服务网格环境下基于整体 QoS 服务匹配策略的研究

摘 要

随着网络技术的不断发展，网格不仅符合OGSI规范，扩展了Web服务技术，秉承了Web服务的优点，而且加入了服务实例的可控性和服务状态等特性。网格服务不仅改善了传统分布式系统的集成性能、提供良好的互操作性，而且使得软件部件的商业化成为可能。因此，不同企业可以通过网格服务来连接商务程序，从而实现企业资源的共享，降低企业运营成本、提高运营效率和投资回报。

同时，网格服务的迅速发展加剧了服务提供者之间的竞争，能够完成相同功能或相似功能的服务越来越多，但是这些服务的质量却不一定相同。另一方面，由于网络条件的限制，网格服务存在很大的不确定性、网格服务的描述也有可能存在出入，甚至还可能存在欺骗性。因此，我们需要建立一个用以评估服务质量的评估机制，并作为选择网格服务的标准。尤其对于复合服务需要基于对元服务评定的评估结果进行综合服务的选择，以便在QoS的充分约束下，使组合服务的整体质量达到最高。

本文根据现有的服务网格环境下的QoS特点，对基于QoS服务发现模型和服务选择匹配策略进行了研究。主要的研究工作主要包括以下几个方面：

(l) 对网格的知识背景、网格中间件、网格服务及QoS的需求描述、网格环境下的QoS分层观点等进行归纳总结，并指出在面向服务的网格环境下针对基于整体QoS服务匹配与选择所需关注的主要问题。

(2) 对现有的基于QoS的网格服务选择策略问题进行了分析与整理，并提出了将其提升为网格服务的必要性和可能性。

(3) 根据网格服务特性设计并阐述了其不同的描述机制，并系统地分析了网格服务所涉及的各类性能指标。提出了一种网格QoS描述机制和基于整体QoS的服务发现模型，可有效提高服务发现的灵活性和可靠性。

(4) 描述服务网格环境下的质量需求，定义服务网格下的基本QoS参数及其量化方法，提出一种QoS参数归一计算方法，并在此基础上提出一种基于整体QoS的服务匹配策略。该策略通过全面比较整体QoS参数进行网格服务发现与选择，加快了服务匹配的速度，充分保证了用户的QoS需求。最后通过实验比较说明了此策略的有效性。

关键词：服务网格　服务质量　服务质量描述 归一计算 服务匹配

RESEARCH ON SERVICE MATCHING STRATEGY BASED WHOLE QOS IN SERVICE-ORIENTED GRID

ABSTRACT

With the development of network technology, grid technology meets the OGSI specificaton as the extension of Web Service technology. It not only inherits the advantages of Web Service, but also appends controllable characteristic of service instances and service state. Grid Service improves the integration performances of the traditional Distributed System, and also provides well interoperability. So it enables the software component's commercialization. The Commerce Software can connect different enterprises by Grid Service, thus the enterprises can share their resources, lower their operation costs, and increase their returns of investment.

With the fierce competition between grid services providers in nowadays,a lot of different grid services can all meet the customers' requirements,but these services which belong to the same type have different qualities.What's more,considering of the unstability of current internet conditions, the quality of grid services may be uncertain,and the descirption on grid services may be different from the fact.So we need create an evaluating mechanism on grid service quality to choose a right service. Meanwhile, we should take an full account of the quality of the composite service based on quality of meta-service in order to get the composite service of the highest quality.

So, according to the characteristics of QoS in service-oriented grid, this thesis focuses on studying the model of grid service discovering and the strategy of service matching based on user's whole QoS request. The main works are as follows:

(1) Generalize and summarize the knowledge background of grid service, grid services, grid middle-wares, the developments and characteristics of QoS requirement and point out the main problem of services choosing and matching based on the QoS in service-oriented grid.

(2) Collect, classify and analyze all the existing Grid Service selection strategy based on the QoS of grid services, and propose that these strategies should be promoted as necessity and probability of Grid Service.

(3) Also analyze all the performance indices which can describe the Grid Service comprehensively, then design and expound the diffierent describing mechanism. Propose a new descriping mechanism based on QoS and a new discovering model based on the QoS in service-oriented grid and greatly improve the effectiveness and reliability for the grid service discovering.

(4) Describe quality requirements of service-oriented grid, define the basic QoS parameters of grid service and their measurements, propose a normalization method to compute the QoS parameters and a strategy of service matching based on user's whole QoS request. This strategy comprehensively compares whole QoS properties to discover and select grid services, so that speeds service matching up and guarantee user's QoS requirements in full. Finally, the effectiveness for service matching of this strategy is illustrated by experiments and comparisons.

Key words: Service-oriented Grid; Quality of Service; Description of Quality of Service; Normalization Computing; Service Matching

广西大学学位论文原创性声明和学位论文使用授权说明

学位论文原创性声明

本人声明：所呈交的学位论文是在导师指导下完成的，研究工作所取得的成果和相关知识产权属广西大学所有。除已注明部分外，论文中不包含其他人已经发表过的研究成果，也不包含本人为获得其它学位而使用过的内容。对本文的研究工作提供过重要帮助的个人和集体，均已在论文中明确说明并致谢。

论文作者签名：廖海东　　2009 年 6 月 24 日

学位论文使用授权说明

本人完全了解广西大学关于收集、保存、使用学位论文的规定，即：

本人保证不以其它单位为第一署名单位发表或使用本论文的研究内容；

按照学校要求提交学位论文的印刷本和电子版本；

学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务；

学校可以采用影印、缩印、数字化或其它复制手段保存论文；

在不以赢利为目的的前提下，学校可以公布论文的部分或全部内容。

请选择发布时间：

☐即时发布　　☐解密后发布

（保密论文需注明，并在解密后遵守此规定）

论文作者签名：廖海东　导师签名：李[illegible]　2009 年 6 月 24 日

第一章 绪论

1.1 论文研究的背景、目的和意义

1.1.1 论文研究的背景

20 世纪 90 年代开始，互联网在全世界范围内迅速发展起来，成为了人们进行沟通和协作的主要工具，利用 Internet/Intranet 网络环境，以团队或小组为基本单位的计算机支持的协同设计(Computer Support Collaborative Design，CSCD)[1]方式应运而生，并且 CSCD 技术也得到了迅速的发展。CSCD 是一个知识共享和集成的过程[2]，也是一个数据和计算密集型的应用，有多个远程的组织和个人参与，这些组织和个人通常分布在不同的地域，使用不同的软、硬件平台，经常会随时的加入或离开设计环境。传统的基于 Internet 的协同设计平台在异构性、互操作性、动态性、可扩展性和服务质量保障(Quality of Service, QoS)等方面均无法满足协同设计的需要[3]。

为了解决 CSCD 中遇到的各种问题，网格计算技术诞生了，它能很好地解决这些问题[4]。网格是高性能计算和信息服务的战略性基础设施，它将地理上分布、异构的各种资源通过高速网络连接并集成起来，共同完成重大的科学研究问题。网格技术做为一门新兴的、功能强大的技术，一经提出就取得了前所未有的发展，众多研究机构以及企业都展开了对网格的研究，他们的研究成果不仅仅能够为信息资源的获取、分布、传输和有效利用带来革命性和结构性的巨大变化，而且将根本改变我们的研究方式、教育方式、生活和生产方式[5]。

随着网格技术研究的不断深入，网格系统间松耦合集成和互操作性等问题日益得到重视。随着 Web 服务技术的应用日益广泛，Web 服务技术标准化工作发展迅速，由于学术界和工业界对面向服务的体系结构（Service-Oriented Architecture，简称 SOA）[6]共识的形成，直接加速了网格与 Web 服务技术的融合，尤其是开放网格服务体系结构(Open Grid Service Architecture, OGSA)[7]和 Web 服务资源框架(Web Service Resource Framework，WSRF)[8]的出现进一步推进了网格技术的发展，并迅速推进了网格中间件和基于 SOA 的服务网格体系结构的发展。OGSA/WSRF 体系结构业已成为服务网格中间件构建的主要参考标准，并形成了两种典型的服务网格中间件：一种是针对特定功能设计的通用网格服务，如用于异构数据库互连的 OGSA-DAI[11]以及用于群组授权的 CAS[12]等；另一种则是支持服务运行和管理的基本软件服务器和模块，如基于 Web 服务的 GT3/4 和 OMII、基于 CORBA 的 GriT[9]和基于 Jini 的 ICENI[10]等。

由于现有的的网格研究及中间件的开发多是针对特定的应用，加之网格是信息技术领域的前沿，也是多学科交叉的热点，尚处于发展阶段，并没有形成一套统一的规范和

标准， 网格系统开发与标准的制定工作尚处与相辅相成的阶段，所以目前的网格系统之间在实现机制、技术支持上异构性大，导致相互之间的资源服务共享性差，缺乏互操作。而且，现有的中间件在资源分配效率、协作效率和 QoS 保障能力等方面并没有充分的考虑。 近两年，网格计算正在日益成为大规模共享资源、合作完成复杂任务以及集成异构系统的主流技术，并正在从科研领域逐渐走向商业应用[13~18] ，为了在进行协同设计时保证运行的服务质量[19]，面向服务中间件[16]可以为协同设计系统的运行和管理提供基础设施支撑环境，并能够动态自适应地进行服务质量目标驱动的服务调整，为具有不同需求的应用提供适宜的服务保障。

随着网格技术研究和应用的更进一步发展，缺乏服务质量保证已成为制约网格应用的瓶颈之一。由于网格具有的异构性、分布性、动态性和自治性等特性，网格应用中的服务质量QoS 的协商和保证是一项非常复杂和具有挑战性的工作，服务质量QoS 的好坏、效率的高低直接关系到网格系统的性能。因此基于网格应用的需求，必须具有非凡的服务质量，即网格应用系统希望、也迫切需要有QoS 的协商和保证[23]。同时，在对网格服务注册和查找中，具有相同功能的网格服务数量日益增加，而一般服务使用者的要求是选择一个或前几个最能满足积极需求的服务。对于不同的QoS属性，服务使用者可能会偏爱其中的部分属性，且很多不同的服务使用者可能会有相同的偏好。如何对基于整体QoS的网格服务进行组织，从而能尽快找到更能满足服务使用者质量需求的网格服务成为研究的关键点。

1.1.2 论文研究的目的和意义

本课题在现有的网格服务质量保证的基础上，主要开展服务网格环境下基于整体 QoS 的服务匹配策略的进一步研究。本课题研究的目的是：扩展现有的网格服务质量体系结构，并提出提取和量化 QoS 参数的方法，并由此形成基于整体 QoS 的服务匹配策略，它不但能够保证所选网格服务的可用性，同时还对服务等级协议进行监控并利用服务质量特征对资源服务进行评估，从而保证服务使用者服务请求的执行质量和执行效率。服务选择策略为服务网格系统提供了资源服务维护的智能化依据，通过策略中的预定方案管理资源服务的调用。

服务选择策略能够依据服务使用者的需求描述自动地完成为服务使用者的服务请求选择合适的资源服务，无需服务使用者在大量的功能等价的服务中逐个尝试，使服务使用者获得所需的资源服务的操作变得透明。因此，它能确保双方的利益最大化，并给服务使用者提供预定的服务，同时希望使之具有更强的适应动态和并发环境的能力，能更好地描述QoS服务，并能更方便地实现计算机与计算机、人与计算机之间的协作，以便能为网格的商业应用提供更好的服务质量保障。本课题的研究对网格的服务质量控制研究理论的发展具有一定的理论意义和学术价值，对网格技术投入实际商业应用也具有一定的现实意义。

1.2 服务网格环境下的服务匹配策略研究现状

1.2.1 网格环境下服务中间件

随着OGSA/WSRF体系结构成为服务网格中间件构建的主要参考标准，面向服务的网格中间件得到了迅猛的发展，并形成了两种典型的服务网格中间件：一种是支持服务运行和管理的基本软件服务器和模块；另一种则是针对特定功能设计的通用网格服务。在这些方面的研究情况大致如下：

基于Web服务的GT3/4和OMII、基于Jini的ICENI[10]等是一些支持服务运行和管理的基本软件服务器和模块，也是一种支持网格计算的底层中间件，能够提供统一的访问、服务聚合、个性化服务、工作流支持、灵活可扩展的体系结构、特别的安全防护措施和满足标准的需求[17]等。

由美国Vanderbilt University软件集成系统研究所的Aniruddha S. Gokhale等人提出的基于CORBA的GriT[9]是利用基于标准的CORBA分布式对象计算和集成技术来实现网格中间件，它不仅实现并加强了已存在的底层的诸如Globus类似的中间件功能，并描述了如何使用实时的、可容错的、数据并行的CORBA特性来给GriT提供预想的服务，而且声称同时解决了当前网格应用中所遇到的可移植性和互操作性等挑战。

英国e-science核心项目OGSA－DAI[20~22]（Open Grid Services Architecture-Data Access and Integration）是数据库网格研究领域的代表作。OGSA－DAI是一种数据访问和集成的中间件平台，它主要实现了把数据库以网格服务的形式，实现了数据共享和服务化的访问。目前它开展了OGSA－DAI ODBC driver方面的研究[20]。

OGSA－DQP(Open Grid Service Architecture-DataQuery Processor) [22]开放网格服务架构－数据查询处理器符合OGSA 的网格标准。它使用了OGSA/OGSI 的参考实现即Globus Tookit 的不同版本，Globus Tookit 是通过把资源虚拟化为网格服务来实现了一种基于服务的框架。目前推出的OGSA- DQP 3.0 是基于Globus Tookit 4 和OGSA- DAI 7.0 的基础上的，它被看作是OGSA- DAI Wrappers 的中间者。实际上，OGSA- DQP 就是一个中间件，它是基于OGSA- DAI 的，必须和OGSA- DAI联合使用才能共同实现网格环境下的分布式查询。

由美国南加州大学信息科学学院的Laura Pearlman等人提出并开发的基于群组授权的CAS[12] (Community Authorization Service)是一种通用的网格服务。它通过引入可信的受虚拟组织管理的第三方机构，当离开对资源所有者的资源访问的最终控制时，根据社区策略，对成员进行细粒度的访问控制，并声称解决了在分布式虚拟组织中待解决的三个关键授权问题，即可量测性、柔性和可表达性，并且迎合了策略层次性的需要。

在国内，北京航空航天大学承担了国家自然科学基金委员会2004年网格重大专项中

的子项目—CROWN[3]，该项目研究遵循OGSA/WSRF体系结构，并充分考虑到OGSA /WSRF 安全结构的局限性和网格应用需求，实现对网格服务的开发、部署、运行、监控和管理等全生命周期的支持。尤其在中间件设计中，特别考虑了网格资源组织与动态管理机制，并提出了与之相适应的分布式资源访问控制技术、安全结构及信任管理机制，对松耦合环境下异构资源的共享与协同提供了强有力的支持。

由清华大学等高校联合开发的中国教育科研网格公共支撑平台CGSP[22]，是为ChinaGrid的建设和发展而研制的网格核心中间件。它符合OGSA规范，是全球第一个基于WSRF技术开发的网格支撑平台，它整合教育和科研系统中的各种资源，屏蔽网格资源的异构性和动态性，提供网格资源管理、数据管理、作业管理、信息管理、网格编程环境和安全等功能支持，为各种科学计算与工程研究提供高性能的、高可靠的、安全方便的透明网格服务。

中南大学正在进行的基于网格面向弹性服务的协同设计项目—GMCD[4]通过定义一组任务关联性规则，对Globus工具包的资源描述语言进行扩展，以保证设计任务合理分解，同时为了有效解决协同环境中的冲突问题，提出了一种三级冲突消解策略。

总的来说，网格环境下面向服务的中间件的相关技术研究已经展开，并且进展很快，但目前这些研究工作尚处于起步阶段，还有许多关键的理论和技术问题有待深入研究与解决。

1.2.2 网格环境下的服务质量保障架构的相关研究

网格的QoS研究出现于2000年前后，但到目前为止尚未形成明确的技术体系，其典型研究工作包括QoS需求的描述、支持QoS的资源(服务)选取与映射、资源的预留与联合分配、支持特定QoS参数的资源调度以及域间的QoS协商与资源管理等机制。在网格计算研究的前期，考虑到计算密集型和数据密集型应用的性能，QoS管理的研究重点[23,24]主要是资源预约(Resource Reservation)和协同分配(Co-allocation)。

随着服务网格概念的提出，网格QoS研究工作的重点迅速转移到面向服务的QoS管理方面。它一方面通过定义统一的服务接口扩展和服务间交互协议，实现服务的QoS 协商和管理；另一方面，继续沿用互联网QoS中“服务级协定(SLA)”的概念，通过基于XML的QoS描述语言描述用户的QoS需求。在网格QoS系统和SLA 管理框架方面的研究工作主要包括G-QoSM[25]、CREMONA[26]、SLM Engine[27]、Gsmon[28] 等，它们最重要的特点就是开放性和标准化，QoS的表示与QoS管理机制往往通过定义标准规范或向OGSA结构中扩充新的QoS支撑服务得以实现。几种主流的网格QoS体系结构研究进展如下：

由Ian Foster等人提出的Globus预留和分配体系结构GARA(Globus Architecturefor Reservation and Allocation)[29~31]，它是一个分层的结构，主要由Globus网格安全体系框架(GSI)、各级应用编程接口API和各种资源管理器(Resource Management,RM)三部分组

成。为了确保端到端的服务质量，它们通过一致的接口对各种类型的网格资源执行提前和立即控制、监视和预留等操作。其中，GARA API主要实现网格资源的预留机制，并提供预留的创建、修改、声明、监视和取消等接口。而GSI主要使用看门人(Gatekeeper)完成鉴别、授权、创建看门人服务，并由LRAM API负责GSI与RM之间的通讯。RM主要针对网格资源的预留请求进行接纳控制，并在其管理的信任域内实现对特定网格资源的预留。

