视为泛洪方法的变形。

路由转发[18][19]RT (routing transponders)：节点以系统中其他部分节点为邻节点，且知道到所有异地资源的路由信息。资源发现请求将被接收节点转发到相应邻节点，并最终被路由到注册有满足条件资源所在的节点。在这种方式下，资源信息的更新需要传播到每个节点。

NEVRLATE[20]：节点被分为若干组，每一个资源都必须在所有组中至少一个节点上进行注册，因此资源发现的时候只要搜索任意一个组即可。但对于资源更新，则需要将其散布到每一个组。除以上几种基本方法外，还有不少混合方法，如，将泛洪与RT相结合：资源信息仅散布到注册节点小范围的邻域，首先使用泛洪方式进行查找，若找到满足条件的资源信息链接，则改用RT方法。

2.3.3 分层式网格资源发现模型

在织女星网格中，设计者提出了服务网格的概念和虚拟计算机的体系结构模型，资源发现机制面对的是封装成服务的资源，而不是各种各样原始的物理资源。虚拟机体系结构基础上，资源发现机制主要通过两个技术来解决资源发现的问题:基于资源信息的路由转发的资源定位模型和3层资源表示模型。

基于三层P2P技术的网格资源发现模型，在模型中有三种角色的节点：普通节点peer、管理节点MP(Manage Peer)、功能节点FP(Function Peer)。MP定时通知FP和其它MP自身资源信息的更新情况。当MP连接FP时直接向FP发送更新信息，否则向其它MP发送更新信息。当MP接受到其它MP更新信息时，如果连接FP则向FP转发，否则继续向其它MP转发。FP接收到MP发送过来的更新信息以后，如果资源信息类型符合自身功能设定的类型则更新本节点资源注册信息，否则向其它FP转发该更新信息。结合P2P技术得以下模型图[21]：

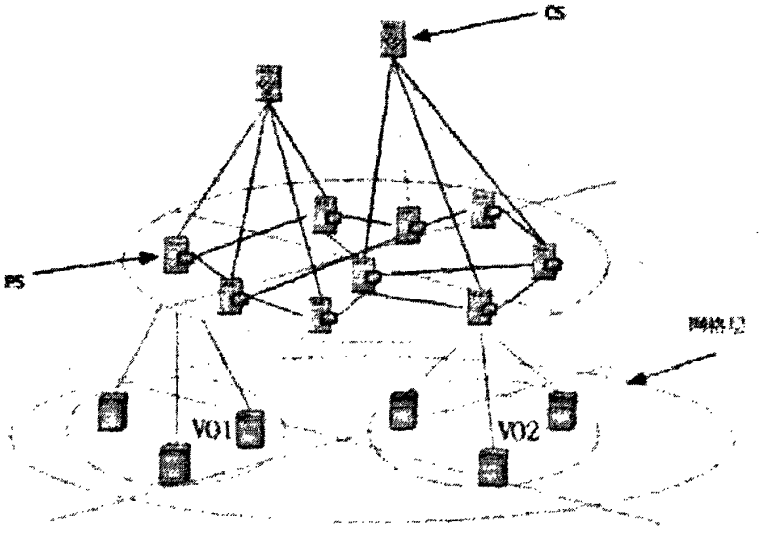

图2-4 应用P2P技术的资源发现模型框架

Fig 2-4 Model framework that application of P2P resources discovery technology

将模型分为上下两层，下层由多个虚拟组织组成。各虚拟组织内部部署的是GT3的索引服务，负责虚拟组织内部的资源发现。将该层称为网格层(或称为GT3层)。上层由多个PS(Peer Service)服务组成，每个PS服务对应一个虚拟组织。PS服务之间可以互相通信构成P2P系统。新的PS可以从CS(Contact Service)中获得邻居节点从而加入系统，该层称为P2P层。P2P层应该实现的功能包括：跨虚拟组织的资源发现和覆盖网络两个方面。

A.Kermarrec等人2000年提出Gossip协议[22][23]，节点之间通过特定协议，维护、更新信息并转发查询请求。基于传统的Gossip算法，可进行两层消息扩散机制[24]：第一层是在同一VO内的各个机构之间进行消息扩散，第二层是在网格内的各个VO之间进行消息扩散。

基于复合拓扑的网格资源发现机制[25]，从资源描述、资源组织等方面进行分析提出把网格建立成大资源库的构想，并结合P2P领域的研究成果，采用分层结构化方法实现了基于三层重叠拓扑的网格大资源库，同时提出了相应的资源发现机制。其中基于系统实现提出了改进的CAN[26](Content Addressable Network)协议、可扩展的域入口、可迁移的用户注册策略、可伸缩的管理服务机制等方法和手段来提高网格系统性能和资源发现效能。

2.3.4 其他网格资源发现模型

(1) 路由转发的资源定位模型[27]，资源定位模型主要在资源路由器上实现，是资源发现机制的核心。负责解决包括路由器网络的生成和维护、路由更新策略、资源信息聚类策略和资源请求处理策略等多个问题。但是路由协议算法还有待完善，资源路由器对节点的链路要求很严，任何关键节点的断路或者是拥塞都会影响到网格资源发现的质量和效率。存在资源路由器的存储和通信开销还过大等缺点。

(2) 有效的网格 Web 服务资源组织体系模型[28]。它将 Web 服务资源的逻辑结构与物理资源组织分开，以层次结构的 Web 服务解析系统 WSNS (Web Services Analysis System)来进行Web服务资源查找与定位，以区域自治系统来进行Web服务资源的维护，它可以在服务资源的异构、复杂与动态性、多级的、负载较小的开销下获得较满意的资源组织与定位性能，能适应网格 Web 负载平衡和良好的可扩展性。但是没有进行大规模范围的实验，而且许多实现还很简单，考虑实践复杂性还不够。

(3) 基于相似性原理的网格资源发现方法[29]提出了社区的思想，虚拟社区是使用同一类网格资源的用户的集合。采用的结构是集中式与分布式相结合的结构。即虚拟社区内采用分布式结构，用泛洪算法或者闲聊(Gossip) [30]算法进行消息扩散。但同时，有一个全球的资源信息查询中心，为无法在本社区内查找到资源信息的虚拟社区用户提供服务。

2.3.5 网格服务发现机制的研究

网格服务虽然是一种扩展的Web服务，但两者有着很大的区别。每一个Web服务都是持久的服务，其服务实例在容器启动时就被创建。而网格服务既可以是持久的，也可以是临时创建的。因此，用于Web服务的发布、查找机制不适合于网格服务。因为，频繁地在UDDI[31](Universal Discovery Description and Integration)中更新临时网格服务需要消耗惊人的网络带宽和处理能力。

最新的网格标准OGSA将网格环境中所有的软件、硬件、网格等资源均以服务的形式进行包装，提供了一个相对统一的抽象界面。网格中的资源极其丰富，但一个特定用户要想从中找出真正适用的服务仍面临不少问题。

Globus项目的MDS（Monitoring and Discovery Service）实现了基于OGSI（Open Grid Services Infrastructure）的网格信息服务系统。功能侧重于服务数据的查询，没有很好地提供对基于服务接口和行为的网格服务类型发现的支持。文献[32]对网格QoS属性作了分类并扩展了UDDI来支持基于QoS属性的搜索。但在实际网格环境中，由于资源本身的动态性和分配造成的QoS频繁更新是不可忽略的，因此在实际应用中必须作大量的改进。

WSRF提出了重新组织OGSI的概念，基于WSRF的网格服务监控与发现系统MDS4拥有以下新特征：用更为强大的Xpath查询语言代替LDAP，实施简单却更健壮，配置简单，对任意信息源提供接口，信息提供者不需预先定义模式。

已出现的相关或类似技术中，较有影响的有GlobusToolkit2中基于LDAP查询的MDS2机制以及Globus3中基于服务数据匹配的Index Service。它们的共同缺点是在实际的应用中，服务提供方与服务需求方往往无法做到事先沟通，因而简单的关键字匹配的方式无法提供足够的灵活性和推理能力，很难表述网格服务的真正能力，检索结果难以令人满意。卡内基梅隆大学的Massimo Paolucci等人将语义网和本体论的技术引入Web Service，使用OWL（Online Writing Lab）语言对其进行描述、匹配。在这些成果的基础上，可在网格服务中使用本体论进行描述和检索[33]。但是，这种算法的主要缺陷是效率问题。对OWL文件本身进行解析需要较大的开销，同时基于本体论的匹配算法需要有较完备的本体库的支持，而本体库的规模越庞大，概念之间的传递关系越复杂，进行大规模搜索时消耗时间也越多。

2.4 网格资源发现模型评价

集中式资源发现机制优点：

(1) 系统拓扑结构简单，构建和维护容易且消耗较小。

(2) 资源信息空间集中且结构不存在资源信息不一致问题，而且资源发现效率较高。

(3) 服务集中，易资源共享并发控制，而且系统安全性较好。

但其有两个最大的缺点：

(1) 系统可靠性很差，当中心服务器出现故障时，系统不能够正常工作，系统没有容错功能。

(2) 系统可扩展性比较差，受限于中央信息节点的能力，因为系统中的资源信息必须集中存储在中心服务节点，并且所有服务也必须通过中心服务节点，使得系统不能够很容易地增加大量资源。

分布式资源发现机制与集中式资源发现机制的特点几乎正好相反，其主要缺点是：

(1) 资源信息空间的无序性和无结构性使得资源发现具有一定盲目性，基于泛洪或广播的资源发现效率较低。

(2) 节点可随时加入或离开使得系统的安全性也难以控制。

但其最大的优点：是可扩展性和可靠性，节点可随时加入网络并将自己的资源共享给其它的用户，同时任何节点的关闭或故障都不会影响系统中其他节点的正常工作。

目前已有的网格资源发现方式大都是完全集中式的或完全分布式的，而巨大数量的资源和用户以及资源具有的异构性、动态性和自治性等特点，需要一种集中式和分布式相结合的资源模型，来进行系统扩展和提高效率，而目前在这方面的研究还不够成熟，移动Agent技术具有智能性、移动性等特点，智能性能够使得移动Agent进行自主运行，移动性则保证代理能够在动态变化的异构环境中操作运行。因此，网格中引入移动Agent可以优化网格中资源发现过程。

2.5 移动 Agent 及相关技术

2.5.1 移动 Agent 简介

我们将移动Agent[34]简明地定义为：包含代码、数据以及执行语言的软件包，它可以在执行过程中，有目的、自主的在网络中移动，利用分布资源的局部交换而完成分布任务的软件实体。

从系统的角度来看移动 Agent 的结构，一个移动 Agent 应当包括代码、状态和属性三个基本部分。

(1) 代码。代码用来实现移动 Agent 的各项功能。为了使移动 Agent 的代码可以在它能到达的任何主机上以相同的方式执行，一般采用直接解释型或者编译成中间代码的基于解释型的语言编写。

(2) 状态。为了保证运行的连续性，移动 Agent 必须保存移动之前的一些状态信息，例如 Agent 的一些全局变量代码的重入等，这样移动 Agent 在移动到另一个主机后可以复位到移动状态，重新开始工作。

(3) 属性。属性描述了移动 Agent 本身的一些特性，包括 Agent 标识符、所有者、

起始点地址以及起始时间等。此外，移动 Agent 还必须带有认证信息，认证信息用来作为确认用户的凭证。

移动Agent主要的特性包括：

(1) 自主性。自主性是Agent最基本的特性，指行动上的独立性。Agent一旦被初始化以后便独立执行，无需后来的直接干预，它控制着自身的内部状态和外部行为，也可以授权去做出某种决定，完成一些重要的事情。

(2) 反应性。是指Agent能感知和作用于其所处的环境从而对环境的变化做出及时的响应。这些环境可以是物理的世界、使用图形接口的用户、其他Agent集合或者所有这些的组合。

(3) 能动性。为了达到目标，Agent不是等着接收指令，而是事先有计划并做一些初始化。Agent能探测到适合用户的有利场景，通知用户这个场景出现的时机。也就是说，Agent不仅能对所处环境做出响应，还能主动地展现面向目标的行为。

(4) 通信性。通信性是指Agent之间的交互。Agent之间的接口和联系不是固定不变的，而是随着任务驱动者的改变而改变。

(5) 移动性。移动性是移动Agent最重要的特性之一，它是指Agent可以在一个网络上随时、随地、自主的从一台主机迁移到另一台主机。正在运行中的Agent状态可以被存储且传送到新的主机上，在那里Agent程序恢复且继续从暂停的地方开始执行。

2.5.2 移动 Agent 的体系结构

不同的移动系统的结构不同，但是所有的移动Agent系统都由移动Agent和移动Agent服务器两部分组成。移动Agent服务器基于Agent传输协议ATP(Agent Transfer Protocol)来实现Agent在主机间的迁移，并为其分配执行环境和服务接口。Agent在服务器中执行，通过Agent通信语言ACL（Agent Communication Language）相互通信并访问移动Agent服务器提供的服务。移动Agent的体系结构定义为下图：

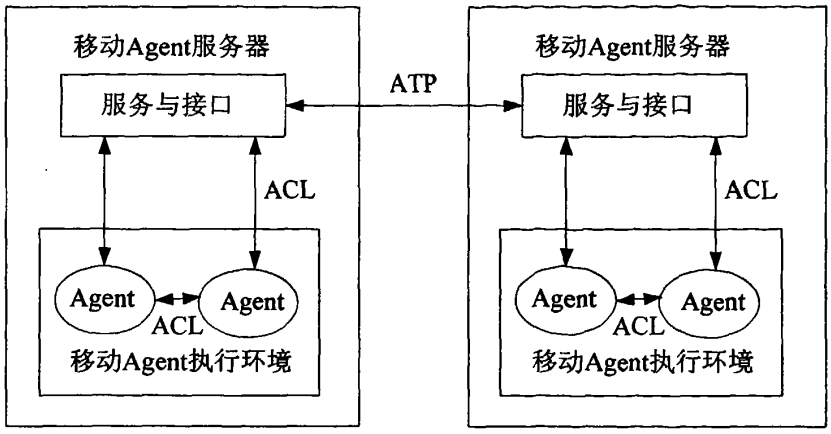

图2-5 移动Agent系统结构图

Fig2-5 Structure of the mobile agent system

移动Agent执行环境(MAE)也被称为移动Agent服务器或移动Agent平台，是一个分布在网络中各种计算设备上的软件系统，一般建立在操作系统之上，为发送、接收、恢复、安全管理和服务调用等提供基础服务设施，利用底层提供的服务为运行于其上的移动Agent提供安全、通畅的运行环境。

移动Agent执行环境(MAE)的模块化结构模型，如图2-6所示。具体来说，由Agent管理器、安全服务、通信服务、移动服务、事件服务、可靠性服务和资源管理模块组成。

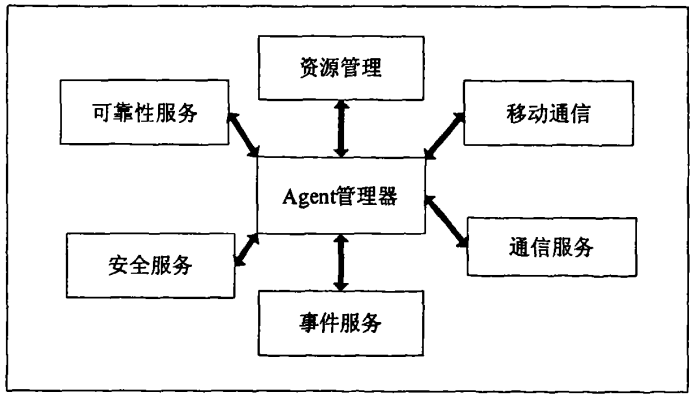

图2-6 MAE的体系结构

Fig2-6 Architecture of the MAE system

Agent管理器。它是整个MAE中最重要的部分，为Agent提供生命周期服务、目录服务，管理服务器中所有Agent的创建、挂起、激活、删除，负责记录Agent的运行状态，回答有关系统状态的查询，协调各模块之间的关系，使Agent可以在权限许可范围内充分利用服务器所提供的服务和资源，从而顺利完成各自的任务。

