第32卷 第22期 计 算 机 工 程 2006年11月
Vol.32 No.22 Computer Engineering November 2006
·软件技术与数据库· 文章编号：1000—3428(2006)22—0055—03 文献标识码：A 中图分类号：TP311.13

基于 CORBA 的并行数据库中间件

蔡建宇，邹 鹏，杨树强，贾 焰
(国防科技大学计算机学院，长沙 410073)

摘 要：集群技术在大型数据库应用系统中得到了越来越多的应用。无共享结构的集群易于实现，具有良好的可扩展性。但是目前的数据库集群工具非常少，往往与数据库相关。针对这一问题，该文提出了一种灵活有效的构建数据库集群的方法，研究并实现了并行数据库中间件 StarTP。StarTP 的基本思想是：屏蔽后端数据库的细节，为应用提供单一的虚拟数据库；通过流水并行加速数据加载；利用数据划分和复制将查询本地化从而实现并行查询。StarTP 支持大型数据库集群，具有容错和负载均衡功能。试验结果证明了 StarTP 的有效性和可扩展性。

关键词：并行数据库；中间件；集群

Parallel Database Middleware Based on CORBA

CAI Jianyu, ZOU Peng, YANG Shuqiang, JIA Yan
(School of Computer, National University of Defense Technology, Changsha 410073)

【Abstract】 Clusters of workstations become more and more popular to large database application systems. A shared-nothing architecture is relatively straightforward to implement and has demonstrated both speedup and scale up to hundreds of processors. But the few tools that exist for clustering databases are often database-specific. Star TP addresses this problem. Star TP is a flexible and efficient middleware for database cluster. It presents a single virtual database to the application through CORBA and does not require applications know the details of backend database. It speeds up data loading through pipelined parallelism. And it implements parallel query processing through query localization based on data partitioning and replication. Star TP is open, configurable and extensible to support large database cluster architectures offering fault tolerance and load balance. Experiment results show the efficiency and scalability of Star TP.

【Key words】 Parallel database; Middleware; Cluster

1 概述

相对廉价的机器通过网络互联组成的集群能够以较低的代价获得与并行机器相当的高性能。集群作为并行机器的替代在数据库领域得到了越来越多的应用，人们开始探讨利用数据库集群来代替并行数据库。集群中每个节点运行独立的传统串行DBMS的机器构成数据库集群。集群系统的结构与并行数据库系统相似，可选择的体系结构包括共享内存，共享磁盘和无共享[1]。其中无共享体系结构通过最小化资源共享来减少干扰，具备良好的可扩展性，被认为是支持并行数据库的最好并行结构。本文所讨论的数据库集群也是无共享结构的。

现有利用数据库集群实现并行数据库系统的主要方法分为两种：(1)构造完整的支持集群的DBMS，通过改造传统串行数据库管理系统增加并行功能[2,3]。(2)开发专用中间件来协调管理独立运行于集群各个节点的DBMS，如Leg@Net[4,5]和PowerDB[6,7]。第(2)种方法在中间件层实现并行处理相关的功能，不需要重复实现数据库基本的操作。这种方法比 方法(1)更为简单，也有利于系统的进一步扩展。不过现有的中间件方法存在以下不足之处：以查询为并行优化对象，忽视了写操作的并行；为了避免物理上划分单一关系而实现数据库的多节点复制，造成空间浪费；中间件没有提供充足的优化工具。

为此，本文研究并实现了构建高性价比并行数据库系统的中间件 StarTP：利用中间件屏蔽集群的复杂性，为用户提供一个单一的视图，保证数据库集群对用户透明；利用数据划分和复制将查询本地化，实现并行查询处理。与以往的工作相比，本文的贡献在于：提出了一个可扩展的并行数据库中间件体系结构；支持数据写入数据库的并行处理；提供多种数据存储策略，为并行查询奠定了良好基础；基于缓存优化查询。

2 StarTP 的体系结构

2.1 基本功能

CORBA是由对象管理组织(OMG)提出的标准化的开放的分布对象计算基础设施[8]。StarBus是国防科技大学计算机学院研制的遵循CORBA标准的分布计算平台。StarTP是基于StarBus实现的并行数据库中间件，将一组数据库集成为一个虚拟的数据库，保持后端数据库对用户透明。

