ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.11, November 2006, pp.2302−2313
DOI: 10.1360/jos172302

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

支持多领域动态数据集成的数据库网格系统*

申德荣+, 于 戈, 聂铁铮, 寇 月

(东北大学 信息科学与工程学院,辽宁 沈阳 110004)

A Database Grid System for Multi-Domain Dynamic Data Integration

SHEN De-Rong+, YU Ge, NIE Tie-Zheng, KOU Yue

(College of Information Science and Engineering, Northeastern University, Shenyang 110004, China)

+ Corresponding author: Phn: +86-24-83687776, Fax: +86-24-23895654, E-mail: shenderong@ise.neu.edu.cn, http://www.neu.edu.cn

Shen DR, Yu G, Nie TZ, Kou Y. A database grid system for multi-domain dynamic data integration. *Journal of Software*, 2006,17(11):2302−2313. http://www.jos.org.cn/1000-9825/17/2302.htm

Abstract: With the richness of common database resources, distributed users in wide areas hope to transparently access and use these data resources on demand. DS_Grid (database grid) is an SOA (service-oriented architecture) based database grid system for data sharing in multiple application domains. DS_Grid adopts a P2P (peer-to-peer) Multi-Chord (MultiChord) architecture to realize the distributed storage, query processing and dynamic data integration of data resources. According to the text similarity, the data resources are registered to the corresponding domains to realize rapidly discovering data resources. The domain ontology knowledge and reasoning rules are used to support the semantics based intelligent query. A multi-root and multi-peer maintenance based data resources replica management mechanism is applied to improve the reliability of the system. A keyword filter based distributed data integration strategy is adopted to reduce the communication cost. A distributed clustering technique is used to summarize the huge data information. The experiments demonstrate the feasibility and effectiveness of the key techniques of DS_Grid.

Key words: database grid; P2P (peer-to-peer); data integration; resource discovery; query processing, replica management

摘 要: 随着公有数据库资源的丰富,广泛分布的用户希望能够按需地、透明地访问和使用这些丰富的数据资源.DS_Grid(database grid)是一个采用 SOA(service-oriented architecture)思想、支持多应用领域数据共享的数据库网格系统.系统采用一种 P2P(peer-to-peer)多 Chord(MultiChord)网格体系结构,实现数据资源的分布存储、查询处理和动态数据集成;基于文本相似性,可分领域地注册数据资源,实现资源的快速发现;根据领域本体知识和推理规则,实现基于语义的智能查询;采用多根节点多点维护的数据资源副本管理机制,提高系统可靠性;基于关键字过滤的数据集成策略,减少通信代价;采用分布式聚类技术,实现大数据量信息的概要显示.通过实验验证了 DS_Grid 中所采用的关键技术的可行性和有效性.

关键词: 数据库网格;P2P(peer-to-peer);数据集成;资源发现;查询处理;副本管理

* Supported by the National Natural Science Foundation of China under Grant Nos.60473073, 60673139 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2003AA414210 (国家高技术研究发展计划(863))

Received 2006-06-10; Accepted 2006-08-25

中图法分类号: TP311 文献标识码: A

随着信息技术的发展,各行业的信息量呈爆炸性增长,其中包括众多公共有效的数据库资源,如高能物理、生物计算等科学研究领域、电子商务领域、深层 Web 数据查询领域等.地理上广泛分布的用户都希望能够按需地、透明地访问和使用这些丰富的数据资源[1,2].数据网格[2,3]是基于广域网对海量、分布异构的数据资源进行管理、访问和共享的系统.数据库网格[3,4]是随着公有数据库资源的丰富而提出的概念,是以数据库为主要资源的数据网格系统,可为上述应用提供良好的支持.一方面,利用网格环境的高效处理能力可以实现海量数据的有效整合,并有效地利用已有的众多的数据库资源;同时,也可以利用数据库管理系统高效的数据管理能力[5],为网格内实现数据库资源的有效管理、分布数据的集成优化以及大数据的分析处理等提供强有力的支持.

然而,目前有关数据库网格的研究和开发还处于起步阶段.几年来,最具有代表性的工作是隶属于全球网格论坛(global grid forum,简称 GGF)的 DAIS(database access and integration service)工作组[6]制定的网格环境下访问数据库的协议和中间件[6−9],以及针对特定应用数据库进行处理的网格系统[10−12].但是,已有工作大多是对静态数据库资源的访问与集成提供支持,在支持网格环境内数据资源的动态性方面讨论得很少.目前已有的针对特定应用的数据库处理的网格系统也是如此.虽然网格环境下针对数据库的处理技术与已有的多数据库、并行数据库以及分布式数据库的处理技术有很多相容之处,但由于网格环境内数据资源的不确定性(动态性),比如:存在哪些满足用户需求的资源需要实时发现;满足用户需求的数据集合大小只能在查询后才可知;异构数据资源的同构化规则事先无法预知等等.可见,已有的支持技术不足以支持构建具有不确定性的数据库网格环境.同时,网格环境下存在的这些不确定性也都给网格支撑环境下的数据资源管理、资源查询处理、数据集成优化、事务调度、大数据量分析等实现机制带来了困难,对研究者们提出了挑战.

本文基于 OGSA-DAI(open grid services architecture-data access and integration)规范,采用面向服务的思想,以数据库为主要数据资源,基于 P2P 框架结构,构建了一个面向多领域并支持动态数据集成的数据库网格系统——DS_Grid(database grid),目的是在网格环境下,借用网格的高效处理能力,为分布、自治、异构的数据库资源的有效管理、动态数据集成和分析处理等提供一个良好的使能环境,透明地为用户按需提供服务.本文主要讨论支持分布数据资源管理的 P2P 框架结构、多领域的资源管理(发布与发现)、数据资源的查询处理与集成优化、数据资源维护、大数据信息的概要可视化等,并给出相应的解决方案和实验验证.