资源管理模块。资源是主机提供的各种服务和信息的总和，包括各种硬件、软件和数据。Agent的移动正是为了有效使用这些资源，资源管理模块负责安全可靠地管理所有资源。

安全服务模块。提供了对MAE及主机的多重保护，通过采取加密、数字签名、安全认证等安全措施，确保它们不受到非法操作，保证安全的运行环境。通信服务模块负责MAE与MA的通信以及移动Agent之间的通信。

移动通信模块负责发送和接收移动Agent的全过程。按照Agent传输协议，两个主机的Agent传输模块在传输前彼此联络，在确认符合接收方安全规范的情况下，建立起一条Agent传输通道，正式启动Agent移动过程，并在移动完成之后再次确认，如不符合安全要求，则放弃本次传输。

可靠性服务模块。负责维护Agent的状态信息以及与之相关的移动Agent系统的状态记录，从而保障Agent免受诸如系统崩溃等严重事件所带来的破坏。

事件服务模块提供了灵活、有力的可以配置的基本能力，负责管理发送给Agent或从Agent发出的注册和通知等事件，从而在相互不了解的Agent之间建立起一条宽松耦合的通信通道。基本的事件服务包括Agent对外界的感知和对外界环境的响应，也包括与其他Agent的联系与协作过程的管理。

MA必须包括任务体、属性(描绘其名字、创建者、来源等信息)和行程表(将要访问的站点列表)以完成用户指定的任务，需要实现移动语义(即目标主机或路由的选择)，安全控制与外界的通信(包括与MAE以及其他Agent的通信)等功能，还可以访问MAE提供的服务。与MAE相似，依据其所应实现的功能提出其模块化结构，如图2-7所示

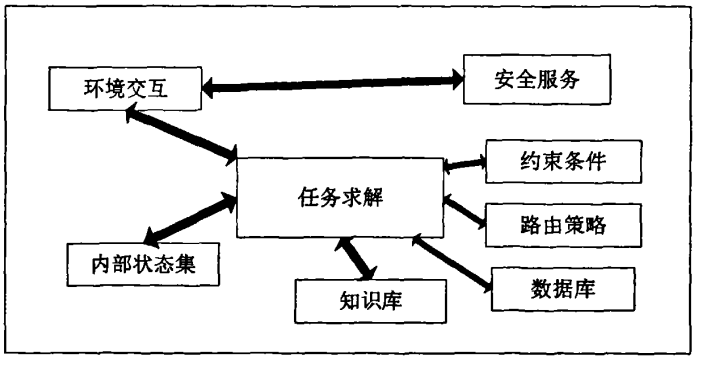

图2-7　MA的体系结构

Fig 2-7　Architecture of the MA system

具体来说由安全服务模块、任务求解模块、环境交互模块、知识库、内部状态集、

约束条件和路由策略构成，模块化结构可以尽可能的减少移动过程中的数据传输量，提高移动Agent系统设计的灵活性。数据库保存移动Agent运行时所需要的数据、中间结果以及由其采集处理并将要发送回用户的数据。

知识库是Agent所感知的世界和自身的模型，并保存移动过程中获取的知识和任务求解结果。

内部状态集记录Agent执行过程中的当前状态，影响Agent的任务求解过程。约束条件是Agent创建者为保证Agent的行为和性能而设置的约束参数的集合，如返回时间、站点停留时间及任务完成程度等。

路由策略决定Agent的迁移路径，即目标主机的有序列表，可以是静态的服务设施列表，但它只适用于简单、明确的任务求解过程，也可能是基于规则的动态路由策略，以满足复杂和非确定性多任务的求解。

安全服务模块执行安全策略、提供安全保障，阻止外界环境(包括其他Agent，恶意主机和恶意MAE)对Agent内部数据的非法访问。环境交互模块负责移动Agent与MAE之间的通信。

任务求解模块包括Agent的运行模块，以及与Agent任务相关的推理方法和规则，负责执行Agent携带的任务，根据知识库中存储的相关知识和移动中得到的中间结果，以及内部状态集中存储的当前状态进行求解，在求解过程中还要满足移动Agent创建者为其指定的约束条件，若任务不能在本地完成，则根据路由策略通过环境交互模块向其他MAE发出迁移请求。

2.5.3 移动 Agent 在网格环境中的应用

路由节点Agent存在于网络中的每个路由节点，负责发现相邻节点、对节点路由信息的初始化、与路由移动Agent进行交互、监听路由更新信息并完成对节点路由信息的更新[35]。

路由Agent的最终作用是为了更好的转发用户数据，所以路由Agent目的节点的选择应该依据节点用户数据的流向决定。

路由Agent在路由节点中移动，收集网络中的路由信息。路由Agent在节点通过节点Agent得到路由信息，根据下面将要提出的路径优化算法进行移动计算，并保存节点信息，然后根据计算结果决定一个移动的目标，直至目的节点，在目的节点，路由Agent将行程中收集到的行程数据发送给途中经过的各个节点，供更新节点路由信息之用。在

各节点进行的计算是移动Agent自主进行的，根据不同的读入节点信息作出不同的决定，这是对周围环境的一种反应。节点路由信息的更新是一种间接方式的路由Agent之间的通信，体现了移动Agent之间的协调性。

2.5.4 移动 Agent 在网格环境中的移动策略

路由算法主要有以下几种：

(1) 启发性迁移：Agent根据本地节点的连接情况，判断邻点状态、网络延迟，再决定迁移方向。$p_{ij}=\frac{(d_{ij})^{-\alpha}}{\sum_{j\in neighbora(i)}(dij)^{-\alpha}}\times\alpha>0$其中，$d_{ij}$为链路的延迟。显然，$\alpha$越大，则Agent越倾向于选择快速的探索路径。

(2) 贪婪性搜索[36]：Agent总是尽量选择新的或从未到达的节点，因此在网络性能较好的情况，Agent总能快速的遍历。Agent所选邻点j_{old}满足：

lastTime(Ma，j_{old})=min{lastTime(Ma，j)|$j\in$neighbor(i)}，其中Ma为Agent的近期轨迹记忆，当$j\notin$neighbor(i)时lastTime(Ma，j)<0。

此处neighbor(i)为已经走过的邻接点，$j\notin$neighbor(i)时lastTime(Ma，j)<0表示Agent从未走过此节点，所以更倾向于选择这个节点。

(3) 遗传算法GA(Genetic Algorithm)[37]：移动Agent通过携带服务环境的信息区的部分信息和服务节点交互的过程中仿照生物界进化，保留有效的服务资源信息，删除过时的、错误的服务资源信息，从而使移动Agent在网格中快速找到所需的服务资源。在此过程中，有效的服务资源信息会有更强的适应性，即这种信息会在网格中保留更长的时间。

(4) 综合策略:将以上若干策略折衷，进行综合判断。

2.5.5 移动 Agent 在网格环境中的优点

在网格计算中引用移动Agent技术将会有以下优点：

(1) 在地域上分布的异构网格计算环境中能自主地将计算任务从一节点迁移到另一节点，并可与其它Agent或资源交互以实现作业和资源的管理和自适应。

(2) 移动Agent通过在局域网服务器之间双向移动来传递对应的资源信息、负载信息、通讯量和任务执行序列等信息。这些信息作为资源管理、负载平衡、通讯调整、任

务调度等的参考依据，移动Agent根据这些数据智能的判断管理域的情况并做出相应处理。这将大大改善系统的性能和智能水平，提高网格计算的可靠性和执行效率。

(3) 在网格计算中，移动Agent不需要统一的调度。由用户创建的Agent可以异步在不同计算节点运行，等任务完成再将结果传送给用户。同一用户或同一计算节点可创建多种Agent，同时在一个或多个节点运行，形成并行求解的能力。

(4) 移动Agent使网格具有更多的弹性以实现安全[38]。一旦资源遇到掉电等突发性灾难被迫离开网格，运行节点还没有将信息交给网格管理机构，这时就没有办法恢复运行现场。但在移动Agent处可以设置状态为"persistent"，使其保存该作业在运行过程中的副本，包括运行程序、输入数据、描述信息等。

2.6 本章小结

网格资源发现是网格的支撑技术之一，是影响网格计算成功与否的关键因素之一。因此，资源发现的研究对网格而言具有重要意义。

本章首先对网格进行了综述，包括网格的定义、体系结构、网格服务、网格技术与发展趋势等。

其次，对网格环境下国内外已有的资源发现模型做了详尽的介绍。

再次，陈述了移动Agent的相关内容，包括移动Agent的概念、特点、应用、发展现状，分析了移动Agent系统分层及模块化结构及其实现的关键技术。

最后，本章介绍了几种传统移动策略以及在网格环境下利用移动Agent技术的优点。

第三章　一种新的网格资源发现机制

效率问题非常重要，尤其对于网格，那么如何能让人们方便快捷的找到网格资源呢？

通过前面章节的讨论可以得出结论:移动Agent能够自行选择运行地点和时机，根据具体情况，中断当前自身的执行，移动至另一设备上恢复执行。基于移动Agent的资源发现机制具有灵活性、高效性和可扩充性，可适应网格的各种基本特征，在实际应用中有较好的执行效果[39]。本文将分布式与集中式发现相结合，对本地网格资源进行快速资源发现并改进移动Agent在网格资源发现中的移动路径算法，开发更为有效适用的网格资源发现机制，使之具有良好的扩展性和动态适应性，大大提高网格资源发现的效率。

3.1　新模型总体设计

本文提出新的网格资源发现模型分为两层：资源层（网格层），虚拟组织管理层(P2P层）。网格环境下资源发现模型的框架结构见图3-1。

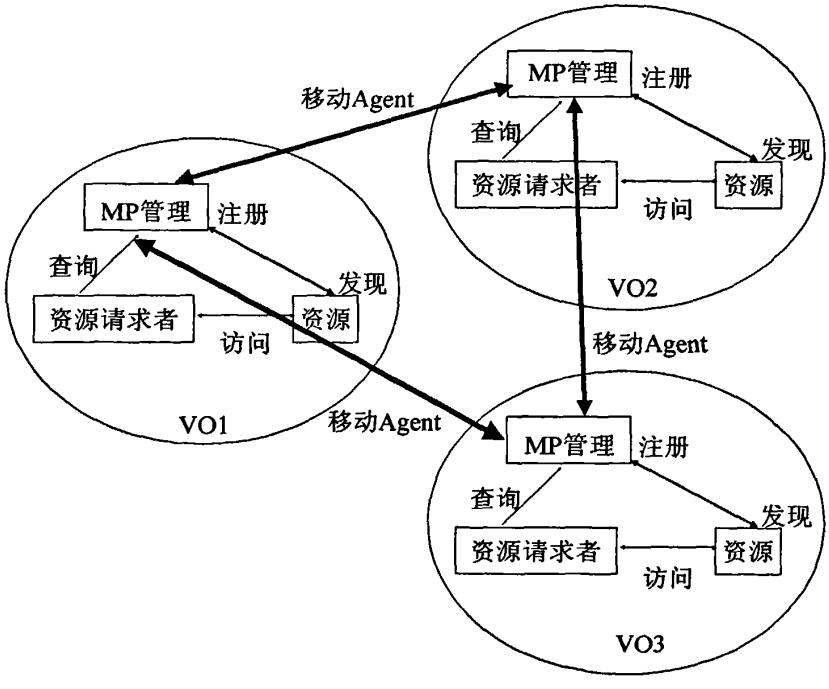

图3-1　网格资源发现模型框架

Fig 3-1　Grid resource discovery model framework

文献[40]等提出网格资源信息组织分域管理模型，本模型是按资源属性分类，资源层由资源提供者和资源请求者按照相似属性组成多个虚拟组织[41]VO(virtual organizations)，但都包含一个资源目录。在网格环境中，组成VO的资源可以理解为地理位置分布、逻辑位置相邻、具有共同需求和属性的相似资源的集合。VO内的系统行为、管理策略是一致的，每个VO内需要维护一个VO控制节点MP（Manage Peer），资源目录、VO内资源发现者等实体都运行在这个节点上。管理层上由多个MP节点组成，每个MP对应一个虚拟组织，MP之间可以互相通信，并且各MP之间构成一个纯P2P网络，完成全局范围内的资源发现。在MP之间，需要基于资源的属性值进行查找，很难将具有多种可变属性的资源形式化为一个名字或者ID，因此采用无结构的P2P结构形式。当移动Agent在网格环境中巡游时，如果所有节点均未有变动，则这是一个旅行Agent问题(Traveling Agent Problem,TAP)，但网格环境是个动态变化的，我们还必须考虑其动态特性。

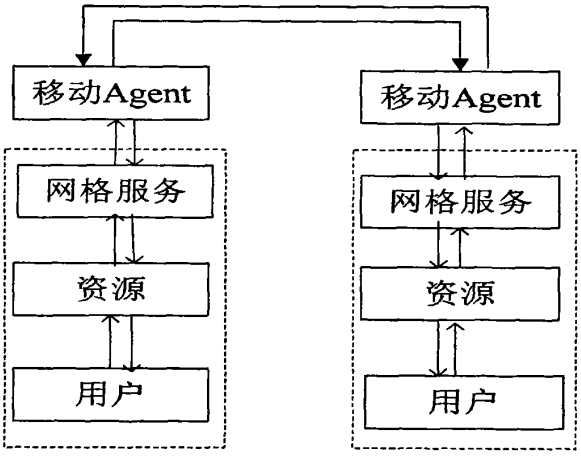

图3-2　网格资源发现模型逻辑结构

Fig 3-2　Logical structure of the grid resource discovery model

3.2　新模型资源组织形式

3.2.1　虚拟组织内的资源组织形式

虚拟组织是多个节点的集合，这些节点的关系是对等的，各个资源提供者也可能成为其他资源的消费者。同时，虚拟组织是动态的，节点可随时加入和删除。

用户节点和本地资源节点设置注册资源信息表。本地计算机上的注册资源信息表存放的是最近该计算机节点访问过的资源信息，注册资源信息表主要属性如下：资源ID、

资源类型、最近访问时间、访问次数。因为表结构较简单，表项数目较少，性能有限，所以无须为本地计算机的注册资源访问信息表建立数据库，可以将注册资源访问信息表以数据结构中循环链表的形式存在于本地计算机的内存中。按最近访问时间排序，当注册资源访问信息表存满时使用近期最少使用算法LRU（Least Resently Used）将一段时间内使用最少的资源信息表项替换出去。这样做可以最大程度的减少访问MP节点的次数，因而，最大程度的降低了网络开销。

虚拟组织服务系统采用GlobusToolkit4.0中的信息服务组件MDS4(Monitoring Discovery Service 4)，在虚拟组织中心节点，即MP节点部署MDS-Index，本地注册资源信息表收集来自信息源的所有资源信息，虚拟组织的MP节点聚合来自本地的资源信息表的信息提供给上层应用。MP集中动态信息存储只保存一个描述信息，真正的信息分散存储在不同的地方，以降低更新信息给MP节点带来的压力。安装MDS4的Globus容器中，有一个默认索引服务实例。运行在这个容器中的任何GRAM、RFT组件及CAS服务，将可以向容器中的索引服务注册。MDS4可提供如下服务信息:网格计算环境存在的资源、资源的状态信息、网格应用的优化信息。