StarTP 支持多种数据分布方式：全复制，部分复制和划分。数据分布的形式依赖于应用的需求，要求查询中的关系至少出现在后端数据库的某个节点上。

StarTP 能够自动将查询重定向到不同的后端数据库，提供了多种负载均衡策略，具有配置支持缓存和容错的选项。

基金项目：国家“863”计划基金资助重点项目(2004AA112020, 2003AA111020, 2003AA115210, 2003AA115410)
作者简介：蔡建宇(1976－)，男，博士生，主研方向：分布计算和数据库；邹 鹏，教授、博导；杨树强，研究员；贾 焰，教授、博导
收稿日期：2006-01-26 **E-mail：**jianyucai@163.com

此外，StarTP 支持构建规模更大、可用性更高的系统。

2.2 体系结构

图 1 为StarTP的体系结构图，包括了StarTP的主要组成部分：语法分析器，数据字典，数据加载服务，并行查询服务，负载均衡服务AFLS和对象事务服务OTS[9]。

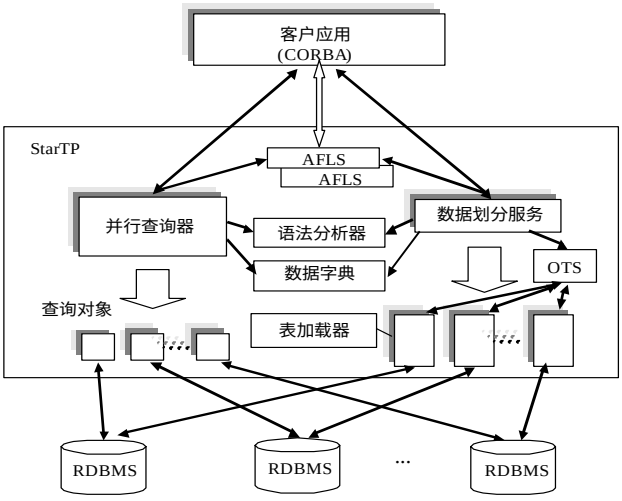

图 1 StarTP 的体系结构

语法分析器是 StarTP 处理 SQL 语句和产生执行规划必不可少的工具。StarTP 实现了支持 SQL-92 标准的语法分析器，同时扩展了建表语句的语法。用户可以在建表语句中指定表的存储和划分策略等。

StarTP 为用户提供了一个虚拟的数据库，StarTP 需要构造自己的数据字典，保存数据库中关系的复制方式、副本的位置和划分策略等信息，该数据字典只有在创建或者删除关系的时候才会被更新。在处理 SQL 语句之前，StarTP 从数据字典中读取所涉及关系的必要信息。

数据加载服务包括数据划分服务和表加载器，通过并行加速数据加载。并行查询服务包括查询服务器和查询对象，通过并行提高查询的速度。在某种意义上，数据加载服务是并行查询服务的基础。数据加载服务确定了数据记录的位置，而并行查询服务得以并行执行查询则依赖于数据记录的适当分布。

语义分析器和数据字典是数据加载服务和并行查询服务的公共基础设施，辅助数据加载和并行查询取得必要的信息。负载均衡服务 AFLS 管理服务中冗余对象，实现服务对象的容错和负载均衡，同时通过双 AFLS 避免单点失效。对象事务服务 OTS 是遵循 CORBA 标准的服务，为 StarTP 中分布事务的一致性提供保证。

3 关键技术

3.1 数据加载服务

数据加载服务实现对数据库的写操作，包括 insert 和 update。它由若干数据划分服务和表加载器组成。表加载器封装对特定数据库的写调用，负责处理写请求。数据划分服务依据给定的划分策略将写请求分发给特定的表加载器，通过对后端数据库并行写数据提高数据加载速度。StarTP 提供 3 种数据划分策略：轮转划分，范围划分和散列划分。范围划分按照预先定义的取值范围将数据记录分发到不同节点。散列划分利用散列函数对每个数据记录的特定属性进行计算，根据取得的值确定元组的存储位置。轮转划分是最简单的划分策略，数据记录以轮转的形式分发给每个数据库节点。当客户应用需要向数据库写数据时，StarTP 按照负载均衡算法选择数据划分服务。