1 相关工作

目前,有关数据库网格的研究和实践还处于起步阶段.典型的工作有 DAIS 工作组制定的网格环境下访问数据库的协议和中间件,如 OGSA-DAI[7],OGSA-WebDB(OGSA Web database)[8],OGSA-DQP(OGSA distributed query processing)[9]等.相关的工作有 MyGrid[10],Polar*[11],GDIS(grid data integration system)[12],POQSEC(parallel object query system for expensive computations)[13],CoDIMS-G(configurable data integration middleware for the grid)[14],PALADIN(pattern-based approach to large-scale dynamic information integration)[15],DartGrid[16],SDG(scientific data grid)[17]等.OGSA-DAI 能无缝地实现数据库与网格的集成,包括关系数据库和 XML 数据库等;OGSA-WebDB 基于 OGSA-DAI 提供访问与集成 Web 数据库能力;OGSA-DQP 是基于 OGSA-DAI,并面向并行处理的查询处理机制;Polar*是支持特定领域的科学网格,也是基于 OGSA 体系结构,并预知数据资源;CoDIMSG 是中间件查询系统,主要基于吞吐率动态协调查询处理节点;MyGrid 是英国 e_science 核心项目的代表,为生命科学研究提供了一套中间件软件,其基于英国 OGSA-DAI 开发的 OGSA-DQP 实现数据库的访问和集成;GDIS 采用 OGSA-DQP,OGSA-DAI 和 Globus Toolkit 3[18]中间件,并基于服务框架实现 XML 数据集成.POQSEC 透明地实现科学数据查询和数据分析,其数据包装为原始数据格式,而不是 SQL 数据库数据,但提供类似 SQL 的查询处理机制;PALADIN 基于图匹配引擎实现数据集成.DartGrid 是针对中医药应用构建的数据库网格环境,实现数据库的服务化访问和数据的分布查询,主要工作在语义层;SDG 是面向科研数据处理构建的数据网格,其基于 JDBC 实现与数据库的连接,并提供统一的访问接口实现异构数据集成.

以上工作大多是针对特定领域并基于静态环境构建的数据库网格系统,没有适应网格环境内数据库资源的不确定性的相关讨论,也没有看到有关完善的支持动态数据集成的数据库网格的报道.本文探讨了一个支持多领域的、面向数据库资源动态集成的数据网格系统.本文的主要贡献在于:提出了基于 MultiChord 的体系结构,有效地支持了分领域管理分布的数据库资源;提出了基于松驰的模式匹配的服务发布和查询机制,有效地提高了获取资源的精度,实现了基于本体的智能查询处理机制;提出了基于过滤关键字的集成策略,通过减少数据通信量,有效地提高了查询处理的效率;验证了多根节点多点维护的副本管理机制的有效性;给出了分布的聚类分析策略,有效地提高了大数据量的分析处理效率.

本文第 2 节介绍 DS_Grid 数据库网格系统.第 3 节介绍支持 DS_Grid 的 P2P 体系结构.第 4 节给出基于本体的智能查询处理机制.第 5 节讨论资源服务的发现与发布.第 6 节为多根节点多点维护的副本管理机制.第 7 节介绍基于关键字过滤的数据集成策略.第 8 节为基于分布聚类分析的数据分析策略.第 9 节给出相关实验.第 10 节总结全文.

2 DS_Grid 数据库网格介绍

面向数据库的数据网格是随着网格的发展和应用需求而提出的.DS_Grid 的目的是利用网格的高效性能实现数据库资源的有效共享,按需为用户提供增值的服务.针对网格环境的动态性、自适应性和高效的处理能力以及数据库资源的动态、自治、异构等特性,DS_Grid 采用如下主要思想:采用面向服务思想,将数据库资源包装为 Grid 服务,方便异构数据资源存取;采用 P2P 体系结构,充分利用网格内的分布资源,提高网格的效率;在 P2P 框架下,基于领域本体知识,分领域管理数据资源,提高资源发现效率;基于相似匹配和松驰的服务发布与发现策略,扩大资源的定位范围;基于领域本体定义全局数据模式,有效地实现异构数据集成,提高集成结果的准确性;基于传输最小数据量规则定义有效的查询和数据集成策略,降低数据集成时间代价;采用副本管理策略增强网格的可靠性和资源查询效率;基于分布的数据挖掘策略,提高数据分析与处理的效率.DS_Grid 数据库网格系统的体系结构如图 1 所示.

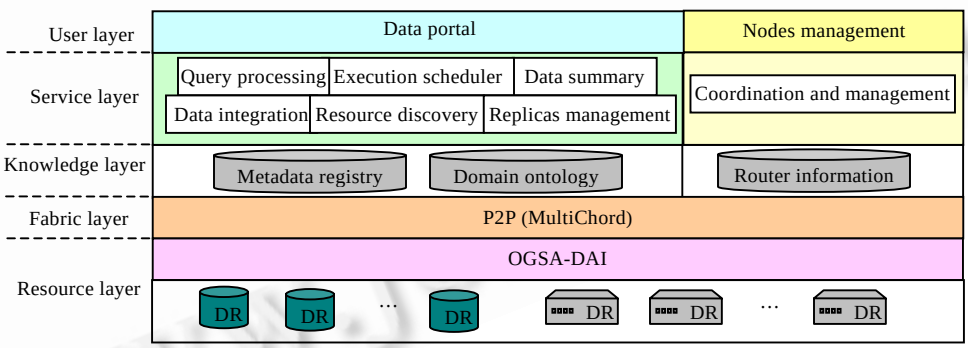

Fig.1 DS_Grid architecture

图 1 DS_Grid 体系结构

基于 OGSA-DAI 将数据库资源包装为 Grid 服务,并注册于元数据仓(metadata registry)中,后文简称数据库资源为 Grid 服务或服务;基于 JXTA(juxtapose)构建 MultiChord 框架结构;采用 Shunsaku XMLManager 为服务资源管理仓.各主要服务组件功能介绍如下:

节点协调与管理(nodes coordination and management):所有管理节点组成双层的 Chord 环结构,即 MultiChord 结构.节点上存储着相应的资源定位元信息,节点之间互相影响,并保存着各自的路由表.随着节点的加入和退出,维护 P2P 结构的正确性和完整性.

查询处理(query processing):基于领域本体知识,扩展全局请求语义,并分解为全局子模式信息;基于所发现的服务资源信息进行查询重写,生成由多个子查询组成的查询执行计划.

资源发现(resource discovery):以每一个全局子模式信息为单位,基于语义实现领域匹配和模式匹配,按需发现资源服务集.

执行调度(execution scheduler):调度查询执行计划中各子查询到相应的执行场地,并全局协调各子查询的执行.

数据集成(data integration):依据子查询执行计划,基于传输最小数据量的启发式集成策略,实现各个场地的数据的最优化集成.

副本管理(replicas management):采用多根节点多点备份思想对数据进行备份,保证服务资源元数据的完整性和有效性,并提高网格数据服务的查询效率.