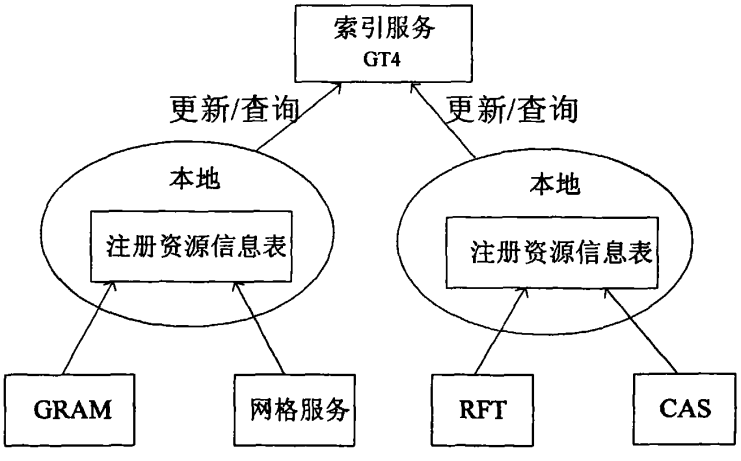

图3-3 虚拟组织内资源组织层次结构

Fig 3-3 The structure of the organization resources within virtual organization

3.2.2 虚拟组织间的资源组织形式

资源的信息在虚拟组织内和虚拟组织之间进行传递，为其他节点共享，为查询请求的处理提供支持。每个虚拟组织有一个MP服务，每个MP与多个MP相连接。MP互为邻居节点以P2P的方式彼此之间交换查询/响应消息，联系在一起。虚拟组织MP节点根据本地管理策略可以选择其他的节点作为邻居，将本地的资源属性信息传递给其他虚拟组

织共享，并将其他虚拟组织共享的资源属性保存在本地。当本地虚拟组织内的资源查询请求不能得到满足时，虚拟组织可以将送出以查找任务为载体的移动Agent，转发给其他有可能满足用户资源请求的虚拟组织MP，在全局范围内进行网格资源发现。

3.3　新模型的资源发现过程

(1) 当用户发送资源请求信息，资源请求节点先查询本地计算机上的目录，如果有满足条件的资源，直接访问最佳资源对应的信息，确认资源可访问后，更新本地所在的虚拟组织所对应的 MP 资源索引目录的信息，并对资源进行预约或分配。如果此时没有满足要求的资源信息，则向虚拟组织发送查询请求。

(2) 用户请求节点所对应的 MP 节点收到资源请求信息后，在 VO 内按照资源索引目录提供集中式资源发现，将结果返回给用户请求端。

(3) 在用户所在虚拟组织内部不能找到所需资源时，移动 Agent 作为资源查找的任务载体在 VO 之间进行分布式资源发现。

以下图 3-4 是本模型进行网格资源发现过程的流程图。

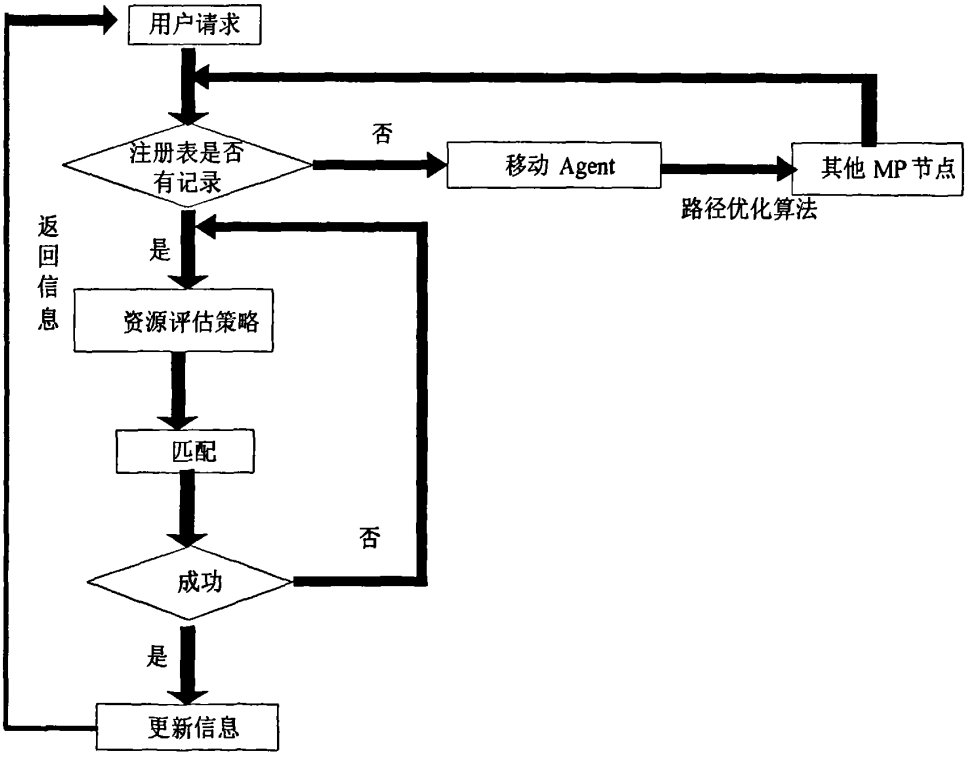

图3-4　资源发现过程的流程图

Fig 3-4　Resource discovery process flow chart

网格动态性过程:本地管理策略决定新加入节点在哪个注册资源信息表注册，然后，在MP上建立相应的索引目录，同时资源会定期向MP发送更新消息，保持联系。资源在一系列的协同工作中会结识若干邻居和好友节点，资源也会定期向这些节点发送更新消息，保持联系。若资源在规定时间没有向MP发出更新消息，MP会向该资源节点发出超时探测消息，如果节点没有响应，则认为该资源不可用，MP会注销该资源的信息:若资源节点正常退出虚拟社区，会向其邻居朋友和MP发出注销消息，其资源信息会被注销。

行为移动Agent BMA (Behavior Mobile Agent)用于代表用户或应用程序与目标环境交互完成某种行为。根据模型特点，本文利用 BMA 作为用户提交所需求资源的载体，携带用户需要的资源类型、属性等信息在网格环境中巡游。BMA 在全局系统按一定的策略巡游，途经每个节点时都会收集当前区域的全局性资源信息，并记录自己的部分信息，以及和其他 BMA 进行合作、交换知识，按照索引目录找到与用户所需资源相匹配的信息。

具体过程：在用户所在虚拟组织内部不能找到所需资源时，MP便按照时间间隔T_{creat}产生有生命周期T_{time}的BMA并激活它。资源请求者用资源描述机制描述自己的资源需求，BMA根据需求描述做出响应。另一方面，资源把描述自己的信息告诉BMA，以便使BMA知道自己的相关信息，进而把与资源匹配的请求引导到资源上。BMA在虚拟组织管理层从大量的可供选择的资源中为请求者寻找所需要的资源，把正在匹配的资源的唯一标识符和使用接口告诉请求端，并替请求端完成协商。当巡游结束找到相匹配的资源时便返回信息给相应的请求用户所在虚拟组织的管理节点MP。当BMA无法感知下一节点或者生命周期结束时便自我销毁。

为了保证网格系统的快速响应,全局各个虚拟组织之间需要一定数量的BMA在进行移动，让它在全局范围内进行网格资源发现，本文主要是针对移动Agent在网格中的移动策略进行深入的研究。

3.4 新模型开发环境的构成

为了更好的实现网格资源发现模型的基本功能，系统的开发工具和开发平台如下：

操作系统：使用稳定性较好的redhat as5.0操作系统。

网格平台：使用目前网格研究中广泛应用的网格工具Globus Toolkit4.0.1(以下简称GT4)。

移动Agent平台：IBM的Aglets2.0.2。

开发语言：采用具有平台无关性和安全性较好的Java语言。

3.4.1 Globus 开发平台

Globus运行时间的组件包提供了一系列用于建立Web服务和非Web服务的各种库和工具。MDS(Monitoring and Discovery Services)，它是信息服务组件包，被称为监测发现服务。包括一系列的组件用来监测虚拟组织中资源的运用情况。GT4中MDS称为MDS4或WS-MDS，主要完成对网格计算环境中信息的发现注册查询修改等工作，对网格计算环境中的各种资源包括数据资源、计算资源等服务和其他主体进行描述。它是网格计算环境中的信息服务中心。下面图3-5是GT4体系结构图。

我们采用GT4主要有以下优势：

安全的实现数据共享，对工具包GSI(Grid Security Infrastructure)的数据进行保护并针对不同的用户有不同的访问权限。

采用WS-Notification机制可以高效的、动态的发现网格内的可用资源进行任务分割。在弱耦合的PC之间实现共享，装有不同的操作系统、具有不同的配置、距离很远的任意的PC之间都可以建立网格环境实现共享和协同操作。

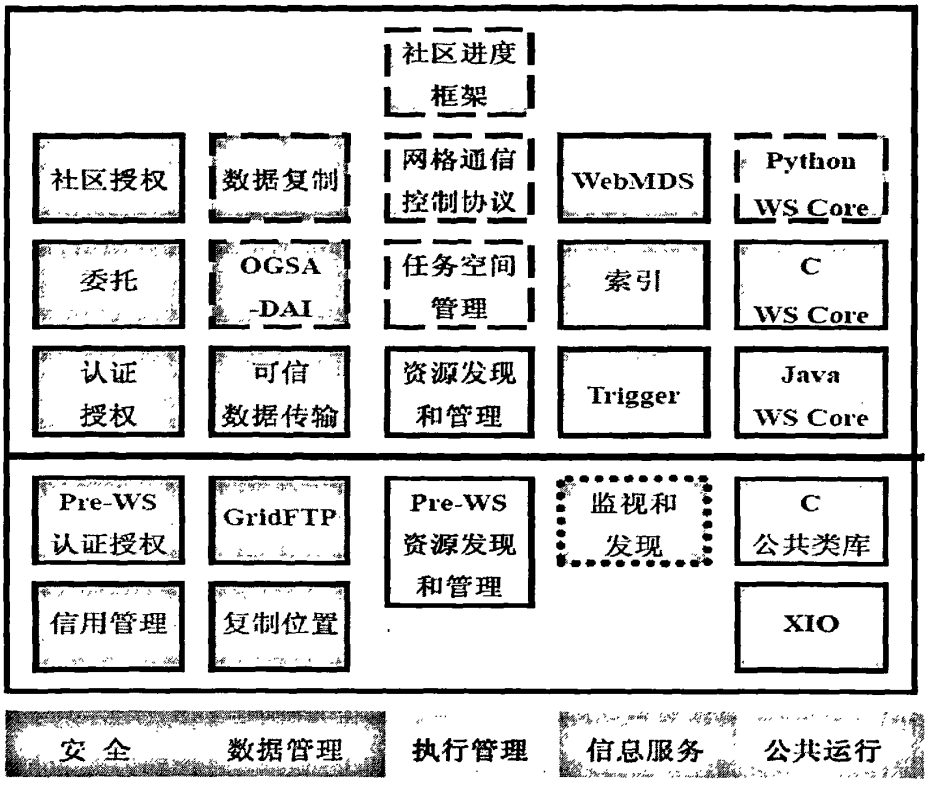

图3-5 GT4体系结构图

Fig3-5 Structure of the GT4 system

3.4.2 Aglets 开发平台

由于系统是基于移动 Agent 的，资源的发现者和管理资源发现过程的节点都必须支持移动 Agent，即安装有移动 Agent 系统。

Aglet 是由 IBM 公司用纯 Java 开发的移动 Agent 技术，并提供实用的平台 Aglet workbench，让人们开发或执行移动 Agent 系统。Aglet 这个字是由 Agent 与 applet 两个字所合成的，简单地说就是具有 Agent 行为的基于移动 Agent 的分布式资源发现研究与设计 Java applet 对象，但 Aglet 同时传送代码及其状态，而 applet 只传送代码。

Agletsl.0 不能支持 Java2，而 Aglets2.0.2 可以，并且 Aglets 在 Linux 系统和 Windows

系统下都可以很好地运行。因此，我们选用了 IBM 的 Aglets2.0.2。

Aglets系统提供一个上下文环境来管理Agent的基本行为，如:创建、复制、分派、招回、暂停、唤醒和撤销等。Aglet与Aglet之间的通信可以用消息传递的方式，Aglet并不让外界直接存取其信息，而是通过一个代理提供相应的接口，这样Aglet的所在位置会透明化，也就是Aglet要与远端的Aglet沟通时，只在本地的上下文环境中产生对应远端Aglet的代理，并与此代理沟通即可，不必直接处理网络连接与通信的问题。

Aglets为软件开发人员提供了一套基于Java语言的开发工具包ASDK(Aglets Software Development Kit)[42]。该工具包具有以下一些特点:

1. 提供了一个相对完整的移动 Agent 编程模型。
2. 为 Agent 间提供了较好的通信机制。
3. 提供了一系列系统开发的类库。
4. 提供了一套比较详细易用的安全机制。
5. 具有较好的扩展机制。

样式就是指从系统中抽象提取出来的一些具有共性的东西，以便于重用。Aglet提供各种样式，其中本文利用主从样式(master-slave)。

主从样式提供这样的机制，即允许主Agent把任务分解并分派给不同的从Agent去执行，从Agent移动到指定的目的地，完成指定任务后返回结果。主从样式主要是抽象出Agent系统中分派任务的共性，提供了一个基本方式，即在类中重用代码。

主从样式的框架代码：

```
Public abstract class Master extends Aglet{
Public void getResult();  //得到从 Agent 返回的信息
}
Public abstract class Slave extends Aglet{
Public void run(){
initializeJob();  //初始化参数
dispatch(destination);  //分派到目的地
doJob();  //在目的地执行任务，返回结果给主 Agent
dispose();  //返回结果后清除
}
}
```

该样式适用于任务的并行执行、远端执行。在网格资源发现中，本文采用的的移动Agent的算法是具有并行特性的蚁群算法。该样式主Agent创建从Agent，从Agent移动到远端机器执行任务，并把执行结果返回给主Agent。

3.5 网格服务系统

OGSA 下，所有资源都抽象为服务，资源发现机制面对的是封装成服务的资源，而不是各种各样原始的物理资源，大大简化了资源发现问题的解决，从而使得资源发现问题归于服务发现问题。

3.5.1 网格系统的建立

网格平台搭建过程中，首先要进行配置静态IP，启动telnet服务，创建Globus用户，安装GT4.0等。

在安装GT4.0之前，必须在系统中顺序安装以下三个软件：JDK1.4.2、apache-ant-1.6.0、数据库软件postgresql-8.0.3。并为这三个软件配置正确的系统环境。

开始安装 GT4.0，本文所用的 GT4.0 安装程序来自于 Globus 官方网站。首先考虑到系统的稳定性，不能直接在超级用户 root 下安装，而是建立一个 Globus 用户，在这个用户下安装 GT4.0，此外这个用户还要用来执行管理任务，如启动和终止 container、部署服务或者管理 SimpleCA 服务器，但是普通用户的权限受到限制，无法完成这些功能，因此在安装过程中会出现错误。为了解决这个问题，在安装之前，以 root 帐号创建安装目录：

/usr/local/globus-4.0.0

mkdir /usr/local/globus-4.0.0

修改 Globus 用户的权限：

chown globus:globus /usr/local/globus-4.0.0

初步安装完GT4.0以后还需要进行一些必要的配置包括对网格安全基础设施GSI的安装和主机证书用户证书的签发。GT4.0SimpleCA安装与配置创建用户：运行客户端的用户。Globus用户，用来执行管理任务，如启动和终止container，部署服务等，也可以用于管理SimpleCA服务器。安装结束后要运行服务程序必须申请并签发host证书，并将其拷贝到适当的目录下。用户也需要申请用户证书。

