Lite/2 中基于数据同步对象的事务性同步技术*

张 孝[1], 孟小峰[2], 王 珊[1,2]

[1](中国科学院 计算技术研究所,北京 100080);
[2](中国人民大学 信息学院 数据与知识工程研究所,北京 100872)
E-mail: zhang_xiao99@263.net; {xfmeng,suang}@public.bta.net.cn
http://www.ict.ac.cn

摘要: 数据同步技术是移动数据库系统中消除移动客户机与数据库服务器之间数据不一致和冲突的主要方法.给出了"小金灵"移动数据库系统 Lite/2 中所采用的基于数据同步对象的事务性数据同步方法,它能够保持数据的语义一致性.此外,Lite/2 将同步失败恢复和队列技术相结合来优化事务性同步过程的总通信代价.实验结果表明,Lite/2 中的同步方案是高效合理的.

关 键 词: 数据同步对象;同步恢复;事务性数据同步;查询重写
中图法分类号: TP311 文献标识码: A

移动计算环境具有用户移动、频繁断接、网络条件多样、电源有限等特点,同时,移动计算环境中的嵌入式移动数据库系统 EMDBMS(embedded mobile DBMS)又具有处理失配、事务处理复杂等特点,移动数据库能因断接而与数据库服务器上的关联数据发生一致性偏差,数据同步是消除偏差、保持数据一致性的主要技术.

同步技术的一个主要领域是研究数据发布中数据项的有效性判断、更新传播等[1,2].此外,还要考虑上载中的两个问题:

(1) 保证数据同步过程的事务性和语义一致,即待同步数据集中蕴含的语义关联性需要保证完整.

(2) 降低同步的通信代价,尤其是要降低因上载中断引起的总代价增加.

在保险业务应用中,EMDBMS 管理的各表之间存在主/外码依赖及其他约束(如图 1 所示).在工作过程中,业务员可能修改客户信息表 Customer 中的某个客户的 CID.如果还修改保单信息表中的某个 CTRID,为保证 EMDB 中数据的一致性,就需要相应修改与该客户及保单对应的保单责任信息和被保险人信息表中的保单信息(后两个表及图中保单信息表统称为 CustInsu),因此同步过程中就需要一次完成相关 4 个表的上载.如果同步过程中断,而 CustInsu 未完全上载,则必须将已经处理的部分结果回滚,否则服务器数据库中某些保单信息会与修改的 CTRID 冲突.如果单独上载这些表并在成功时保留在数据库服务器中,则割裂了 Customer 和 CustInsu 中应该反映到数据库服务器中的数据的内在语义依赖关系.

考虑数据同步的语义后,中断时的回滚将增加总的同步代价.上例如果立即进行一次数据同步,Customer 的数据需要再次上载.如果这次同步成功,成功的代价应包含上次失败时的通信代价,则总的通信代价增长了.

对于数据同步问题,两级缓存机制[3]中的移动客户机 MC(mobile client)和主机上都保持着某些数据主本,可以各自更新并按一定规则通过同步实现数据一致.它从理论上描述移动数据库系统中的数据分布和管理方法.Bayou 的 BXMH[4]支持移动的电子邮件管理,提供多种处理冲突的方法,但对数据的同步性要求不太严

* 收稿日期: 2001-02-08; 修改日期: 2001-05-25
基金项目: 国家自然科学基金资助项目(60073014);国家 863 高科技发展计划资助项目(863-306-ZD12-12-1)
作者简介: 张孝(1972－),男,辽宁新民人,博士生,主要研究领域为数据库系统,移动计算;孟小峰(1964－),男,河北邯郸人,博士,副教授,主要研究领域为数据库系统,移动数据管理,Web 数据管理;王珊(1944－),女,江苏无锡人,教授,博士生导师,主要研究领域为数据库与知识库系统,移动数据库.

© 中国科学院软件研究所 http://www.jos.org.cn

格.MobiLink[5]可进行基于时标的数据同步,考虑了增量同步问题,但没有更多地处理数据间的语义一致问题.