网格QoS管理体系结构(G—QoSM，Grid QoS Management Architecture)[32]是英国Cardif大学Rashid Al—Ali等人在开放网格服务体系结构OGSA(Open Grid Service Archi—tecture)环境下提出来的。它是一种能进行资源预留和自适应调节的QoS管理体系结构，主要提供三项功能：建立SLA，并在中间层提供服务质量保证机制；自适调节已分配的网格资源；基于QoS属性的服务发现。G—QoSM包括三个主要部件：应用QoS管理器(AQoS)，它存在所有的域中，主要与服务使用者、其它域的AQoS、NRM、RM进行交互，并与服务使用者及RM协商SLA，确保已分配资源SLA的一致性，同时也具有冲突消解的能力；资源管理器(RM)，它是UDDIe和Globus的融合且只存在管理域中，Globus根据特定的QoS规则管理网格服务和创建执行环境，而UDDIe作为注册中心，主要实现基于QoS属性的网格服务注册与发现；网络资源管理器(NRM)，它主要依据协商成功的SLA在给定域内管理QoS资源，负责域间NRM的通信集跨域的SLA的协调，也能根据通信信息决定当前的配置和网络的状态。

在国内，北京航空航天大学的胡春明、怀进鹏等人提出了一种支持端到端QoS的服务网格体系结构，即QESA[33](QoS-enabled service grid architecture)，这是一种支持服务质量的服务网格层次体系结构，它主要通过对网格服务容器、信息服务与调度服务的服务质量的扩展来实现了该体系结构，并设计了基于资源能力管理与服务预留的服务质量保证方法，讨论了服务质量感知的服务发现和服务调度的实现机制。并声称基于QESA的服务网格中间件可为网格的服务请求提供确定的服务质量保证能力。

同时，一些局部研究也取得了一定的进展，如在资源管理分配机制方面。文献[34]将网格资源的分配划分为两阶段进行，即首先在网格这个宏观环境下分配，然后再由具体的资源管理者在资源所在系统进行微观分配，它指出了网格环境下资源分配的特点，但是它并没有充分考虑QoS要求。文献[35]提出并建立了一个通用的支持资源预留和分配的体系架构GARA。GARA为不同类型的资源提供了统一的QoS预留机制，向服务使用者和应用开发者提供了方便的End-To-End的QoS控制机制。在GARA的基础上，A. Roy等学者在文献[36]中提出了一种新的消息传递架构MPICH-GQ，它综合了QoS描述、预留和实现技术，从基于消息传递编程的角度给出了网格QoS的一些实现机制。虽然在MPICH-GQ中，A. Roy涉及到了网格QoS层次的问题，但没有深入研究网格QoS的层次结构，也没有系统分析以服务为中心的网格QoS的特性。

另外，有一些研究在网格资源管理和分配方面引入了经济模型，突出以市场竞争为基础的资源管理和分配，利用市场中的供求原则对资源的所有者和使用者进行调节，以

保证双方均获取最大利益。这方面的相关研究还处于起步阶段，尤其在商业化阶段的过程中，在这一领域还有待于进行大量的理论研究与应用实践；再次，随着服务网格的发展，对网格服务的QoS描述、具有QoS属性的网格服务的注册与发现、服务网格中的QoS代理服务等方面的研究也变得尤为紧迫。

总之，目前面向网格作业的 QoS 管理机制还处在起步阶段，网格 QoS 的研究仍缺乏完整、清晰的技术体系，尤其对作为服务网格核心的网格服务的 QoS 控制机制还很少有研究。目前还没有采用 OGSA / WSRF 结构的服务网格中间件能够很好地支持网格应用的 QoS 需求。

1.2.3 基于 QoS 的网格服务选择策略

现有的研究主要集中在资源预留及服务等级协定的管理上[37]，对于具体的QoS参数如何提取、控制、计量等，还没有形成统一的方法，有待提出新的概念和机制，以便更好地适应服务网格环境，并为网格服务的选择提供便利。基于QoS属性是一种较好的思路，能应用到服务的选择与合成中去，并形成QoS管理框架G-QoSM[25]，符合了与Web Service结合的潮流，对服务网格QoS研究起到了很大的促进作用。但是，目前有效解决这方面具体问题的方法不多。

文献[38]提出的模型和方法对QoS参数在服务请求者和提供者间进行逐一比较，直至匹配成功，该方法的逐一匹配过程较复杂且不一定能匹配到质量较好的服务。文献[39]提出的方法利用具有代表性的QoS属性建立属性矩阵，通过局部优化算法进行服务匹配，速度快但是不能确保提供较优质的服务。文献[40]在文献[39]基础进行了改进，采用全局优化算法，充分考虑了全局QoS限制，能给服务使用者提供满意的质量，但是提出的方法速度慢、负载大。

总之，尽管目前针对网格环境下的元服务选择取得一定的进展，但是基于整体QoS的复合网格服务选择还处于起步阶段，尤其是对于QoS属性的提取、控制、计量，复合服务的子服务间的服务关系，以及服务匹配策略等方面，还有待更进一步的发展和完善。

1.3 本文的主要研究内容

近年来，随着网格业务迅速发展并逐渐向服务网格的过渡，由于竞争的加剧，通常存在多个同类的网格服务均能满足用户需求，但是各个服务的服务质量(QoS)却并不相同；同时由于网络条件的限制，网格服务存在很大的不确定性、由网格服务提供商提供的服务描述也有可能存在出入，更为严重的是还可能存在欺骗性。因此我们需要建立一个服务评估机制，用以评估服务的质量，为选择服务提供一个标准。

现有的 QoS 模型都忽略了复合服务的特性，即需要将一个复合服务作为一个有机整体向服务使用者提供完整的功能，满足服务使用者对复合服务的要求，使得所包含的基

本服务能够相互协作以完成复合服务目标。复合服务选取的目标就是根据基本服务提供的服务质量以及服务使用者对服务质量的要求，选取出质量良好或最优的复合服务，那么这必须得要求基于 QoS 的服务选取模型应用于复合服务的选取当中。

本文认为，网格服务选择要确保服务使用者的最基本 QoS 需求，而服务排序则是在此基础上的进一步选优。如果服务使用者没有任何偏好，则只要在满足最基本的可用性的基础上进行整体 QoS 指标评价，并依据整体 QoS 指标较高的服务优先反馈给服务使用者。

基于上述理由，本文提出基于整体 QoS 的网格服务发现模型和一种基于整体 QoS 的网格服务匹配方法，该方法能够在较短时间内有效地实现服务匹配成功，并确保给服务使用者提供满意的服务质量。

1.4 论文的组织结构

本文分为五章，各章内容安排如下：

第一章 绪论。主要论述课题的研究背景，当前国内外研究现状，以及本课题主要研究的内容、目的和意义等。

第二章 网格及 QoS 相关理论和技术。主要介绍服务网格环境下 QoS 相关的理论和技术，包括 Web Service 技术、开放网格服务架构(OGSA)、Globus、OGSI.NET 以及 QoS 相关技术等。

第三章 基于整体 QoS 服务网格发现模型。主要阐述基于整体 QoS 服务网格模型提出的背景、动机，以及服务网格的需求活动、网格 QoS 需求描述、网格 QoS 层次结构等，并着重介绍基于整体 QoS 服务网格发现模型的结构以及每个功能模块的具体功能和实现。

第四章 基于整体 QoS 的服务策略的设计实现。简要介绍服务选择匹配策略的功能和执行条件，定义和量化基于整体 QoS 匹配算法中的 QoS 参数，阐述基于整体 QoS 匹配算法。最后通过仿真实验对所提出的算法进行验证，并对实验后的数据结果进行分析。

第五章 总结与展望。对本文的研究成果进行总结，并指出了需要改进或有待发展的地方。

第二章　网格及 QoS 相关理论和技术

本章主要介绍服务网格环境下QoS相关的理论和技术，包括Web Service技术、开放网格服务架构(OGSA)、Globus、OGSI.NET以及QoS相关技术等。

2.1 Web Service

Web Service [41]利用简易的标准 Web 协议 (如 XML[42]等)作为组件界面描述和协同描述规范，实现了服务的描述、发布、定位以及调用。W3C(World Wide Web Consotium，万维网联盟)组织定义了一系列与 Web Service 相关的标准，其中有四个重要的协议标准，即 XML 和 XSD、SOAP [43]、WSDL[44]、UDDI[45]。

SOAP(Simple Object Access Protocol)是一种轻量的、简单的、基于 XML 的不依赖传输协议的表示层协议，用来在服务提供者与服务请求者之间进行消息传递。它能和现存的许多因特网协议和格式结合使用，在 SOAP 的下层，可以是 HTTP/HTTP，也可以是 SMTP/POP3，还可以是为一些应用而专门设计的特殊的通信协议。它主要包括三个部分：SOAP 封装、SOAP 编码规则、SOAP RPC 表示。

WSDL 是一个用来描述 Web 服务和说明如何与 Web 服务通信的 XML 语言。它用一种和具体语言无关的抽象方式定义了给定 Web Service 的有关接口和访问方法。

UDDI 提供了一组基于标准的规范用于描述和发现服务，还提供了一组基于互联网的实现。它定义了 Web Service 的目录结构。

2.1.1 XML 和 XSD

随着 HTML 的广泛应用，仅仅依赖于 HTML 这一种文件类型来处理千变万化的文档和数据已经力不从心，开发一种新的 Web 标记语言显得十分必要。W3C 建议使用一种精简的 SGML 版本，XML 就这样诞生了。XML 是 Web service 平台中表示数据的基本格式，除了易于建立和易于分析外，XML 主要的优点在于它与平台和厂商都无关。

XML 解决了数据表示的问题，但它并没有定义一套标准的数据类型，更没有对这套类型进行扩展。例如，整型数到底代表 16 位，32 位，还是 64 位？在实现互操作时，这些细节显得尤为重要。W3C 利用 XML Schema(XSD)定义了一套标准的数据类型，并给出了一种语言来扩展这套数据类型。Web Service 平台就是用 XSD 来作为其数据类型系统的。当你用某种语言(如 JAVA 或 C#)来构造一个 Web Service 时，为了符合 Web Service 标准，所有你使用的数据类型都必须被转换为 XSD 类型。我们可以利用现有的工具直接完成这个转换，但很可能会根据你的需要修改一下转换过程。

2.1.2 WSDL

WSDL（Web Service Description Language）是一种描述 XML Web 服务的标准语言，由 Ariba、Intel、IBM 和微软等开发商作为一种基于 XML 的标准而提出的。虽然它用一种与具体语言无关的抽象方式定义了给定 Web 服务收发的有关操作和消息，但是我们不能把 WSDL 当作一种对象接口定义语言，例如，CORBA 或 COM 等应用程序体系结构就会用到对象接口定义语言。WSDL 保持协议中立，但它确实内建了绑定 SOAP 的支持，因而同 SOAP 建立了不可分割的联系。

WSDL 文档由一组描述服务数据类型的元素、服务可以接受的消息以及关联消息的 SOAP 绑定组成。WSDL 文档以包含了一个版本标识的标准的 XML 头开头，而根元素则被称为 definitions，它可以采用多种可选属性，这些属性不仅说明文档，同时定义文档其余部分使用的命名空间(NameSpace)。同时定义被分配了一个名字(StockQuote)，某些命名空间定义是根据以下常规前缀缩写制定的：

tns—“this namespace”的缩写，包含被定义服务的主命名空间

xsdl—XML Schema(XSD)命名空间，用于定义文档中的类型

soap—SOAP 绑定采用的命名空间

接下来，WSDL 还定义了七种元素(element)来描述 Web Service，如表 2-1 所示：

表 2-1 WSDL 文档结构

Table 2-1 Document structure of WSDL

元素名称	说明
Types	数据类型定义的容器，它使用某种类型系统(一般使用 XML Schema 中的类型系统)
Import	类似于 XML 模式文档中的 import 元素，用于其它 WSDL 文档中导入 WSDL 定义
Message	由一个或多个逻辑 Parts 组成，每个 Parts 由 Types 所定义的类型来定义
PortType	和 operation 元素描述了 web 服务的接口并定义了它的方法
Operation	需要定义多个或一个 message 类型来定义它的输入和输出有效负载
Binding	特定接口类型的具体协议和数据格式规范的绑定
Services	此元素负责将一个 internet 地址赋给一个具体的绑定

2.1.3 UDDI

2.1.3.1 UDDI 概述

UDDI[46]是统一描述、发现和集成的缩写，全称是 Universal Description, Discovery, and Integration。它是一个基于 XML 的跨平台的描述规范，可以使世界范围内的企业在互联网上发布自己所提供的服务。UDDI 提供了一组基于标准的规范用语描述和发现服

务，还提供了一组基于互联网的实现，大大地加速 Web Service 的推广，极大地加强 Web Service 的互操作能力。另外，UDDI 对服务描述的支持具有很强的扩展性，几乎可以使用任何接口。同时 UDDI 规范允许服务提供者在注册中心发布服务，而其它的服务提供者允许在注册中心检索信息，这样极大地方便了服务的发现，并大大地促进了服务网格的商业化进程。UDDI 还能动态地发现相关的 Web 服务，并及时地将它们整合到相关的业务过程中去。

UDDI 的分层 Web 服务协议栈如图 2-1 所示：

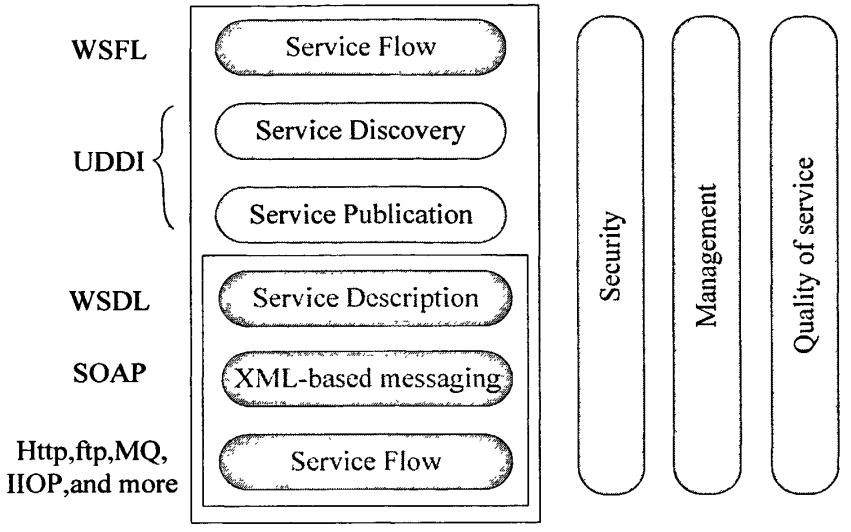

图 2-1　UDDI 的分层 Web 服务协议栈

Fig. 2-1 The UDDI's layered protocol stack for web service

2.1.3.2 UDDI 工作原理

UDDI 注册中心不仅提供了企业所提供服务的相关信息描述，还包含了对各类 Web 服务所支持的规范、分类法定义以及表示系统的引用。UDDI 也具有很强的可扩展性，它提供了一种灵活的编程模型和模式，定义了各部分之间的通信规则，而且它所有的 API 都用 XML 来定义且封装在 SOAP 信封中，在 HTTP 上传输。这些便利的可扩展的功能对服务的运行提供了强大的支持。

图 2-2[46]说明了 UDDI 消息的传输流程。首先，通过 HTTP 从客户机的 SOAP 请求传送到注册中心节点(UDDI registry node)，然后再反向传输。注册中心服务端的 SOAP 服务器接受 UDDI SOAP 消息、进行加工与处理，然后把 SOAP 响应通过 APIs 返回给客户机。为了确保消息的有效性，注册中心必须规定客户机发出的要修改数据的请求必须是安全的、经过验证的事务。

UDDI 注册中心的数据从概念上可以分为四类，每一类表示 UDDI 最上层的一种实体。每个实体都有自己的 UUID，在 UDDI 注册中心，利用这个标识符就能对这些实体进行定位。

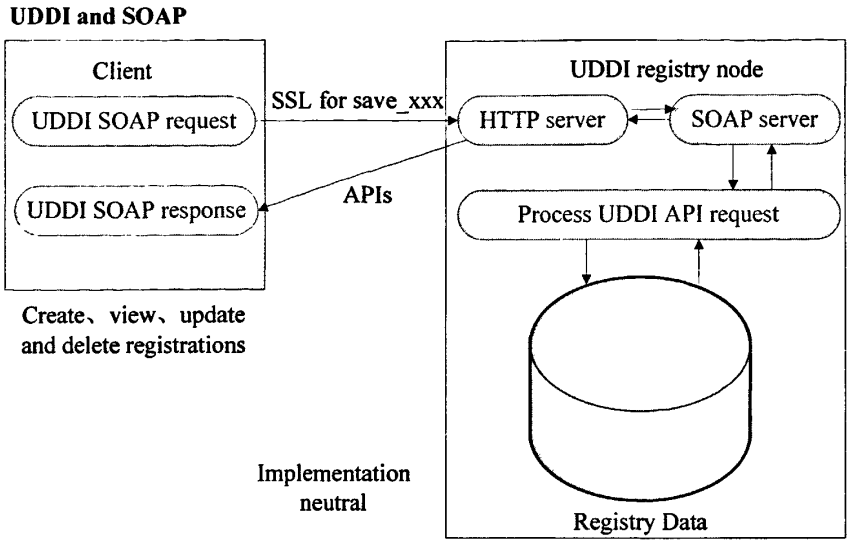

图 2-2 UDDI 消息在客户端和注册中心之间传递

Fig. 2-2 The transferring of the UDDI's message between a client and a registral center

UDDI 把企业(服务提供者)与服务的注册信息分成三类：

(1) 白页：表示有关企业的基本信息，如企业名称、经营范围描述、联系信息等。

(2) 黄页：包括基于标准分类法的行业分类。

(3) 绿页：提供商家披露的有关服务的技术信息，主要包括提供 Web 服务的规范参考，及对各种文件和基于发现机制的 URL 提供指示标志。

2.2 OGSA

2.2.1 OGSA 思想

开放网格服务体系结构 OGSA(Open Grid Service Architecture)[7]是在原来“五层沙漏结构”的基础上，结合最新的 Web Service 技术提出的，被称为下一代的网格体系结构。OGSA 是以服务为中心的“服务结构”，这里指的服务涵义极其广泛，包括各种计算资源、存储资源、程序、数据库等具有特定功能的网络化实体。这种抽象资源、数据、信息等统一起来，有利于实现灵活、一致、动态的共享机制，为分布式系统管理提供了标准的接口和行为。因此，在本文中提到的“服务”等同于“资源”，例如服务发现就等同于资源发现等。