数据概要(data summary):基于分布的数据处理策略,对集成结果数据进行分析,抽象概要数据并可视化.

网格门户(data portal):用户通过门户提交请求,并通过门户可视化数据的集成结果.

3 MultiChord 系统结构

P2P 对等网络结构是目前分布式系统的典型支撑结构,也是 DS_Grid 的首选框架结构.支持 P2P 框架的 Chord 算法[19]以其简单性、可证明性备受关注.然而,数据库网格中通常一个查找包含多个模式,如果使用 Chord 算法,需要多次大范围的多节点查找,效率低下.同时,也无法区分领域信息.为此,本文基于分领域管理资源的思想,从结构上对 Chord 进行改进,提出了双层的 Chord 算法——MultiChord.基于 MultiChord,可方便支持资源分领域管理,并通过缩减查找的总 hop 数,达到提高查询效率、减轻系统负载的目的.

MultiChord 把查找过程分为两步:首先定位领域所在的小环,缩小下一步对模式信息查找的节点范围;之后,在领域所对应的小环内查找匹配的模式信息.具体查找过程如图 2 所示.假设 MultiChord 环中有 20 个节点,每个节点的标识符由 5 比特组成,所有的节点组成一个大 Chord 环,节点标识符前 2 比特相同的节点组成小 Chord 环,形成双层环.每个节点拥有两个后继和前驱节点表.

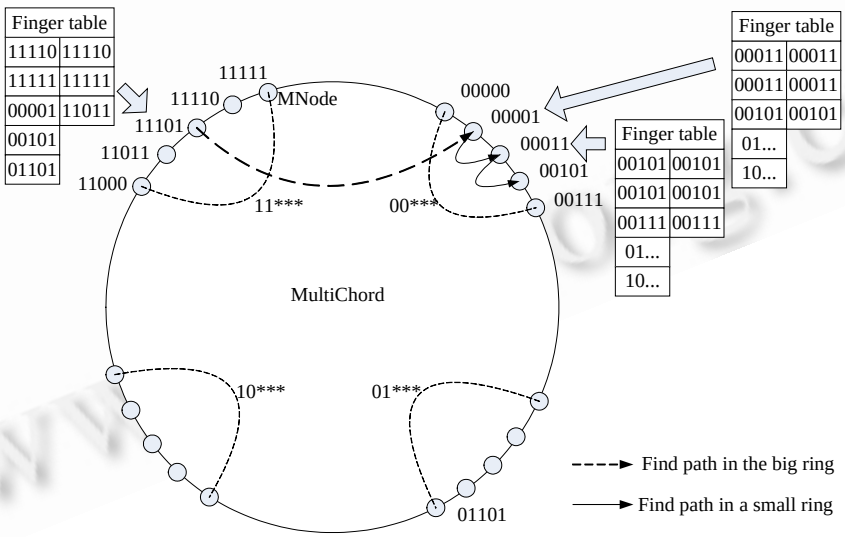

Fig.2 Structure sketch map of MultiChord

图 2 MultiChord 结构示意图

设 N 为节点个数,K 为领域关键字标识的比特数,2^k 为小环个数,S 为一次查询请求的模式个数,则使用 Chord 算法的平均 hop 数为 $S\times\log_2 N$,用 MultiChord 算法的平均 hop 数为 $\log_2 N+S\times\log_2(N/2^k)$.假设 N 不变,当 $S>(\log_2 N/k)$ 时,使用 MultiChord 的效率高于使用 Chord.

在 DS_Grid 内的数据资源发现过程中往往涉及多个模式、多个属性,即所有资源发现都是基于多元素 Hash 实现的,因此采用 MultiChord 是合理的.

4 基于本体的智能查询处理机制

面对分布环境下大量的自治、异构并具有丰富语义的数据库资源,若无任何遵循标准,实现异构数据集成则很难达到良好的效果.为此,DS_Grid 中基于领域本体(domain ontology)为数据资源的语义一致性规范.领域本体是由领域专家定义的领域概念,如文献领域、微型轿车领域等.本文基于领域本体定义领域概念和全局模式,通过 XML 异构数据源到全局模式的映射实现异构转换,并基于本体实现智能查询.

4.1 领域概念和全局模式本体

领域概念基于领域本体定义,具体定义为 $DO=\{(D_1,DD_1),(D_2,DD_2),\ldots,(D_n,DD_n)\}$,其中,$DO$ 为领域概念集合,DD_i 为领域描述信息,D_i 为对应的领域关键字.

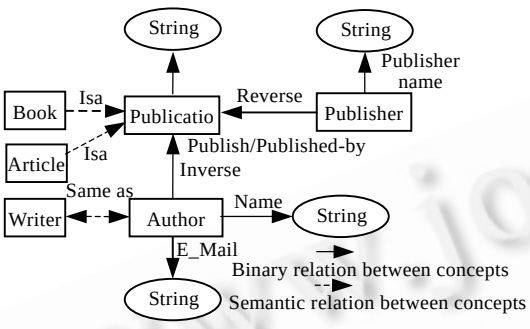

Fig.3 Publication domain ontology

图 3 Publication 领域本体

全局模式本体[20]基于特定领域本体定义,本文将领域内的全局模式本体定义为一个四元组 $SO=(C,R,A,r)$,其中:C 是特定领域的一个概念集;R 是概念间双向的二元联系;A 是概念属性集;r 是概念间的语义关系集,用来描述该领域的概念及其之间的关系.具体语义关系规则有:类的等价关系(same_as)、类的继承关系(isa)、逆反关系(inverse)、传递关系(transitive)、对称关系(symmetry).图 3 表达了一个文献领域的本体,当在文献领域进行查询时,其作为全局模式,基于以上语义规则以及本体知识进行推理,实现基于语义的智能查询.

4.2 XML异构数据源到全局模式的映射

全局查询引擎接受来自于用户接口的基于全局模式本体的查询语句,基于本体知识进行查询转换,并将基于本体的查询转化为基于 XML 的查询.当本体与 XML 进行映射时,主要基于以下规则:

(1) XML 的复杂元素映射为本体的概念,对应于本体概念关系图中的非叶子节点;

(2) 本体概念关系图中的叶子节点为概念的属性,对应于 XML 文档的简单数据类型.概念间的二元联系用于表达 XML 文档中复杂数据类型之间的关系;

(3) 通过全局本体对概念抽象,并实现本体与 XML 文档之间的映射,从而屏蔽不同 XML 文档间的结构异构,给用户以统一的查询界面.

