基于全局目录的分布式数据库数据一致性算法

李章兵，车乌江
（湖南科技大学 计算机科学与工程学院，湖南 湘潭 411201）

摘 要：针对分布式数据库中数据的分布性、事务执行的分布性和一些不可预测的软件错误和硬件故障等原因导致的数据不一致性问题，指出解决这些问题尚存在的困难，分析了消息队列法、事务控制法、复制控制法等分布式数据库中维护数据一致性的三种主要方法的缺点。在全局目录和分布式两阶段提交协议的基础上，提出了基于全局目录的分布式数据库数据一致性控制算法。该算法增强了分布式数据库中数据的一致性，节省了网络带宽，也提高了事务处理能力。

关键词：数据一致性；分布式数据库；惰性复制；全局目录；事务串行化
中图分类号：TP393 **文献标识码**：A **文章编号**：1673-629X(2011)09-0077-04

Data Consistency Algorithm Based on Global Directory in Distributed Database

LI Zhang-bing, CHE Wu-jiang
(Computer College of Hunan University of Science and Technology, Xiangtan 411201, China)

Abstract: Aiming at problems of data inconsistency which are generated by data distribution, the distribution of transaction execution and some other unexpected problem that are caused by software and hardware faults in distributed database, this paper explored the demerits of maintaining data consistency which exist in message queue algorithm, transaction control algorithm and copy control algorithm. It puts forward the data consistency algorithm based on global directory in distributed database in the basis of global directory and two-phase committing protocol. The algorithm enhances data consistency in the distributed database, saves network bandwidth and improves the transaction process capabilities.

Key words: data consistency; distributed database; inertia copy; global directory; transaction serialization

0 引 言

分布式数据库是物理分布在计算机网络上的多个逻辑上相互关联的数据库的集合[1]。在分布式数据库环境下，同一逻辑数据可能被存储在多个节点上，从而引起许多不同于集中式数据库的数据一致性问题。主要原因如下：

(1)数据的分布性，即同一个数据在每个数据源的表现值不同[2]，这种不一致性严重地影响了数据质量。

(2)事务执行的分布性，即一个全局事务的执行分解成在许多节点上执行的子事务，增加了实际执行中的并行能力，但在数据的分布性和事务的分布性处理中，由于全局事务与本地事务存在着并发控制问题，多个用户“同时”对数据的不同站点的多个副本进行读、写操作就有可能引起数据不一致。

(3)而一些不可预测的软、硬件故障及操作失误引起的事务重试，也会引起数据不一致。保持分布式数据库中数据的一致性，可以提高系统的可靠性和可用性；提高“读”事务的本地性，降低通信成本。

为解决分布式数据库的一致性问题，文献[3]提出了复制控制法，复制方法主要思想是：需要的通信开销高且存在操作延时，引起操作事务的大量堵塞和死锁。文献[4]提出了懒惰拷贝，惰性复制方法主要思想是：只能完成主本数据到副本数据之间数据对象的更新。文献[5]提出了消息队列法，消息队列法主要思想是：在每一个数据块上有一个最大尺寸限制，同时在系统中所有消息队列上的块尺寸上也有一个最大尺寸限制。文献[6]提出了事务控制法，事务控制法主要思想是：在一些数据量大、用户对数据操作范围大的情况，会遇到频繁的事务失败，重起事务太多，应用程序运行效率低下。

针对上述算法的缺点，文中提出基于全局目录的

收稿日期：2011-03-01；**修回日期**：2011-06-09
基金项目：国家自然科学基金项目(90818004)；湖南省科技计划项目(2007FJ3091)
作者简介：李章兵(1965-)，男，硕士，副教授，研究方向为数据库、分布计算、信息安全；车乌江，硕士研究生，研究方向为数据库、分布计算。

分布式数据库数据一致性控制算法，用全局目录维护所有站点信息，使用主本数据的更新时间作为所有数据的时间戳并在全局目录中登记，有效保持了各副本数据库中数据的一致性。

1 数据一致性的相关研究

1.1 现有的分布式数据一致性算法研究

强一致性是指对数据的任意修改，都将同时作用到该数据的所有副本上，所有的副本在任何时刻都保持一致[7]；弱一致性则不执行同步修改，修改消息先传递给一个副本，然后异步传送给其他副本，最终每个副本都会接收到修改消息，从而达到数据一致[8]。