Globus完全安装之后，需要启动容器(container)来测试安装是否正确。如果未启动

pgSQL数据库服务器终端，将导致GT4.0在启动时无法打开数据库引擎，部分功能无法实现，当以root用户输入下列命令globus-start-container启动容器时出现错误，提示RFT (Reliable FileTransfer)功能无法实现。因此，在启动容器之前，需打开postgre终端输入命令：postmaster–i–D /usr/local/pgSQL/data启动数据库服务器。

正确启动的过程如下图3-6

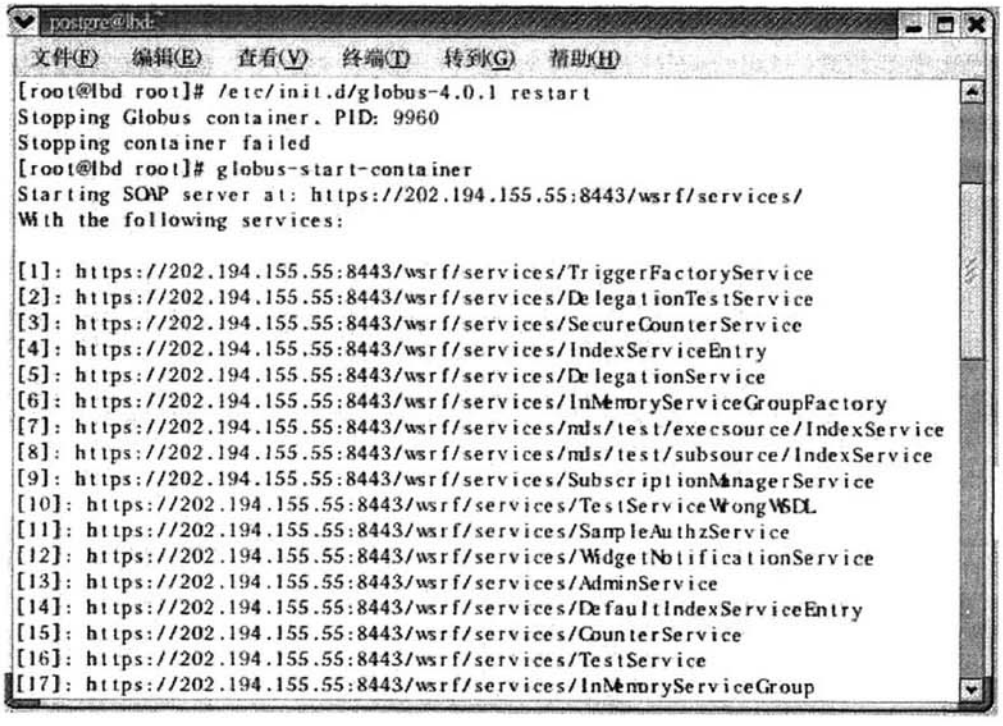

图3-6　Globus启动

Fig3-6　Start Globus

安装总结：安装过程中由于解压的文件、拷贝的文件、安装的文件的权限都有些不同，用户对不同的文件有着不同的权限，在进行某些操作时常常因为权限的问题造成无法执行。因此，我们一定要对用户和对应文件/目录的权限很清楚，针对不同的情况作不同的权限修改。

3.5.2　网格服务的生成方法

由于网格服务是定义了一组接口，这些接口的定义明确并遵守特定的惯例，用于解决服务器发现，动态服务创建、服务生命周期管理、通知等与服务生命周期有关的问题。网格服务标记语言，简称GSML (Grid Service Markup Language) 是一种基于XML的网格应用编程（描述）语言，支持网格环境中用户端的“按需”编程，通过基于可复用组件

和事件驱动的方式来描述网格应用灵活多变的处理逻辑和协同工作机制。

使用GT4.0建立的网格环境，是以网络服务资源框架WSRF(Web Service Resource Framework)为基础的。WSRF通过增加许多特征，详细描述了Web Service服务状态。

因为资源种类繁多，我们要对每一种资源进行标记，把所有资源区分开来，形成网格服务句柄GSH(Grid Service Handle)。资源在网格服务工厂被署上唯一的名字，当客户端想调用某个资源，它将联系网格服务工厂，网格服务工厂将返回给端口一个网格服务参照GSR(Grid Service Reference)，GSR中包含访问对应服务实例所需所有信息，有了一个服务实例的有效GSR，用户就可以访问相应的服务实例。

实现一个网格服务主要经历以下五个步骤：

(1) 定义服务接口使用WSDL(Web Service Describe Language)；

(2) 通过Java编写程序，实现服务；

(3) 使用WSDD(Web Service Deployment Descriptor)和JNDI(Java Naming and Directory Interface)定义部署参数；

(4) 使用ANT工具编译生成GAR文件；

(5) 使用GT4提供的工具globus-build-service来部署服务。

3.6 网格服务实例

本节利用GlobusTookit4.0和IBM Aglet等工具进行环境配置，将移动Agent作为应用程序和网格服务发现的任务的载体。移动Agent的目标是根据它从网格环境中获得的信息，确定对网格资源节点的访问顺序来实现花费最少的迁移时间，优先访问易于发现资源的节点，完成自己的查找任务。本文利用IBM Aglet中提供的主从样式予以实现。

3.6.1 服务接口定义

服务接口确定我们的服务将要向外界提供什么服务，WSDL(Web Service Description Language)是一种特殊的XML语言，它可以被用来定义web service操作。GT4使用WSDL文档来描述服务所能提供的操作，以及有状态的资源属性。这个服务接口在web service中被称为端口类型(Port Type)。

我们用WSDL描述服务、定义所必需的类型、定义匹配操作。WSDL比应用Java编写接口更难，我们用它的主要原因是，虽然Java接口更容易读写，但从长远来看它比WSDL更容易出现问题。因此，用WSDL编写接口更好一些。定义网格资源匹配服务类

型模式如下：

<types>

<xsd:schema

targetNamespace=http://www.globus.org/namespaces/examples/core/MateService_instance>

xmlns:tns="http://www.globus.org/namespaces/examples/core/MateService_instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<xsd:element name="mate" type="xsd:float "/>

<xsd:element name="mateResponse">

<xsd:complexType/>

</xsd:element>

<xsd:element name="Value" type="xsd:string"/>

<xsd:element name="MathResourceProperties">

<xsd:complexType>

<xsd:sequence>

<xsd:element ref="tns:Value" minOccurs="1" maxOccurs="1"/>

</xsd:sequence>

</xsd:complexType>

</xsd:element>

</xsd:schema>

</types>

3.6.2 请求查找 Agent

请求查找 Agent (Request Search Agent)在用户端所在的管理节点上，获取要查找的信息、分析用户请求。

Agent<Request Search Agent>

Private

Service, Data Source,Result Dest,lnitProgress, MateProgress,Knowledge_base... ...

Process<>

On <Request Search Item>Do<Get search Item >//获取要请求查找项目

Process<>

On <User Request>Do<Request Analyze>//分析用户请求

Process<>

On <Request Search Info>Do<Get Search Info>//获得查找信息

Process<>

On <Request Create RSA >Do<Create RSA>//创建查找请求Agent

Process<>

On < Search RSA >Do< Search >//进行查找

Process<>

On <Search Finish>Do <get Result From RSA>//获得结果

Action <.....>

End

3.6.3 行为移动 Agent

行为移动 BMA（Behavior Mobile Agent）由用户所在的 MP 节点创建、配置、派发和管理，代表用户在提供所需网格资源节点间移动并访问其上的网格资源，进行匹配，并在远端与移动到的接受服务请求节点的网格服务 Agent（Receive Service Agent）进行交互。

Agent< Behavior Mobile Agent>

Private

Service Item ,Data Source, Result Dest, Init Progress, Service Progress, Knowledge_Base

Process<>

On <Request Search Item>Do<Get Search Item>//获得查找任务

Process<>

On <Request Create BMA >Do<Create BMA>//创建行为移动Agent

Process<>

On <Initial>Do<Initial Parameter>//初始化参数

Process<>

On <Request Dispatch>Do<Dispatch>//路径迁移

Process<>

On <Request Mate From RRA >Do<Mate>//向接受服务Agent请求匹配

Process<>

On <Mate Finish>Do<Finish Mate>//结束匹配

Process<>

On <Save Result>Do<Save Result>//保存结果

Process<>

On <Request Return>Do<Return With Result>//找到服务后返回用户发送节点

Process<>

On <Dispose> Do<Dispose>//清除信息

Action<.....>

End

3.6.4　接受请求 Agent

接受请求Agent（Receive Request Agent）获得需求信息后，管理节点MP在本地进行快速资源查找，获得请求信息中所需的资源信息进行匹配。

Agent< Receive Request Agent>

Private

Service Item,Data Source,Result Dest,Init Progress,Service Progress,Knowledge_Base

Process<>

On <Search Request>Do<Request Analyze>//对请求分析

Process<>

On < Process Case Search>Do<Case Search>//执行本地查找

Process<>

On < Resources Data>Do<Get Resources Data>//获取资源数据

Process<>

On < Initial> Do< Initial Parameter >//初始化参数

Process<>

On <Request Mate From BMA >Do<Mate>//进行匹配

Process<>

On < Mate Finish>Do<Finish Mate>//结束匹配

Action <.....>

End

3.7 本章小结

本文提出了一种基于 Agent 技术的网格资源发现机制，该模型结合 P2P 技术，在资源组织形式上将集中式和分布式相结合，在本地和虚拟组织中均有注册资源信息表，进行快速资源发现，并介绍了网格系统和服务生成过程，利用 GlobusTookit4.0 和 IBM Aglet 等工具生成网格服务，对于网格服务接口的类型进行了定义，并对不同类型的 Agent 行为进行定义。

第四章 新模型中路径优化算法的改进

4.1 网格资源发现中的 TAP 问题

在网格资源发现中移动Agent的路由规划问题，即怎样在所有可能的或必须访问的网格节点中选择最有利的访问顺序和路径是移动Agent要解决的问题。如果P2P层的节点不加入和删除，这种如何决定在多个固定目标节点中移动顺序以获得最高效率的问题类似旅行商问题TSP(Traveling Salesman Problem)，但是移动Agent在网格中的访问和迁移又有其特殊性。

(1) 提供网格资源的网格节点所提供的资源类型不同，网格环境具有异构性的特征。

(2) 移动Agent不需要遍历所有的网格节点，只要完成查找所需资源的任务即可。

(3) 网格环境的动态性很强，网格节点的数目是动态变化的，网络负载也经常出现波动，网络传输速度不同。

(4) 网格资源的数量十分巨大，进行有效的资源发现需要有一定的经验积累。

所以，在网格资源发现中移动Agent出发前，是不能为移动Agent规划迁移路径的，即移动Agent采用的路由选择策略不能是固定路由策略，而是采用可变路由或动态路由，即在选择下一个要访问的节点时，根据网络负载情况和提供所需资源节点的匹配状况来决定。

我们在此称其为网格资源发现中的旅行Agent问题TAP（Traveling Agent Problem）。

4.2 蚁群算法研究

针对移动 Agent 在分布式 P2P 网格环境中的路径查找方面的算法，以往，主要是利用各种启发式算法[43]，对路径进行优化。在解决分布式路径优化策略中，意大利学者M. Dorigo[44]等人首先提出蚁群算法解决旅行商问题。

4.2.1 蚁群算法描述

蚁群算法[45]是模拟自然界蚂蚁寻找食物的行为方式而提出的一种算法。根据蚁群算法原理，某条路径越短，经过的蚂蚁数越多，则在该路径上留下的信息素也越多，说明沿着该路径寻找到食物的概率也越大，因此其他蚂蚁选择该路径的可能性也越大。参考蚁群算法原理，系统对Agent进行调配时，Agent k从节点i移动到节点j上的转移概率函数为:

$$p_{ij}(k)=\begin{cases}\dfrac{[\tau(i,j)]^{\alpha}\times[\eta(i,j)]^{\beta}}{\sum_{u\in ADJ_k(i)}[\tau(i,u)]^{\alpha}\times[\eta(i,u)]^{\beta}} & j\in ADJ_K(i)\\ 0 & otherwise\end{cases} \tag{4-1}$$

其中$p_{ij}(k)$表示Agent k从节点i转移到节点j的概率，$ADJ_k(i)$表示Agent从节点i处经过一步转移可以到达的下一个位置的集合，在网络拓扑结构中也就是i的邻居节点，$\tau(i,j)$表示路径（i,j）上的信息素，$\eta(i,j)$为一个启发因子。α和β是用于控制信息素与启发因子对于Agent转移概率的影响的相对重要性的参数。

对路径上的信息量作动态调整，缩小最好和最差路径上的信息量的差距，并适当加大随机选择的概率，以利于对解空间更完全地搜索。算法按下式的选择规则来确定蚂蚁k由节点i转移到下一节点j

$$j=\begin{cases}\arg\max_{u\in allowed_k}\left\{\tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)\right\} & \text{如果}r\le p(t)\\ \text{否则就依照4-1式}\end{cases} \tag{4-2}$$

路径上的外激素会随着时间的推移而挥发，并且当所有蚂蚁都返回初始主机之后会对各自走过的路径激素进行外激素浓度更新，更新的幅度与各自的总时间长度成反比。

全局路径信息素动态调整规则如下：

$$\tau_{ij}(t+n)=(1-\rho).\tau_{ij}(t)+\rho.\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t) \tag{4-3}$$

用参数1-ρ表示信息消逝程度，$\rho\subset[0,1]$其中$\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t)$表示本次循环中路径$i$，$j$上的信息素数量的增量。$\Delta\tau_{ij}^{k}(t)$表示第$k$只蚂蚁作为本次循环中留在路径$i,j$上的信息量。

我们采用蚁圈模型[46]，进行调整：

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}Q/L_k & \text{第}k\text{只蚂蚁在本次循环中经过}i,\ j\\ 0 & \text{否则}\end{cases} \tag{4-4}$$

Q为信息素强度，L_k表示第k只蚂蚁在本次循环中所走的路径总长度。

蚁群算法具有很强的发现较好解的能力[47]。这是因为该算法不仅利用了正反馈原理，在一定程度上可以加快进化过程，而且是一种本质并行的算法，个体之间不断进行

信息交流和传递，有利于发现较好解。单个个体易收敛于局部最优，多个个体通过合作，很快收敛于解空间的某一子集，有利于对解空间的进一步搜索，从而不容易陷入局部最优，有利于发现较好解。

但是，这种算法初期信息素匮乏，求解速度慢，存在搜索时间较长的缺陷。这是因为在进化的初级阶段，各个路径上信息量相差不明显。通过信息的正反馈，使得较好路径信息量逐渐增大，经过较长的一段时间，才能使得较好路径上的信息量明显高于其他路径上的信息量，随着这一过程的进行，差别越来越明显，从而最终收敛于较好的路径，这一过程一般需要较长的时间。并且，该算法容易出现停滞现象（Stagnation Behavior），即搜索到一定程度后，所有个体所发现的解完全一致，不能对解空间进行进一步的搜索，不利于发现更好的解。

4.2.2　已有蚁群优化算法

1996年，Gambardella和Dorigo提出一种修正的蚁群算法，增加了局部路径信息素更新规则。德国学者Holerhoos和Thomastuzle提出的一种最大最小系统，在一定程度上克服了蚁群算法存在的收敛速度慢、容易出现停滞现象。

1999年，吴庆洪等人提出了具有变异特征的蚁群算法。姜彤艳[48]等人针对网格环境提出了对蚁群算法的改进，主要是针对网格环境的节点动态加入和删除的特点，对选路规则、激素更新规则、动态规则和初始化规则等方面进行了改进。但是总体来说蚁群算法缺点是:初期信息素匮乏，求解速度慢，易陷入局部最优。