具体映射规则定义为 $R:p–>u/q:v$,其中,R 是规则的标签,u 是规则的根,q 是 XPath 的局部路径,p 是本体的模式路径.规则的根(u)既可以是变量,也可以是 URI.如果 u 是变量,则 p 是角色路径;否则,p 是概念路径.如果 u 是变量,则规则 R 称为相对映射规则;否则,称为绝对映射规则.如图 4 中数据源 Source1,则全局模式本体(如图 3 所示)路径与局部模式路径之间的映射如图 5 所示.

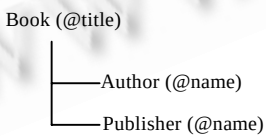

Fig.4 XML data source Source 1

图 4 XML 数据源 Source 1

R1:Book–>doc(source1)/book:*v*1
R2:title–>*v*1/@title:*v*2
R3:written_by, Authour–>*v*1/Authour:*v*3
R4:name–>*v*3/name:*v*4
R5:published_by, Publisher–>*v*1/publisher:*v*5
R6:publisher_name–>*v*5/name:*v*6

Fig.5 Mapping of global schema to Source 1

图 5 全局模式本体到 Source 1 的映射

4.3 基于本体的智能查询处理

本体所提供的丰富语义信息和推理能力为用户的查询处理提供了强大的智能性.如图 3 所示,Author 和 Writer 是同一个概念,Writer 可以拥有 Author 类的全部特性.Book,Article 作为 Publication 的子类,对 Publication 的查询可以自动地扩展到所有的子类中去.

基于本体的智能查询处理步骤如下:

(1) 接收用户发出的基于概念(全局模式)的查询,根据本体中概念间的语义关系,生成所有相联系的概念集合 $C=\{c_1,c_2,\ldots\}$;

(2) 对概念集 C 中所有概念,根据其语义关系进行标准化与泛化处理.如等价概念之间的规范化,父子关系、包含与被包含关系等的扩展查询,实现对查询条件的扩展;

(3) 对概念集 C 中所有概念间的二元联系建立二元联系集合 $R=\{r_1,r_2,\ldots\}$;

(4) 对集合 R 中的所有联系,根据已定义的语义规则进行推理.如逆反、对称、传递等规则间的变换与推理,获得更深一层的语义关系;

(5) 通过前 4 步的处理,生成新的语义扩展后的查询;

(6) 以查询子模式为单位发现相应的服务资源,并分解为多个子查询模式的集合.

基于本体与发现的资源的 XML 之间的映射关系,将基于全局模式的查询重写为基于各数据源的 Xpath 查询,并生成查询执行计划.

5 资源发现与发布

发现恰当的服务是网格为用户提供高质量集成数据信息的基础保证.DS_Grid 中的服务发现与发布过程基本一致.首先,进行等价语义扩展,并基于相似匹配度确定资源所属的领域,即找到对应的小环;之后,在领域内基于松驰定位策略,确定相应的全局模式本体和相应的 Peer 点,即在小环内找到模式信息对应的节点;最后,实现资源复制发布或基于模式匹配结果找到所有相似匹配的 Grid 服务的元信息.

5.1 基本概念

定义 **1**. 单词条元素相似匹配度($editSim(a,b)$)[21].基于编辑距离定义相似度,具体定义为

$$editSim(a,b)=1-\frac{ed(a,b)}{\max\{|a|,|b|\}} \tag{1}$$

其中,a,b 分别为两个词条;$\max\{|a|,|b|\}$表示词条 a,b 中较长一个字符的长度.

定义 **2**. 向量元素相似匹配度($EleVecSim(V_1,V_2)$)[21].设有向量元素 V_1,V_2,σ_l 为向量元素相似匹配阈值,则向量元素相似度匹配定义为

$$EleVecSim(V_1,V_2)=\sum_{\langle t,s\rangle\in Close(\sigma_l,V_1,V_2)} w(V_1,t)\times w(V_2,s)\times editSim(t,s) \tag{2}$$

其中,$Close(\sigma_l,V_1,V_2)=\{\langle a,b\rangle|\exists a\in V_1\wedge\exists b\in V_2\wedge editSim(a,b)>\sigma_l\}$,$w(V,r)$表示词条 r 在元素向量 V 中的权重,$w(V,r)=\log(tf_{V,r}+1)\cdot\log\left(\frac{N}{df_r}+1\right)$,$tf_{V,r}$ 表示词条 r 在 V 中的出现频率,N 表示该模式对应的模式元素个数,df_r 表示词条 r 在该模式所对应的所有模式元素向量中的出现频度.

定义 **3**. 相似匹配矩阵($CMSim(s,p)$).令 S 和 P 是两个向量元素,$S=\{s_1,s_2,\ldots,s_n\}$,$P=\{p_1,p_2,\ldots,p_m\}$,s_i 表示 S 中的第 i 个元素;p_j 为 P 中第 j 个元素,矩阵元素 $m_{ij}=EleVecSim(s_i,p_j)$,若 $EleVecSim(s_i,p_j)<\sigma_l$,令 $EleVecSim(s_i,p_j)=0$,则

$$CMSim(s,p)=\begin{bmatrix} m_{11} & \ldots & \ldots & m_{1m} \\ \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ m_{n1} & \ldots & \ldots & m_{nm} \end{bmatrix}.$$

定义 **4**. 矩阵相似匹配度($\sigma_{MSim}(S,P)$).由矩阵中各相似元素的平均相似度值来描述,并充分考虑元素间的一对一关系.基于定义 3,令 h_i 为矩阵第 i 行中不为 0 的元素个数,v_j 为矩阵第 j 列中不为 0 的元素个数,则矩阵相似度定义为

$$\sigma_{MSim}(S,P)=\frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{m} m_{ij}/h_i+\sum_{j=1}^{m}\sum_{i=1}^{n} m_{ij}/v_j\right) \tag{3}$$

5.2 服务资源发布与发现

为提高发现服务的效率和精度,在发布模式(service provider,简称 SP)元信息时,计算 SP 与匹配的本体模式(S_i)的相似匹配度阈值.在发现服务时,只需计算请求模式(service requester,简称 SR)与领域内所有本体模式的相似匹配度,同时利用各发布模式的相似匹配度阈值,获得满足 SR 的服务集合.免去了 SR 与所有 SP 的繁琐相似度计算.