StarTP 的数据加载的流程如下：

(1)客户应用利用 StarTP 的负载均衡服务取得数据划分服务的对象引用；

(2)根据对象引用向数据划分服务发送写请求；

(3)数据划分服务调用负载均衡服务，根据划分策略选择适当的表加载器；

(4)选定的表加载器调用数据库执行写操作。

数据加载的各个阶段可以通过流水并行执行来提高数据加载处理的速度。数据划分服务对象和表加载器的冗余也提高了数据加载的并行度。

StarTP 中的关系可被划分或者复制。如某个关系被划分到指定节点集，则与其连接的其它关系应该按照相同方式划分或者复制到相同指定节点集合。通常数据库应用中存在一些很少被更新的关系。如果这些关系能够复制到集群中的多个节点，那么与这些关系进行的连接操作可以在每个节点执行。此外，容错也要求复制某些关系。因此，StarTP 将数据同步写入复制所需的节点，由 OTS 维护分布事务的一致性。

3.2 并行查询服务

并行查询服务通过一组查询接口将一个单一视图展现给用户。并行查询服务包括若干查询对象，查询对象负责执行对特定数据库的查询。查询对象屏蔽了后端数据库的复杂性，起到了数据库驱动的作用。并行查询的基础是基于数据加载服务对关系进行划分或者复制，使得查询能够分解为可并行执行的多个子查询，同时不需要多个节点之间相互传送数据。

当并行查询服务接收到查询请求，它首先对语句进行分析，然后取得查询涉及到关系的信息并以此为依据判断是否需要在多个数据库上执行此查询。这个过程依靠语法分析器和数据字典完成。若查询需要在多个数据库上执行，则称该查询为并行敏感的。为了能够并行执行查询，并行查询服务需要变换并行敏感的查询语句，使查询转换为等价的若干可独立执行的子查询。并行查询服务按照关系的分布状况创建一定数量的查询对象，这些查询对象并行执行子查询语句。所有查询对象取得的查询结果经过合成之后再返回给客户。

并行查询服务提供了 3 类缓存优化查询。表缓存将包含很少元组而频繁被访问的关系存储在内存中，使应用能够快速访问缓存的表。游标缓存将查询结果存储在两个缓冲区中：一个缓冲区由用户直接访问，保存用户访问的数据；另一个缓冲区是备用缓存区，用于预取查询结果。并行查询服务从这两个缓冲区中交替取得数据，提高了查询处理效率。StarTP 还提供优化聚集查询的语义缓存[10]，通过重用以往的查询结果处理查询。聚集查询是海量数据库应用中的典型查询，执行聚集查询往往需要较长时间，消耗较多的系统资源。因此基于语义缓存优化聚集查询可显著提高整个系统的性能。

4 性能分析

现在 StarTP 已经能够支持 Oracle，Sybase 和 IBM DB2 等同构的数据库。下面通过数据加载和查询测试验证基于 StarTP 构建的数据库集群的性能和可扩展性

4.1 实验环境

我们以 Oracle8.1.7 为数据库服务器。所有机器都运行 Redhat Linux7.3。系统包括 5 个后端数据库。每台机器配置为 2 个 Intel Xeon 2GHz CPU，4GB RAM，2 个 70GB 的 SCSI 硬盘。所有机器通过 100Mbps 以太网互联。

4.2 数据加载

本实验的目的是评估数据加载服务的可扩展性。通过一个客户应用分别将 1 万、2 万和 10 万条记录写入一个关系。为了避免干扰，实验采用轮转划分策略将记录分发到所有节点。由图 2 可以看到数据加载速度随着后端数据库节点数增加呈近似线性递增。这说明 StarTP 在数据加载时能够获得近似线性的加速比。

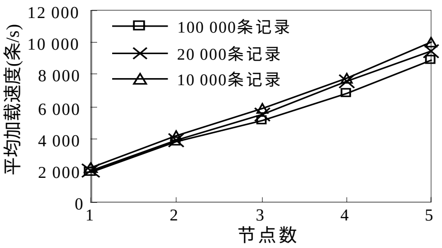

图 2 数据加载的可扩展性

4.3 并行查询

本实验的目的是评估并行度对查询处理性能的影响。我们构造一个 StarTP 客户应用执行两组查询，一组查询为简单查询，不包含连接操作，另外一组为对两个关系的连接查询。简单查询处理的关系包括 3 000 万条记录。参与连接的两个关系分别包含 300 万和 30 万条记录。从图 3 可以看到查询响应时间随着后端数据库节点数增加而下降，这说明并行查询能够加速查询处理，缩短查询响应时间。