复制控制法使用复制机制保证数据库中数据的一致性，有两种常用的方法[9]。一是主本更新法：对每一个有多个副本的数据，指定其中一个为主本，其他为副本；不同的数据主本在不同的站点，对一个数据的更新，只要对主本进行更新，就完成了对该数据的更新；再由拥有主本的站点负责把对主本所作的更新，发送到副本站点进行更新。该方法的优点是：各站点的负荷较均衡，减少了站点间控制信息的传输量；缺点是更新传播必须在短时间完成，否则会得到过时数据；主本站点出现故障时，副本站点也不可用。二是快照方法：快照方法不考虑数据的副本，只关心每一数据的主本和在主本上定义的快照；当主本更新时，需刷新快照。此方法优点是既能避免并发控制的开销，又便于复杂查询的完成；缺点是快照只是一个只读关系，数据只能读而不能写。

惰性复制方法是副本在访问的时候才去更新，可以节省网络带宽而不用当副本变化时传输实时数据，然而惰性复制在数据更新的时候，存在访问延迟[10]。主要使用于对数据的实时性要求不高的场合。

消息队列是进程间通信的一种机制，两个或者多个进程通过访问共用的系统队列来交换信息，从而保持事务的ACID特性，达到数据的一致性[11]。其通信量少，控制方式灵活。但它有一个最大尺寸限制。

事务控制法中事务是一系列不可分割的操作系列，将数据从一个一致性状态转换到另一个一致性状态。由于全局事务与局部事务存在并发执行，可能造成数据副本不一致。用两阶段协议保证全局事务执行的可串行性，用两阶段提交协议同步更新副本数据。对数据操作涉及的记录不多、事务保持时间不长的数据库系统有效。一些数据量大、用户对数据操作范围大的情况，会遇到频繁的事务失败，事务重起多，应用程序运行效率低下。

1.2 系统体系结构

分布式数据库系统的体系结构如图1所示，本地事务管理器(LTM)、事务预处理器(TP)和附加的控制表都属于局部成员。而全局事务管理器(GTM)是全局的、发送给GTM的全局事务被划分成子事务，再查找全局目录确定LTM，然后相应的子事务传递给LTM。LDBMS之间并不通信，GTM从整体上负责一个全局事务的执行，它充当协调器，利用2PC协议的原理来协调其执行。TP位于LDBMS之上。局部事务首先发送给TP，对局部用户来说TP是透明的。LTM是一个相当复杂的软件层，作为2PC协议的参与者，它负责大量复杂功能的处理。

1.3 两阶段封锁协议(2PL)

两阶段封锁协议是事务分两个阶段对数据项加锁和解锁，即获得封锁阶段和释放封锁阶段。在获得封锁阶段，事务可以申请获得数据项上任何类型的锁，但是不能释放锁。在释放封锁阶段，事务可以释放数据项上任何类型的锁，但是不能再申请任何锁。