服务资源发布由以下 5 步实现:

步骤 1. 确定领域.令 DR 为请求描述信息,D_r 为获得的匹配领域关键字,则 $D_r=\{D_i|EleVecSim(DD_i,DR)=\max\{EleVecSim(DD_1,DR),\ldots,EleVecSim(DD_n,DR)\}\}$,$D_r$ 为根据 DR 描述信息确定的领域标识,即与 DR 相似度最大的领域关键字.

步骤 2. 确定候选的全局本体模式集合($CMS(SP)$).令 $SO=\{S_1,S_2,\ldots,S_n\}$ 描述领域本体全局模式,$S_i=\{s_{i1},s_{i2},\ldots,s_{in}\}$,其中,$s_{ij}$ 表示本体模式 S_i 中的第 j 个属性元素;发布资源 $SP=\{sp_1,sp_2,\ldots,sp_m\}$,$sp_j$ 为 SP 中第 j 个属性元素,$(\sigma_{ls},\sigma_{la})$为模式匹配下线阈值对,则 $CMS(SP)$定义为 $CMS(SP)=\{S_i|editSim(S_i,SP)>=\sigma_{ls}\wedge\sigma_{MSim}(S_i,SP)>=\sigma_{la}\}$.

步骤 3. 确定 SP 的匹配模式集合($MS(SP)$).令 $\sigma_i=\sigma(S_i,SP)=EleVecSim(S_i,SP)^*w_1+\sigma_{MSim}(S_i,SP)^*w_2$,并按 σ_i 降序排列 $CMS(SP)$中的本体模式,σ_k 是第 k 个模式的综合匹配度,取前 k 个本体匹配模式组成为 SP 的匹配模式集合 $MS(SP)$,则 $MS(SP)=\{S_{ji}|S_i\in CMS(SP)\wedge S_i=S_{ji}\wedge\sigma_i>=\sigma_k\wedge\sigma_{ji}=\sigma_i\wedge\sigma_{ji}>=\sigma_{(j+1)i}\}$.

步骤 4. 基于 $MS(SP)$中的全局本体模式实现多副本发布.

步骤 5. 计算发布模式 SP 的当前本体模式匹配阈值对$(\sigma_s(SP),\sigma_a(SP))$.

$(\sigma_s(SP),\sigma_a(SP))=\{(\min(EleVecSim(S_i,SP)),\min(\sigma_{MSim}(S_j,SP)))|S_i,S_j\in MS(SP)\}$,$(\sigma_s(SP),\sigma_a(SP))$为后续发现 SP 提供推荐的模式匹配阈值.

基于以上服务资源发布过程,服务发现过程具体由以下 4 步实现:

步骤 1. 令请求模式为 $SR=\{sr_1,sr_2,\ldots,sr_m\}$,首先确定所在的领域 Dr(同发布过程).

步骤 2. 令 $SO=\{S_1,S_2,\ldots,S_n\}$为领域 Dr 所包括的领域本体模式,计算所有本体模式与请求服务模式的相似匹配度($EleVecSim(S_i,SR)$)和矩阵相似匹配度($\sigma_{MSim}(S_i,SP)$).

步骤 3. 确定候选的全局本体模式集($CMS\text{-}Set(SR)$)[22].

$$CMS\text{-}Set(SR)=\{RM(S_1),RM(S_2),\ldots,RM(S_n)\},$$

$$RM(S_j)=\{SP_i|(\sigma_s(SP_i)+EleVecSim(S_j,SR))/2>=0.5\wedge(\sigma_a(SP_i)+\sigma_{MSim}(S_j,SR))/2>=0.5\}.$$

步骤 4. 得到最后的模式匹配集($RMS(SR)$).$RMS(SR)=\{RM(S_1)\cap RM(S_2)\cap\ldots\cap RM(S_n)\}$.

6 基于关键字过滤的数据集成策略(filter based distributed data integration,简称 FDDI)

网格环境中存在着众多分布、动态、自治的数据库资源,其中存在大量冗余的数据信息,因此,按需发现的数据资源也必然会存在大量的冗余数据.若不经筛选把所有数据直接返回到任务的发起节点进行数据集成,则需要传输许多重复数据,造成带宽浪费,也增加了任务的执行时间.为此,我们基于传输最小数据量的启发式优化思想[23],提出了基于关键字过滤的数据集成策略,目的是通过降低数据的传输时间代价,达到提高数据集成效率的目的.FDDI 的基本思想是:(1) 按需传输数据.在节点之间不是直接传送 XML 数据结果,而是先将结果中的关键字传送回任务发起节点,经过任务发起节点筛除之后,将需要的数据关键字返回,再按需传输所需要的数据.(2) 综合考虑节点之间的数据传输的时间代价,基于"能者多劳"的思想,传输性能越好的传输路径传输的数据量越多.具体描述如下:

假设有 n 个节点参与完成一项任务,$D=\{D_1,D_2,\ldots,D_n\}$为各点返回的数据的偏序集,D_i 是第 i 个节点返回的 XML 数据结果,$K=K_1\cup K_2\cup\ldots\cup K_n$ 是对应 D 中各数据集的关键字集合,K_i' 是第 i 个点需要返回的关键字集,$K_i'=\left(K-\bigcup_{j=1}^{i-1}K_i\right)\cap K_i$,使得越早返回数据的节点需要传输的数据越多,从而降低数据传输的总的时间代价.

7 多根节点、多点维护的副本管理机制

在动态网格环境下,Peer 点可以随时加入和退出,数据复制通常是保证资源有效的主要手段.在结构化的P2P 系统中,单根节点存在单点失效问题,为此,我们提出了多根节点[24]概念.DS_Grid 中采用多根节点、多点维护的副本管理策略.其基本思路是:每个节点都维护 M(M>=1)个与自己节点 ID 值最接近的节点(称为 M 邻居集),并将自己主管的对象索引都存放在这些节点上.如果原根节点一直在系统中,那么,对象定位请求和对象索引发布都能路由到该节点上,所有节点的原有功能都不会发生变化,只是增加了根节点定时地维护 M 邻居集,并向它们发布对象索引更新信息的操作.当原根节点退出时,路由过程会自动定位到新的根节点,只要新的根节点属于原根节点的 M 邻居集,则对象定位会无缝地完成.当下次新的根节点维护自己的 M 邻居集时,会将对象索引信息扩散到自己的 M 邻居集节点中.因此,只要控制好节点维护 M 邻居集的时间周期,就能保证动态网络中对象定位的高成功率.