遗传算法[49]GA是Holland于1975年受生物进化论的启发而提出的，在移动Agent和资源节点交互的过程中仿照生物界进化，保留有效的资源信息，删除过时的、错误的资源信息，资源信息在网格资源发现系统中呈现出“适者生存”的现象。但是缺点是：对于系统中的反馈信息利用不够，当求解到一定范围时往往做大量无为的冗余迭代，求精确解效率低。

因此，在网格环境中，如果将遗传算法和蚂蚁算法相融合应用于移动Agent的路径优化算法中，可以克服各自的缺陷，汲取两种算法的优点，优势互补，在时间效率上优于蚁群算法，在求精解效率上优于遗传算法。

将以上两种算法相融合的方式[50][51]有很多，意大利学者Fabio，Abbattista等人受遗传算法的启发提出蚁群算法和遗传法相结合的思路，对参数的初值进行优化，并在TSP问题中进行验证。文献[52]将遗传算法加入到蚁群系统的每一次迭代过程中，取全局最优

解。文献[53]采用遗传算法对蚁群算法的参数进行优化，并在TSP问题中进行验证。文献[54]先用MMAS（Max Min Ant System）改进传统蚁群算法的寻径行为，然后用遗传算法对蚁群算法得到的有效路径优化。但是，网格是动态变化的，与TAP问题是有区别的，在蚁群算法每一次迭代加入遗传算法开销太大，而且每次迭代如果求的不是最精确解，误差会越来越大，则优化算法参数也不会取得较好结果。

本文主要是从初始信息素和改进节点动态变化规则方面考虑改进网格中的移动Agent路径优化算法。

4.2.3 蚁群算法在网格中的不足

蚁群算法在网格资源发现中存在着如下不足：

首先，在对于TAP问题的研究过程中，发现传统蚁群算法解空间绝大部分的解都无法满足用户需求，而且劣质解会干扰对最优路径的寻找，如果剔除这些影响最优路径的劣质解，不但可以缩小空间的搜索范围，而且还可以减少劣质解对最优解的干扰，进而迅速发现全局最优解。

其次，以固定的节点服务能力和主机的运行时间作为衡量节点能力的标准，不能实时反应节点完成匹配任务能力的变化。

最后，没有考虑到网格中经常有节点加入或删除的动态变化特点。真实网格环境是动态变化的，资源的动态加入、动态退出、资源故障、网络故障等问题不可避免，且无法预测，使得网格资源发现变得极其复杂。传统蚁群算法没有考虑到各种突发情况的变化。

综上所述，传统的蚁群算法在解决动态网格资源发现问题的时候，还存在很多不适应性，没有综合考虑节点间的距离、节点的资源匹配率的大小及其动态变化的特点，不能使得移动Agent尽快完成查找任务。

4.2.4 解决移动 Agent 路径迁移问题

蚁群算法解决移动Agent路径迁移问题时，在t时刻蚂蚁选路的算法见公式(4-5)，蚂蚁k(k= 1,2,...,m)在运动过程中，由一个随机变量q与参数q_0。决定是“利用”已有的经验直接选择目前最优(延迟最短，完成查找任务的概率最大)路径，以便加快算法的收敛速度，还是根据信息素浓度按公式(4-6)以一定的概率选择其他路径来探索其他的路径，以便能找到最优路径，避免出现路径的搜索停滞到目前的局部最优解的情况。

$$j=\begin{cases}\arg\max_{u\in allowed_K}\left\{[\dfrac{\tau_{ij}(t)}{t(i,j)}]^{\alpha}\left[u_j(t)\right]^{\beta}\right\} & q\le q_0\\ \text{公式}4-2 & \text{否则}\end{cases} \tag{4-5}$$

按下式的选择规则来确定蚂蚁k由城市i转移到下一城市j

$$p_{ij}^{\ k}(t)=\begin{cases}\dfrac{[\dfrac{\tau_{ij}(t)}{t(i,j)}]^{\alpha}\left[\mu_j(t)\right]^{\beta}}{\sum_{r\in allowed}[\dfrac{\tau_{ir}(t)}{t(i,r)}]^{\alpha}\left[\mu_j(t)\right]^{\beta}} & j\in allowed\\ 0 & \text{否 则}\end{cases} \tag{4-6}$$

$p_{ij}^{\ k}(t)$表示在t时刻移动Agent选择由节点i转移到节点j的概率，$u_j(t)$是前面访问过节点j的所有移动Agent对它的总体评价程度，我们称为节点信息素，$\tau_{ij}(t)$是前面经过节点i和节点j间路径的所有移动Agent对该段路径信息素的总体评价程度，$t(i,j)$是节点i和节点j间通过需要的时间，用于表示该段路径当前负载状态，*allowed*表示移动Agent下一步允许选择的节点的集合，参数α表示蚂蚁在运动过程中所积累的信息及路径信息素在蚂蚁选择路径中所起的不同作用。参数β表示节点当前节点资源匹配能力和路径当前状态对移动Agent进行路径选择影响力大小。

4.2.5　选择蚁群算法的可行性

蚁群算法是一种随机的离散化的启发式的优化方法，尤其是在蚁群寻优的过程中遇到障碍物时，不仅能避开障碍物，而且能够利用其正反馈机制(或称增强型学习系统)，自适应的重新寻找最优路径，并且其寻优过程就是蚁群移动过程中信息素的数量积聚过程。因而，无论从蚁群算法的过程，还是蚁群算法的所形成的机制和效果来看，用于处理当前网格的动态路径选择都具有很强的针对性。

移动Agent的每个节点作为蚂蚁，P2P层的每个MP节点作为城市，MP节点间的距离作为路径长度，给定节点的资源匹配率作为初始节点信息素，各蚂蚁对每个节点选中的次数进行节点信息素的调整，以遍历所有节点的路径最短为优化目标[55]。

4.3　改进蚁群算法

在进化的初级阶段，各个路径上信息量相差不明显，蚁群算法存在初期信息素匮乏，

求解速度慢，搜索时间较长的缺陷。不同类型移动Agent对节点提供的服务的满意程度的衡量标准有所不同，以往进行节点更新的蚁群算法[56]没有考虑在网格资源发现中，资源匹配率和查找服务的完成率对节点的影响。

本文提出改进蚁群算法的思想：根据已有的经验值，利用遗传算法得到较优路径的解，然后利用蚁群算法进行精确解的查找。移动Agent作为蚂蚁，利用以往的记录优先访问资源匹配率比较高的节点，进而尽快找到用户所需的资源。

4.3.1 初始化问题的改进

以往蚁群算法初始化，所有节点激素和路径激素都是一个（0,1）间的固定值，这样蚂蚁(移动Agent)开始选路就是随机的，需要较长时间才能收敛。也就是说，在网格资源发现问题中，需要较长时间才能找到用户所需的资源。

为了解决这种算法初期求解速度慢，存在搜索时间较长的问题，本文提出采用遗传算法生成初始信息素分布，来改进蚁群算法的初始化规则，从而，缩短移动Agent寻找到所需资源的时间。算法具体过程如下：

我们将每个虚拟组织MP节点信息组成可行域，以n个节点的遍历次序作为遗传算法个体的编码，适应度函数取为哈密顿圈的长度的倒数[57]。

初始化种群生成与染色体选择:利用Rand函数随机生成一定数量的十进制实数编码种群，根据适应值函数选择准备进行交配的一对染色体父串。

交叉算子[58]:采用顺序交叉方法。具体交叉过程如下:

(1) 随机在父串上选择一个交配区域，如两父串选定为：

a l= 1 2|3 4 5 6 | 7 8 9， a2=9 8 |7 6 5 4|3 2 1

(2) 将a2的交配区域加到a1前面，将al的交配区域加到a2的前面:

a1'=7 6 5 4|1 2 3 4 5 6 7 8 9， a2'=3 4 5 6|9 8 7 6 5 4 3 2 1

(3) 依次删除a1'，a2'中与交配区相同的数码，得到最终的两子串:

b1=7 6 5 4 1 2 3 8 9， b2=3 4 5 6 9 8 7 2 1

变异算子:采用逆转变异方法。所谓“逆转”，如染色体(1-2-3-4-5-6)在区间2-3和区间5-6处发生断裂，断裂片段又以反向顺序插入，于是逆转后的染色体变为(1-2-5-4-3-6)。对选定的一个串，如果通过逆转，路径信息素的适应值有提高的才能接受下来，否则逆转无效。

这样通过选择和交叉、变异就产生了下一代编码组。重复上述选择、交叉和变异过

程，直到满足一定的条件为止。

将最后一次迭代中各染色体的适应度值按正比例关系转化为蚁群优化算法中源点到各目的节点路径的初始信息素值。所以把路径信息素的初值设置为：

$$\tau_S=\tau_C+\tau_G \tag{4-7}$$

这里τ_C是一个根据具体求解问题规模给定的一个信息素常数，τ_G是遗传算求解结果转换的信息素值。

遗传算法求解结果向蚂蚁信息素值转换:我们选取遗传算法终止时种群中适应值最好的前10%个体作为遗传优化解集合。

利用遗传算法对路径信息素的初始值设定后，给定的资源节点的资源匹配率作为节点信息素的初始值，然后进行蚁群算法的精确计算。

4.3.2　节点更新算法的改进

不同类型蚂蚁（移动Agent）对节点提供的资源的满意程度的衡量标准有所不同，对于网格资源发现中的蚂蚁（BMA）而言，由于其任务是查找用户所需资源，则对节点满意程度由在该节点进行资源发现的匹配程度决定，计算公式如下：

$$\Delta u_i^k(t)=\begin{cases} mate_i^k(1-(u_i(t))) & \text{满意节点} \\ mate_i^k(1-\frac{1}{t_i})(1-(u_i(t))) & \text{不满意节点} \end{cases} \tag{4-8}$$

其中，$mate_i^k$是蚂蚁（BMA）在节点i的匹配程度，当蚂蚁访问一个节点时候，因为蚂蚁的功能是访问它查找所需要的与用户请求节点相匹配的资源，把获得的信息返回给用户，所以如果该节点与所要查找的资源匹配率高，则它对获得的资源信息就“满意”，否则就“不满意”。蚂蚁因为在主机i上因尝试进行匹配任务，而造成的时延为t_i，如果蚂蚁对本节点的匹配率不够“满意”，即本节点上所提供的资源与用户请求匹配的资源匹配率不高，则此蚂蚁在该节点处理时间便造成“浪费”，即减少后续所需此类资源的蚂蚁对节点的满意程度，作为惩罚。否则，便增加后续所需此类资源节点的“满意”程度。随着“满意”程度的不断变化，资源节点匹配率也是在不断变化的。

但是，针对不同资源需求的用户，其满意程度的评价标准不一样，如果蚂蚁根据用户需求，发现对此类节点不够“满意”，则后续需求此类资源的蚂蚁便不再访问该节点。若对此类资源节点“满意”则后续需求此类资源的蚂蚁便优先访问较优的节点，从而优

先访问用户所需资源较多的节点，可达到提高网格资源发现效率的目的。

蚂蚁对节点的匹配满意程度的更新算法如下：

$$u_i'(t)=\begin{cases}u_i(t)+\rho\Delta u_i^k(t) & \text{如果蚂蚁满意此节点}\\ u_i(t)-\rho\Delta u_i^k(t) & \text{如果蚂蚁不满意此节点}\\ 0 & \text{其它情况}\end{cases} \tag{4-9}$$

其中，$\Delta u_i^k(t)$ 表示蚂蚁在t时刻访问节点i时对它的资源匹配率的满意程度，我们用这个满意程度来更新节点i的总体满意程度，参数ρ决定每个蚂蚁对节点满意程度的影响。

4.3.3 网格资源的动态性问题

由于网格环境的动态性特征，节点可能由于意外情况变得不可访问，即从网格地图中删除一个节点，也可能有新的节点出现提供可访问的资源，这些都是动态的。把这些节点从路径中去掉或加入路径中，需要在路径优化算法中做出响应。本文针对网格的动态特征对蚁群算法的动态规则进行改进。

当一个蚂蚁(移动Agent)发现一个网格节点变得不可达时，它令当前所在节点i和这个节点j间的路径激素τ_{ij}设为0，并且该节点的节点信息素$u_j(t)$设为0，根据公式(4-9)，后序访问的移动Agent对此节点不“满意”，便优先访问其他资源匹配概率高的节点。这样该路径信息素也就迅速减少，蚂蚁便不再对该节点进行访问，从而增加访问其他较优路径的路径信息素。

当一个新的资源节点加入时，为了让其尽快加入到网格环境中，与其他网格节点得到较公平的访问概率，它到其他节点的路径激素则由周围与该节点相邻的最临近N个的节点间的路径激素的平均值决定。节点信息素也由该网格系统的资源匹配概率平均值进行设定。

4.3.4 局部路径信息素更新

由于计算机网络的动态特性，网格节点间的距离(迁移时间)是随时间动态变化的，蚂蚁(移动Agent)必须对这些变化做出及时响应。所以蚂蚁应对它走过的路径上的路径激素进行更新，使路径激素能及时反映当前路径的拥塞程度。由于蚂蚁是在走过两个节点间的路径后立即对其路径激素进行更新，而不是完成旅行后才更新所有路径激素，所以该部分叫做局部更新算法。

局部路径调整准则：$\tau_{ij}'(t)=\tau_{ij}(t)+\rho\Delta\tau_{ij}^k$ (4-10)

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{L_k l_{ij}}(1-\tau_{ij}) & \text{蚂蚁}k\text{在本次路径中经过节点}i,\ j\text{之间}\\ 0 & \text{否则}\end{cases} \tag{4-11}$$

Q为信息素强度，L_k表示第k只蚂蚁在本次循环中所走的路径总长度，l_{ij}是蚂蚁当前走过路径i,j之间的长度。$\Delta\tau_{ij}^{k}(t)$表示第k只蚂蚁作为本次循环中留在路径i,j上的信息量，路径激素增加量与节点间的当前距离成反比。同时，蚂蚁对路径激素的更新量还与它当前已经走过的路径量成反比，即蚂蚁所走过的节点路径越长，这条路径是最优路径的可能性越低。