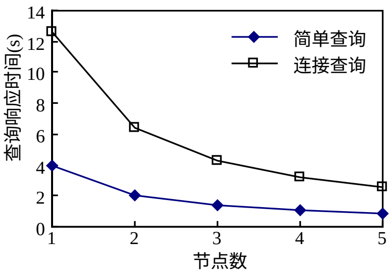

图 3 并行度对查询性能的影响

5 总结

当前数据库领域缺乏灵活有效的数据库集群构造方法。针对这一问题，本文研究并实现了基于 CORBA 的并行数据库中间件 StarTP，期望能够以较低的代价构建用于大规模数据库应用的数据库集群。本文通过实验对 StarTP 的有效性和可扩展性进行了初步验证。在某国家大型工程应用中的实践也表明 StarTP 是开发高性能数据库系统的有效平台。

参考文献

1 Dewitt D, Gray J. Parallel Database Systems: the Future of High Performance Database Processing[J]. Comm. of the ACM, 1992, 35(6): 85-98.

2 Tamura T, Oguchi M, Kitsuregawa M. Parallel Database Processing on a 100 Node PC Cluster : Cases for Decision Support Query Processing and Data Mining[C]. Proc. of SC'97 on High Performance Networking and Computing, 1997.

3 Exbrayat M, Brunie L. A PC-NOW Based Parallel Extension for a Sequential DBMS[C]. Proc. of IPDPS Workshops, 2000: 91-100.

4 Gançarski S, Naacke H, Pacitti E, et al. Parallel Processing with Autonomous Databases in a Cluster System[C]. Proc. of Int. Conf. on Cooperative Information Systems, Los Angeles, California, 2002.

5 Gançarski S, Naacke H, Valduriez P. Load Balancing of Autonomous Applications and Databases in a Cluster System[C]. Proc. of WDAS' 02, 2002: 159-170.

6 Röhm U, Böhm K, Schek H. OLAP Query Routing and Physical Design in a Database Cluster: Advances in Database Technology[C]. Proc. of the 7th Int. Conf. on Extending Database Technology, 2000: 254-268.

7 Akal F, Böhm K, Schek H. OLAP Query Evaluation in a Database Cluster: A Performance Study on Intra-query Parallelism[C]. Proc. of ADBIS' 02, 2002: 218-231.

8 Object Management Group. The Common Object Request Broker: Architecture and Specification(Version 3.0)[Z]. 2002.

9 Object Management Group. The Object Transaction Service(Revision 1.2.1)[Z]. 2001.

10 Ren Q, Dunham M, Kumar V. Semantic Caching and Query Processing[J]. IEEE Transactions on Knowledge and Data Engineering, 2003, 15(1): 192-210.

(上接第 54 页)

4 结论

构件组装是基于构件的软件开发的核心和最终目的，二进制构件组装不需了解内部细节和修改，对于使用者来说是最方便使用的。本文根据程序控制结构的基本形式，定义了二进制构件组装的 3 种方式，即顺序组装、条件组装和循环组装，并给出了其形式化定义。为一般意义下的二进制构件的组装提供了一种具有柔性机制的方法，也为进一步的研究和实现提供了方便。

参考文献

1 McIlroy M D. Mass-produced Software Components[C]. Proc. of NATO Conference on Software Engineering: Concepts and Techniques, 1976: 88-98.

2 Cai Xia, Lyu M R, Wong K F, et al. Component-based Software Engineering: Technologies, Development Frameworks, and Quality Assurance Schemes[C]. Proceedings of the 7th Asia-Pacific Software Engineering Conference, 2000: 372-379.

3 Brown A W, Wallnau K C. The Current State of CBSE[J]. IEEE Software, 1998, 15(5): 37-46.

4 杨芙清. 软件复用及相关技术[J]. 计算机科学, 1999, 26(5): 1-4.

5 Szyperski C. Component Software[M]. Addison-Wesley, 1998.

6 任洪敏，钱乐秋. 构件组装及其形式化推导研究[J]. 软件学报, 2003, 14(6): 1066-1074.

7 Arregui D, Pacull F, Riviere M. Heterogeneous Component Coordination: The CLF Approach[C]. Proc. of the 4th International Enterprise Distributed Object Computing Conference, 2000: 194.

8 Illback J, Sholberg J. Application Integration in the Boeing Enterprise[C]. Proc. of the 4th International Enterprise Distributed Object Computing Conference, 2000: 4.

9 Smith G. Component Adaptation for Web Services[C]. Proc. of the 13th Australian Software Engineering Conference, 2001: 137.