为防止同步过程中可能出现的语义不一致,我们在"小金灵"嵌入式移动数据库系统 Lite/2 中使用基于数据同步对象 DSO(data synchronization object)的事务性同步技术,同步服务器采用模型 DTSM(DSO-based transactional synchronization model).数据同步对象 DSO 中的关系形成一个语义依赖封闭的关系组作为事务性同步的基础.同步服务器保证同步对象的所有成员关系在一次同步内要么全部完成同步,要么全部不进行,从而将 EMDB 中一致的依赖关系反映到数据库服务器中,防止可能发生的语义偏差,反之亦然.

另外,DTSM 中通过队列技术(文献[6~8]中有详细的论述)进行同步恢复以减小同步的整体通信代价.

Table A. Trader. Insurance salesperson's information①

TID Salesperson's identifier②	TName Salesperson's name③	TPassWord Password④	TBriefAcc Brief⑤
dzm	丁治明	dzm	Mobile User

Table B. Product. Insurance product information⑥

PID Product identifier⑦	Pname Product name⑧	Ptype Product class⑨	PCAttr Insurant's attributes⑩	Pduty Duty covered by product⑪	PbriefAcc Brief⑫

Table C. Customer. Customer's information⑬

CID Customer's identifier⑭	Cname Customer's name⑮	CCo Corporation name⑯	CAddr Address⑰	CTel Phone call number⑱	CbriefAcc Brief⑲	TID Salesperson's identifier⑳

Table D. Contract. Insurance contract information㉑

CTRID Contract identifier㉒	CTRDate Date of underwriting contract㉓	CTRBriefAcc Brief㉔	TID Salesperson's identifier

Note: The detail information of the insurance contract is kept in other two tables CTRDuty and CTRCSTM. These three tables are unified and relations can be illustrated by the contract identifier㉕

Table E. CTRDuty. Duty information in the contract㉖

CTRID Contract identifier	**CTRDTID** Duty identifier㉗	CTRDuty Duty entry㉘	CTRDTBriefAcc Brief㉙

Note: One contract can consist of many duty entries, but one entry belongs to only one contract㉚

Table F. CTRCSTM. Insurant's information㉛

CTRID Contract identifier	**CID** ID: Insurant identifier㉜	CTRCBriefAcc Brief㉝

Note: One contract can consist of many insurants, and one insurant can belong to many contracts㉞

①保险业务员信息,②业务员标识,③业务员姓名,④口令,⑤简介,⑥险种信息,⑦险种编号,⑧险种名称,⑨险种所属大类,⑩被保险人属性,⑪责任,⑫险种说明,⑬客户信息,⑭客户编号,⑮客户姓名,⑯单位,⑰通信地址,⑱电话,⑲简要说明,⑳业务员 ID,㉑保单信息,㉒保单编号,㉓保单日期,㉔保单说明,㉕注:D 表中只放了保单总体信息,保单的责任信息及被保险人信息分别放入 E 表和 F 表,并通过保单编号实现它们之间的联系.D 表~F 表形成一个整体,共同表示保单信息,㉖保单责任信息,㉗责任编号,㉘责任项,㉙责任项说明,㉚一个保单可对应多个责任项,一个责任项只对应一个保单,㉛保单被保险人信息,㉜被保险人,㉝简要说明,㉞一个保单可对应多个被保险人,一个被保险人也可对应多个保单.

Fig 1. Entity-Relationship model in insurance

图 1 保险业务关系依赖图

本文第 1 节介绍 DTSM 模型和 DSO 的概念.同时,给出应用数据模式的定义及 DSO 的设计规则和查询重写方法等.第 2 节描述了 DTSM 控制同步过程的有限状态自动机以及事务性同步算法.模拟实验结果表明,我们的同步方案是有效的.文章最后总结 Lite/2 的同步技术特点.