OGSA 主要包括网格技术和 Web Service 技术。在网格应用中，大量的服务都是临时性的和短暂性的；而 Web Service 所面对的通常是永久服务。OGSA 针对网格环境的特点在 Web Service 服务概念的基础上，提出了“网格服务(Grid Service)”的概念，以便解决服务发现、动态服务创建等与临时服务有关的问题。基于网格服务的概念，OGSA

将整个网格看作是“网格服务”的集合，但这个集合是动态多变的，是可以扩展的。

2.2.2 Grid Service

网格服务提供丰富的接口，包括必需的和可选的，它们能方便地实现各种功能，而服务数据是关于网格服务实例的基本信息，因此我们可以简单地将网格服务表示为“网格服务=接口/行为+服务数据”。图 2-3 是对网格服务的简要描述。在 OGSA 的定义中，只有 GridService 接口是必须的，而其余的接口如服务创建、服务注册、授权、通知、管理、并发等都是可选的。

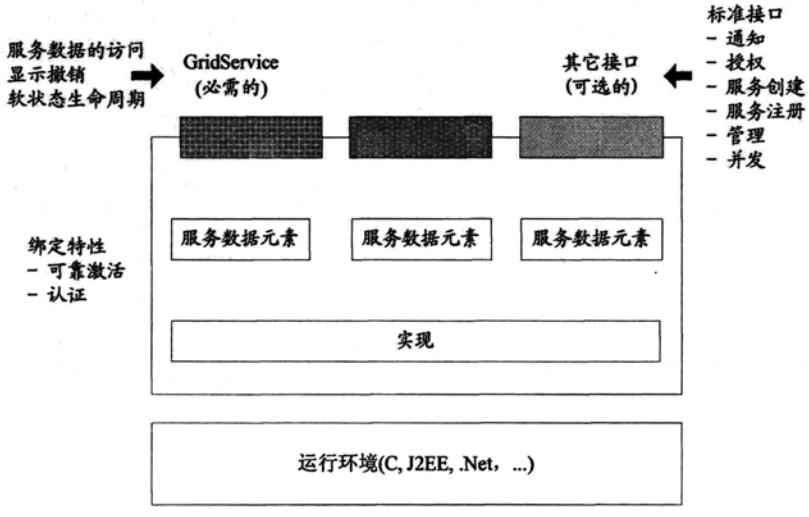

图 2-3 网格服务描述

Fig. 2-3 The description for grid service

表 2-2[47]列出了目前 OGSA 定义的一些标准的 Grid Service 接口，下一部将考虑扩充安全认证、策略管理等方面的内容。

表 2-2 Grid Service 标准接口

Table 2-2 A standard interface for Grid Service

接口	操作	描述
GridService	SetTerminationTime	设置并获取网格服务实例的结束时间
	FindServiceData	查询网格服务实例的各种信息
	Destroy	终止网格服务实例
NotificationSource	SubscribeToNotificationTopic	向通知发送者进行登记
	UnSubscribeToNotificationTopic	取消登记
NotificationSink	DeliverNotification	异步发送信息
Registry	RegisterService	服务句柄的软状态注册
	UnRegisterService	取消注册的服务句柄
Factory	CreateService	创建新的服务实例
HandleMap	FindByHandle	返回与网格服务句柄相联系的服务实例

2.2.2.1 服务工厂(Factory)与服务实例(Instance)

服务工厂(Factory)是指实现了 Factory 接口且能够通过此接口创建服务实例的服务。它用 CreateService 接口实现创建服务实例，并返回服务实例的 GSH(Grid Service Handle)和初始 GSR(Grid Service Reference)。服务工厂是永久性的服务，一旦启动，就会一直保持运行。

服务实例(Instance)是一个临时性的服务，它通过 WSDL 描述服务的接口、属性等，实现服务发现、通知、生命周期管理等。当用户找到所需要的网格服务后，首先请求服务工厂创建一个具有指定生命周期的服务实例，此实例在生命周期内执行用户的服务请求。一旦生命周期结束，服务会自动终止运行，同时用户也可以通过向服务发送 keepalive 消息使得服务实例能够继续存活。

2.2.2.2 服务数据(Service Data)

由于网格服务扩展了现有的 Web 服务，增加了服务状态性这一特点，因此，我们需要建立一种机制，当服务实例的状态数据发生变数时，服务请求者能够收到可靠的通知。此处我们使用“服务数据”来表示这些体现服务实例状态的数据。

网格服务的接口，不仅要对其内部元素进行描述，还应该对向外界表示其状态的元素进行描述。向外界表示状态，指的是在服务本身的内部实现不考虑具体的客户端的情况下，通过预先声明的服务的接口向服务的客户端展示服务的状态。我们可以将服务数据作为服务对外接口的一部分进行声明，类似于在面向对象的接口定义语言中将对象的属性作为面向对象接口的一部分进行声明，对外展示的服务数据可用于读取、更新或者作为发送消息通知的依据。

在 WSDL 中，外界对服务状态只能通过在服务的接口中为每个 portType 所定义的服务操作和消息来进行访问，并且所有的网格服务都实现了接口 Grid Service portType，它提供了通过名称处理服务数据的基本操作，避免了针对每个服务数据元素定义特定的服务数据操作。

例如，接口 fop 包含三个操作 op1、op2、op3，同时还有三个可以被外界访问的数据元素 de1、de2 和 de3。我们可以使用标准的 WSDL 描述 fop 接口中的操作，而使用 OGSI 中对 WSDL 在服务数据方面的扩展部分来描述接口 fop 中数据元素 de1、de2 和 de3 的状态：是否可以被外界全部或部分地进行访问。对于实现了上述接口定义的有状态服务实例，将可以大大地简化了与服务状态有关的操作的执行过程。

总之，我们可以通过如表 2-2 所定义的服务接口中的操作，对服务数据元素进行访问。对于服务数据的声明，只将对外可访问的服务状态的元素作为服务接口的一部分进行描述，而私有的内部状态不作为服务接口的一部分，不会通过声明而对外展现。

2.2.3 WSRF

由于OGSI过分地强调网格服务与Web服务的差别，而导致了二者不能很好的融合，为了解决OGSI的不足，IBM、BEA与微软公司于2004年3月联合发布了WS-Addressing协议。该协议提出后不久，Globus联盟和IBM迅速推出了Web服务资源框架WSRF(Web Services Resource Framework)[60]。WSRF是OGSA的最新核心规范，它的提出是对应于OGSI 1.0版的推出，并试图解决OGSI和Web服务之间存在的矛盾。

WSRF 采用了完全区别于网格服务的定义：资源是有状态的，服务是无状态的。为了充分兼容现有的 Web 服务，WSRF 利用 WSDL 1.1 定义 OGSI 中的各项能力，把网格服务分割成 Web 服务和资源文档两部分。WSRF 的推出，在于定义一个通用、开放、可扩展的架构，并利用 Web 服务存取具有状态属性的资源、包含描述状态属性的机制以及如何将机制延伸至 Web 服务中的方式。

2.2.3.1 WSRF 的优势

Web 服务资源框架 WSRF 的提出加速了网格和 Web 服务的融合，以及科研界和工业界的接轨。和 OGSA 的最初核心规范 OGSI 相比，WSRF 的优势表现为如下三点[48]：

(1) OGSI 中的术语和结构让 WS 组织感到困惑，因为 OGSI 错误地认为 Web 服务一定需要很多支撑的结构。WS-Resource Framework(框架)通过对消息处理器和状态资源进行分离来消除上述隐患，明确了其目标是允许 Web 服务操作对资源进行管理和操纵。

(2) 将网格服务完全融入 Web 服务标准，同时更全面地扩展了现有的 XML 标准，在目前的开发环境下，使其实现简单化。

(3) OGSI 规范的规模过于庞大，读者难以充分理解其内容并明确具体任务中所需的组件。在 WSRF 中通过将功能进行分离，使之简化并拓展了组合的伸缩性。

2.2.3.2 WSRF 的技术规范

WSRF 的规范主要包括以下方面的内容：

(1) Web 服务资源生命周期管理(WS-ResourceLifetime)，它定义了 WS-Resource 消除管理机制，服务请求者可以立即消除或定时消除 WS-Resource。

(2) Web 服务资源属性(WS-ResourceProperties)，它用 XML 文档定义一个 WS-Resource，提供了管理 WS-Resource 属性的机制。

(3) Web 服务服务组(WS-Service Group)，它定义了描述、管理不同种类的 Web 服务方法。

(4) 可更新的 Web 服务引用(WS-RenewableReferences)，它说明了 WS-Resource 如何通过某种通知策略重新找回失效的端点。

(5) Web 服务的基本错误服务(WS-BaseFaults)，它定义了在 Web 服务信息交换发生错误时，可使用的 XML 错误类型。

针对 OGSI 规范的主要接口和操作提出了 WSRF 规范，它保留了 OGSI 中的所有基本功能，并对部分语法进行了改进，并且使用不同的术语进行表达。随着 OGSA 和 WSRF 技术的不断完善，可以预见未来的所有的网格应用都将在基于基于 OGSA 和 WSRF 的服务网格平台和规范协议的基础上建立并实施。考虑到 WSRF 提出时间比较短暂，其规范还有待在实践中得到进一步的证明并逐步加以完善。

2.3 GLOBUS

2.3.1 GLOBUS 的起源和发展

Globus 项目发起于 20 世纪 90 年代中期，是目前国际上最有影响力的与网格计算相关的项目之一。它是美国 Argonne 国家实验室等科研单位的研发项目，当初的目的是希望把美国境内的所有的高性能计算中心通过高性能网络连接起来，方便美国的大学和研究机构使用，以便提高高性能计算机的使用效率。由于来自世界各地关注网格技术的研究人员和开发人员共同努力，Globus 项目取得了辉煌的成就。Globus 对网格计算的关键理论和技术进行了广泛的研究，尤其在信息安全、资源管理、信息服务、数据管理以及应用开发环境等方面研究比较深入，并开发出了能在多种平台运行的网格计算工具包，即 Globus Toolkit。它可以帮助规划和组建大型的网格试验和应用平台，开发适合大型网格系统运行的大型应用程序，对促进网格的商业化起到了很大的推动作用。

Globus Toolkit 是 Globus 最重要的实践成果，近些年来其版本更新很快。第一版于 1999 年推出，其后依次推出 1.1.3，1.1.4，并于 2002 年底推出了 2.2 版。2005 年 4 月底，Globus 推出了一个全新的版本，即基于 OGSA 的 4.0 版，得到了 IBM、HP、Intel、Sun 等许多国际 IT 企业的支持，为网格技术的研究和应用展现了更为广阔的前景，大大地

加速了网格商业化的进程。

2.3.2 GT4 架构

GT4[49]结合了 Web Service 技术和 Grid 技术，是 OGSA 的一种实现。与以前的版本相比，有如下方面的优势：

(1) GT4 遵守最新的 Web Services Interoperability Organization（WS-I） Web 服务标准，在不同环境之间提供了最大的互操作性。

(2) GT4 包含对一些重要授权标准的初步支持，其中包括 Security Assertion Markup Language（SAML）和 eXtensible Access Control Markup Language（XACML）。

(3) GT4 实现了 WSRF 和 WS-Notification 规范。

(4) GT4 具有复杂的授权和安全功能。Globus 在网格安全性方面一直非常努力，从安全性的观点来看，GT4 已经是“满足企业需求的”了。

(5) GT4 提供了高级的执行和数据管理功能。

GT4 的结构见图 2-4：

GT4 Core 主要实现 OGSI 规范中的 Gird Service 的接口和行为。

GT4 Security Services 实现了网格服务的访问控制。GT4 采用了 WS Authentication Authorization 取代了原来的 Grid Security Infrastructure (GSI)来实现开放网络中的安全认证和交流，其被再分为 Message（消息）级的安全性和 Authorization（授权）框架。Message 级安全性实现了两个标准：WS-Security 和 WS-SecureConversation。这两个标准提供 SOAP 消息加密、完整性和重放保护。

GT4 Base Services 利用 GT4 Core 实现已有的 GT4 功能并扩展新功能，例如：资源管理、数据传输和信息服务等。GT4 Base Service 主要包括：File Streaming Service ，Managed Job，Index Service 等服务。

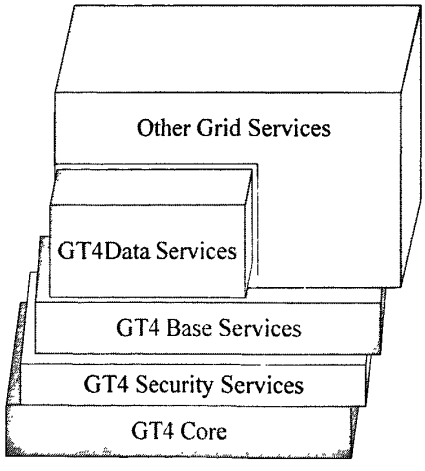

图 2-4 GT4 结构

Fig. 2-4 The structure of GT4

2.4 OGSI.NET

OGSI.NET[50]是基于 OGSI 标准的开放式网格计算框架，其目标是尽可能地利用 Microsoft .Net 核心的技术去实现 OGSI，实质也是对.NET 平台网格特性的补充。它的设计原则是尽量让框架结构中的层尽可能的薄，方便框架的扩展和新技术的使用。其包含的主要功能和工作流程分析如下：

(1) 核心层。OGSI.NET 的工作主要集中在核心层，它在实现 OGSI 规范时也根据自身的特点进行了扩充。它支持两类服务：一类是网格服务，它完全遵从 OGSI 规范并通过 Factory 方式产生的；另一类为轻量级服务，它不通过 Factory 产生，没有自己的内部状态数据，不需要具备网格服务的可配置性，但是具有部分网格服务的接口。它们与 GT4 有明显的区别，GT4 所有服务实例的服务数据都统一存放在容器中，而它们中的每一个服务都有自己的区域 AppDomain，即使是同一个 Factory 的不同服务实例也是互不干扰的。OGSI.NET 在网格服务的开发方面提供了一种基于属性的编程模型，把方法、数据和政策定义成元数据，通过描述一定数量的元数据（即服务逻辑）这种简单的编码方式来实现网格服务的开发，这些便利极大地加速了软件开发人员的进程。

(2) OGSA 服务层。该层主要提供安全方面的服务，例如消息签名、加密、身份认证等，它们完全遵从 WS-Security 规范，另外也提供了到本地用户的身份映射来完成服务实例的执行。

(3) 工作流程。首先，客户端向 OGSI.NET 发送请求，实际上就是向 IIS 服务器发送信息，OGSI.Net 使用 ISAPI 过滤器在请求信息进入 IIS 请求链的前期就把它拦截并重写了请求，并将请求分派给 OGSI.NET 的 ASP.NET HttpHandler。这个 HttpHandler 将消息路由给 OGSI.NET 的容器进程。容器进程拥有一个线程池，每一个 IIS 请求都会引起一个线程执行分派器。分派器会决定将消息路由给 GridServiceWrapper 类，留待以后进行处理。

在 AppDomain 里面，控制被转移到一个叫做网格服务包装器(GridServiceWrapper，GSW)的对象中。GSW 不仅包括了服务的实例，还包括它的方法实现和服务数据。GSW 维护着服务所支持的 portType 的一个列表(包括 OGSI 指定的由 OGSI.NET 提供的 portType，以及自定义的 portType。比如 GridService portType 和 NotificationSource portType 等)，另外还有两个是为了满足服务支持多种消息协议(比如 SOAP 或.Net Remoting)的需要而存在的序列化器和逆序列化器。GSW 通过对特定信息进行处理，并对请求信息进行序列化，从而得到即将调用的方法的名字及参数，并通过特定的消息数据(比如 SOAP 标头数据)来调用方法。当调用返回数据时，GSW 再将这些数据序列化成一个二进制 array 并传回给分派器，分派器再将这个 array 传给 IIS 并返回给客户。

图 2-5 为 OGSI.NET 的体系结构和请求流程。

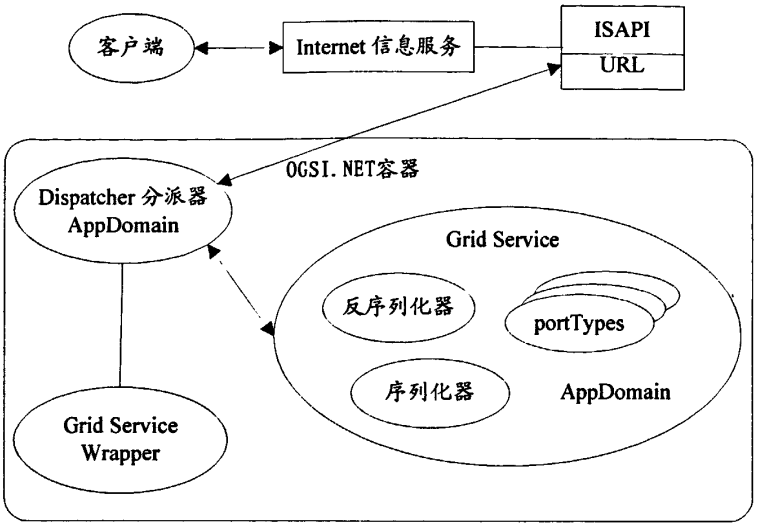

图 2-5 OGSI.NET 的体系结构和请求流程

Fig. 2-5 Both the architecture and the requested flow of OGSI.NET

2.5 QoS

2.5.1 QoS 的一般描述

QoS(Quality of Service，服务质量)指决定服务使用者满意程度的服务性能的综合效果[51]。也就是说，服务使用者(即用户)和服务提供者之间，以及他们与传输信息的继承服务网络之间关于信息传输的质量约定。该约定可被解释为服务提供者与服务使用者间的一份服务契约，即服务提供者承担支持给定的服务质量层次的义务当且仅当服务使用者按照约定的信息流量特征提供数据时。也就是说，服务质量包括两个方面：一是用户服务请求，即用户对所提供服务的基本要求，如性能、服务类型、服务质量等；二是集成服务提供者的行为，它是指一般的服务所能提供和达到的服务性能和质量，以及访问该服务的网络系统的综合性能。