4.3.5　算法描述

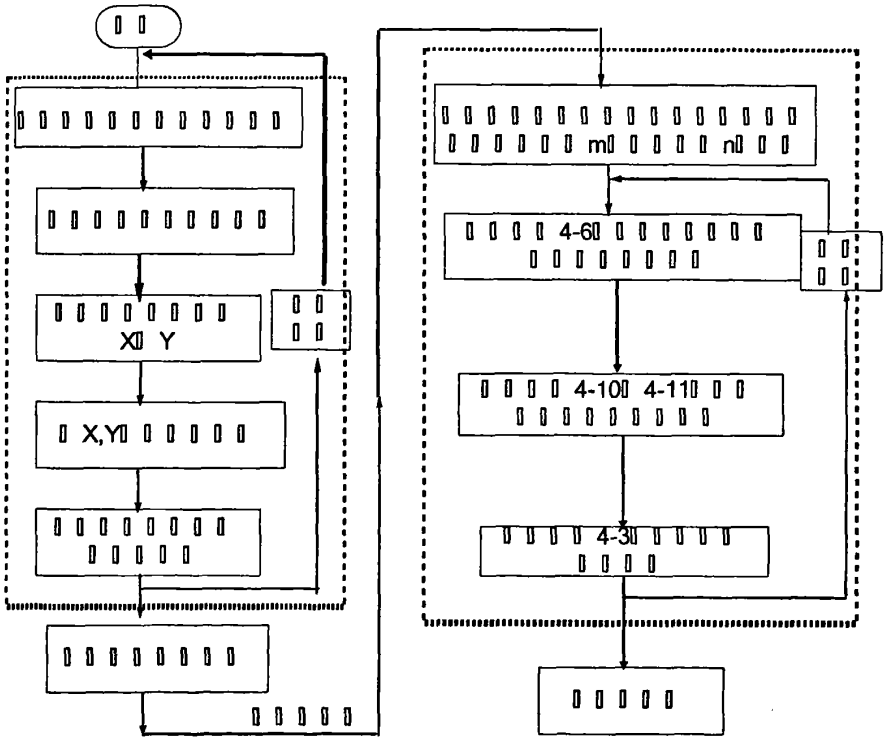

图4-1　算法流程图

Fig4-1　Algorithm flow chart

下面给出改进遗传蚁群算法的描述过程：

①初始化参数；

对参加进化的每个个体的MP节点信息抽象为编码；

设置结束条件；

随机生成初始种群P(0)，g=0；

计算P(0)中个体的适应值；

②根据个体适应值及精英选择策略确定P(g)代解群；

以概率P_c执行交叉操作；

以概率P_m执行变异操作；

计算P(g+1)中个体的适应值，采用精英策略保存最优解；

If满足算法结束条件调用蚁群算法

Else转②；

③从P(g)中选择适应能力强的前n%个体放入集合中，作为优化解集合；

利用公式4-7将遗传算法染色体的适应度转化为所有边（*i,j*）上的初始信息素；

t=0；//t是计数器

N_C=0；//N_C是循环计数器

将m只蚂蚁随机的放在n个节点上；

④*s*=1；//变量s是蚂蚁k在一次寻径过程中走的步数

For k=1 to m do

将第k只蚂蚁的初始节点放在数组$tabu_k(s)$；

$J_k(s)$={1,2,3,4…n}-$tabu_k(s)$；//$J_k(s)$是蚂蚁k在第s步时尚未经过的节点的集合

⑤s=s+1；

For k=1 to m do

根据公式4-5，4-6选择蚂蚁*k*的下一跳节点*j*；//蚂蚁*k*在t时刻处在节点*i*=$tabu_k(s-1)$

将蚂蚁k放置于节点*j*，并将节点j插入数组$tabu_k(s)$

$tabu_k$=*j*；$J_k(s)$= $J_k(s-1)$-*j*；

利用公式4-8，4-9对节点信息素进行更新；

利用公式4-10，4-11对路径进行局部信息素更新；

重复⑤直到s=n；

⑥For *k*=1to m do

将蚂蚁*k*从节点$tabu_k(n)$移到$tabu_k(1)$//蚂蚁回到初始节点，完成一次回路寻径；

计算蚂蚁k的路径长度L_k并比较其大小；

求得最短路径长度L_{best}及其对应的k_{best}和$tabu_{kbest}$；

用公式4-3，4-4对路径进行全局信息素更新；

得到任一条边的信息素浓度值$\tau_{ij}(t+n)$；

⑦$t=t+n$；

$N_C=N_C+1$；

If($N_C<N_{Cmax}$)and没有停滞行为

Then{清空所有tabu列表，转跳至③}；

Else将寻径后的最短路径长度作为适应度函数；

End for

End for

End

输出最优结果；

计算最短时间；

4.4　本章小结

本章根据网格环境的特点，研究用移动Agent作为网格中的分布式资源发现的载体，发挥移动Agent自主移动性和智能性的特点，提出能充分反映网格特点的TAP模型来描述移动Agent在网格环境中的路径问题，采用蚁群算法作为移动Agent的路径算法进行各方面的改进：

为了提高网格的资源发现效率，利用遗传算法来改变蚁群算法的初始规则。

根据用户对节点的满意程度，改进蚁群算法的节点更新规则。

针对网格动态性特征，介绍了网格的动态性更新规则和局部路径信息素更新规则。

其中，将遗传算法融合蚁群算法的初始化问题用于网格资源发现中，并改进蚁群算法的资源节点匹配更新算法，是本文的精髓所在。

最后，用伪代码完整描述了算法的全过程。

第五章　算法验证及分析

5.1　理论分析

在网格复杂环境中进行资源发现的过程中，网格资源的信息适应度越高，则在此节点匹配成功的可能性越大，适应度越低则在此节点匹配成功的可能性越小。移动Agent巡游路径上的信息素浓度越大，说明找到用户所需资源的概率越大。反之，则此路径不够最优[59]。

在分布式的网格环境中，如果采取传统的资源搜索方法，则需要在每一个虚拟组织中移动Agent每经过一个节点会向虚拟组织中k个相连节点分别进行搜索，如果移动Agent在资源发现过程中，经过r个节点，那么移动Agent需要进行$k^r \sim n$ 次搜索（n为资源节点的数目），此外，这种方法没有考虑到网格资源节点的加入和删除，即不能适应网格的动态性。

如果移动 Agent 用遗传算法和改进的蚂蚁算法在网格环境中进行资源发现，则只需选择适应度高的虚拟组织的目录节点 m，$m \sim \ln n << k^r \sim n$。然后，利用初始信息素的值按照改进的蚁群算法寻找资源，并且本文对蚁群算法的改进也充分适应了网格环境的动态性特性。

网格资源发现的任务是尽快找到用户所需求的资源，一个好的算法评价标准有很多，为了充分验证本文算法在网格资源发现中的优势，我们将从算法的有效性、能否适应网格动态性、算法的普适性方面对算法进行比较。

5.2　实验设计

目前已有的网格仿真软件，如 GridSim[60]等，基本上都是针对网格中资源调度的仿真而设计的，不适合于在大规模、动态环境下对网格资源发现中的路径优化算法的仿真。本文用于验证算法，在对已有的基本蚁群算法研究的基础上，把 TSPLIB（http://softlib.rice.edu/softlib/tsplib）地图上城市作为节点，并对每一个节点设置一个资源匹配率的初始值，设计成仿真的动态网格环境。

本实验使用的PC机配置如下：使用Pentium(R)4 CPU3.06GHz，内存512M，80G的硬盘空间，WindowsXP Professional操作系统，编程环境：VC++6.0。

在现实情况中，网格中各种类型的资源数量比例一般不均匀，呈现出Zipf特性。但

是，为了体现蚁群算法的随机性，客观评价本文方法的实验效果，我们研究虚拟组织大小相对比较均匀的情况，在P2P层各个节点是对等的，进行分布式的对等资源类型分布。这样，在相对比较平均的资源分布状态下，对改进后的算法与基本算法进行比较时更能体现算法的客观性。

在整个网格环境中节点拥有资源数量分布方面，我们研究较均匀分布的情况，参照文献[61]中的Emulated Grid，仿真设定资源总数为50000，信息节点总数为10000，资源类型数为100。对应的本地节点拥有资源种类的数量分布如图5-1所示。

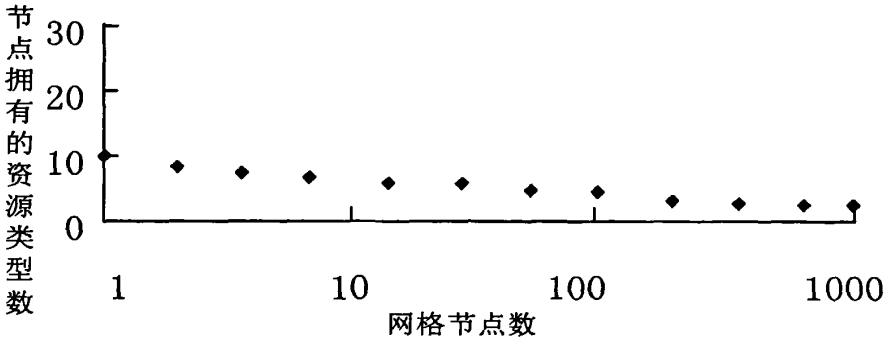

图5-1 节点拥有资源种类的数量分布图

Fig5-1 The distribution number of the nodes have resources species

参照文献[62]进行参数选取，遗传算法交叉概率P_c=0.6，变异概率P_m=0.2，最大迭代次数30，最小进化率3%，连续代数3。蚁群算法中各路径信息素初值[63]τ_C设为60，遗传算法求解结果转换的信息素值是经过的路径加2，主机数n=50，蚂蚁数m=25，q_0=0.93，N=15，Q=1000，实验20次取平均值。

我们采用改进算法分别对典型的Oliver30城市TSP问题(附录1)和Eil51问题(附录2)进行了节点信息素初始值设置，模拟网格环境进行实验。

对实验一、实验二、实验三和实验四在实验过程中模拟网格环境分配资源，给定一组资源匹配率初始值如下：

表5-1 30节点资源分布匹配情况1

Table 5-1 Resources distribution matching situation of 30 nodes 1

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
1	0.42	2	0.63	3	0.51	4	0.27	5	0.18
6	0.36	7	0.74	8	0.88	9	0.96	10	0.79

表5-1 30节点资源分布匹配情况1（续）

Table 5-1 Resources distribution matching situation of 30 nodes 1（continue）

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
11	0.74	12	0.56	13	0.41	14	0.18	15	0.37
16	0.20	17	0.14	18	0.67	19	0.76	20	0.34
21	0.47	22	0.51	23	0.33	24	0.16	25	0.25
26	0.56	27	0.61	28	0.81	29	0.75	30	0.42

表5-2 51节点资源分布匹配情况1

Table 5-2 Resources distribution matching situation of 51 nodes 1

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
1	0.82	2	0.63	3	0.71	4	0.67	5	0.58
6	0.66	7	0.54	8	0.88	9	0.56	10	0.69
11	0.64	12	0.36	13	0.41	14	0.58	15	0.37
16	0.20	17	0.54	18	0.57	19	0.66	20	0.57
21	0.13	22	0.53	23	0.53	24	0.16	25	0.25
26	0.86	27	0.41	28	0.81	29	0.58	30	0.82
31	0.86	32	0.29	33	0.67	34	0.48	35	0.89
36	0.98	37	0.51	38	0.52	39	0.35	40	0.47
41	0.91	42	0.27	43	0.18	44	0.66	45	0.59
46	0.43	47	0.38	48	0.25	49	0.72	50	0.11
51	0.67								

5.3 比较算法的有效性

5.3.1 实验方案

实验方案一：我们利用TSPLIB中典型地图环境Eil51进行设置作为Agent迁移的移动环境，在给定的一组资源匹配率初值的情况下，比较基本蚁群算法、遗传蚁群算法和改进后算法的性能。

实验方案二：同上，利用TSPLIB中Eil51作为Agent迁移的地图环境，为了使算法更具有说明性，固定迭代次数，我们进行500次迭代实验，比较其优化路径拓扑图。与文

献[64]类似，在仿真实验中，我们忽略一次资源发现过程中信息节点拓扑的动态变化，并假设所有资源信息有效。

5.3.2 实验结果及分析

实验一：在 51 个节点资源分布匹配情况 1 下，对几种算法分别运行 20 次，取算法最好结果如图 5-2。图中上面黑色实线为基本蚁群算法的实验结果，中间虚线为遗传蚁群算法的实验结果，最下面虚线为改进后的蚁群算法的实验结果。

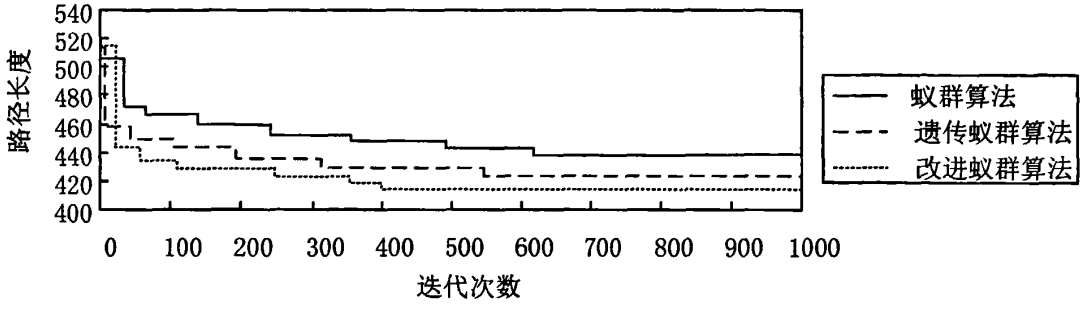

图 5-2 三种算法比较的实验结果

Fig 5-2 Compare the experimental results of 3 algorithms

实验一的结论：由图5-2通过比较可以看出本改进算法用在网格模拟环境中，在算法初期搜索速度较快，而且能够以较好的迭代次数达到最优解。而基本蚁群算法出现了停滞现象，而且搜索时间比较长，从实验结果明显看出该改进算法无论从收敛速度上还是收敛路径上都有较大的优势，这就意味着在网格环境中移动Agent可以更快的找到较短的路径完成资源发现的任务，从而提高网格资源发现的效率。

实验二:图5-3是基本蚁群算法的最优路径结果，图5-4是改进蚁群算法的最优路径结果。

实验二的结论：由结果图5-3、5-4可以看出由于移动Agent不需要遍历所有的网格节点，只要完成查找任务即可，则在改进的蚁群算法中不需要访问所有的节点，移动Agent也不一定访问距离最近的节点，而是根据已有经验访问节点资源匹配率高、节点间距离又短、完成查找任务好的节点即可。相比较基本蚁群算法的最优路径，改进蚁群算法的最优路径比较简单。

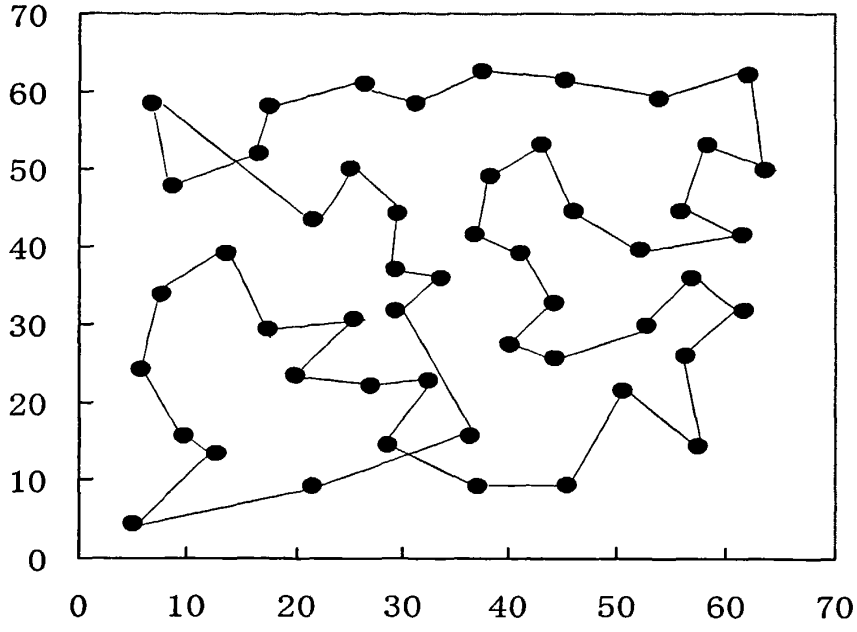

图5-3 基本蚁群算法的选路结果1

Fig 5-3 The result of basic ant colony algorithm election road

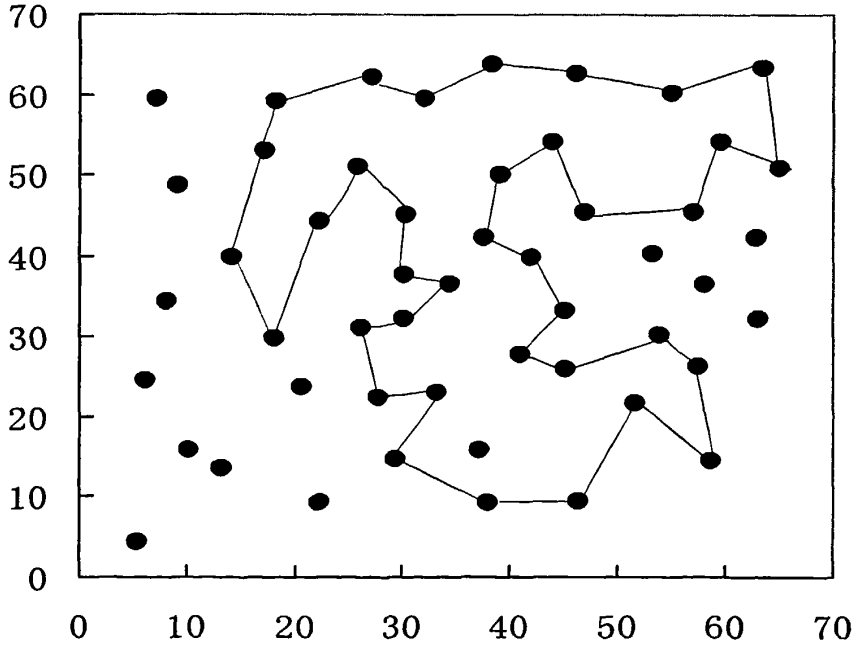

图5-4 改进蚁群算法的选路结果1

Fig 5-4 The result of improved ant colony algorithm election road

5.4 网格动态适应性

5.4.1 实验方案

实验方案三：在给定网格资源节点一组资源的匹配率初始值的情况下，在设计的动态网格资源环境中删除一个节点，固定迭代次数，用改进的蚁群算法与基本蚁群算法相比较，进行多次模拟实验，比较选路结果。