●集中式数据库两阶段封锁(见图2)的特点：

(1)锁管理器(LM)负责管理数据加锁；

(2)数据字典协助LM完成锁的分配。

具体步骤如下：

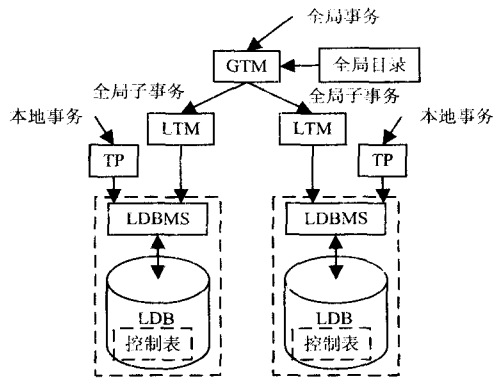

图1 系统体系结构

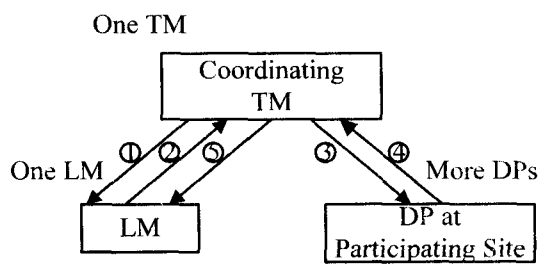

图2 集中式数据库2PL

①加锁请求；

②同意加锁；

③数据操作；

④结束数据操作；

⑤释放锁。

●分布式数据库两阶段封锁(见图3)的特点：

(1)每个站点有一个LM；

(2)并发控制由各站点的锁管理器写作完成。

具体步骤如下:

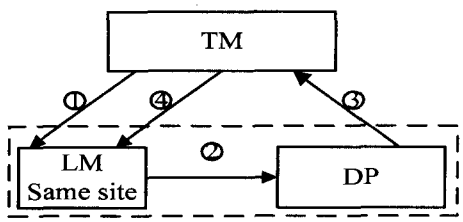

图3　分布式数据库2PL

①加锁请求;

②同意加锁;

③结束数据操作;

④释放锁。

其中:TM表示事务管理器;LM表示锁管理器;DP表示数据处理。

2　基于全局目录和全局事务的一致性控制算法

根据文献[12]的定义,本地事务是一个仅仅访问或更新一个站点上数据的事务。全局事务是一个要求访问或更新多个站点上数据的事务,可由不同站点上的若干本地事务组成。全局目录是存放与分布式数据库有关的对象和各种控制信息的场所。时间戳是由全局事务管理器(TM)产生并发布的。

定义1　主本数据(MD)是原始创建并在全局目录中注册的数据库中的数据。

定义2　副本数据(RD)是与主本数据相同而与主本数据存储地点不同的数据库中的数据。

定义3　更新队列(RQ)是全局事务管理器所维护的,当有数据更新时,相应副本进入更新队列,依次复制主本数据到副本的队列。

定义4　延迟队列(DQ)是将主本数据复制到站点副本时,发生网络故障,将副本从更新队列移入延迟队列,当网络连通时再完成数据更新的队列。

定义5　更新域(RU)是更新的数据区域。

定义6　时间戳由全局目录服务器中的事务管理器统一分配,用来记录主、副本数据最后被修改的时间,并集中控制。

2.1　数据一致性保持算法

(1)全局事务运行的时候,先到全局事务管理器注册。

(2)全局事务管理器将全局事务分解成子事务(本地事务)。本地事务将需要使用的数据加锁,并将加锁信息传递给所有的副本。

(3)所有的数据加锁完成后,比较全局目录中主RD的时间戳,如果时间戳一致,使用本地RD运行事务;如果时间戳不一致,将MD复制到本地副本并更新全局目录中RD的时间戳,再运行事务。如果尚有数据不能加锁,撤销事务。

(4)所有的本地事务运行完成后,都同意提交,全局事务管理器将本地事务的RU合并,再更新MD。

(5)MD更新完成后,将全局目录中该数据的时间戳修改为MD更新完成时的时间戳。

(6)查看全局目录中数据的所有副本,并将副本依次进入RQ。全局事务管理器将更新数据,按照出队的顺序,依次复制到副本并将RD的时间戳改为主本的时间戳。

(7)如果复制过程中出现通信故障,必产生网络延时,此时将相应的副本出队并进入DQ,当网络连通时重新完成更新操作。所有的副本更新完成后,全局事务提交,释放所有的锁。