QoS 与服务使用者，服务提供者以及 Internent 之间的关系如图 2-6 所示：

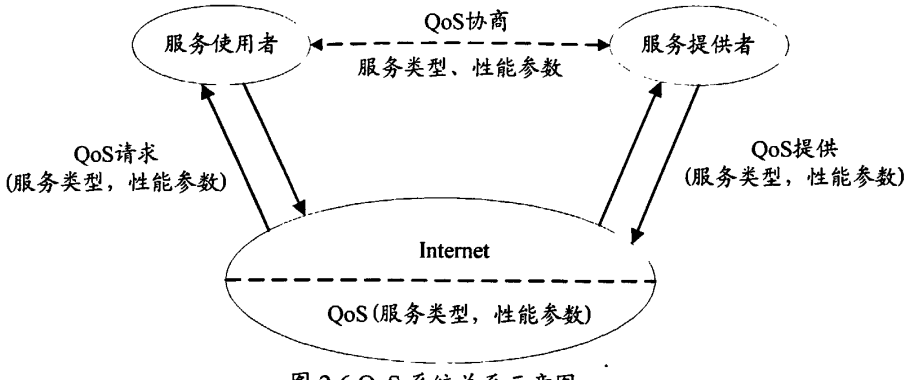

图 2-6 QoS 系统关系示意图

Fig. 2-6 The associated diagram of QoS System

由图 2-6 可以看出，QoS 不单单是网络中某个个体或元素的行为描述，还涉及到服务使用者与服务提供者，以及他们与网络、网络内部节点的整体行为。所以，为了保证服务的质量，在双方进行相互通信时，服务使用者与提供者事先必须相互协调通信时的服务类型以及相应的性能参数。我们以实时的语音通信为例，假定服务提供者以 50 帧/秒的速率传输信息，而服务使用者的接受能力低于 50 帧/秒，如果事先双方没有经过协商，那么信息会由于服务使用者的接受能力不够而丢失，从而无法达到满意的实时通信。相反，传输前如果双方商定由服务提供者降低传输速度，与服务使用者的接受能力处于相当水平，那么传输的质量将会明显提高且网络负载也会大大地减轻。

为了保证服务的质量，我们必须加强各部分之间的协调与合作，如服务使用者与服务提供者之间的积极协商，以及两者与网络、网络中各元素之间的 QoS 协商和管理。现实中网格服务提供者和使用者对 QoS 的要求很可能不一致，那么双方必须及时协商以调整到适当的水平，否则将会影响到网格服务的运行。尤其当服务使用者的 QoS 要求太高而网络无法提供相应的集成服务时，将会自动降低服务使用者的要求，甚至拒绝请求。

2.5.2 网格环境下的 QoS 分层观点

通常情况下，我们将 QoS 划分为三个层次：应用层 QoS，中间件 QoS 和网络 QoS[52]。在应用层，网格用户参与协作和分布式应用，需要访问和使用分布式资源，必须满足相应的 QoS 需求，使用 QoS 管理方法协调与确保应用和服务之间的及时交互。例如，面向服务的网格应用层可能进行下列操作：

(1) 发送控制信息到远程服务；

(2) 向远程服务传送数据；

(3) 执行远程服务；

(4) 返回数据结果。

应用层 QoS 管理，就要干预到上述的每一个操作当中，测量和记录关联操作的相应 QoS 参数，并根据具体情况灵活应变。中间件如 Globus 等，提供描述 QoS 度量的服务执行环境，及随后的管理和控制，实施 QoS。在网络层有两种 QoS 协议支持高端的应用，IntServ 和 DiffServ，在网格环境下的网络层 QoS 更加复杂，需要实现不同管理域、不同资源的聚合。网络资源的聚合，过去都是建立在 Inernet 的“尽力服务”基础上，现在不同，它需要软实时约束和更加严格的传输保障。如何管理这些聚合资源的 QoS 需求，如何监控和测量应用的网络流，并建立自适应机制，是亟待需要解决的问题。而且，操作系统层对网格 QoS 控制也会产生影响，很多诸如请求、调度等处理，部分需要在操作系统的内核中完成，如果这个层次的处理不精确，会带来负面效应。

综上所述，从分层的观点看，网格 QoS 和网络 QoS 是不同的两个问题，前者要复杂得多，网络 QoS 只是网格 QoS 中的一个部分。

2.6 本章小结

本章介绍了论文研究的相关知识，包括 Web Service 技术、开放网格服务架构(OGSA)、WSRF 协议、Globus、OGSI.NET 以及 QoS 相关技术等，为基于整体 QoS 的服务网格发现模型和基于整体 QoS 的服务匹配策略的实现奠定坚实的基础。

第三章 基于整体 QoS 服务网格发现模型

服务网格环境下的质量管理是一项非常复杂的工程，也是体现服务网格概念的核心内容之一。要在服务网格系统中实现真正意义上的 QoS(Quality of Service，服务质量)管理，必须与网格体系结构层面的系统管理结合起来，扩展现有的系统结构，并引入 QoS 管理模块，以便及时、准确地反映网格服务的 QoS 属性。基于此，本章提出一种基于整体 QoS 服务网格发现模型，为基于整体 QoS 的服务选择匹配策略提供便利。

3.1 研究背景分析

3.1.1 问题的提出

近年来，网格服务在全球各大研究机构与 IT 企业的不断研究下，显示出强劲的发展动力，尤其在商业系统中的应用日益凸现其价值。自网格服务在 Web 服务基础上增加了服务实例可控性和服状态特性后，允许使用的资源服务为短暂的和临时的，因此，服务提供者可以随时发布或撤销服务。在网格服务应用系统中，随着不断有新的服务注册到网格系统中，各应用领域中功能上等价的网格资源服务越来越多，因此，为了给用户提供优质的服务，我们需要进行更全面和多方位的选择，并充分考虑网格服务的整体 QoS 质量。网格服务选择的主要目的是在网络的应用集成中提供动态的服务质量保证，提高网络性能和效率，并为服务使用者提供优质的服务。

为了进一步推动网格的应用并解决其面临的相关问题，服务网格有逐渐取代主流的网格应用趋势，并遵循面向服务的网格技术和体系结构以及服务体系的规则、推动服务网格相关技术的发展。在服务网格中，主要事件活动就是一系列的服务，即每一个请求进入到这个网格之后就以流的形式在各相关的服务之间转换，流程结束请求即被完成，这一过程对服务使用者来说应该是透明的。然而，随着网格服务提供者之间竞争的加剧，通常存在多个同类的服务均能满足用户需求，但是它们的服务质量(QoS)却并不相同，同时由于网络条件的限制，网格服务存在很大的不确定性、网格服务的描述也可能跟实际的服务存在出入，甚至可能存在欺骗性的服务。

在服务网格中，服务质量现已变成衡量服务网格是否成功的重要因素。尽管“提供非凡的服务质量”是判断网格的三个准则之一[53]，但由于实际服务网格应用需要使用异构的、类型多样的资源，使得资源的调度分配、任务执行变得异常复杂，因而服务质量得不到保障。一方面，如何从服务提供者系统和服务使用者系统上获取资源，并能实时地获得服务实例运行状态以及保证服务使用者应用的服务质量，并提供动态组合的服务质量保障成了服务网格中的核心问题之一；另一方面，为了推进服务网格的匹配和执

行，可靠的服务选择策略与服务选择的自动执行业已成为当今服务网格的研究热点。

服务网格环境下的质量管理是一项非常复杂的工程，因为在服务网格中的资源对象可以是一个以网格服务形式发布的一个计算网格系统，甚至可以是集合了多个服务以提供一套更高级功能的复合服务，它的含义非常广泛。要想在服务网格系统中实现真正意义上的 QoS 管理，必须与服务网格体系结构层面的系统管理结合起来，解决如下的基本问题：

(1) 如何有效描述服务使用者对网格服务的要求。在不同的网格服务应用领域，服务评估性能指标和评估标准不一定相同，甚至不同的服务使用者对同一个网格服务也会有不同的服务质量要求。而对于服务使用者来说，不管评估标准是否相同，环境是否复杂，他们都希望能够按照自己的质量要求使用网格服务。

(2) 如何形成 QoS 统一的规范。在整个服务网格活动中以及各个活动阶段，服务使用者对网格服务的质量需求都能被正确地理解、传递和反馈。

(3) 如何保证 QoS 的质量。在服务网格活动过程中，QoS 需求的实现要有统一的 SLA(Service Level Agreement)协商机制；而且，必须监控 SLA 的结果，处理 SLA 违规，进行实时的自适应控制。

(4) 网格服务自动化匹配问题。为了给服务使用者提供透明的网格服务选择机制，在保证所选择网格服务质量的同时，应尽可能地减少服务使用者的干涉，为服务使用者使用服务提供便利。另外，我们必须对服务结果集通过合理的服务匹配选择算法进行过滤，最终为用户提供可靠的、最佳性价比的服务。

(5) 基于 QoS 的复合服务的选取方法。复合服务选取的目标就是根据基本服务提供的服务质量以及用户对服务质量的要求，选取出质量良好或最优的复合服务，那么这必须得要求基于 QoS 的服务选取模型应用于复合服务的选取当中。

3.1.2 相关研究的现状

伴随着 OGSA 的出现，一些相关的 Web Service 技术被引入到网格应用中，大大地推动了网格应用中对服务质量的考虑。尽管目前已经存在一些相关的考虑服务质量的网格管理框架，还是不能很好地解决上述的各类问题。

目前基于 QoS 的网格系统有 General-purpose Architecture for Reservation and Allocation(GARA)，GARA 为客户端和系统应用设计了通用的访问接口和端到端的 QoS，对于不同类型的网格资源例如网络、计算机和存储等提供统一的资源预留方式。虽然 GARA 已经得到了一定的应用，但还有很大的局限性。例如它不支持开放式网格服务体系(OGSA)，不支持服务水平协议(Service Level Agreements， SLAs)，同时由于 GARA 没有设计 QoS 监控器和适应器，不能提供即时的 QoS 服务。

在 QoS 管理框架 G-QoSM[38]中，把 QoS 属性应用到网格服务的选择与合成中，并提出了资源预留、预留策略和服务水平协议的解决方案，符合了与 Web Service 结合的

潮流，但是该框架并没有考虑如何使网格应用提供预期的服务质量。例如，当用户访问由多个网格服务构成的应用，并提出诸如响应时间要求或价格要求时，目前的网格资源管理很难保证网格应用的服务质量。

针对制造业的服务网格应用，在 MG-QoS[54]中提出了与 G-QoSM 中类似的解决方案，为用户提供了一个基于 QoS 的查询服务策略，并提出了两阶段网格服务发现与选择机制，其首先根据服务的功能性属性进行服务发现，在选择阶段，直接根据 QoS 性能指标进行筛选。

在这些众多的关于 QoS 的服务网格研究中，关于网格服务选择的研究工作主要集中在满足用户对服务功能性需求的服务匹配和通过服务等级协议 SLA 来保证服务质量，并没有充分考虑到用户的非功能性需求，以及复合服务环境下的整体 QoS 服务选择策略，因此，基于上面提到的几个问题，以及通过对现有的基于服务质量的网络管理框架进行改进，提出了基于整体 QoS 的服务发现模型，为基于整体 QoS 的服务选择和匹配奠定坚实的基础。

3.2 基于整体 QoS 网格服务发现模型

3.2.1 服务网格 QoS 管理活动

在服务网格环境下，QoS 管理活动包含的内容非常丰富，例如：QoS 描述、QoS 映射、QoS 协商、QoS 反馈、QoS 监控、资源选择与分配、资源释放、自适应、容错处理等。假设把 QoS 管理分成一系列的行为，构成一次 QoS 会话，QoS 会话包含三个阶段，每个阶段都有特定的 QoS 管理活动。在建立阶段，主要的行为包括 QoS 描述、QoS 映射、QoS 协商、资源预留和记账等；活动阶段的行为包括：资源分配、QoS 监控、QoS 协商和重协商、自适应和记账等；终止阶段，负责结束 QoS 会话，资源预留终止、协定违规、服务完成和资源释放的行为，都可以导致 QoS 会话终止。这三个阶段对应的 QoS 行为如下表所示：

表 3-1 QoS 管理活动

Table 3-1 Management activity of QoS

阶段	行为
建立阶段	QoS 描述、QoS 映射、QoS 协商、资源预留和记账等
活动阶段	资源分配、QoS 监控、QoS 协商和重协商、自适应和记账等
终止阶段	资源预留终止、协定违规、服务完成和资源释放

3.2.2 服务网格 QoS 的层次结构

在服务网格环境下，资源设备层对服务使用者来说是透明的，而应用层的整体 QoS 必须依赖于资源设备层 QoS 性能指标的调整来保证，而完成这二者 QoS 属性之间的映射和转换的过程还必须依赖于中间层，即虚拟组织层。由于现有的抽象网格服务不仅仅局限于单个的网格服务，而且很有可能由多个抽象级别较低的子网格服务组织构成，因此，如果仅对其提出简略的 QoS 需求，那么系统必须能将这种简略的 QoS 请求分解成各个子服务的具体 QoS 性能指标，同时要充分考虑到服务提供者的义务和服务使用者对 QoS 需求的不同形式，在对虚拟组织层 QoS 性能指标分类的基础上，我们采用了如图 3-1 所示的服务网格 QoS 的层次结构模型[55]。

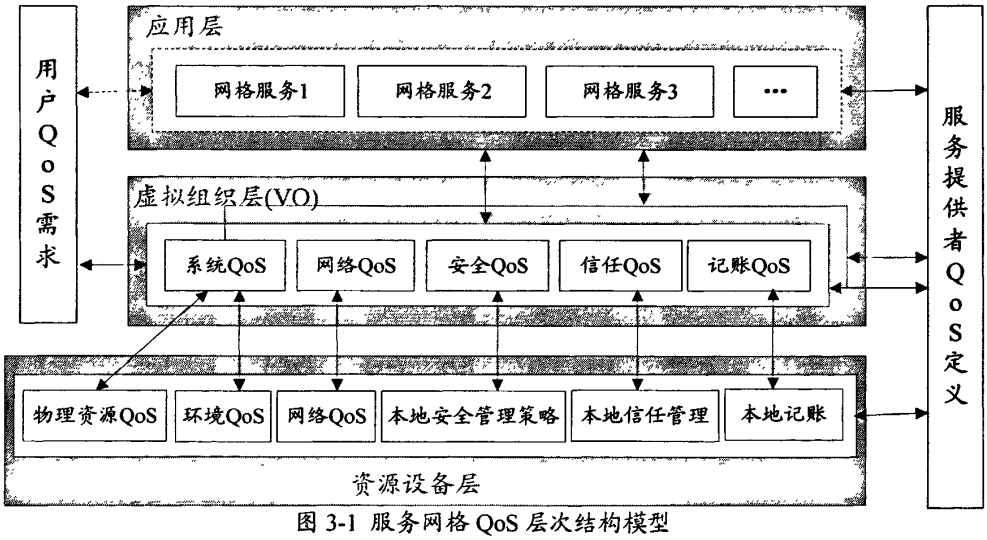

图 3-1 服务网格 QoS 层次结构模型

Fig. 3-1 Layered structure model of QoS in service-oriented grid

最顶层为应用层，对 QoS 性能指标主要提出两方面的要求：一是给网格服务粗略地确定不同的 QoS 等级，如优等、良好、一般、较差等，以满足终端用户在功能性 QoS 方面可能的缩略请求；二是网格服务提供者要提供有关服务安全、记账等描述性 QoS 性能指标。同时，为了应对服务使用者在功能性 QoS 方面可能会提出具体的 QoS 性能指标请求，我们必须对网格服务在系统 QoS 和逻辑资源 QoS 方面的具体性能指标和不同的 QoS 等级建立对应关系。现有的网格服务，可能是一个元服务，也可能是由一系列工作流组成的复合服务，对于复合服务的整体 QoS 性能指标，我们必须充分考虑到各子服务的 QoS 性能指标，确保复合服务能获得较高的整体质量。

第二层为虚拟组织层，本层的 QoS 性能指标是网格顶层 QoS 性能指标在 VO 中的映射。在虚拟组织层，服务提供者对功能性的 QoS 性能指标需要分别给出每个子服务不同 QoS 等级所对应的系统 QoS 和逻辑资源 QoS 的性能指标值；对于描述性 QoS 性能指

标，复合网格服务的每个子服务可以基于语义映射成各自的安全 QoS、信任 QoS 和记账 QoS 等三种类型的 QoS 性能指标。

最下层为资源设备层，此层的 QoS 性能指标是 VO 中各 QoS 性能指标在物理平台上的映射。

在此模型中，服务的解析过程采用了从上到下的映射转换顺序。而现实中，最下层与中间层之间是由资源提供者与虚拟组织之间来共同完成 QoS 性能指标和 QoS 协商的映射转换，而最顶层与中间层间是网格服务机构根据服务使用者的需求与虚拟组织之间进行的 QoS 协商和并完成 QoS 性能指标的映射转换，当服务提供者向虚拟组织中提供资源共享时，就完成了最下层与中间层间的 QoS 性能指标映射转换，因此，二者在时间上是可以异步进行的。

3.2.3 服务网格服务需求描述

在服务网格中，我们通常都是以定量的方式来反映网格服务的服务质量[61]，这些质量信息我们称之为服务质量参数。网格服务提供者可以针对提供的服务提出相关的服务质量信息，而服务网格使用者针对这些服务属性提出的需求描述就是网格服务质量需求描述，根据它们的来源、状态等的不同分为：