通过实验(见第 9.3 节)确定,DS_Grid 中采用副本数量 M=2,根节点数 N_r 为 4 的副本管理策略.

8 基于分布聚类分析的数据概要策略

面对由大量的数据库资源返回的海量的集成结果数据,需要进一步抽象概要结果,之后,再由网格门户将概要结果展示给用户.在 DS_Grid 中,我们采用简单的基于特定属性的 k-平均聚类分析算法(根据应用需求可选择相应的数据分析算法)进行聚类分析.由于聚类分析在系统的数据处理过程中占有重要比重,若集中进行聚类分析,随着数据量和并发用户数的增加,系统的性能急剧下降.为此,DS_Grid 中提出了采用分布式聚类分析数据的处理策略[25],并将数据处理分为数据合成层和数据分析层.由数据合成层实现数据的整合,保证合成后的数据满足用户的模式需求,之后,在相同模式的基础上实现数据的一次聚类分析和二次聚类分析,通过利用 P2P 的分布计算能力,达到缓解集中处理瓶颈和提高网格内数据处理效率的目的.

如图 6 所示,n 层(二次聚类)和 n–1 层(一次聚类)分别实现一次聚类和二次聚类;n–1 层以下存在的多层(1~n–2)为数据合成层,保证 n–1 层的数据具有完好的模式信息.令 $S(n)$表示 n 层的数据模式,则 $S(n)$=$S(n$–1),$S(i)=\overset{m}{\underset{l=1}{\infty}}S_l(i-1)$,即在 n–2 层以下进行数据整合.当满足数据结果模式需求($S(n)$)时,进行一次聚类分析,并将一次聚类结果提交给上一层,进行二次聚类分析,得到最终的聚类分析结果,返回给服务器.

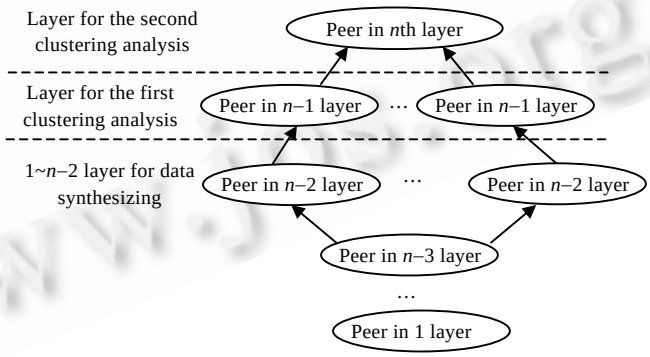

Fig.6 Distributed cluster analyzing process

图 6 分布的聚类分析过程

9 实验测试

本节针对 DS_Grid 中提出的部分关键技术进行实验测试,主要包括对比 MultiChord 和 Chord 性能、比较 FDDI,CDI(centralized data integration)和 DDI(distributed data integration)集成策略,确定多根节点(N_r)和多副本(M)的数量,比较不同节点数时分布式聚类分析的时间代价.

9.1 MultiChord和Chord的性能比较

本实验对 MultiChord 和 Chord 方法进行了仿真测试.我们测试了小环个数分别为 50,100,150 的 hop 数的变化情况.横坐标为节点个数,纵坐标为平均查找每个模式需要的 hop 数,如图 7 和图 8 所示.

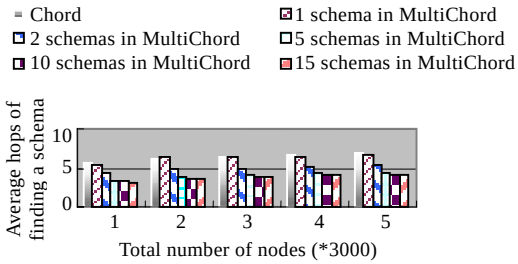

Fig.7 Average hops of querying one schema in a small chord with 50 nodes

图 7 查询一个模式的平均 hop 数(小环内 50 个节点)

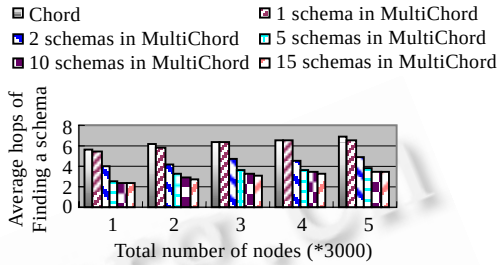

Fig.8 Average hops of querying one schema in a small chord with 150 nodes

图 8 查询一个模式的平均 hop 数(小环内 150 个节点)

由图 7 和图 8 可知:在一次查询包含多个模式时,使用 MultiChord 比 Chord 效果明显要好,并随着模式个数、属性个数的增加,优势越发显著;小环的个数对平均查找每个模式需要的 hop 数没有明显影响.但小环越多,就越会降低系统中的负载平衡.

鉴于领域个数有限,并且在数据服务资源的发现过程中往往涉及多个模式、多个属性,因此,采用 MultiChord 是合理的.

9.2 FDDI,CDI和DDI策略的比较

本实验将关键字过滤的数据集成策略与另外两种集成策略(集中的集成策略(CDI)和分布的集成策略(DDI))进行了对比.CDI 是将一个执行任务的所有数据网格服务都在一个节点上运行调用和集成.DDI 是将数据网格服务的调用分散到多个节点上执行,调用的结果数据再传送给分发任务的节点进行数据集成.图 9~图 11 分别表示了数据冗余度为 0%,10%,20%的情况下,执行同一个任务所需要的执行时间.横坐标为数据量,纵坐标为时间.可以看出,随着数据冗余度的增加,基于关键字过滤的集成策略显示了其优势.即通过减少大量的数据传输,有效缩短任务的执行时间.可见,FDDI 策略应用于存在大量冗余信息的数据网格中是可行的.