实验方案四：在给定网格资源节点一组资源的匹配率初始值的情况下，在网格模拟环境中加入一个节点，复位重新选路。基本蚁群算法、已有的遗传蚁群算法和改进的蚁群算法在设计的网格资源环境中，分别进行多次模拟实验，从最优路径长度和收敛速度方面对这些算法进行比较，找到最优路径的长度和收敛速度。

5.4.2 实验结果及分析

实验三：本文在地图TSPLIB的Eil51地图环境中删除一个节点，删除节点39(59, 15)，进行500次迭代实验，得到如下的路径结果图5-5和图5-6。图5-5是在地图TSPLIB Eil51地图环境中删除一个节点后，基本蚁群算法的最优路径结果。图5-6是改进的蚁群算法的最优路径结果。

由实验三所得结论：在基本蚁群算法中删除一个节点后，最优路径变得非常复杂。而改进的蚁群算法由于完成资源发现任务后，便不再访问其他节点，产生较优路径。

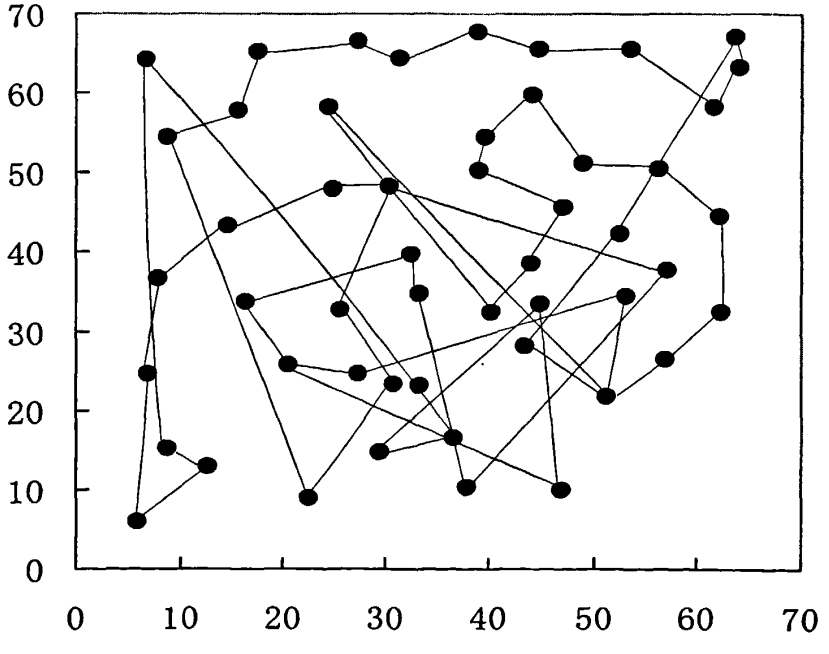

图5-5 基本蚁群算法的选路结果 2

Fig5-5 The result of basic ant colony algorithm election road

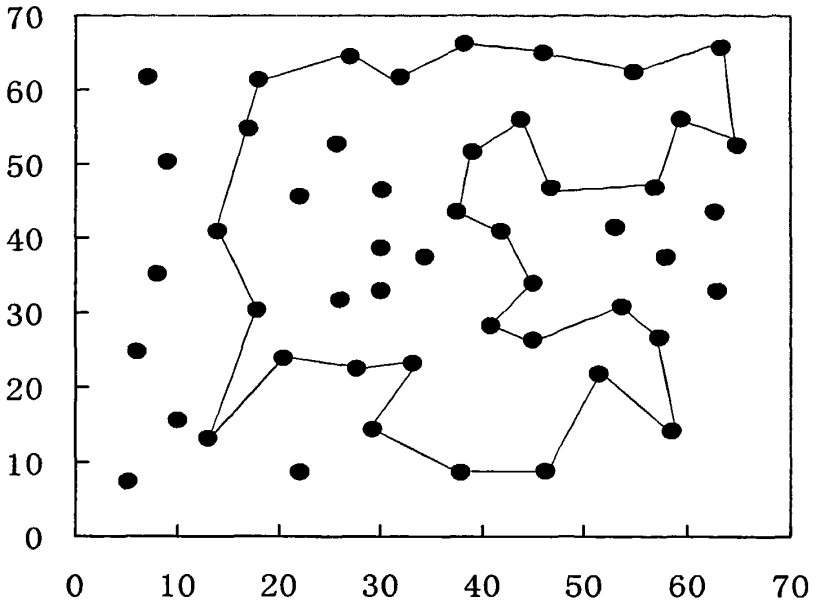

图5-6 改进蚁群算法的选路结果 2

Fig 5-6 The result of improved ant colony algorithm election road

实验四：利用Oliver30TSP地图是改进算法后获得的收敛路径和加入新节点后使路径上的激素复位重新选路，获得新的收敛路径的结果。实验进行20次取平均值。

参数α表示蚂蚁在运动过程中所积累的信息及路径信息素在蚂蚁选择路径中所起的不同作用。参数β表示节点当前资源匹配能力和路径当前状态对移动Agent进行路径选择影响力大小。参数$0<\rho<1$决定每个Agent对节点匹配满意程度的影响以及每条路径上信息素的衰减因子。

表5-3 基本蚁群算法的实验结果

Table 5-3 The experimental results of basic ant colony algorithm

α	β	ρ	最短路径长度	进化代数
2	2	0.5	424.80	350
2	2	0.9	427.01	344
1	2	0.5	423.76	342
5	2	0.9	430.50	338
5	2	0.5	445.02	347

表5-4 遗传算法与蚁群算法相融合的实验结果

Table 5-4 The experimental results of integration genetic algorithm and ant colony algorithm

α	β	ρ	最短路径长度	进化代数
1	1	0.8	426.60	30+11
1	2	0.8	424.69	30+10
2	1	0.8	424.46	30+16
2	2	0.8	423.74	30+13
2	3	0.8	424.67	30+21
3	3	0.8	425.65	30+19
3	2	0.8	425.52	30+13
5	2	0.8	424.90	30+9
5	3	0.8	426.90	30+11
3	5	0.8	429.79	30+9
5	5	0.8	430.13	30+11

表5-5 改进蚁群算法后实验结果

Table 5-5 The experimental results of improved ant colony algorithm

α	β	ρ	最短路径长度	平均进化代数
1	5	0.063	424.05	36.50
5	1	0.063	423.55	36.00
5	3	0.063	424.70	34.50
3	5	0.063	424.55	36.55

表5-6 复位重新选路的实验结果

Table 5-6 The experimental results of reset re-routing

α	β	ρ	最短路径长度	平均进化代数
1	5	0.063	422.76	38.76
5	1	0.063	423.55	39.66
5	3	0.063	424.70	38.50
3	5	0.063	424.55	37.50

表5-1是基本蚁群算法的针对Oliver30TSP[65]实验结果表5-2是传统遗传蚁群算法[66]针对Oliver30TSP的实验结果。表5-2和表5-3是改进算法后获得的收敛路径和加入新节点后使路径重新选路的实验数据。

实验四的结论：由这些实验数据我们可以得出结论：改进的蚁群算法与基本的蚁群算法和遗传蚁群算法相比，能够以较少的迭代次数，取得较短的路径。

5.5 比较算法的普适性

5.5.1 实验方案

实验方案五：随机给定几组节点资源的匹配率初值，比较在不同类型资源分布情况下，比较三种算法性能。本文利用 TSPLIB 中 Eil51 作为移动 Agent 迁移实验环境的地图。

表5-7 51节点资源分布匹配情况 2

Table 5-7 Resources distribution matching situation of 51 nodes 2

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
1	0.82	2	0.33	3	0.41	4	0.67	5	0.98
6	0.66	7	0.34	8	0.48	9	0.16	10	0.39
11	0.34	12	0.26	13	0.81	14	0.88	15	0.67
16	0.60	17	0.84	18	0.37	19	0.26	20	0.64
21	0.87	22	0.11	23	0.66	24	0.86	25	0.75
26	0.26	27	0.31	28	0.21	29	0.35	30	0.82
31	0.66	32	0.89	33	0.87	34	0.28	35	0.35
36	0.32	37	0.21	38	0.82	39	0.25	40	0.27
41	0.17	42	0.27	43	0.28	44	0.36	45	0.21
46	0.82	47	0.68	48	0.85	49	0.42	50	0.90
51	0.77								

表5-8 51节点资源分布匹配情况 3

Table 5-8 Resources distribution matching situation of 51 nodes 3

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
1	0.52	2	0.73	3	0.41	4	0.37	5	0.28

表5-8 51节点资源分布匹配情况 3（续）

Table 5-8 Resource distribution matching situation of 51 nodes 3（continue）

编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率	编号	匹配率
6	0.26	7	0.84	8	0.78	9	0.86	10	0.69
11	0.84	12	0.46	13	0.51	14	0.28	15	0.27
16	0.10	17	0.24	18	0.77	19	0.56	20	0.44
21	0.57	22	0.61	23	0.43	24	0.46	25	0.35
26	0.66	27	0.51	28	0.91	29	0.85	30	0.32
31	0.26	32	0.39	33	0.37	34	0.48	35	0.65
36	0.52	37	0.41	38	0.32	39	0.65	40	0.67
41	0.82	42	0.77	43	0.88	44	0.56	45	0.61
46	0.50	47	0.28	48	0.35	49	0.12	50	0.51
51	0.17								

5.5.2 实验结果及分析

我们对于给定的几组数据进行 20 次模拟实验，进行分析，得出如下实验结果图。

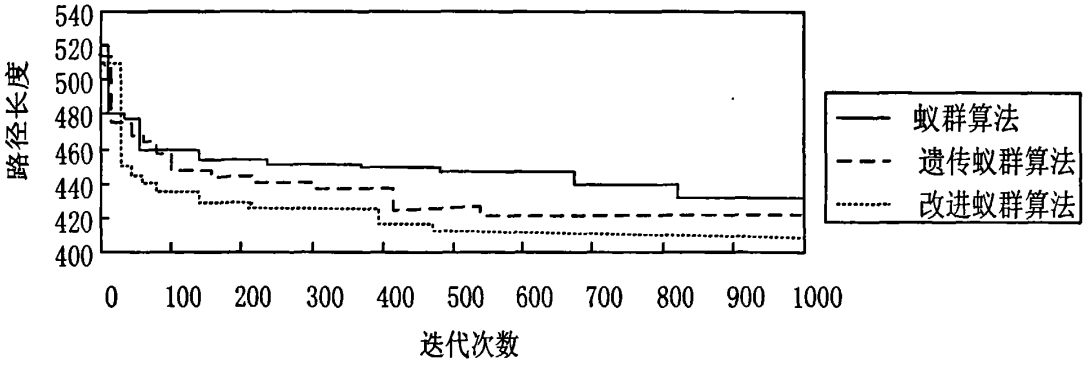

图 5-7 资源分布情况 2 下资源算法比较的实验结果

Fig 5-7 The comparison algorithm experimental results of resources distribution situation2

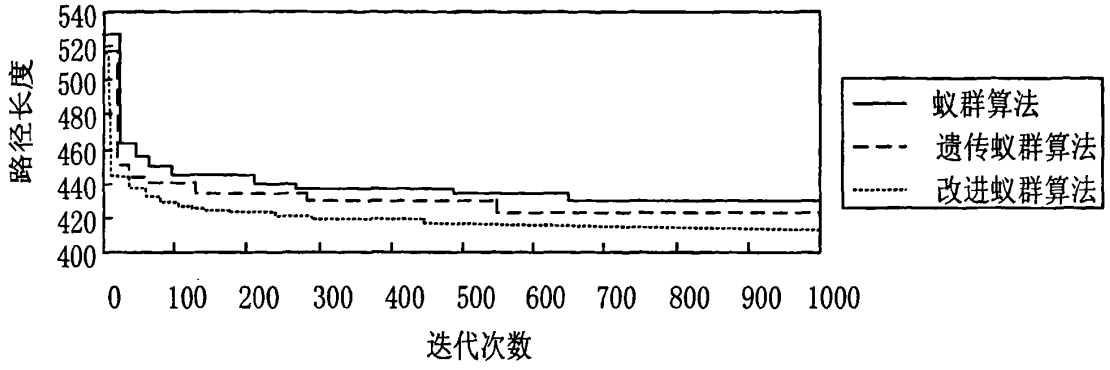

图 5-8　资源分布情况 3 下算法比较的实验结果

Fig 5-8 The comparison algorithm experimental results of resources distribution situation 3

图5-7和图5-8是验证网格资源节点在不同的资源类型分布状态下，本文所改进算法与其他算法的比较。实验五的结论：本文改进后的算法在随机给定的网格资源类型分布状态下，即：随机给定网格节点的资源匹配状态的情况下，依然保持良好的性能。

5.6　实验结论

通过以上几组实验，可以看出本文的改进算法相比较基本蚁群算法和遗传蚁群算法用在网格资源发现中有较优性能。当网格节点出现动态特征时候，相比较其他算法更加适应网格环境的动态性。并且，在选路过程中充分的考虑到了不同类型的资源分布情况，可在较短的时间内发现较好的资源，实现高效的网格资源发现效率，为目前网格资源发现机制中的移动 Agent 路径选择问题提供了一种智能化的方案。

5.7　本章小结

本章首先对改进算法进行理论分析，然后设计实验，通过对蚁群地图环境的设置，编程实现模拟网格环境，对改进后的蚁群算法分别从算法的有效性、网格的动态性以及算法的普适性方面进行了效果验证，最后对结果进行分析。

结论

本文通过对已有网格资源发现模型、移动 Agent 技术、蚁群算法及其改进策略的深入研究，在进行多次模拟实验和对大量实验结果进行分析的基础上，获得了如下结论：

(1) 网格资源的分布性、动态性和多管理域使得已有网格资源发现机制存在效率不高、不能适应网格动态性等问题，因而，传统网格资源发现机制不能满足用户需求。

(2) 将遗传算法和蚁群算法相融合的方法，用在网格资源发现中移动Agent的路径优化方面，可以克服各自的缺陷，优势互补，使移动Agent在较短时间内发现较好的路径。