(1) 静态参数

无论网格服务是否被调用，此类参数一般都不会发生变化。此类参数通常由网格服务提供者在网格服务注册时提供，如果要更改此类参数，必须手动修改服务的相关注册信息。此类常见的参数如“服务费用”等。

(2) 动态参数

此类参数通常随着网格服务的运行而动态改变，但是在初始化时或者在服务注册时，服务提供者通常会提供这些质量参数的参考值。此类参数典型的包括提供计算网格服务中的处理器使用率、剩余内存量等，通常是可检测的。这些能够动态改变的服务质量参数为基于服务质量的服务选择带来诸多不便 ，为了准确地评定当前服务的整体质量，这类参数必须加以监控并不断获取最新的参数信息。有时为了保证提供参数的公平、公正性，参数监控工作可以交由服务监控代理商(通常是第三方)提供，服务监控代理商通常有多个，服务使用者可以根据实际需求的情况动态地选择这些质量参数的来源。

(3) 统计参数

此类参数主要特点就是其参数值无法由网格服务提供者提供，必须由第三方提供的监控代理程序通过在服务使用者使用网格服务的过程中，统计每次调用时收集到的相关信息来获得。此外，这类服务质量参数通常也需要经过一段时间累计后才具有参考价值，在网格服务注册时，由于没有调用记录，因此监控代理会给统计型质量参数设定一个默认值。此类参数常见的有网格服务的调用成功率、调用正确率等，它们反映了资源服务的可靠性和可用性等。此类参数反映了网格服务所共有的 QoS 特征，对服务的选择来说

是一个不可或缺的影响因素。

(4) 反馈参数

此类参数通常是指服务使用者在执行完服务请求后，根据执行结果好坏的程度对网格服务做出相应的评价，此类参数常见的如信誉度等。通过收集网格服务的反馈信息，我们可以分析服务注册信息是否真实可靠、资源服务的描述是否真实等，通过了解这些信息有利于服务使用者和服务评估系统对网格服务质量的正确评价。反馈参数与统计参数，都需要经过一段时间的数据积累才具有参考价值，少量的数据容易产生片面性。

3.2.4 基于整体 QoS 的网格服务发现模型

为了克服现有网格模型缺乏对网格服务质量选择的不足，更加方便、实时、准确地反映各类 QoS 参数值，本文结合已有的 G-QoSM，MG-QoS，QSWS 等模型，并根据 QoS 参数的分类，提出一种基于整体 QoS 网格服务发现模型，其具体的模型如图 3-2 所示：

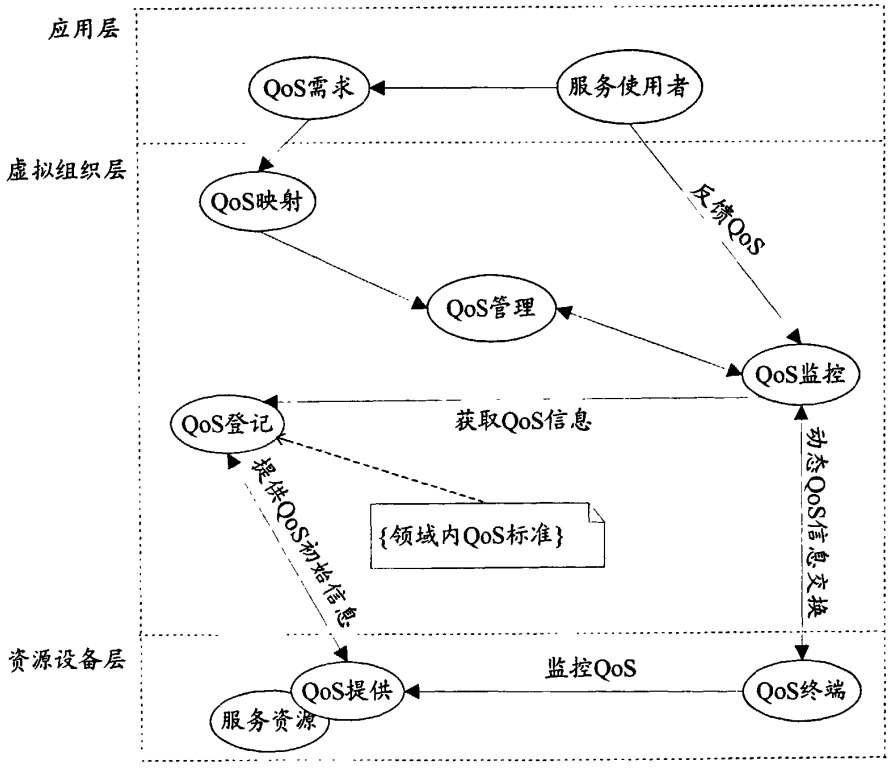

图 3-2 基于整体 QoS 网格服务发现模型

Fig. 3-2 Grid service discovering model based on the whole QoS

3.2.4.1 QoS 提供

按照 OGSA 规范，现有的在因特网上发布的网格服务基本上都配有相关的 WSDL

文档，这个文档通常包括服务接口描述文档和服务实现文档，但是并没有任何网格服务的 QoS 信息。因此，在我们的模型中借鉴了文献[59]提出的 Q-WSDL 在 WSDL 中添加 QoS 信息的表述方式，对于静态的 QoS 属性直接发布在 WSDL 文件中，对于动态的 QoS 属性，则不予提供，由 QoS 终端系统实时监控，并在发出请求时，与 QoS 监控系统实时交换信息。

QoS 描述过程：首先定义 QoS 属性性能指标的 XMLSchema，把它放在一个如图 3-3 所示的 QoSDT.xsd 文件中；然后声明网格服务接口中所定义的 QoSDataType 类型的 serviceData (如图 3-4 所示)。对 QoS 属性性能的描述基本完成，但是为了支持 SDE 的创建和更新，还得添加适当的代码。

```
<wsdl:definitions name="QoSProperty" targetNamespace="GridService">
  <wsdl:types>
    <xsd:schema>
      <complexType name="QoS_Res">
        <parameter>
          <element name="CPU" type="float"/>
          <element name="Memory" type="float"/>
          <element name="Disk" type="float"/>
        </ parameter >
      </complexType>
      <complexType name="QoS_Cost">
        < parameter >
          <element name="Cost" type="float"/>
        </ parameter >
      </complexType>
      ......
    </xsd:schema>
  </wsdl:types>
</wsdl:definitions>
```

如图3-3　QoS属性XMLSchema(QoSDT.xsd)

Fig. 3-3 XMLSchema of QoS property(QoSDT.xsd)

```
<gwsdl:portType name="WholeQoSService" extends="ogsi:GridService">
    <sd:serviceDataSet>
        <sd:serviceData name="CPU" type="QoS_Res" minOccurs="2" maxOccours="2"
                mutability="mutable">
        </sd:serviceData>
        <sd:serviceData name="Memory" type="QoS_Res" minOccurs="1024"
                maxOccours="1024" mutability="mutable">
        </sd:serviceData>
        <sd:serviceData name="Disk" type="QoS_Res" minOccurs="200" maxOccours="200"
                mutability="mutable">
```

```
        </sd:serviceData>
        <sd:serviceData name="Cost" type="QoS_Cost" minOccurs="5600" maxOccours="7600"
                mutability="mutable">
        </sd:serviceData>
        ......
    <sd:serviceDataSet>
</gwsdl:portType>
```

如图3-4　QoS属性值描述(QoSProperty.gwsdl)

Fig. 3-4 The description of QoS property value (QoSProperty.gwsdl)

3.2.4.2 QoS 终端

QoS 终端由服务注册商提供，当网格服务提供者向服务注册商成功注册服务后，会自动下载 QoS 终端到服务资源上，并在后台实时运行。当服务资源一旦有效运行时，由 QoS 终端实时与 QoS 监控系统交换 QoS 信息。QoS 终端主要监控 QoS 动态属性，为了保证提供的 QoS 属性的公平、公正，QoS 终端监控系统必须由授权机构授权之后发放证书 ID，携带证书 ID 与 QoS 监控系统交换信息。

3.2.4.3 QoS 登记

QoS 登记，即在网格服务注册登记时，完成当前网格服务所有的 QoS 属性(主要是静态 QoS 属性参数)的登记。对于动态参数、统计参数和反馈参数，可以不予登记或者也可以赋予初始默认值。

3.2.4.4 QoS 监控

QoS 监控系统在此模型中的地位非常重要，它主要完成如下功能：

(1) 收集网格服务使用者的反馈信息，并根据一段时间以来的反馈数据分析出服务注册信息是否真实可靠、资源服务是否按其描述的那样实现了相应功能。

(2) 从 QoS 登记及系统处获取 QoS 参数信息，主要是指静态类型的 QoS 参数。

(3) 在需要时，从 QoS 终端获取相关 QoS 参数信息，主要针对动态的 QoS 参数，例如可用性、可靠性等；

(4) 对所有的统计参数进行分析，按照一定算法给出当前的网格服务在一定时间内的总体服务质量水平；

(5) 与 QoS 管理系统进行数据交换，为 QoS 管理系统提供及时、准确的 QoS 参数，确保 QoS 管理系统根据提供的 QoS 参数对当前服务提供一个相对公正的评价。

3.2.4.5 QoS 管理

QoS 管理系统是此模型中的核心模块，它主要实现如下功能：

(1)在进行服务匹配选择时，实时地从 QoS 监控系统中获取所有的 QoS 参数。

(2)获取由 QoS 映射系统映射过来的 QoS 需求，并规范成系统中的标准的 QoS 参数。

(3)完成服务费用以及别的相关属性的谈判。

(4)根据现有的 QoS 属性评估策略客观地评价网格服务的 QoS 等级，并完成网格服务的选择和匹配。

3.2.4.6 QoS 映射

QoS 是一个综合指标，用于衡量使用一个服务的满意程度。为了能够给服务使用者提供较优质的服务通常都需要将服务需求参数化。满足客户对不同网格服务的不同 QoS 需求，应尽量提供灵活的 QoS 映射方式。通常情况下，服务使用者的 QoS 需求方式有两种：一是简略的 QoS 描述，如“优质”、“良好”、“一般”、“较差”等；二是具体的 QoS 性能需求。因此，QoS 映射的功能就是完成这两种 QoS 需求描述到应用层映射、应用 QoS 到系统 QoS 的映射、系统 QoS 到网络 QoS 的映射。本文着重关注服务使用者 QoS 需求描述到应用层映射。为了确保 QoS 管理系统能够准确地根据提供的 QoS 需求进行服务选择匹配，除了 QoS 需求描述必须遵循统一的原则外，QoS 需求描述中的 QoS 属性必须与 QoS 监控系统中获取的 QoS 属性建立一种一一对应的关系，确保相应的 QoS 属性映射的一致性和稳定性。

3.2.4.7 QoS 需求

服务网格资源对服务使用者来说是透明的，因此服务使用者无须关心网格资源的具体情况，只需要在应用层或者虚拟组织层(VO)按照预定的格式提出自己的 QoS 需求，选取合适类型的服务即可。为了满足服务使用者多样性需求，允许服务使用者在不同的层次提出 QoS 需求，尤其在应用层，服务使用者既可以在功能上提出简略的 QoS 需求，如“优质”、“良好”、“一般”、“较差”等，也可以提出具体的 QoS 性能指标需求，例如服务费用 Q_{cost} 不超过 300 元、网格服务的信誉度不小于 6 等。但是，如果是在虚拟组织层，则只能提出具体的 QoS 性能指标需求。因此，网格服务提供者必须提供灵活多变的易管理的 QoS 需求，以满足不同服务使用者对网格服务的质量要求。

同时，考虑到由网格服务提供者确定不同层次上的 QoS 性能指标及层次间 QoS 映射转换关系，那么对于服务的跨层抽象，使得复合服务中的子服务可能由不同的子服务提供者提供。因此，服务提供者需要定义每层的 QoS 性能指标，并且不同子服务的 QoS 性能指标可能由不同的子服务提供者确定。为了提高映射效率并确保映射成功，我们需要对服务请求发出的 QoS 需求规则化，统一格式，具体的可参照如图 3-5 所示，其中 SimpleQoS 代表简略 QoS 需求，如果其中的参数 SimpleIsValid 的值为 1，则启用简略 QoS 需求，无需要考虑具体的 QoS 性能指标需求；如果 SimpleIsValid 的值不为 1，则意味着简略 QoS 需求无效，直接启用具体的 QoS 性能指标需求。为了让服务使用者拥有更多的决策权，引入 WeightIsValid 参数，当其值如果为 1 的时候，则启用 QoS 需求参数表中的权重值，如图 3-5 中的 CPUWeight、MemoryWeight、DiskWeight 等，否则按照服务匹配策略的既定方法选择 QoS 参数的权重向量。

```
<GridQoSRequirement>
  <SimpleIsValid>1</SimpleIsValid >
  <SimpleQoS>
    <SimpleQosLevel>Best</SimpleQosLevel >
  <SimpleQos>
  <Q_Res>
    <Disk>
      <Disk_QoS>2</Disk_QoS>
      <Disk_Unit>G</Disk_Unit>
      <Disk_Weight>4</Disk_Weight>
    </Disk>
    <Memory>
      <Memory_QoS>0.5</Memory_QoS>
      <Memory_Unit>G</Memory_Unit>
      <Memory_Weight>6</Memory_Weight>
    </Memory>
    <CPU>
      <CPU_QoS>1</CPU_QoS>
      <CPU_Unit>GHz</CPU_Unit>
      <CPU_Weight>8</CPU_Weight>
    </CPU>
  </Q_Res>
  <Q_Cost>
    <Cost_QoS>300</Cost_QoS>
    <Cost_Unit>元</Cost_Unit>
    <Cost_Weight>30</Cost_Weight>
  </Cost>
  ......
  <WeightVsValid>1</WeightVsValid >
</GridQoSRequirement>
```

图 3-5　QoS 需求表示(QoSRequirement.xsd)

Fig. 3-5 The expressment for QoS requirement (QoSRequirement.xsd)

3.3 本章小结

本章主要对基于整体 QoS 服务网格模型提出的背景、动机，以及服务网格的需求活动、网格 QoS 需求描述、网格 QoS 层次结构等进行简单阐述，着重提出了基于整体 QoS 服务网格发现模型，并介绍了每个功能模块的具体功能和初步实现。

第四章 基于整体 QoS 的服务匹配策略的实现

本章在前一章介绍的基于整体QoS网格服务发现的模型基础上，结合现有的相关工作，提出一种适合商业化服务网格系统的基于整体服务质量的服务选择策略。

4.1 服务匹配策略的功能

Web 服务的迅速发展，大大改善了传统的分布式系统集成性能，在提供良好互操作性的同时，使服务资源的商业化也成为了可能，因特网上的服务终端都可以通过 Web 服务来连接商务程序。随着 OGSI 规范和 OGSA 规范的出现和不断完善，Web 服务逐渐被扩展成为网格服务，并添加了服务实例的可控性和服务状态性。Web 服务和网格服务的迅速发展和日益普及，导致了网格服务数量的迅速增长，给服务使用者选择使用网格服务带来了极大的挑战性。事实上，服务使用者为了节约支出、降低人力成本、提高竞争优势，他们通常都希望能依照自定的需求使用网格服务。

因此，为了能给服务使用者提供一种随需应变地使用网格服务的机制，需要有相应的基于网格服务质量的选择策略，它能按照服务使用者的需求描述为服务使用者作业执行配置资源服务。希望籍此选择策略来保证所选网格服务的可用性、监控服务等级协议并进行自适应调节、通过 QoS 属性对网格资源服务进行评估，从而提高网格服务的执行质量和执行效率。基于网格服务质量的选择策略为服务网格系统提供了资源服务维护的智能化依据，通过策略中的预定方案可灵活地管理资源服务的调用。

基于网格服务质量的选择策略主要基于服务使用者的 QoS 性能需求描述，完成网格服务自动的匹配和选择，减少服务使用者和服务提供者的干预，使服务使用者获得网格服务的操作变得透明。

4.2 服务匹配策略的执行条件

服务使用者为了选择合适网格资源处理作业请求，可以借助于服务网格系统中的服务选择策略，而执行服务选择策略通常需要如下部分内容：

(1) 服务使用者的资源服务需求描述。其主要包括标志服务级别的服务质量需求描述、语义约束描述，以及提供可用性服务的功能性需求描述。

(2) 服务选择规则。同基于服务质量的服务评估算法相对应，需要有一系列的服务选择规则，为在候选服务中选择合适的资源服务、统计服务质量参数信息、创建资源服务实例等操作提供执行依据。

(3) 基于服务质量的服务评估算法。由于现有服务网格环境下，完成相同或类似功

能的网格服务比较多，因此，我们需要使用有效的服务评估算法，按照服务使用者对服务质量需求间的匹配程度对这些网格资源排序，以完成网格服务的顺利选择。同时，为了减少系统维护对象、提高服务匹配效率，在服务评估前过滤掉那些不符合服务质量需求以及服务质量需求匹配度较低的网格服务。

在网格系统的服务选择策略中，服务使用者通过在资源服务需求描述里定性地指定服务选择中使用的服务评估算法，从而与服务网格系统建立相应的带有定量描述的服务等级协议，服务等级协议内容的描述主要由服务选择规则实现。

4.3 基于整体 QoS 的服务匹配算法

4.3.1 QoS 参数定义

虽然目前不同标准和研究组织(如 ITU，ISO，IETF 等)对 QoS 认识还存在一定差别，但国际标准组织 ISO 8402[56] 和 ITU E.800[57] 对 QoS 的定义较好地反映了服务网格的特点。根据这些标准的定义和网格环境下的特定因素，QoS 由一些非功能性属性组成，包括服务费用、执行时间、可用资源、可用性、可靠性、信誉度等，这些属性从一定的程度上反映了服务使用者的需要或者影响着服务使用者的满意程度。为了能够较客观地评定整体服务质量，下面对这些非功能属性进行如下定义：