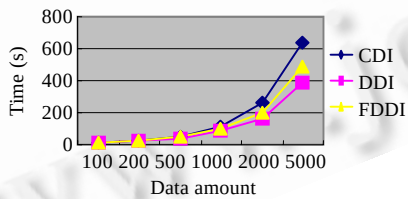

Fig.9 Comparison with no redundancy

图 9 无冗余数据的比较

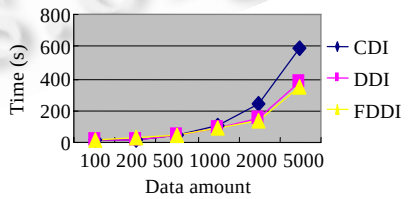

Fig.10 Comparison with 10% redundancy

图 10 10%冗余数据的比较

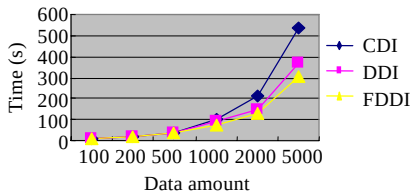

Fig.11 Comparison with 20% redundancy

图 11 20%冗余数据的比较

9.3 多根节点(N_r)和多副本(M)的数量的选择

为确定适当的多根节点和多副本的数量,我们基于 NS 模拟网络环境进行了一系列实验,通过实验得到单根节点下副本(M)为 2 和 4 时的访问效率接近(图略).为此,我们选 M=2 和 M=4,并分别测试了根节点和维护节点之间的相互影响.如图 12 和图 13 所示.

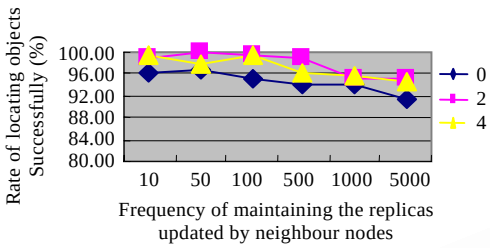

Fig.12 Index of object republished per 100 seconds, M=2

图 12 每 100 秒重新发布一次对象索引,M=2

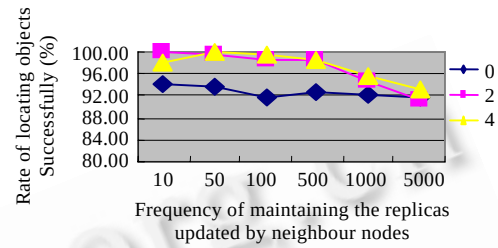

Fig.13 Index of object republished per 1 000 seconds, M=4

图 13 每 1 000 秒重新发布一次对象索引,M=4

从图 12 和图 13 可知,多根节点可极大地提高系统的访问效率.当 N_r 为 4 时效果较好,并没有因为邻居节点维护时间变大而发生很大的变化.可见,多根节点、多点维护使得系统访问成功率有了很大的提高.当 N_r 为 2 和 4 时,曲线受邻居节点更新时间的变化比单根节点时要小.当 M=4,N_r=4 时,系统中共有 16 个该数据的备份,可以有效地提高系统的健壮性和对象索引定位的成功率.但当副本过多时,则增加了维护副本一致性的代价.

图 14 为不同副本和根节点数的平均维护代价.横坐标是备份的方案,如(1,0)表示 1 个根节点,0 个维护节点;纵坐标为代价.假设每一次数据访问的代价为 P,t 为邻居节点更新的时间,$N/2^k$ 为小环的节点个数,M 为副本数量,N_r 为根节点数量,则平均每秒的代价为 $C(t)=[(P/\log_2(N/2^k))/t]\times(M+N_r)$.若$\sigma(t)$为访问成功率,则总的评价模型定义为$\Omega=\Sigma(C(t)\times\sigma(t))$,实验中,$t$=[10,50,100,500,1000,2000],单位为 s,N_r=M=[0,2,4].

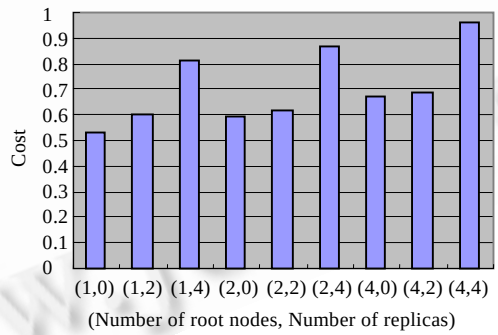

Fig.14 Average maintaining cost in different cases

图 14 不同情况下的平均维护代价

通过模拟实验可以得出,较合适的副本方案是 N_r 为 4,M 为 2.该方案的代价适中(如图 14 所示),而且访问效率的曲线很平稳(如图 12 和图 13 所示).

9.4 分布式聚类分析时间代价比较

在局域网环境下,我们模拟了 20 000 条数据进行聚类分析,测试了数据聚类分析的响应时间,测试结果如图 15 和图 16 所示.

从图 15 和图 16 可知,随着数据集的记录数的增加,分层聚类分析具有时间优势.若在广域网环境中,由于数据文件的传输代价较大,则将更能体现分层聚类的优势.

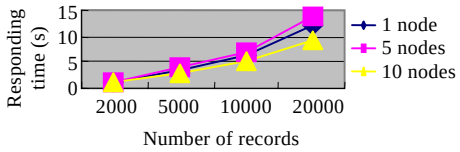

Fig.15 Comparison of time cost of cluster analyzing (5 clusters)

图 15 聚类分析的时间代价比较(5 个聚类)

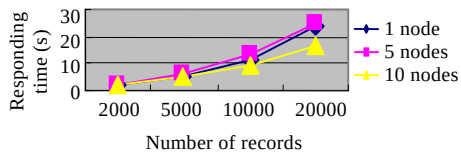

Fig.16 Comparison of time cost of cluster analyzing (10 clusters)

图 16 聚类分析的时间代价比较(10 个聚类)

10 结 论

本文介绍了一个面向多领域的、支持动态数据集成的数据库网格系统 DS_Grid.在数据资源管理中,系统采用面向服务的思想,将数据库资源包装为 Grid 服务,并基于 P2P MultiChord 体系结构,实现了数据资源的分领域管理,同时,采用多根节点多点维护的数据资源副本管理机制,有效地提高了系统可靠性;在服务发布和发现中,基于文本相似性确定资源所属的领域和对应的本体模式,并将资源松弛定位到相应的领域小环中的相应节点,实现了服务分领域注册和服务快速发现,有效地提高了资源发现的精度.在查询处理过程中,以本体为领域全局模式,基于领域本体知识和推理规则实现异构消解和语义扩展,并结合发现的服务资源实现了查询重写和查询分解.在数据资源合成过程中,基于关键字过滤冗余信息,有效地提高了数据集成效率,并基于分布的数据挖掘技术实现了大数据量信息的概要抽象.DS_Grid 网格系统由国家高技术研究发展计划(863)资助,该系统由东北大学软件研究所研发,并应用于中国科学院自动化所研发的网络化故障监测(condition-based maintenance,简称 CBM)平台[26]的数据管理中.CBM 平台实时监控设备的运行状态信息,并按数据类别分别存储于不同的数据库中.各数据库资源包装为 Grid 服务,其描述信息注册于 DS_Grid 的元数据仓中.由 DS_Grid 环境对监测到的数据进行处理与分析,主要实现设备的故障分析与预测.