(3) 改进的蚁群算法，不但考虑了网格的动态性特征，而且考虑到了节点资源匹配能力大小不同的特点。通过进行模拟实验，将各种算法在模拟实验环境中进行比较，实验结果表明：本文改进的算法是合理的，该算法能够提高网格资源发现的效率，尽快满足用户资源发现的需求。

(4) 改进后的算法具有普适性。本文随机给定节点不同资源匹配率的初始值，发现在不同类型资源分布情况下，改进后的算法都具有良好的性能。

本文创新点如下：

(1) 将遗传算法与蚁群算法相融合的思想应用于网格资源发现机制中，优化移动 Agent 的初始路径选择。

(2) 改进了蚁群算法中的节点更新算法，并设计了实验对算法进行验证。结果是改进后的算法比传统的蚁群算法和遗传蚁群算法更适合网格的动态环境，能够更好的应用于网格资源发现中。

(3) 对网格资源发现模型中的虚拟组织进行设计，提出了在虚拟组织内利用注册资源信息表的方法，实现了从本地存储到虚拟组织管理节点的快速资源发现。

进一步的工作：

本文主要是解决网格资源发现中的效率问题，对于网格资源发现其他方面，例如安全问题等由于时间和条件的限制未能作深入研究。此外，还没有完全实现本文所提出的资源发现模型的全部功能，对于算法中的参数选择问题已还有待于大量的实验研究，并且受仿真环境的限制只考虑了资源较均匀分布的情况，现实情况中，网格中各种类型的资源数量比例有些极不均匀，还应进一步的进行各种情况的研究。

由于作者水平和时间有限，文中难免出现疏漏和错误之处，恳请各位专家学者批评指正。

参考文献

[1] (美)约瑟夫、弗莱斯汀(著)，战晓苏(译). 网格计算[M]. 北京：清华大学出版社. 2005：32-82

[2] 徐志伟，冯百名，李伟. 网格计算技术[M]. 北京：电子工业出版社，2004：15-98

[3] 徐志伟，李伟. 织女星网格的体系结构研究[J]. 计算机研究与发展，2002，39(8)：923-929

[4] Czajkowski K, Fitzgerald S, Foster I, et a1.Grid Information Services for Distributed Resource Sharing[A].Proceeding of 10th IEEE International Symposium on High-performance Distributed Computing(HPDC-10)[C].Washington, DC: IEEE Computer Society Press, 2001:181-194. [EB/OL],http://www.chinagrid.net/grid/paperppt/Globus Paper/MDS- HPDC.pdf

[5] 张云勇，等. 移动Agent及其应用[M]. 北京：清华大学出版社，2002：20-109

[6] 薛宏全，廖建明，周小兵. 基于移动代理的网格资源发现与监控模型研究[J]. 计算机应用，2004，24(6)：54-57

[7] 福斯特. 网格计算[M]. 金海，等译. 第2版. 北京:电子工业出版社，2004:149-162

[8] Foster I. The anatomy of the grid: enabling scalable virtual organizations[J].International joumal of Supercomputer Applications ,2001,V15(3):6-7

[9] Foster I. Ian foster on recent changes in the grid community[J].IEEE Computer Society , 2004,V5(2):2-3

[10] 都志辉，陈渝，刘鹏，等. 网格计算[M]. 第2版. 北京:清华大学出版社，2002：22-30

[11] Foster I, Kesselman C.The physiology of the grid : An open grid services architecture for distributed systems integration.Globus,2002.[EB/OL],http://www. globus.org/research/ papers/ogsa.pdf

[12] Foster I, Kesselman C. Grid services for distributed system integrationl[J]. Computer,2002,V35 (6):37-46

[13] Tuecke S, Czajkowski K, Foster I, et al. Open Grid Services Infrastructure (OGSI) Version 1.0[EB/OL], http://www.ggf.org/ogsi-wg 2003

[14] Globus alliance.GT information services : Monitoring Discovery System (MDS) [EB/OL],http:// www.glob2us.org/mds/default.html,2005-04-30/2005-05-22

[15] Frey J , Tanenbaum T , Livny M , et al. Condor-G: A Computation Management Agent for Multi institutional Grids[J].Cluster Computing, 2002 ,V5(3):237-246

[16] Iamnitchi A . Resource discovery in large resource-sharing environments[D].Chicago: University of Chicago,2003

[17] Huang L, Wu Z, Pan Y.Virtual and dynamic hierarchical architecture for E-science grid[J].International Journal of High Performance Computing Applications,2003, V7(3):329-347

[18] Andrzejak A, Xu Z . Scalable, efficient range queries for grid information services[A]. Proc of the 2nd Int'l Conf on Per-to-Per Computing(P2P2002)[C].Los Alamitos,CA: IEEE Computer Society Press,2002:33-40

[19] 李伟，徐志伟，卜冠英. 网格环境下一种有效的资源查找方法[J]. 计算机学报，2003，26(11):1564-1549

[20] Vanthournout K, Deconinck G, Belmans R. A Taxonomy for resource discovery[A].Proc of the Int'l Conf on Architecture of Computing Systems(ARCS 2004)[C].Berlin: Springer, 2004:78-91

[21] 叶哲丽，何秀强，王寅峰. 基于P2P技术的资源发现机制的研究[J]. 计算机工程与应用，2005，4(21)：152-155

[22] Kermarrec A, Massoulie L, Ganesh A.Reliable probabilistic communication in large-scale information dissemination systems[R]. Tech Rep:Microsoft Research Cambridge, 2000

[23] Hawick K A, James H A. Modelling a Gossip Protocol for Resource Discovery in Distributed Systems[A].Proc of the Int Conf on Parallel and Distributed Processing Techniques and Applications(DHPC-102)[C].Las Vegas, USA:CSREA Press,2001:23-26

[24] 毛涛，徐云，胡自林. 基于Gossip协议的网格资源发现机制[J]. 计算机工程与应用，2005，5(14)：133-137

[25] 刘星，肖卫东，徐磊，等. 基于复合拓扑的网格资源发现机制[J]. 计算机工程与应用，2005，5(9)：132-136

[26] Ratnasamy S, Francis P, Handley M,et al.A scalable content-addressable network[A]. Proceeding of ACM SIGCOMM[C].New York,USA:ACM press, 2001:161-172

[27] 李春林，卢正鼎. 一种分层式计算网格资源定位路由协议[J]. 武汉理工大学学报. 2003，27(5)：593-597

[28] 孙雨婷. 网格环境中资源信息服务的研究与设计[D]. 济南：山东大学，2005

[29] 朱承，张维明，刘忠. 一种基于资源类型的网格资源发现方法[J]. 计算机研究与发展，2004，41(12)：2156-2162

[30] 范小鹏，郭敬林，刘西洋，等. 一种分析闲聊协议的数学方法[J]. 计算机工程与应用，2004，3(21)：157-166

[31] UDDI Executive White Paper.UDDI.org [EB/OL],http:// www.uddi.org/pubs/ UDDI-Executive-White-Paper. pdf, 2001

[32] Al-Ali R, Rana O, Walker D, et al. G-QoSM: Grid Service Discovery Using QoS Properties [J].Computing and Informatics Journal, Special Issue on Grid Computing, 2002, V21(4):363-382

[33] 罗洋，曾国荪. 基于本体语义的网格服务能力匹配算法[J]. 计算机应用，2004，24(9)：53-76

[34] 何炎祥，陈萃萌. Agent和多Agent系统的设计与应用[M]. 武汉：武汉大学出版社. 2001：5-67

[35] 刘大有，杨鲲，陈健中. Agent研究现状与发展趋势[J]. 软件学报，2000,11(3)：315-321

[36] Zhu C, Liu Z, Zhang W, et al.Analysis on greedy search based service location in P2P service grid[A].Proc of the 3rdInt'l Conf on Peer-to-Peer Computing[C].LosAlamitos, CA:IEEE Computer Society Press,2003:110-117

[37] 张文修，梁怡编. 遗传算法的数学基础[M] . 西安：西安交通大学出版社. 2003：112-130

[38] 王汝传，徐小龙，郑小燕. 移动代理安全机制的研究[J]. 计算机学报，2002，25(12)：1294-1301

[39] 李徐焰，郝克刚，葛玮，等. 基于Agent的网格的资源发现机制的研究[J]. 微机发展，2005，15(7)：50-53

[40] 殷锋，李志蜀，付强，等. 基于关联规则的网格资源分域管理[J]. 四川大学学报，2006，38(3):129-134

[41] 尚尔凡，都志辉. 基于虚拟组织和小世界模型的高效网格服务定位机制[J]. 计算机研究与发展，2003，40(12)：1743-1748

[42] Bunn. J, Lingen F. JClarens:a Java framework for developing and deploying Web services for grid computing[A].Proceedings of IEEE International Conference[C]. Washington,DC:IEEE Computer Society press ,2005:141-148

[43] 董健全. P2P网络中应用移动Agent进行资源搜索的研究[J]. 计算机工程与设计，2005，26(1):27-30

[44] Dorigo M, Bonabeau E, Theraulaz G. Ant algorithms and stigmergy[J].Future Generation Computer System , 2000,16(8): 851-871

[45] 段海滨. 蚁群算法原理及其应用[M]. 北京:北京科学出版社，2005：24-44

[46] 吴启迪，汪镭. 智能蚁群算法及其应用[M]. 上海:上海科技教育出版社，2004：66-80

[47] 李士勇，陈永强，李研. 蚁群算法原理及其应用[M]. 哈尔滨:哈尔滨工业大学出版社，2004：61-83

[48] 姜彤艳，罗四维. 网格环境中移动Agent的设计及其路由规划研究[D]. 北京：北京交通大学，2005

[49] 宋凤龙，刘方爱，基于遗传算法的资源节点选择策略[J]. 微机发展，2005，15(10)：62-67.

[50] Peng P.F,Lin Y.P,Zhang G.F. The Research On Optimization of Linearly Direct Current Servo fixed system by Genetic Algorithm[J].Journal of Natural Science of Hu Nan Normal University, 2004,V4 (6):48-52

[51] 陈乔礼，吴怀宇，程磊. 一种遗传蚁群系统的研究[J]. 计算机应用研究，2007，24(12)：44-50

[52] 毛宁，顾军华，谭庆，等. 遗传蚁群混合算法[J]. 计算机应用，2006，26(7)：1692–1696

[53] Pilat M.L.,White T. Using Genetic Algorithms to Optimize ACS-TSP[A].Proceedings of Ant Algorithms ANTS[C].Brussels Belgium:Springer,2002:282–287

[54] 朱玉平，叶大振，王锁萍. 基于蚁群—遗传算法的QoS路由选择[J]. 计算机工程与应用，2006，25(3)：113-147

[55] 邓小清，周竹荣，程向荣. 基于蚂蚁算法的网格资源发现模型[J]. 计算机应用，2007，27(10)：2430-2432

[56] 张至柔，罗四维，陈歆，等. 移动Agent在网格中的路径优化算法研究[J]. 计算机研究与发展，2006，43(5):791-796

[57] 丁建立. 基于蚂蚁算法的智能优化算法研究[D]. 天津：南开大学，2004

[58] 赵扬帆. 基于遗传算法和蚁群算法的网格任务调度策略[D]. 青岛：中国海洋大学，2006

[59] 张大陆，林晨. 一种基于遗传算法的快速服务发现方法[J]. 同济大学学报(自然科学版)，2006，34(2)：260-263

[60] Buyya R, Murshed M. GridSim: A Toolkit for the Modeling and imulation of Distributed Resource Management and Scheduling for Grid Computing[J].Journal of Concurrency and Computation: Practiceand Experience,2002,V14(13):1175-1220

[61] Iamnitchi A. Resource discovery in large resource-sharing environments[D].Chicago: University of Chicago, 2003

[62] 张毅，梁艳春. 蚁群算法中求解参数最优选择分析[J]. 计算机应用研究，2007，24(8)：70-83

[63] Randall M, Lewis A. A parallel implementation of ant colony optimization[J].Journal of Parallel and Distributed Computing, 2002 , 62 (9) : 1421-1432

[64] Lv Q, Cao P, Cohen E, et al.Search and replication in unstructured peer-to-peer networks [A].Proc of the 16th Int'l Conf on Supercomputing[C].New York: ACM Press, 2002: 84-95

[65] 王凌. 智能优化算法及其应用[M]. 北京:清华大学出版社，2001：154-159

[66] 丁建立，陈增强，袁著祉. 遗传算法与蚂蚁算法的融合[J]. 计算机研究与发展，2003，40(9):1351–135

附录 1　30 个城市 TSP 问题坐标

编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标
1	41	94	2	37	84	3	54	67	4	25	62	5	7	64
6	2	99	7	68	58	8	71	44	9	54	62	10	83	69
11	64	60	12	18	54	13	22	60	14	83	46	15	91	38
16	25	38	17	24	42	18	58	69	19	71	71	20	74	78
21	87	76	22	18	40	23	13	40	24	82	7	25	62	32
26	58	35	27	45	21	28	41	26	29	44	35	30	4	50

附录 2　51 个城市 TSP 问题坐标

编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标	编码	坐标	坐标
1	37	52	2	49	49	3	52	64	4	20	26	5	40	30
6	21	47	7	17	63	8	31	62	9	52	33	10	51	21
11	42	21	12	31	32	13	5	25	14	12	42	15	36	16
16	52	41	17	27	23	18	17	33	19	13	13	20	57	58
21	62	42	22	42	57	23	16	57	24	8	52	25	7	38
26	27	68	27	30	48	28	43	67	29	58	48	30	58	27
31	37	69	32	38	46	33	46	10	34	61	33	35	62	63
36	63	69	37	32	22	38	45	35	39	59	15	40	5	6
41	10	17	42	21	10	43	5	64	44	30	15	45	39	10
46	32	39	47	25	32	48	25	55	49	48	28	50	56	37
51	30	40												

攻读学位期间取得的成果

在读期间发表的学术论文

[1] 刘素芹，冯雪丽，邵红李. 网格资源发现中移动Agent的路径优化的研究[J]. 计算机工程与设计，200805，29（5）

[2] 刘素芹，邵红李，冯雪丽. 网格经济模型下基于信誉度的资源调度策略[J]. 计算机工程与设计，200804，29（4）：836-838

[3] 刘素芹，冯雪丽，邵红李. 一种有利于资源选择的网格资源发现机制[A]中国通信学会，第五届中国通信学会学术年会收录论文集[C]，北京：电子工业出版社，2007：64-68

[4] 刘素芹，孟令芬，邵红李，冯雪丽. Research and Implement of Grid Security Authentication Model based on Globus[A]. 中国通信学会，第五届中国通信学会学术年会收录论文集[C]，北京：电子工业出版社，2007：259-262

[5] 刘素芹，冯雪丽，邵红李. 一种快速的网格资源发现机制[J].计算机系统应用

致谢

在此论文完成之际，我要衷心地感谢我的导师刘素芹在我的课题研究以及论文写作过程中所给予的悉心指导和帮助。在三年的研究生学习期间，在导师对我的关怀和指导下，我各方面的能力得到了很大的提升，也为我将来的工作学习打下了良好的基础。导师严谨的治学态度、渊博的理论知识和一丝不苟的敬业精神对我影响深刻，将使我终生收益。

感谢刘老师在平时的学习和生活中给予我的帮助和支持。即将毕业之际，谨向恩师致以衷心的感谢和深深的敬意！

论文完成过程中，得到了中国石油大学（华东）计算机与通信工程学院各位老师和同学们的帮助，正是我们共同的努力和探讨，才能有今天的成果。在此向各位老师和同学们表示诚挚的谢意。

感谢母校中国石油大学（华东）对我的培养！