(1) 服务费用 Q_{Cost}：是指使用某个服务所需支付或收取的服务费用。在服务网格市场，服务费用的确定应遵循人们对商品市场的认知规律，在同等条件下，不同的商品应该有不同的价格，质量好的商品具有较高的价格。Q_{Cost} 在服务匹配的过程中可以由服务使用者和服务提供者协商确定，双方的期望 Q_{Cost} 共同遵循着服务网格市场中的价格规律。

服务网格资源的提供者和使用者之间协商价格是一个很复杂的问题，R.Buyya 等学者参照经济学的原理对网格中的价格协商机制进行了研究[58]。本文假设服务网格使用者已经以某种价格机制认可服务网格资源的提供者所提供的价格。

(2) 执行时间 Q_{ET}：服务请求发出与该请求结果收到之间的时间延迟，包括通信时间 T_{com}、中间处理时间 T_{mid} 和传输时延 T_{tran}，即 $Q_{ET}=T_{com}+T_{mid}+T_{tran}$。

(3) 信誉度 Q_{Rep}：指合法的授权用户(服务使用者)对服务的综合评价，其计算公式定义为：

$$Q_{Rep}=\frac{1}{n}\sum_{k=1}^{n}\delta_k(S_i) \tag{4-1}$$

对于服务 S_i，定义 $\delta_k(S_i)=A_i$，A_i 为服务使用者对本次服务的评价分，$A_i \in [A_{min}, A_{max}]$，其中 A_{min} 表示评价分的下限，A_{max} 表示评价分的上限，且 $A_{min}< A_{max}$，例如某些服务提供者常设定评价分范围为[1，5]；

(4) 可用性 Q_{Ava}：定义为过去 θ 秒内服务可用时间 T_a 与 θ 的比值，其计算公式定义为： $Q_{Ava}= T_a / \theta$；

(5) 可靠性 Q_{Rel}：表示服务请求能否在声明时间内被正确地发送、调用并成功返回结果的程度，其计算公式定义为：

$$Q_{Rel} = \frac{1}{n}\sum_{k=1}^{n} Inv_k(S_i) \tag{4-2}$$

其中 $Inv_k(S_i)$表示服务 S_i 在所声明时间内是否成功调用，若成功调用其值为 1，否则为 0；

(6) 可用资源 Q_{Res} ：表示可用的节点资源，包括可用的 *CPU*，内存以及外存；可利用资源用基于时间的变化函数，可表示为 *F(T)*：→(CPU_{idle}，MEM_{idle}，$DISK_{idle}$)= Q_{Res} ，为了便于计算，把 *F(T)*归一为度量值，其值越大，表示可利用资源越多。

由于 Q_{Res} 属于服务质量动态参数，不能依靠服务网格者在服务发布时所提供的 QoS 属性描述来获取，而有赖于 QoS 监控系统实时地或者周期性地与服务提供者进行通信，以动态地获取当前节点资源的相关 QoS 参数。假定在 T 时刻 *CPU*，内存以及外存的已使用大小分别是 R_{cpu_used}、R_{mem_used} 和 R_{disk_used}，它们的总大小依次是 R_{cpu_total}、R_{mem_total} 和 R_{disk_total}，据此，Q_{Res} 计算公式不妨定义为：

$$Q_{Res}=a_1*(R_{cpu\_total}-R_{cpu\_used})/\ R_{cpu\_total} + a_2*(R_{mem\_total}-R_{mem\_used})/\ R_{mem\_total} + a_3*(R_{disk\_total}-R_{disk\_used})/\ R_{disk\_total}$$

其中，$\sum_{i=1}^{3} a_i = 1$ 且 $a_i>=0$。

4.3.2 QoS 参数归一计算

在进行 QoS 服务匹配时，面对数目众多的 QoS 参数，尤其是某些参数本身还嵌套集合、度量单位也不一致，使得匹配困难，影响服务的选择。

定义 1：网格服务 S_i 的 QoS 描述向量 Q=(Q_{i1}， Q_{i2}, Q_{i3} , …, Q_{im})，其中 m 为向量维度。该向量支持对宿主服务的实际 QoS 指标的度量，也能表达服务使用者对目标服务 QoS 指标的需求。

假设有一系列提供相同功能的服务 S(S={S_1,S_2,…,S_n})。根据定义 1，可以得到服务 S 的 QoS 服务矩阵 Q_S：

$$Q_S = \begin{pmatrix} Q_{11}, Q_{12}, \cdots, Q_{1m} \\ Q_{21}, Q_{22}, \cdots, Q_{2m} \\ \cdots \\ Q_{n1}, Q_{n2}, \cdots, Q_{nm} \end{pmatrix} \tag{4-3}$$

下面是对矩阵(4-3)进行归一计算的过程：

Step1：引入数组$N=\{n_1,n_2,\ldots,n_m\}$，其中n_j=1或0，$j\in[1,m]$，当n_j=1时，表示当前服务属性属于效益型属性，否则表示成本型属性，按照如下方法对矩阵(4-3)进行标准化计算：

$$V_{i,j}=\begin{cases}Q_{i,j}/(\frac{1}{n}\sum_{i=1}^{n}Q_{i,j}) & \text{当}n_j=1\text{时；}\\ (\frac{1}{n}\sum_{i=1}^{n}Q_{i,j})/Q_{i,j} & \text{当}n_j=0\text{且}Q_{i,j}\neq 0\text{时；}\end{cases}$$

特别地，当(n_j=0 且 $Q_{i,j}$=0 时)或$\frac{1}{n}\sum_{i=1}^{n}Q_{i,j}$=0 时，$Vi,j$=0；按照上述规则对矩阵(4-3)进行标准化后，得到矩阵(4-4)，如下所示：

$$\begin{pmatrix}V_{11},V_{12},\cdots,V_{1m}\\V_{21},V_{22},\cdots,V_{2m}\\\cdots\\V_{n1},V_{n2},\cdots,V_{nm}\end{pmatrix}\tag{4-4}$$

由于量纲的不一致，对矩阵(4-4)再次进行转化，其规则$V'_{i,j}=n*V_{i,j}/\sum_{i=1}^{n}V_{i,j}$($\sum_{i=1}^{n}V_{i,j}\neq 0$)，经过转化后得到矩阵(4-5)；

$$Q'=\begin{pmatrix}V'_{11},V'_{12},\cdots,V'_{1m}\\V'_{21},V'_{22},\cdots,V'_{2m}\\\\V'_{n1},V'_{n2},\cdots,V'_{nm}\end{pmatrix}\tag{4-5}$$

Step2：引入权重矩阵 W，$W=(W_1,W_2\ldots,W_m)$；$W_i(1\leq i\leq m)$为列向量，为了确保权重选择的公平、公正，它的确定由服务代理商从已注册的服务提供者所提供的权重标准中选择。经过 Q'×W 计算以后，得到矩阵(4-6)：

$$G=\begin{pmatrix}G_{11},G_{12},\cdots,G_{1m}\\G_{21},G_{22},\cdots,G_{2m}\\\cdots\\G_{n1},G_{n2},\cdots,G_{nm}\end{pmatrix}\tag{4-6}$$

最后经过归一计算得到一维矩阵：$G'=(\sum_{j=1}^{m}G_{1,j},\sum_{j=1}^{m}G_{2,j},\cdots,\sum_{j=1}^{m}G_{n,j})^T=(G'_1,G'_2,\cdots,G'_n)^T$。

4.3.3 基于整体 QoS 服务匹配的相关描述

上一节提到的归一计算仅仅解决了单一服务的匹配选择问题，即优先选择归一计算后其值较大者对应的服务。实际上，在服务网格环境下，通常一次服务请求(可以分解为多个任务)的完成可能需要 2 个或 2 个以上的服务协同完成。为了能客观地确定本次服务的整体质量，不仅需要考虑宿主服务的服务质量，而且还要考虑与之协同合作服务者的服务质量。

假设本次服务 S_i 的请求可以分解为 n 个任务，T_j ($j \leq n$)表示对应的第 j 个任务，S_{ij} 表示任务 T_j 对应的服务。为了形式化描述任务分解图，引入如下定义：

定义 2(任务分解)：服务 S_i 可分解为 n 个任务的集合$[T_1, T_2, \ldots, T_n]$，其中每个任务状态 $T_i (1 \leq i \leq n)$ 应满足如下条件：

(1) T_1 是状态起点，T_n 是状态终点；

(2) T_i 是状态集合$[T_1, T_2, \ldots, T_{i-1}]$中任意一元素的直接后继；

(3) T_i 不是状态集合$[T_{i+1}, T_{i+2}, \ldots, T_n]$中任意一元素的直接后继；

假设某服务请求，可分解为 7 个任务，其逻辑执行顺序如图 4-1 所示的任务分解图。

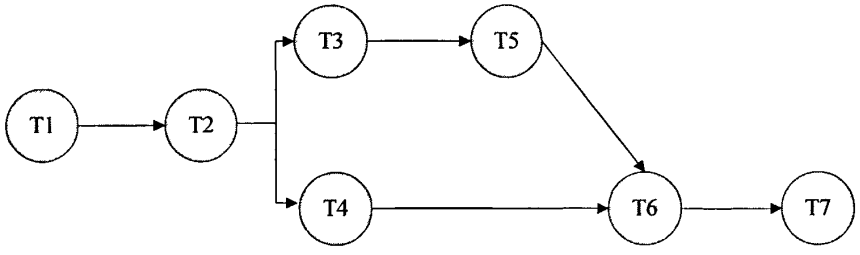

图 4-1 任务分解图

Fig. 4-1 Task decomposed diagram

定义 3：相互合作的服务之间逻辑执行关系有三种：(1)顺序关系；(2)并行关系；(3)分支关系。其中：

顺序关系：对于基本服务 S_i 和 S_j，如果执行完 S_i 后就执行 S_j，则 S_i 和 S_j 之间是顺序关系，表示为[S_i ，S_j]；图 4-1 中顺序关系服务集合={[S_1，S_2], [S_3 ，S_5], [S_6 ，S_7]}；

并行关系：对于基本服务 S_i 和 S_j ,服务 Si 的执行不依赖于 S_j ，或者服务 S_j 的执行不依赖于 S_i ，则服务 S_i 和 S_j 之间是并行关系，表示为<S_i ，S_j>；图 4-1 中并行关系服务集合={<S_4，[S_3，S_5]>}；

分支关系：对于基本服务 S_i 和 S_j ，如果根据前面的结果，或者执行 S_i ，或者执行 S_j ，则服务 S_i 和 S_j 之间是分支关系，表示为(S_i ，S_j)；图 4-1 中具有分支关系服务集合={([S_3,S_5]，S_4)}；

仅仅用这三种逻辑执行关系并不能充分表示服务执行之间的所有的逻辑关系，必须对此扩展，即对于服务 S_i 和 S_{i+1}，都允许为复合虚拟服务，即 S_i (或 S_{i+1})可以表示为顺序或并行执行后的服务。假设 S_i 表示 S_a 和 S_b 串行执行后的服务，则串行执行符号串表

示为[[S_a ,S_b],S_{i+1}]，并行执行符号串表示为<[S_a ,S_b],S_{i+1}>。依此方法，可对满足定义 2 的任意任务分解状态图都可以形成序列化符号串。例如，图 4-1 中的任务分解图序列化符号串后变成 [[[[S_1,S_2],{([S_3,S_5],S_4),[S_3,S_5],S_4,<[S_3,S_5],S_4>}],S_6],S_7] (说明：在具体的任务请求分解下，分支处要么执行期中的一条分支，要么以一定的概率执行其中的部分分支，因此大括号区域在实际任务请求下只能在集合{([S_3,S_5],S_4), [S_3,S_5], S_4, <[S_3,S_5],S_4>}中选择一个序列串)。

定义 4：假设有相邻的服务 S_a 与服务 S_b。合作关系数是指 S_a 与 S_b 之间合作的关联程度，用 R_{ab} 表示。

假设 k 为相邻的服务 S_a 与服务 S_b 协同合作执行成功的次数，令 $\lambda=k^{\xi}$(实验表明，ξ =0.1 时效果较好)，则定义 $R_{ab}=2/(1+e^{-\lambda})$。显然，$S_a$ 与 S_b 合作执行成功的次数越多，即 k 越大，则 λ 越大，合作关系系数 R_{ab} 也越大。在这里 S_a 与 S_b 都可以是串行或并行执行后的复合虚拟服务；合作关系系数的大小对于服务整体质量的评定具有重要的作用。

定义 5：服务间的商业实体关系及关联度

对于两个网格服务提供者商业实体 A 和实体 B，如果他们处于同一个商业联盟中，要么具有合作关系或处于对立的竞争关系，要么没有任何关系。我们用实体关联度来衡量商业实体间关联关系的好坏，用符号 Q_R 表示，计算方法如下：

$$Q_R(S_i,S_j)=\begin{cases}0.5 & \text{服务Si和服务Sj的商业实体间是竞争关系}\\ 1 & \text{服务Si和服务Sj的商业实体间没有关系}\\ 2 & \text{服务Si和服务Sj的商业实体间是合作关系}\end{cases} \tag{4-5}$$

式(4-5)表示如果两个商业实体间是合作关系，则它们的实体关联度最大；如果是竞争关系，则关联度最小。商业实体关联度的大小对于服务整体质量的评定具有重要的作用。

定义 6：服务属性评定函数。下面是一组评定函数的定义：

(1) $FN_S(S_a, S_b)$ 表示服务 S_a 和 S_b 顺序关系的属性评定函数；

(2) $FN_P(S_a, S_b)$ 表示服务 S_a 和 S_b 并行关系的属性评定函数；

(3) $FN_F(S_a, S_b)$ 表示服务 S_a 和 S_b 分支关系的属性评定函数；

假设服务 S_a 和 S_b 对应的属性向量分别为 Q_a 和 Q_b，经评定后得到 Q_{ab}，向量中每个属性的计算方法定义如下：

服务费用(顺序、并行)：

$Q_{ab}.Cost= Q_a.Cost + Q_b.Cost$

服务费用(分支)：

$Q_{ab}.Cost= P_a*Q_a.Cost + P_b*Q_b.Cost$

其中，P_a 和 P_b 分别是基本服务 S_a 和 S_b 的执行概率，且 $P_a+P_b=1$，可以通过历史记录确定 P 的值。

执行时间(顺序)：

$Q_{ab}.ET = Q_a.ET + Q_b.ET$

执行时间(并行)：

$Q_{ab}.ET = Max(Q_a.ET, Q_b.ET)$

执行时间(分支)：

$Q_{ab}.ET = P_a * Q_a.ET + P_b * Q_b.ET$

信誉度(顺序、并行)：

$Q_{ab}.Rep = (Q_a.Rep + Q_b.Rep) * R_{ab} / 2$

信誉度(分支)：

$Q_{ab}.Rep = (P_a * Q_a.Rep + P_b * Q_b.Rep) * R_{ab}$

可靠性(顺序、并行)：

$Q_{ab}.Rel = Min(Q_a.Rel, Q_b.Rel) * R_{ab}$

可靠性(分支)：

$Qab.Rel = Pa * Qa.Rel + Pb * Qb.Rel$

可用性(顺序)：

$Q_{ab}.Use = SQRT(Q_a.Use * Q_b.Use) * R_{ab}$

可用性(并行)：

$Q_{ab}.Use = Min(Q_a.Use, Q_b.Use) * R_{ab}$

可用性(分支)：

$Q_{ab}.Use = P_a * Q_a.Use + P_b * Q_b.Use$

可用资源(顺序、并行)：

$Q_{ab}.Res = Min(Q_a.Res, Q_b.Res) * R_{ab}$

可用资源(分支)：

$Q_{ab}.Res = (P_a * Q_a.Res, P_b * Q_b.Res) * R_{ab}$

商业实体关联度：

$$Q_{ab}.ERel = \begin{cases} \sum_{i=0}^{n}\left(\sum_{j=i+1}^{n+1} Q_R(S_i, S_j) / \sum_{j=i+1}^{n+1} 1\right) & \text{串行} \\ \min_{i=1}^{n}\left\{\left(Q_R(S_0, S_i) + Q_R(S_0, S_{n+1})\right)/2 + Q_R(S_i, S_{n+1})\right\} & \text{并行} \\ \sum_{i=1}^{n} P_i * \left(\left(Q_R(S_0, S_i) + Q_R(S_0, S_{n+1})\right)/2 + Q_R(S_i, S_{n+1})\right) & \text{分支} \end{cases}$$

服务S_a和S_b经顺序(或并行)执行后用S_{ab}表示，计算后的QoS属性向量Q_{ab}=($Q_{ab}.Cost$, $Q_{ab}.ET$, $Q_{ab}.Rep$, $Q_{ab}.Rel$, Q_{ab}.Use, $Q_{ab}.Res$, $Qab.ERel$, R_{ab})，其中R_{ab}为服务S_a 和S_b合作关系系数。

4.3.4 算法描述

基于整体 QoS 的服务匹配算法的基本思想是：服务使用者提出网格服务的作业请求，服务代理商根据请求对作业进行任务划分并形成相应的任务分解图；然后根据任务分解图形成序列化字符串，此时序列化串中 T_i 都用相应功能的服务 S_i 名称所取代；然后按照 QoS 约束的一致性对满足最小要求的服务作进一步的筛选，从而减小其解空间，提高服务匹配效率；最后通过服务选择的核心算法完成复合服务的整体服务质量的 QoS 属性的计算。计算前，为了减少服务一致性判断的次数，可以预先去掉服务区域中不满足条件的服务实例，从而降低了比较次数、提高了执行效率。

匹配算法的核心过程形式化描述如下：

Input(输入)：序列化符号串 *STR*；

Output(输出)：综合评定以后对应的 *QoS* 属性向量 Q_{whole}；

参数：*bFlag* 表示服务区分标志(初始为 *false*)；

　　SerA 和 *SerB* 表示服务名称；

for 序列化符号串 *STR* 中每个字符 *Chr* {

if(*Chr*<>">" *or Chr*<>"]"or Chr<>")") {

　对应字符入栈；*Continue*；

}

else {

　while(!(当前字符出栈并赋给 *Ch*)){

　switch (Ch){

　　case "[":

　　bFlag=false;

