文章编号：1001-2486（2002）03-0076-05

分布式数据库系统中数据一致性维护方法研究*

陈 珉，喻丹丹，涂国庆
（武汉大学计算机学院，湖北 武汉 430079）

摘 要：分布式数据库系统是数据库领域中的一个主要研究方向，数据一致性维护是分布式数据库系统中的一个非常关键的技术问题。在深入分析和比较各种维护数据一致性方法的基础上，提出了一种较为有效的维护数据一致性的方法，详细论述了该方法如何解决并发执行引起的冲突问题，以及如何利用消息队列机制保持各数据副本的一致。

关键词：分布式数据库；数据一致性；消息队列；主副本
中图分类号：TP311.133.1 文献标识码：A

Research of Maintaining Data Consistency in Distributed Database Systems

CHEN Min，YU Dan-dan，TU Guo-qing
（College of Computer，Wuhan University，Wuhan 430079，China）

Abstract：Distributed database system is a main research direction in the database field. Maintaining the data consistency is a critical technical problem in the distributed database system. Based on analyzing in depth and comparing with many kinds of methods in data consistency maintenance，an effective method of maintaining the data consistency is presented. How to resolve the conflict caused by the parallel execution and how to maintain the consistency of all data copies by message queue are discussed in details.

Key words：distributed database system；data consistency；message queue；host copy

随着计算机技术的发展，网络、分布式处理、并发执行技术成为计算机研究的热点，在数据库领域中，能处理分散地域的、具有数据库管理特点的分布式数据库系统成为一个主要研究方向。

分布式数据库系统是一组数据集，其数据逻辑上属于同一系统，物理上分散在用计算机网络连接的多个结点上，由一个统一的分布式数据库管理系统管理[1]。对于分布式数据库系统来说，为了提高检索效率，数据分布在各个不同的结点，同一数据被存储在多个结点上，即存在多个副本（数据冗余）。同时，分布式数据库系统中的事务执行也具有分布性，即一个全局事务的执行将被划分成在许多结点上执行的局部事务，从而增加了实际执行中的并行能力，提高了系统资源的利用率[1]。在数据的分布性和事务的分布性处理中，由于全局事务与局部事务存在着并发控制问题，多个用户“同时”对数据的不同结点的多个副本进行读、写操作引起的数据不一致；以及由于一些不可预测的软、硬件故障及操作失误引起的事务重试而引起的数据不一致。此时，分布式数据库系统必须采用各种策略保证数据库状态和各副本的一致性，系统才能正确有效地运行。

目前，在分布式数据库领域，人们提出了许多方法来维护数据一致性，如事务控制法、复制控制法以及消息队列法等，但它们都有一定的适用局限性，因而在何种环境采用何种方法来维护数据一致性是研究分布式数据库的一个非常重要而迫切的问题。

1 数据一致性方法

1.1 事务控制法

* 收稿日期：2002-01-10
基金项目：国家教委科技司骨干教师资助计划项目（15099）
作者简介：陈珉（1964—），男，教授。

1.1.1 传统的事务控制法

在分布式数据库系统中，“事务”是一系列不可分割的操作序列，将数据库从一个一致性状态转变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，可能会造成数据副本不一致。传统的事务控制法通过分布式两段锁协议（2PL 协议）来保证全局事务与局部事务执行的可串行性，即可保证事务的一致性调度，以及通过分布式两段提交协议（2PC 协议）来同步更新各副本数据[2]。这对数据操作涉及的记录不多、事务保持时间不长的分布式数据库系统有效。而在一些数据量大、用户对数据的操作范围大的情况下，事务保持时间长，若采用 2PL 协议，则会严重地影响事务并发程度，不能满足实际需要，同时，2PC 协议或 3PC 协议的方法在网上通信量很大，而由于网络速度有限，因此，会使用户陷入长时间的不可忍受的等待状态，或遇到频繁的事务失败，重新启动事务太多，造成应用程序运行效率低下。

1.1.2 扩展事务控制法

在分布式环境下，2PC 协议是实现事务原子性的一个主要原则，然而大多数原有的数据库系统并不支持这一协议，扩展事务控制法则不依赖于局部数据库系统是否支持 2PC 协议，而是对全局子事务和局部事务进行扩展，并为各个局部数据库引进了锁表和日志表，它在局部数据库和控制表之上执行扩展事务，它们负责维护全局可串行性和实际事务的恢复。通过扩展事务来检测锁，以避免冲突发生和进行恢复[3]。该方法在一定的基础上维护了局部事务的自治性，但它增加了事务的负担而降低了事务本身的执行效率。