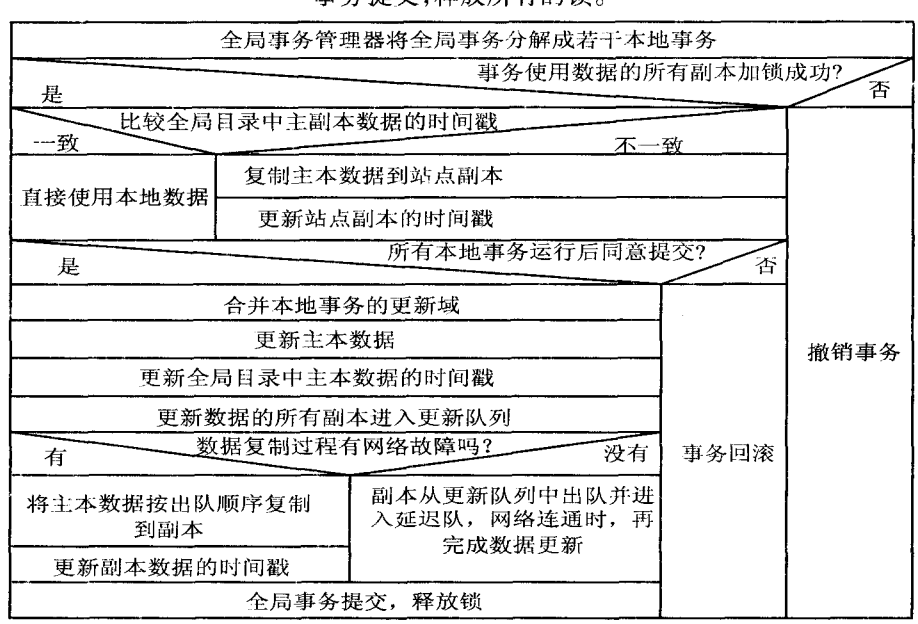

图4为算法流程图。

图4　算法流程图

2.2　访问冲突的解决

由于分布式数据库中各个RD库有可能保持同一数据的多个副本,以及事务执行的分布性,就会存在访问冲突。冲突的主要形式有写-写冲突和读-写冲突。当发生冲突时可以采用下面的解决方法:

任意指定一个数据库为某个数据对象的MD库,

当有两个或两个以上的 RD 库以更新事务的形式访问同一个数据对象，该访问请求同时到达 MD 库，这时 MD 库不管该数据对象的事务队列是否为空，都将撤销本次访问请求，通知相应的 RD 库；访问请求依次到达，则入队，当出队时通知 RD 库起动更新事务，完成更新操作。

在分布式数据库中，最好的数据项粒度大小取决于参与事务的类型。

(1)如果参与事务都访问同一记录的大量属性，则最好采用记录作为粒度；

(2)如果参与事务都访问少量的记录属性，那么选择属性作为数据项粒度。

3 算法分析

本算法使用全局目录维护所有站点信息，使用 MD 的更新时间作为所有数据的时间戳并在全局目录中登记，本地 RD 库在需要读取或更新数据的时候，

If Ts(MD)<>Ts(RD)

Then Update(RD) by CopyAgrithm;

Else Read(RD)。

其中：Ts(MD)是主本数据的时间戳；Ts(RD)是副本数据的时间戳；Update(RD)表示更新副本数据；CopyAgrithm 表示数据复制算法；Read(RD)表示读取副本数据。

全局事务管理器对事务本地化分解执行，以保持 ACID 特性；全局目录管理和维护了所有数据库节点信息，当新增数据库节点时，必须在全局目录上进行注册；删除数据库节点时，必须删除全局目录中相应的数据项；数据库节点信息更改时，也必须更改全局目录的相应项。在 MD 更新时，更改全局目录中相应数据项的时间戳，同时查找 RD 地址，将数据副本放入 RQ，同步更新 RD。算法特征之一是充分利用全局事务管理器 TM，全局事务管理器对事务进行统一的管理，产生并管理事务和数据的时间戳；特征之二是利用全局目录管理副本数据更新并保持一致性，使用 MD 的时间戳作为所有相应数据的时间戳进行注册并将时间戳作为判断数据是否一致的标志；特征之三是算法具有一定的容错性，当主 RD 库因网络故障而致通信中断时，通过周期性地访问网络来链接通信，维持数据的一致性。与其他算法相比较，本文算法的具体特点见表 1。