　　获取服务*SerA*和*SerB*的合作关系系数;

　　计算$FN_S(S_a, S_b)$，得到新的复合虚拟服务S_{ab}和属性向量Q_{ab};

　　复合虚拟服务*SerA*+*SerB*入栈;

　　break;

　case "<":

　　bFlag=false;

　　获取服务*SerA*和*SerB*的合作关系系数;

　　计算$FN_P(S_a, S_b)$,得到新的复合虚拟服务S_{ab}和属性向量Q_{ab};

　　复合虚拟服务*SerA*+*SerB*入栈;

　　break;

　case "(":

　　bFlag=false;

```
            获取服务SerA和SerB的合作关系系数;
            计算FN_F(S_a ,S_b),得到新的复合虚拟服务S_ab和属性向量Q_ab;
            复合虚拟服务SerA+SerB入栈;
            break;
          case ",":
             bFlag=true;
             break;
          default:
          根据bFlag标志获取服务SerA和SerB的名称;
        }
        if (!bFlag)
            break;
    }
}
```

利用本算法可计算所有候选服务的整体 QoS 属性向量，然后利用第 2 节中的 QoS 参数归一计算方法，得到 G'，再根据 G' 中的数值进行排名，排名高的服务将被优先选择。

4.4 实验分析

实验目的是验证所提出的基于整体 QoS 服务选择算法在服务网格环境下的可行性以及存在 QoS 约束下算法的有效性。在这里我们用服务选择后执行成功率来表示算法有效性，一个服务执行成功，必须满足如下条件：

（1）满足服务应用的功能性需求；

（2）满足服务使用者 QoS 的非功能性需求。

服务执行成功率表示成功调度的次数占总调度次数的百分比。我们采用随机选择服务路径算法和上面所提到的算法实现的两种服务选择方法作为比较，下面首先介绍实验方法，然后对结果进行分析。

4.4.1 实验环境

在实验中，采用.Net 平台中的 C#语言编程实现核心的基于整体服务匹配选择算法。为了方便起见，与网格服务相关联的所有信息和跟网格服务提供的 QoS 属性相关的数据，全部存储在 MSSQL Server 2000 数据中，通过 OLEDB 来实现算法对数据库的访问。实验的软硬件环境如下：

硬件环境：HP PC，Pentium® 4 CPU 3.00GHZ，512MB RAM

软件环境：Windows Server 2003，OGSI.NET 2.0，Visual Studio 2003，MSSQL Server2000 数据库。

4.4.2 数据准备

在实验中，为了使仿真的服务网格环境更加接近实际的网格系统，研究中增加了网格服务构成的复杂性，让动态到达的网格服务请求由不同数量的子服务组成，并且这些子服务有不同的资源负载容量和 QoS 要求。假设源服务总共有 100 类，每类服务完成相同的功能，但是提供的服务质量不尽相同。所有的网格服务都采用随机生成的模拟 Web 服务以及服务之间的关联关系数据作为测试用例，参照本身的 QoS 属性以概率的形式决定服务网格资源的加入或离开。实验时按照每类源服务提供总数为 5 个、50 个、300 个三种情况进行比较。

服务相关的基本信息，分别保存在数据库对应的数据表中。每个服务包括前面提到的 6 个属性，假定每个 QoS 属性的值都是在某一时刻的瞬时值，具体属性值按照正态分布分别在一定的范围内随机生成，对于动态属性值允许在一定的时间内按照其特定属性的概率大小进行变化。实验中涉及到主要的数据表的结构及关系如图 4-2 所示：

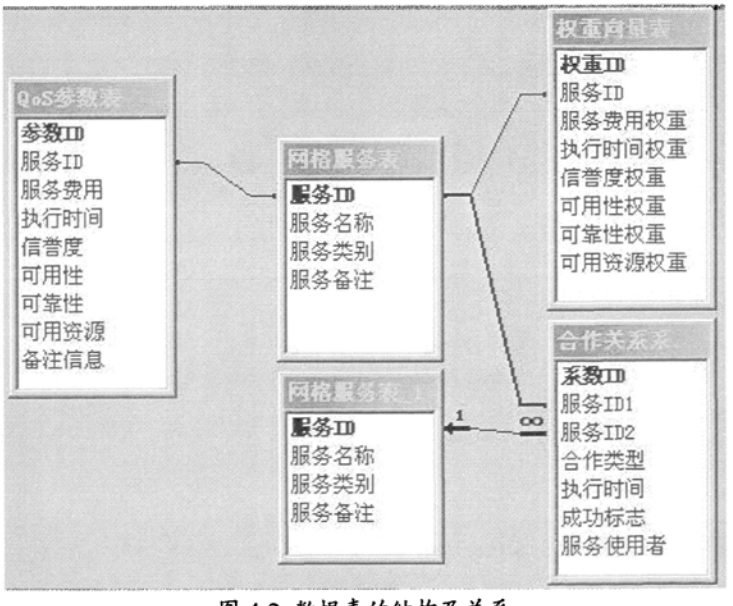

图 4-2 数据表的结构及关系

Fig. 4-2 Both the structure and the relation of data tables

4.4.3 实验结果与分析

实验中分别对具有 3、6、30 个服务的复合服务进行实验，我们采用随机选择算法(一种不考虑服务质量的最普遍的算法)和本文中提到的算法进行比较，同时根据源服务总数量(5、50、300)的不同分别进行比较。

4.4.3.1 用户请求服务成功率实验情况比较

假定每类源服务的总数各为 50 个，然后分复合服务中所包含服务的个数的不同和申请总次数这两种情况分别进行实验，实验结果后的数据表如下：

表 4-1 实验一数据结果

Table 4-1 Result of the first experiment for testing data

每类源服务总数为 50 个					
复合服务包含 3 个服务	申请的服务次数	本文的算法		随机选择算法	
		成功	成功率	成功	成功率
	5	5	100	5	100
	50	48	96	44	88
	300	280	93.3	247	82.3
6	5	5	100	4	80
	50	46	92	38	76
	300	267	89	200	66.7
30	5	5	100	4	80
	50	45	90	35	70
	300	256	85.3	170	56.7

为了更加直观地显示不同情况下的数据比较效果，用折线图显示如下：

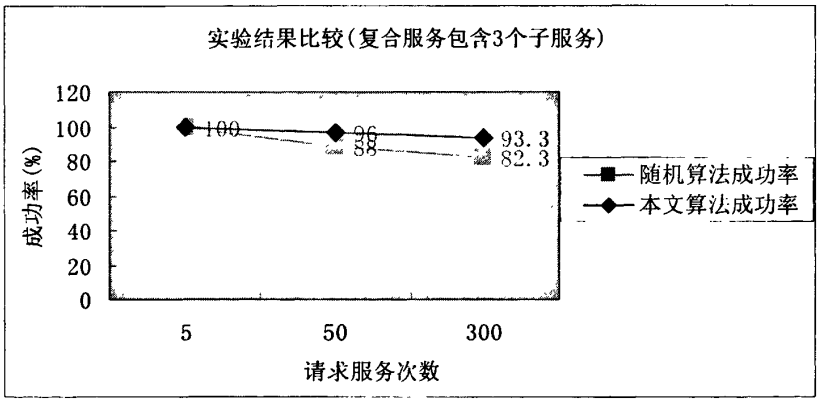

图 4-3 复合服务包含 3 个服务时的对比

Fig. 4-3 Comparision of composition services including 3 sub-services

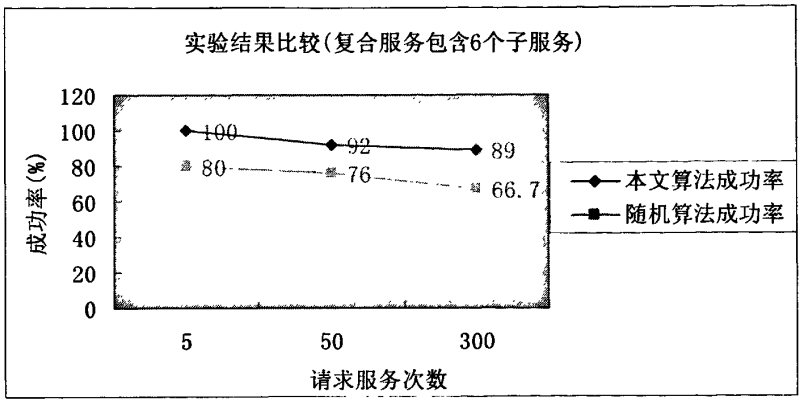

图 4-4 复合服务包含 6 个服务时的对比

Fig. 4-4 Comparision of composition services including 6 sub-services

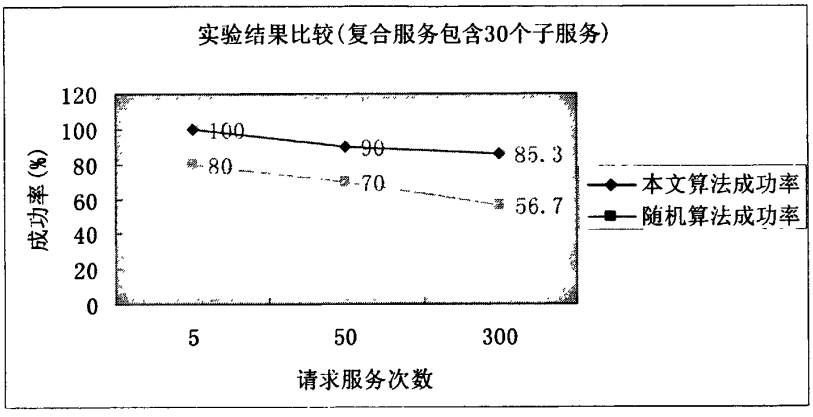

图 4-5 复合服务包含 30 个服务时的对比

Fig. 4-5 Comparision of composition services including 30 sub-services

通过对表 4-1 的实验数据和上述的各种折线图进行分析，可以得出如下结论：

(1) 在复合服务所包含的子服务和请求服务次数都相同的情况下，本文提到的算法的成功率明显高于随机选择服务路径算法；

(2) 在复合服务所包含的子服务一定的情况下，随着请求服务次数的增加，这两种方法的执行成功率都有明显的下降，本文提到的算法的成功率下降的趋势明显要低于随机选择服务路径算法；

(3) 在请求服务次数一定的情况下，随着复合服务所包含的子服务的增加，这两种方法的执行成功率都有所下降，但是本文算法的执行成功率下降比随机选择算法要缓慢；

(4) 随着复合服务所包含的子服务数和服务请求次数的同时增加，这两种方法执行成功率都在不断下降，但是本文算法的执行成功率下降比随机选择算法要缓慢。

4.4.3.2 执行时间实验结果比较

假定在复合服务包含子服务不同和每类源服务总数不同的情况下发送复合服务请求一次，然后根据调度次数和所耗时间(ms)进行数据对比，实验后的数据表如下：

表 4-2 实验二数据结果

Table 4-2 Result of the sencond experiment for testing data

复合服务的一次请求：					
复合服务包含3个服务	每类源服务总数	本文的算法		随机选择算法	
		调度次数	所耗时间	调度次数	所耗时间
	5	1	98	1	77
	50	2	234	4	457
	300	3	795	10	1986
6	5	1	125	2	163
	50	2	376	5	677
	300	4	583	15	2759
30	5	1	358	4	896
	50	3	986	8	3782
	300	6	1756	27	>5000

为了更加直观地显示不同情况下的数据比较效果，用折线图显示如下：

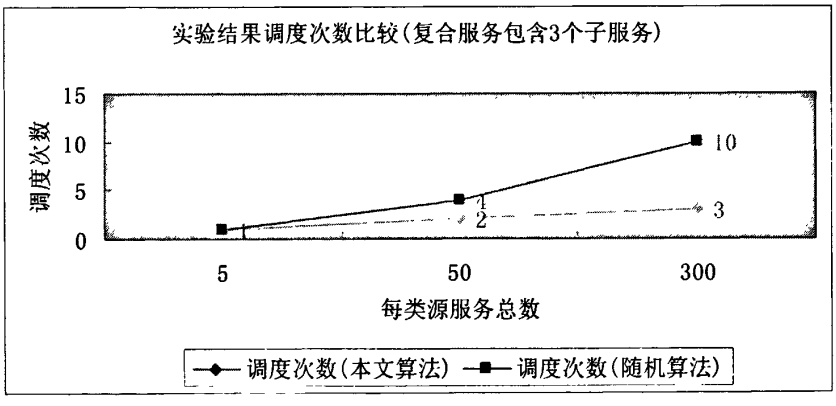

图 4-6 复合服务包含 3 个服务时调度次数对比

Fig. 4-6 Comparision of scheduling times for composition services including 3 sub-services

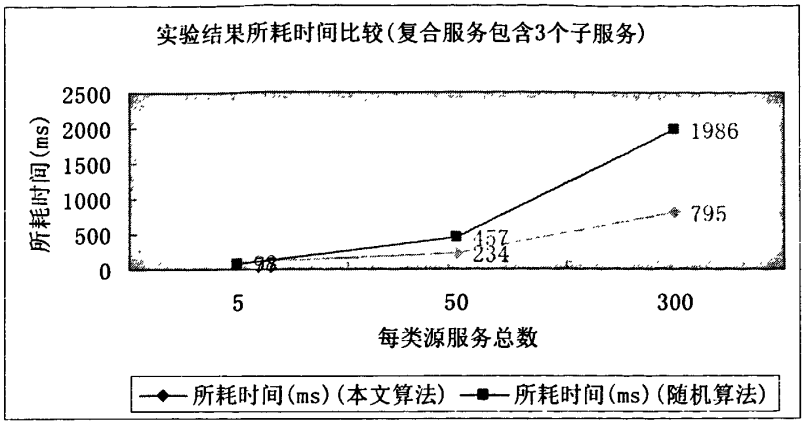

图 4-7 复合服务包含 3 个服务时所耗时间对比

Fig. 4-7 Comparision of executed time for composition services including 3 sub-services

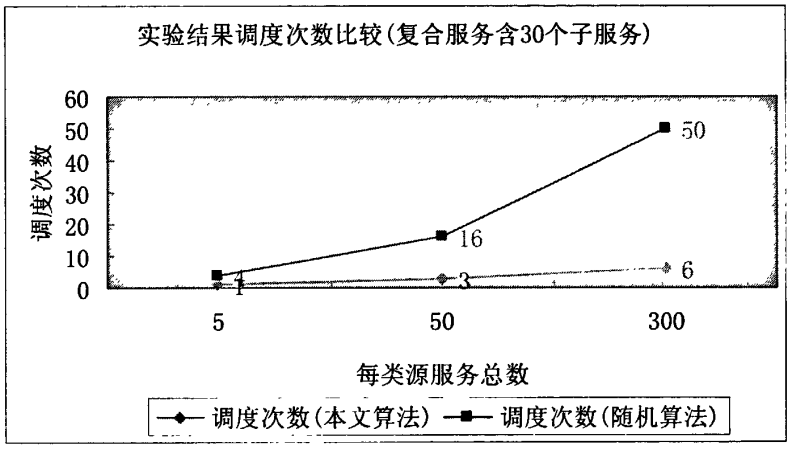

图 4-8 复合服务包含 30 个服务时调度次数对比

Fig. 4-8 Comparision of scheduling times for composition services including 30 sub-services

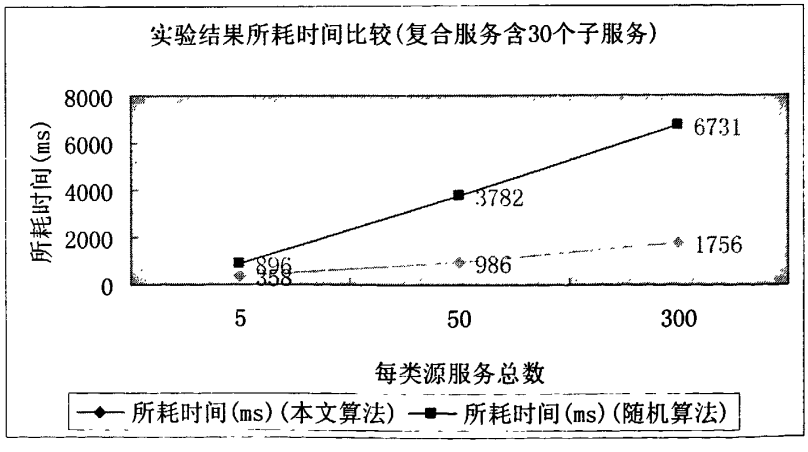

图 4-9 复合服务包含 30 个服务时所耗时间对比

Fig. 4-9 Comparision of executed time for composition services including 30 sub-services

通过对表 4-2 的实验数据和上述的各种折线图进行分析，可以得出如下结论：

(1) 在源服务数量和复合服务所包含的子服务数量都很少的情况下，本文算法执行效率略低于随机选择算法，但是随着其中任意一方数量的不断增加，本算法所耗时间明显小于随机选择算法；

(2) 若复合服务包含子服务数不变，随着源服务数的增加，两者算法的整体性能都有所下降，但本文算法的调度次数和所耗时间明显小于随机选择服务路径算法；

(3) 若源服务数不变，随着复合服务包含子服务数的增加，两者算法的整体性能都有所下降，但本文算法的调度次数和所耗时间明显小于随机选择服务路径算法；

(4) 随着源服务数和复合服务包含子服务数的增加，两种方法性能都有所下降，但是本文提到的算法下降的趋势明显趋缓。

从上述的实验分析，我们可以得出如下结论：无论是本文提出的基于整体 QoS 的服务选择算法还是最普通的随机选择服务路径算法，它们随着源服务总数量和复合服务包含的子服务的数量的增加，性能都有所下降，但是，在调度成功率、调度次数、所耗时间等方面本文算法比随机选择算法更趋向于稳定。因此，就整体性能而言，本文提到的算法明显优于随机选择服务路径算法。

4.5 本章小结

本章对服务选择匹配策略的功能和执行条件进行了简要介绍，并对基于整体 QoS 服务选择算法中的 QoS 参数进行定义和量化，同时对基于整体 QoS 服务选择算法进行了充分的阐述。最后通过实验对本算法进行验证，通过实验及其结果的分析，我们可以得出如下的结论：