1.2 复制控制法

1.2.1 同步复制控制法

利用同步复制使得各数据副本保持紧一致性，即任何时刻不同站点的副本的值总是一致的。这样，全局事务管理系统要明确数据的各个副本所在的站点，当事务对某一数据副本进行了更新时，全局事务管理系统必须向其他站点发送强制用户修改数据副本的消息以保持数据各副本一致。但是对于出现故障的站点，则必须等待其恢复以后再进行复制，因此数据并非严格一致。由于对数据的更新操作频繁，从而向系统发送的复制消息也频繁，系统通信量大，降低了系统处理速度。而且对于很多数据库系统来说，它并不要求严格的数据一致，如空间数据库一般允许访问时间相对滞后的数据，采用同步复制控制并不实用。

1.2.2 异步复制控制法

利用异步复制使得各数据副本保持松一致性，即数据的更新与其各副本所做的修改复制有一段时间间隔，在某一时刻不同站点的副本的值可能是不一致的，数据暂时不同步。它主要解决两个问题，即如何捕捉到更新数据和什么时候复制[4]。一般采用基于事务日志方法和基于触发器方法捕捉系统更新操作，大多数数据库系统一般采用事务日志来捕捉数据更新，如 Sybase，Informix，Microsoft SQL Server 等。但由于事务日志格式可能不相同，设计通用的数据库日志读取程序相当困难，因而一些系统采用触发器方法。对于触发器法，一旦被跟踪的对象被更新，会触发相关事件发生，记录数据源与数据更新的内容。对于被触发器所捕捉的更新数据，由复制服务器对各副本进行更新，复制服务器需要保证被复制的数据是最后的更新结果。

1.3 消息队列法

消息队列是进程间通信的一种机制，两个或多个进程间通过访问共用的系统消息队列来交换信息，这里将消息队列的概念扩展到位于分布式环境下的不同站点间的进程间通信[5]，由消息管理机构在 Internet 上实现可靠的消息传送，其中涉及的消息队列有发送队列、接收队列、应答队列和管理队列。

应用程序在本地数据库上完成数据更新后，将更新信息以及本地应答队列和管理队列的地址存放到消息中，发送到远端接收队列中，消息交给消息管理机构后首先进入本地发送队列等待发送，如一切正常，消息将送到应用程序指定的远端接收队列中，同时，一个后台处理程序一直在监视着自己的接收队列，一旦有消息到达，它将读消息，并对本地数据库实施消息中所描述的更新操作，如果更新

成功，则处理结束，否则依应答队列的地址信息发送出错消息，另一个后台处理程序一直在监视应答队列，根据收到的错误信息，它将在本地数据库中试图撤消相应的更新操作。

该方法提供的是数据库间异步更新，而不是同步更新，这样可能读到的不是最新数据，所以它不适合要求实时数据同步的分布式数据库系统。

通过上面的分析，对各方法的特点和局限性进行比较如表1所示。

表1 数据一致性方法的比较

Tab.1 Comparison of methods of maintaining data consistency

数据一致性方法	特 点	局限性
传统的事务控制法	理论成熟，适用于一般的分布式数据库系统	不适用于数据量大、事务保持时间长的分布式数据库
扩展的事务控制法	不依赖于局部数据库是否支持2PC协议，自治性强	事务执行的效率低
同步复制控制法	能维护各数据副本的紧一致性，适用于实时数据同步的分布式数据库系统	数据的更新操作频繁，系统处理速度降低
异步复制控制法	能维护各数据副本的松一致性，适用于空间分布式数据库系统	可能读取的不是最新数据，不适用于要求实时数据同步的分布式数据库系统
消息队列法	适用于基于Internet分布式数据库系统	提供的是异步更新副本，不适用于要求实时数据同步的分布式数据库系统