算法的系统开销为：$TS = T_{I/O} + T_{wait} + 3 * T_{com}$

其中：

$T_{I/O}$ 为读、写数据元素的 I/O 时间；

T_{wait} 是为了判断数据的一致性，在全局目录中比较数据元素的时间戳的时间；

T_{com} 为通信时间，可用如下公式估算：

$$TC(X) = C_0 + C_1 * X$$

其中：X 为数据的传输量，通常以 bit 为单位；

C_0 为两站点间通信初始化一次所花费的时间，由通信系统确定，近似一个常数，以秒为单位；

C_1 为传输率，即单位数据传输的时间，单位是 b/s。

表 1 相关算法的比较

名称	全局目录相关	全局事务相关	并发性
事务控制法	否	是	是
消息队列法	否	否	是
复制控制法	否	否	是
本算法	是	是	是

4 结束语

数据一致性是分布式数据库广泛研究的课题之一。文中提出用全局目录来维护所有站点的信息，用全局事务管理器对所有事务进行管理，使用主本数据的时间戳作为判断相应数据是否一致性的标志，节约了网络带宽同时对事务处理能力也有所改善。下一步的工作主要集中在重新计算和数据传输的速度等方面的问题。

参考文献：

[1] Bernstein P A, Goodman N. Concurrency Control in Distributed Database Systems[J]. Computing Surveys, 1981, 13(2): 185-221.

[2] Yadav A K, Agarwal A. An Approach for Concurrency Control in Distributed Database System[J]. International Journal of Computer Science & Communication, 2010, 1(1): 137-141.

[3] Pacitti E, Minet P, Simon E. Fast Algorithms for Maintaining Replication Consistency in Lazy Master Replicated Databases [C]//Proc of 25th Int'l Conf on Very Large Databases. Edinburgh, Scotland: Morgan Kaufmann Publisher, 1999: 126-137.

[4] Sun Yuzhong, Xu Zhiwei. Grid Replication Coherence Protocol [C]//The 18th International Parallel and Distributed Processing Symposium. Santa Fe, USA: [s. n.], 2004: 232-239.

[5] Bernstein P A, Goodman N. An algorithm for concurrency control and recovery in replicated distributed databases[J]. ACM Trans. on Database Sys., 1984, 9(4): 596-615.

[6] Phatak S H, Badrinath B R. Mulitiversion Reconciliation for Mobile Database[C]//Proc. of the 15th Int'l Conf. on Engineering. Sydney, Australia: IEEE Computer Society Press, 1999: 582-589.

[7] Davidson S B, Garcia-Molina H, Skeen D. Consistency in partitioned networks[J]. ACM Computing Surveys, 1985, 17

(下转第 84 页)

的行为当作入侵,即误检。漏检率和误检率是评价IDS性能的重要指标[12]。

实验条件:引入待测数据集 test data=10000,自体规模 $S=10000$,数据串长度 $l=32$,进制 $m=2$,匹配阈值 $r=8$,$NR=3$, 10, 14, 18, 22, 30, 35, 39, 43及49,算法各独立运行50次。

3.2 实验结果分析

(1)穷举检测器生成算法在自体集 Ns 较小的情况下,检测器的生成效率较高,但效率下降很快;在自体集 Ns 较大的情况下,穷举检测器生成效率较低,效率下降的速度变缓,检测器生成效率受自体集 Ns 规模的影响较大。改进算法的检测器生成效率受自体集 Ns 规模的影响较小,其效率基本保持在一定的范围之内,比较稳定。

(2) 随着自体规模 Ns 的增大,穷举检测器生成算法的时间开销增长过快,而改进算法生成检测器的时间开销受自体规模的影响较小,随自体规模的增大而稳定增长;当自体规模较小时,适合使用穷举法,当自体规模较大时,改进算法的时间开销明显少于穷举算法。