下一步,我们将针对支持数据集成的动态、自适应的多目标的代价模型,在按需获取高质量数据的评价模型以及大数据量结果概要等方面进行深入研究.

References:

[1] Wang S, Zhang KL. Database system on the grid. Computer Applications, 2004,24(10):1−3 (in Chinese with English abstract).

[2] Yang DH, Li JZ, Zhang WP. Join algorithm based on data grid. Journal of Computer Research and Development, 2004,41(10): 1848−1855 (in Chinese with English abstract).

[3] Ren H, Li ZG, Xiao N. Database grid: The multi-database built on the grid. Computer Engineering and Applications, 2006,42(2): 171−175 (in Chinese with English abstract).

[4] Meng XF, Zhou LX, Wang S. State of the art and trends in database research. Journal of Software, 2004,15(12):1822−1836 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/1822.htm

[5] Watson P. Databases and the grid. 2002. http://www.nesc.ac.uk/technical_papers/

[6] Global grid forum: Grid data service specification. 2004. https://forge.gridforum.org/projects/dais-wg

[7] The OGSA-DAI project. 2005. http://www.ogsa-dai.org.uk/

[8] Kojima I, Pahlevi SM. Design and implementation of OGSA-WebDB. 2004. http://www.nesc.ac.uk/events/GGF10-DA/

[9] OGSA-DQP. 2004. http://www.ogsa-dai.org.uk/dqp/

[10] MyGrid. 2001. http://www.mygrid.org.uk/

[11] Smith J, Gounaris A, Watson P. Distributed query processing on the grid. The Int'l Journal of High Performance Computing Applications, 2003,17(4):353−367.

[12] Comito C, Talia D. XML data integration in OGSA grids. In: Pierson JM, ed. Data Management in Grids: 1st VLDB Workshop, DMG 2005. Heidelberg: Springer-Verlag, 2005. 16−29.

[13] Fomkin R, Risch T. Framework for querying distributed objects managed by a grid infrastructure. In: Pieon JM, ed. Drsata Management in Grids: 1st VLDB Workshop, DMG 2005. Heidelberg: Springer-Verlag, 2005. 58−72.

[14] Porto F, Silva VFV, Dutra ML, Schulze B. An adaptive distributed query processing grid service. In: Pierson JM, ed. Data Management in Grids: 1st VLDB Workshop, DMG 2005. Heidelberg: Springer-Verlag, 2005. 45−57.

[15] Göres J. Towards dynamic information integration. In: Pierson JM, ed. Data Management in Grids: 1st VLDB Workshop, DMG 2005. Heidelberg: Springer-Verlag, 2005. 16−29.

[16] Dartgrid. 2003. http://ccnt.zju.edu.cn/projects/dartgrid/

[17] Scientific data grid (SDG). 2005. http://www.sdg.ac.cn/document/sdg/

[18] Sandholm T, Gawor J. Globus toolkit 3 core—A grid service container framework. 2003. http://www-unix.globus.org/toolkit/3.0/ogsa/docs/gt3_core.pdf

[19] Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for internet applications. 2001. http://pdos.csail.mit.edu/papers/chord:sigcomm01/chord_sigcomm.pdf

[20] Falbo RA, Guizzardi G, Duarte KC. An ontological approach to domain engineering. In: Proc. of the 14th Int'l Conf. of Software Engineering and Knowledge Engineering. New York: ACM Press, 2002. 351−358.

[21] Bilke A, Naumann F. Schema matching using duplicates. In: Kawada S, ed. Proc. of the 21th Int'l Conf. on Data Engineering. Los Alamitos: IEEE Computer Society Press, 2005. 69−80.

[22] Madhavan J, Bernstein PA, Rahm E. Generic schema matching with cupid. 2001. http://research.microsoft.com/~philbe/CupidVLDB01.pdf

[23] Ng WS, Ooi BC, Tan KL, Zhou AY. PeerDB: A P2P-based system for distributed data sharing. In: Dayal U, ed. Proc. of the 19th Int'l Conf. on Data Engineering (ICDE). Bangalore: IEEE Computer Society Press, 2003. 633−644.

[24] Zhao BY, Kubiatowicz J, Joseph AD. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report, UCB/CSD-01-1141, Berkeley: University of California, 2001.

[25] Keim DA, Panse C, Schneidewind J, Sips M, Hao MC, Dayal U. Pushing the limit in visual data exploration: Techniques and applications. In: Günter A, *et al*., eds. KI 2003: Advances in Artificial Intelligence. Heidelberg: Springer-Verlag, 2003. 37−51.

[26] Guo QJ, Yu HB, Wu K. Research & application of distributed condition-based maintenance open system. Computer Integrated Manufacturing Systems, 2005,11(3):416−421 (in Chinese with English abstract).

附中文参考文献:

[1] 王珊,张坤龙.网络环境下的数据库系统.计算机应用,2004,24(10):1−3.

[2] 杨东华,李建中,张文平.基于数据网格环境的连接操作算法.计算机研究与发展,2004,41(10):1848−1855.

[3] 任浩,李志刚,肖侬.数据库网格:基于网格的多数据库系统.计算机工程与应用,2006,42(2):171−175.

[4] 孟小峰,周龙骧,王珊.数据库技术发展趋势.软件学报,2004,15(12):1822−1836. http://www.jos.org.cn/1000-9825/15/1822.htm

[26] 郭前进,于海斌,徐皑.基于状态维修的开放系统研究与实现.计算机集成制造系统,2005,11(3):416−421.

申德荣(1964－),女,辽宁铁岭人,博士,教授,CCF 高级会员,主要研究领域为数据网格,Web 服务.

聂铁铮(1980－),男,博士生,主要研究领域为分布式数据处理.

于戈(1962－),男,教授,博士生导师,CCF 高级会员,主要研究领域为数据流,数据挖掘,分布式数据库.

寇月(1980－),女,博士生,主要研究领域为数据网格.