2 主副本更新消息队列法

主副本更新消息队列法是本文提出的一种改进的维护数据一致性方法，它基于乐观的同步复制控制技术与客户/服务器结构，让事务尽可能地在各结点并发执行，同时对每一数据的多个副本中指定一个副本为主副本，一般而言，不同的数据其主副本在不同的结点。对一数据的更新，只要对它的主副本进行了更新，则认为完成了对该数据的更新，然后再将主副本的更新信息发送到其他副本结点进行更新，以保持各副本之间的一致性。这样，既能充分发挥局部站点的独立性，又能在不影响数据操作正确的前提下提高事务的并发操作。这里主要解决以下两个问题：(1) 各事务在不同结点并发执行，当要将更新结果提交给主副本时，要进行冲突检测，若没有冲突发生，则将更新结果写入主副本，否则要撤消事务和重新启动；(2) 若有与主副本未连通的其他副本，应该如何使该副本得到最新的更新数据。

要解决这两个问题，我们采用有效性检测算法[6]来处理冲突，以及引用消息队列机制来同步更新各副本。

2.1 有效性检测算法

在有效性检测算法中，事务要加盖时间戳（Timestamp），当一个事务被启动时，全局事务计数器就把当前的值作为该事务的时间戳，然后事务计数器自动加1。同时还考虑到事务的优先级，先被系统分配资源的事物的优先级较高。在此讨论写—写冲突和读—写冲突。

2.1.1 写—写冲突：当事务 T_j 向数据 x 发出写请求之前，事务 T_i 已经向数据 x 发出写请求

如果当事务 T_j 向数据 x 发出写请求之前，事务 T_i 已经完成了向数据 x 的写操作，则事务 T_j 可以向数据 x 写入新值，否则，则要进行有效性冲突检测。如果事务 T_j 比事物 T_i 先启动（即Timestamp（T_i）> Timestamp（T_j）），若 T_i 的优先级比 T_j 的优先级高，则 T_j 被拒绝，T_i 向 x 写入新值，否则看事务 T_i 处于提交的哪一阶段，若处于表决阶段，则撤消 T_i，并允许 T_j 向数据 x 写新值，若 T_i 处于不可撤消的提交阶段，则撤消 T_j，使得 T_i 向数据 x 写新值。如果事务 T_i 比事物 T_j 先启动，且 T_i 的优先级比 T_j 的优先级高，则 T_i 向 x 写入新值，T_j 被撤消，否则若事务 T_i 处于表决阶段，则撤消 T_i，并允许 T_j 向数据 x 写新值，若 T_i 处于不可撤消的提交阶段，则撤消 T_j，使得 T_i 向数据 x 写新值。此时算法为：

```
BEGIN
    IF（Timestamp（Ti） > Timestamp（Tj））
      IF（Ti 的优先级 > Tj的优先级）BEGIN  Tj被撤消，Ti 向 x 写入新值  END；
      ELSE  BEGIN  IF  Ti 处于表决阶段
                          BEGIN  Ti 被撤消，Tj 向 x 写入新值  END；
                   ELSE IF  Ti 处于提交阶段
                          BEGIN  Tj 被撤消，Ti 向 x 写入新值  END；
            END；
    IF（Timestamp（Ti） < Timestamp（Tj））
      IF（Ti 的优先级 > Tj 的优先级）BEGIN  Ti 向 x 写入新值，Tj 被撤消  END；
      ELSE IF（Ti的优先级 < Tj 的优先级）
            BEGIN  IF  Ti 处于表决阶段
                          BEGIN  Ti 被撤消，Tj 向 x 写入新值  END；
                   ELSE IF Ti 处于提交阶段
                          BEGIN  Tj 被撤消，Ti 向 x 写入新值  END；
            END；
END.
```