(3) 定义改进算法生成的未成熟检测器集为集合 A,进化得到的成熟检测器集为集合 B,穷举检测器生成算法生成的检测器集为集合 C。

集合 A 和 C 检测非自体待测数据时得到的漏检率和误检率的变化趋势基本一致,随检测器个数 NR 的增多而减小。集合 A 和 C 得到的漏检率和误检率差别不大,集合 B 的漏检率和误检率明显低于集合 A 和 C,尤其是当检测器个数较少时。当检测器达到一定个数之后,漏检率和误检率降低的趋势变缓。在 $NR=4$, 10, 14, 18, 23, 30, 36, 40, 45, 49, 53, 56, 63, 67, 71 及 76 的情况下,集合 B 的漏检率平均低于集合 A 12.69%,平均低于集合 C 14.96%;集合 B 的误检率平均低于集合 A 13%,平均低于集合 C 15.19%。

4 结束语

通过实验可以发现,该算法能够有效地生成未成熟检测器,提高生成效率,减少生成检测器的时间开销。并且进化后生成的成熟检测器集对非自体数据的漏检率和对自体数据的误检率明显优于穷举检测器生成算法。将小生境策略应用于生成未成熟检测器的过程中,利用小生境进化的思想对初始种群进行优化,提高了初始种群的整体适应性,提高了生成未成熟检测器的效率以及检测器识别"非自体"的能力。从实验结果来看,该算法基本达到算法最初的设计目标,即在一定条件下,降低生成检测器的时间开销,并尽可能地提高成熟检测器的性能。

参考文献:

[1] 张 虹,蔡焕夫,高平安. 自适应检测器生成算法研究[J]. 微计算机信息,2007 (21):72-73.

[2] 陈贤富,路 烽. 基于免疫和模糊模式识别的检测器生成模型[J]. 计算机仿真,2009(3):125-128.

[3] Kim J, Bentley P. Negative Selection and Niching by an Artificial Immune System for Network Intrusion Detection[C]// Proceedings of the 2002 Congress on Evolutionary Computation. [s. l.]:[s. n.],2002:1015-1020.

[4] 池元成,方 杰,魏 鑫,等. 基于小生境和交叉选择算子的改进粒子群优化算法[J]. 系统仿真学报,2010,22(1):111-114.

[5] Smith R E, Forrest S, Perelson A S. Searching for Diverse, Cooperative Populations with Genetic Algorithms[J]. Evolutionary Computation,1993,1(2):127-149.

[6] 刘晓峰. 一种基于小生境的克隆选择算法[D]. 太原:太原理工大学,2007.

[7] 章 军. 小生境粒子群优化算法及其在多分类器集成中的应用研究[D]. 合肥:中国科学技术大学,2007.

[8] 莫宏伟. 人工免疫系统原理与应用[M]. 哈尔滨:哈尔滨工业大学出版社,2002.

[9] 马 莉. 基于免疫原理的网络入侵检测器生成算法的研究[D]. 南京:南京理工大学,2006.

[10] 刘星宝,蔡自兴. 负选择算法中的检测器快速生成策略[J]. 小型微型计算机系统,2009,30(7):1263-1267.

[11] 雷英杰,张善文,李旭武,等. MATLAB 遗传算法工具箱及应用[M]. 西安:西安电子科技大学出版社,2005.

[12] Fabio G A, Dipankar D. Anomaly Detection Using Real-valued Negative Selection[J]. Journal of Genetic Programming and Evolvable Machines,2003,4(4):383-403.

(上接第80页)

(3):341-370.

[8] Schroeder M D, Birrel A D, Needham R M. Experience with Grapevine: the growth of a distributed system[J]. ACM Transactions on Computer Systems,1984,2(1):3-23.

[9] 王春晓,杨立国,赖杰贤. 分布式数据库数据复制技术的研究[J]. 中山大学学报,2009,48(3): 366-368.

[10] 朱泓丞,徐志广. 分布式数据库数据副本最优安置问题的研究[J]. 计算机应用,2009,29(9):2509-2511.

[11] 宋长宏, 刘宇栋,朱 劼. 基于消息的主从式数据一致性更新策略[J]. 计算机工程,2004,30:92-94.

[12] Ozsu T M, Valduriez P. Principles of Distributed Database Systems(2E)[M]. 北京:清华大学出版社,2002:120-135.