(1) 在源服务总数和复合服务所包含的子服务数量都比较小的情况下，两者算法的整体性能相差不大；

(2) 在源服务总数和复合服务所包含的子服务数量相同的情况下，本文的算法的整体性能明显优于随机选择服务路径算法；

(3) 随着源服务总数和复合服务所包含的子服务数量的增加，二者的性能都有所下降，但是本文提到的算法下降速度明显比随机选择服务路径算法要慢。

实验结果与我们预期结果基本吻合，有效地说明了本文提出的基于整体 QoS 服务选择算法在服务网格环境下的可行性以及存在 QoS 约束下算法的有效性。为基于 QoS 的服务匹配选择的研究在服务网格环境下的应用迈出了坚实的一步。

第五章　总结与展望

本章对整个论文的研究工作进行总结，讨论研究工作的局限性和服务匹配策略算法进一步优化的可能性；最后对未来的研究工作进行展望。

5.1 现有研究成果总结

在广域网的网格环境中存在大量网格服务实例，如何才能从这些实例中给服务使用者选择最佳性价比的网格服务业已成为一个关键而具挑战性的问题。由于网格系统的复杂性，使得广域下的网格服务发现和匹配存在着很多困难和挑战，其涉及的内容极为广泛。本文仅对基于整体 QoS 网格服务发现模型和基于整体 QoS 网格服务选择匹配策略进行了较为深入的研究, 具体的研究工作包括：

(l) 对网格背景知识、网格中间件、网格服务及 QoS 的需求描述、网格环境下的 QoS 分层观点等进行归纳总结，并指出在面向服务的网格环境下针对基于整体 QoS 服务匹配与选择所需关注的主要问题。

(2) 详细的整理并分析了现有的基于 QoS 的网格服务选择策略问题，并提出了将其提升为网格服务的必要性和可能性。

(3) 根据网格服务特性设计并阐述了其不同的描述机制，并系统地分析了网格服务所涉及的各类性能指标。提出了一种网格 QoS 描述机制，可有效提高服务发现的灵活性和可靠性。重点提出了一种基于整体 QoS 的网格服务发现模型，并介绍每个功能模块的具体功能和实现，为基于整体 QoS 的网格服务选择与匹配奠定了坚实的基础。

(4) 描述服务网格环境下的质量需求，定义服务网格下的基本 QoS 参数及其量化方法，提出一种 QoS 参数归一计算方法，并在此基础上提出一种基于整体 QoS 的服务匹配策略，并提出了详细的算法。初步实验结果显示基于整体 QoS 服务选择算法在服务网格环境下的可行性以及存在 QoS 约束下算法的有效性。

5.2 现有研究成果的局限性

本文在取得一定的研究成果的同时，研究工作还存在着许多不足之处。

首先，基于整体 QoS 网格服务选择模型的建立是在理论分析的基础上，根据应用的需要建立起来的，虽然对它进行了算法实现并进行了实验比较和分析，但没有建立相应的数学模型来验证该模型的性能。

其次，在基于整体 QoS 的网格服务选择机制的研究中，对于基于整体 QoS 属性的评定方法以及服务间合作关系程度对整体 QoS 属性评定的影响仅仅是初步的，还存在一定的局限性；

最后，实验分析还有待进一步完善。

5.3 未来研究工作展望

针对我们已有研究工作的局限性以及服务网格的发展状况，我们计划在将来进一步开展如下几方面的工作：

(1) 在服务网格的理论研究和形式化定义方面做进一步的工作，通过深入分析实际的服务网格应用，抽取其共性特征，对服务网格行为及服务质量保证特征进行更深层次的形式化描述。

(2) 进一步研究基于整体QoS属性的评定方法以及服务间合作关系程度对QoS属性评定的影响程度，进一步优化基于整体 QoS 网格服务匹配策略算法。

(3) 进一步研究更为普适性的基于 QoS 的网格服务选择算法，以应对不同类型的网格服务对网格 QoS 的不同要求，提高网格服务的匹配效率。

参考文献

[1] Regli W. Internet-enabled computer-aided design [J]. IEEE Internet Computing, 1997,1(1): 39—51

[2] Y. V. Ramana Reddy. Computer Support for Concurrent Engineering Systems [J]. IEEE Computer, 1993. 26(1): 12—16

[3] 怀进鹏,胡春明,李建欣,孙海龙,沃天宇. CROWN:面向服务的网格中间件系统与信任管理 [M],中国科学, 2006, 36(10): 1127—1155

[4] Access Grid [EB/OL]. [2007-04-01]. http://www.accessgrid.org.

[5] 金海, 吴松. 网格技术[R]. 中国计算机科学技术发展报告(2006), 2007. 187—191

[6] Heinz Stockinger, Distributed Database Management Systems and the Data Grid [C]. 18th IEEE Symposium on Mass Storage Systems and 9th NASA Goddard Conference on Mass Storage Systems and Technologies, San Diego, April 17—20, 2001

[7] I. Foster, The Physiology of the Grid —— An Open Grid Service Architecture for Distributed Systems [C], Integration. Open Grid Service Infrastructure WG, Global Grid Forum. 2002

[8] Web Service Resource Framework (Version 1.0) [EB/OL], 2004. [2008-06-01] http://www.globus.org/wsrf/ specs/ ws-wsrf.pdf.

[9] Gokhale A S, Natarajan B. GriT: A CORBA-based Grid middleware architecture [C]. In:Proceedings of the 36th Hawaii Int'l Conference on System Sciences (HICSS'03). Washington DC: IEEE Computer Society, 2002. 319—322

[10] Furmento N., Lee W., Mayer A., et al. ICENI: An open Grid service architecture implemented with Jini [J]. Parallel Computing (PARCO), 2002, 28(12): 1753—1772

[11] Antonioletti M., Atkinson M. P., Baxter R., et al. The design and implementation of Grid database services in OGSA-DAI [J]. Concurr Comput: Prac Exp, 2005, 17(2-4): 357—376

[12] L. Pearlman, V. Welch, I. Foster, et al. A community authorization service for group collaboration [C]. In: Proceedings of the 3rd IEEE International Workshop on Policies for Distributed Systems and Networks, Los Alamitos: IEEE Computer Society Press, 2002

[13] I. Foster, C. Kesselman(editors). The Grid: Blueprint for a Future Computing Infrastructure [M]. Morgan Kaufmann Publisher, USA. 1999

[14] D. Gannon, R. Ananthakrishnan, S. Krishnan ,etc. Grid web services and application factories , Chapter 9, in Berman, F ., Fox,G . and Hey, T . (editors) Grid Computing: Making the Global Infrastructure a Reality. Chichester: John Wiley & Sons

[15] Pattnaik P., Ekanadham K. and Jann J.. Autonomic computing and the grid, Chapter 13, in Berman, F ., Fox, G . and Hey,T . (editors) Grid Computing: Making the Global Infrastructure a Reality. Chichester: John Wiley & Sons

[16] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations [J]. International Journal of Supercomputer Applications, 2001, 15(3): 200—222

[17] Berman F., Fox G. and Hey T. Grid Computing Making the Global Infrastructure a

Reality [M]. Published by Wiley Series in Communication Networking & Distributed Systems. 2003

[18] I. Foster, C. Kesselman, J. M. Nick etc. Grid Services for Distributed System Integration [J]. IEEE Computer, 2002, 35(6): 37—46

[19] 胡志刚,阎朝坤. 基于网格的现代协同设计方法 [J]. 中南大学学报(自然科学版), 2004, 35(6): 988—992

[20] Database Access and Integration Services Working Group [EB/OL], [2008-07-03] http://www.cs.man.ac.uk

[21] OGSA-DAI:A look under the hood: Part 1: Architecture and database access,Neil Hardman, Andrew Borley, James Magowan, IBM United Kingdom Limited,06 January, 2004

[22] OGSA-DAI: A look under the hood: Part 2: Activities and results, Neil Hardman, Andrew Borley, James Magowan , IBM United Kingdom Limited,21 January, 2004

[23] Wolf LC, Steinmetz R. Concepts for reservation in advance. Kluwer Journal on Multimedia Tools and Applications. 1997.4(3) http//www.Kom.e-technik.tu-darmstadt.de/ publications/abstracts /WS97-1.html

[24] I. Foster, C. Kesselman, C. Lee, B. Lindell, K. Nahrstedt, A. Roy. A distributed resource management architecture that supports advance reservations and co-allocation. In: Proc. of the Int'I Workshop on Quality of Service(IWQoS'99). 1999.27—36. http:// ieeexplore.ieee.org/search/

[25] R. Al-Ali, O Rana, D Walker, S Jha, S Sohail. G—QoSM: Grid service discovery using QoS properties [J]. Computing and Informatics Journal. Special Issue on Grid Computing, 2002, 21(4): 363—382

[26] Ludwig H., Dan A., Kearney R. Cremona: An architecture and library for creation and monitoring of WS Agreements [C]. In: Aiello M ,Aoyama M, Curbera M, et al, eds. Proc. of the 2nd Int' 1 Conf. on Service Oriented Computing(ICSOC 2004). New York: ACM Press, 2004. 65—74

[27] Keller A, Ludwig H. The WSLA framework: Specifying and monitoring service level agreements for web services [R]. IBM Research Report, RC22456(W0205-171), 2002.

[28] He C,Gu L,Du B Huang ZC,Li SL.A W SLA—based monitoring system for grid service-GSMon [C].In: Proc. of the 2004 IEEE Int' I Conf.on Service Computing(SCC 2004).2004. http://ieeexplore.ieee.org/search/freesearchresult.jsp?history=yes&que

[29] I Foster, C Kesselman. C Lee et al. A Distributed Resource ManagementArchitecture that Supports Advance Reservation and Co-Allocation [C]. In: proceedings of the International Workshop on Quality ofService(IWQoS), 1999: 27—36

[30] I Foster, A Roy, V Sander, A quality of service architecture that tom—bines resource reservation and application adaptation[C]. In: proceedingsof the International Workshop on Quality of Service(IWQoS), 2000: 181—188

[31] A Roy. End-to-End Quality of Service for High—End Applications[D]. PhD Dissertation. The University of Chicago, 2001

[32] R. Al-Ali. K Amin. G Laszewski et al. An OGSA-based Quality of Service Framework[C]. In: Proceedings of the Second International Workshop on Grid and Cooperative Computing(GCC2003), Shanghai, China, 2003—12

[33] 胡春明,怀进鹏,沃天宇,雷磊. 一种支持端到端 QoS 的服务网格体系结构 [J]. 软件学

报,2006,17(6):1448-1458. http://www.jos.org.cn/ 1000-9825/17/1448.htm
[34] Chen Hongtu，M Mabeswaran. Distributed dynamic scheduling of composite tasks on srid computing systems[C]. In: Parallel and Distributed Processing Symposium，Proceedings International，IPDPS 2002: 88-97
[35] I Foster，C Kesselman，C Lee et al. A Distributed Resource Man — agement Architecture that Supports Advance Reservations and Co—Allocation[C]. In:Intl Workshop on Quality of Service. 1999
[36] A Roy. I Foster，W Gropp et al. MPICH—GQ: Quality—of-Service for Message Passing Programs[C]. In: Proceedings of the IEEE / ACM SC2000 Conference. 2000-11
[37] 梁泉,张宏正等. 网格环境下服务质量(QoS)的研究[J].计算机科学,2006,33(7): 42-46
[38] Al-Ali R, Rana O, Laszewski G, et al. A Model for Quality of Service Provision in Service Oriented Architectures[J].International Journal of Grid and Utility Computing, 2005,1(3):72—86
[39] Liu Y., Ngu A., and Zeng L. QoS Computation and Policing in Dynamic Web Service Selection [C]. Proceedings of the 13th International World Wide Web Conference. New York: ACM, 2004. 66-73
[40] Zeng L.Z., Benaltallah B. et al. QoS-aware middleware for Web service composition[C]. IEEE Transactions on Software Engineering, 2004, 30(5) : 311—327
[41] Web Service 工作组 [EB/OL], [2008-07-03]. http//www.w3.org/2002/ws/
[42]D.C.XML Schema Part 0:Primer [EB/OL]. W3C, Recommendation,2001, [2008-05-03] http://www.w3.org/TR/xnilschema-O/
[43] D.Box, D.Ehnebuske, G. Kakivaya, Simple Object Access Protocol (SOAP) 1.1, W3C, Note 8, 2000
[44] E. Christensen, F. Curbera,G. Meredith, S. Weerawarana,Web Services Description Language(WSDL) 1.1 [EB/OL], W3C, Note 15, 2001. [2008-05-03]. http://www.w3.org/TR/wsdl
[45] UDDI:Universal Description,Discovery and Integration [EB/OL]. [2008-06-08]. http://www.uddi.org
[46] Tom Bellwood, Understanding UDDI [EB/OL]. [2008-06-08]. http://www-900. ibm.com /developerWorks/ cn/webservices /ws-featuddi/index_eng.shtmt
[47] I. Foster, C. Kesselman, J. Nick, S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration[EB/OL]. January, 2002. [2008-05-08]. http://www.globus.org/research/papers/ogsa.pdf
[48] Morgan, M. and Wasson, G. 2004. WSRF.Net Developer Tutorial[EB/OL]. [2008-05-03]. http://www.cs.virginia.edu/gsw2c/WSRFdotnet/WSRF.NET_Developer_Tutorial.pdf
[49] I. Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems [J].Journal of Computer Science and Technology,2006,21(4): 513—520
[50] OGSI.Net: OGSI-compliance on the .NET Framework [EB/OL]. 2004,12. [2008-05-03]. http://www.cs. virginia.edu /humphrey/GCG/ogsi.net.html,
[51] ITU-T Recommendation E.800, Terms And Definitions Related To Quality Of Service And Network Performance Including Dependability, August, 1994
[52] R. Al-Ali, O. Rana, D. Walker, S. Jha, and S. Sohail, G-QoSM: Grid service discovery using QoS properties [M], Computing and Informatics Journal, Special Issue on Grid Computing, 2002, 21(4): 363—382

[53] Foster I. What is The Grid? A Three Point Checklist [EB/OL]. GRIDToday,July 20, 2002. http://www.chinagrid.net/grid/paperppt/GlobusPaper/ WhatIsTheGrid.pdf

[54] R.Al-Ali,A.Hafid,O. Rana, D.Walker. QoS adaptation in service-oriented grids [C]. The 1# International Workshop on Middleware for Grid Computing (MGC2003) at ACM/IFIP/USENiX Middleware 2003,Rio de Janeiro,Brazil, 2003

[55] 罗军舟, 王小志等, 网格 QoS 层次结构模型的研究 [J]. 华中科技大学学报:自然科学版, 2005, 33(z1): 51—53

[56] ISO 8402. Quality management and quality assurance-Vocabulary

[57] ITU-T Recommendation E.800. Terms and Definitions Related to Quality of Service And Network Performance Including Dependability,1994

[58] Buyya R, Abramson D, Venugopal S.The grid economy [J]. Proceedings of the IEEE, 2005, 93(3): 698—714

[59] Andrea D Ambrogio. A model-driven WSDL extension for describing the QoS of Web services [C] //Proc of IEEE Int Conf on Web Services (ICWS'06). Los Alamitos: IEEE Computer Society, 2006

[60] K. Czajkowski. WSRF – The WS-Resource Framework [EB/OL]. 2004. [2008-03-01]. http://www.globus.org/WSRF/

[61] 聂铁铮,于戈. 服务网格中资源需求描述方法与服务选择策略的研究[D]. 东北大学硕士学位论文. 2005.

致 谢

值此毕业论文完成之际，不禁想起三年来在广西大学攻读硕士学位期间这段宝贵的学习时光，离不开一些人的帮助与支持。我谨向所有指导、关心和帮助过我的师长、同学和朋友表示最诚挚的谢意和最真诚的祝福！

在攻读硕士这三年里，导师李陶深教授无私地用自己渊博的学识、孜孜不倦的严谨治学态度和学术上的敏锐洞察力，教会了我进行科学研究以及设计开发应用系统的方法与途径。并使得本人的研究设计工作得以顺利进行，而且使自己解决实际问题的研究思维方法以及个人技术能力的提高上都大受裨益。在此，特向李老师致以最真挚的敬意和最衷心的感谢。

感谢葛志辉副教授谆谆教导和亲切关怀，在论文的撰写和定稿过程中，葛老师提出了许多宝贵中肯的意见。在此，特向葛老师致以最衷心的感谢。

同时感谢计算机与电子信息学院的钟诚教授、韦兆文副教授、唐天兵副教授、顾平副教授、宋玲副教授、陈燕副教授、黎展荣副教授、严毅老师、廖平光老师、黄文玲老师等全体学院的老师，谢谢你们在学习和生活中给予我的热情帮助和关心。

感谢我的同窗，李长才，武燕华，魏伟，熊琴，刘辽广，陆宇旻，陈涛，洪锡清，韦修喜，谭运宝等，感谢实验室的师弟师妹廖其耀，李林，王春霞，卢在盛，梁宝龙等，和他们共同的学习生活是我人生中最宝贵的经历，他们，也将成为我永远的挚友，愿大家今后的工作学习一帆风顺，百尺竿头。

感谢陪伴我度过愉快研究生生活的 2006 级全体研究生同学，在与他们的交流中，我得到了许多帮助与收获。

最后，向一贯爱护和支持我的父母、兄弟、嫂子致以衷心感谢，感谢他们多年来给予我的关爱与支持。

廖海乐

2009 年 6 月于广西大学

攻读硕士学位期间参与的科研项目

[1] 参与了南宁大应软件“物流行业信息化”项目
[2] 参与了广西自然科学基金项目“网格环境下支持协同设计的分布式事务管理机制与技术研究”（桂科自 0832056）

攻读硕士学位期间发表的学术论文

[1] 廖海乐,李陶深. 服务网格环境下一种基于整体 QoS 的服务匹配策略. 计算机应用，2008,28(S2):17-19