2.1.2 **读—写冲突：当事务 T_j 向数据 x 发出读请求之前，事务 T_i 已经向数据 x 发出写请求**

如果事务 T_j 比事物 T_i 先启动（即 Timestamp（T_i） > Timestamp（T_j）），若 T_i 的优先级比 T_j 的优先级高，则 T_j 被拒绝，T_i 向 x 写入新值；否则 T_j 读 x 的旧值，T_i 向 x 写入新值，但 T_i 必须等待 T_j 结束后才能提交。如果事务 T_i 比事物 T_j 先启动，且 T_i 的优先级比 T_j 的优先级高，则 T_i 向 x 写入新值，T_j 等待；否则若事务 T_i 处于表决阶段，则撤消 T_i，并允许 T_j 读 x 的旧值，若 T_i 处于不可撤消的提交阶段，则 T_j 读 x 的旧值，T_i 向 x 写入新值，但 T_i 必须等待 T_j 结束后才能提交。

```
BEGIN
    IF（Timestamp（Ti） > Timestamp（Tj））
        IF（Ti 的优先级 > Tj 的优先级）BEGIN  Tj 被撤消，Ti 向 x 写入新值  END；
        ELSE BEGIN  Tj 读 x 的旧值，Ti 向 x 写入新值，
                    但 Ti 必须等待 Tj 结束后才能提交
             END；
    IF（Timestamp（Ti） < Timestamp（Tj））
      IF（Ti 的优先级 > Tj的优先级）
          BEGIN  Ti 向 x 写入新值，Tj 等待 Ti 提交后再读 x  END；
      ELSE  IF（Ti 处于表决阶段）BEGIN  Ti 被撤消，Tj 读 x 的旧值 END；
            ELSE IF（Ti 处于提交阶段）BEGIN  Tj 读 x 的旧值，Ti 向 x 写入新值，
                                              但 Ti 必须等待 Tj 结束后才能提交
                                        END；
END.
```

通过上面的有效性检测算法可以较好地解决事务的冲突问题，有效地保证了数据的一致性。

2.2 消息队列机制

在消息队列机制中，其核心为消息管理机构，它负责消息的传送：当某一结点客户端事务要对某一数据更新时，将描述这些更新的消息以及本地应答队列和管理队列的地址存放到消息中，发送到服务器的接收队列的流程图如图 1 所示。

当服务器接收到来自主副本所在的结点的更新成功消息时，系统则认为此次更新成功，服务器将同步发送更新消息到其他副本的客户端的流程图如图 2 所示。

图1 客户端发送消息到服务器的流程图

Fig.1 Client sends message to Server

图2 服务器发送消息到客户端的流程图

Fig.2 Server sends message to Client

在某一结点客户端事务要对某一数据更新时，可乐观地认为不会发生任何冲突，首先在该结点对本地数据副本进行更新，然后将描述这些更新的消息以及本地应答队列和管理队列的地址存放到消息中，发送到服务器的接收队列中。服务器中保存了各数据的主副本以及其他副本所在结点的接受消息队列地址，此时，若有多个事务对同一数据要求更新时，服务器管理系统采用上面的有效性检测算法来处理冲突，将有效的更新消息传送给主副本所在的结点，同时将被撤消的消息传送给相应的客户端，试图撤消相应的更新操作。另外，当服务器接收到来自主副本所在的结点的更新成功消息时，系统则认为此次更新成功，服务器将同步发送更新消息到其他副本的客户端，从而使得各数据副本保持一致。在此，我们考虑当某一数据副本结点发生网络故障时，为了使得该结点的数据副本与其他副本保持一致，则在其连通后，由服务器将该数据的主副本最近结果通知该结点进行更新。可以看到，消息队列机制在网络故障发生频繁的情况下也可以有效地保持各副本的一致。

3 结论

本文在深入分析和比较各种维护数据一致性方法的基础上，提出了一种较为有效的维护数据一致性的方法，在各结点并不都遵循2PL协议或2PC协议的情况下，也可最大程度地保证系统的利用率，同时让事务尽可能地在各结点并发执行的基础上保证数据一致性。该方法的缺陷在于当主副本所在的结点发生网络故障而不可用时，会导致其他副本也不可用，从而降低了系统的利用率。同时，它提供的是数据库间的异步更新，不适用于要求实时数据更新的分布式数据库系统。要解决该问题，可以采用交换主副本场地法，当主副本所在的结点发生网络故障时，可将下一个副本所在的结点作为主副本，这是切实可行的。

参考文献：

[1] 贾焰，王志英，韩伟红，李霖．分布式数据库技术［M］．北京：国防工业出版社，2000.

[2] 郑振楣，于戈，郭敏．分布式数据库［M］．北京：科学出版社，1998.

[3] 卢正鼎．多数据库系统中的一致性维护［J］．计算机研究与发展，2000，38（2）：157.

[4] 胡启平．利用基于空间对象的分割与合并技术实现GIS数据共享的研究［D］．武汉大学博士学位论文，2001.

[5] 张斌．虚拟专用网环境中保持数据库一致性的一种方法［J］．计算机工程，2000，26（10）：134.

[6] Ceri S，Owicki S.On the Use of Optimistic Methods for Concurrency Control in Distributed System［Z］.Proc.6th Int. Conf.On distributed Data Management and Computer Networks.Berkeley，California，1986.