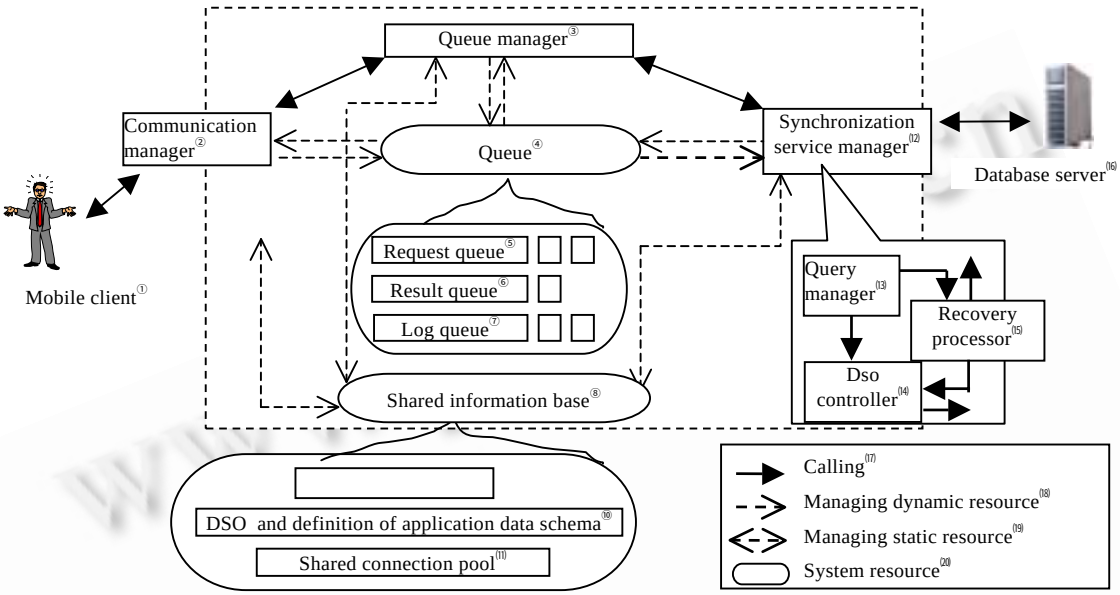
1 DTSM 模型与数据同步对象

移动数据库系统一般由数据库服务器 DBS(database server)维持一致的数据状态,通过同步服务器同步各 MC 与 DBS 中数据的状态.本节介绍 Lite/2 中基于 DSO 的事务性同步模型 DTSM、模型使用的 DSO 以及应用数据模式.

© 中国科学院软件研究所 http://www.jos.org.cn

1.1 DTSM模型及构成

图 2 是 Lite/2 采用的 DTSM 模型,它的主要部件有:(1) 支持 MC 的通信管理器,能够转发 MC 的同步请求并返回同步结果.(2) 具有恢复能力的队列管理器,它管理多种消息队列,其中日志队列可供 DTSM 的恢复处理器进行同步失败恢复.(3) 同步服务管理器.它利用队列信息和共享信息区中的信息处理 MC 的同步请求.



①移动客户,②通信管理器,③队列管理器,④队列,⑤请求队列,⑥结果队列,⑦日志队列,⑧共享信息区,⑨UserSession 控制,⑩DSO 及应用数据模式定义,⑪共享连接池,⑫同步服务管理器,⑬查询管理器,⑭DSO 控制器,⑮恢复处理器,⑯数据库服务器,⑰调用关系,⑱管理动态资源,⑲管理静态资源,⑳系统资源.

Fig. 2 Functional model of synchronizer

图 2 同步服务器功能模型

DTSM 的核心控制部件是同步服务管理器 SSM(synchronization service manager),它主要由 3 个部件构成,即 DSO 控制器 DSOC(DSO controller)、恢复处理器 RecvProc 和查询管理器 QM(query manager).接收到同步请求后,SSM 的 DSOC 利用共享信息区 SIB(shared information base)中的会话信息 UserSession 和 DSO 信息解析请求中的 DSO. RecvProc 将利用日志队列对请求的 DSO 进行失败恢复,以尽可能降低系统的总通信代价.第 2.3 节将详细说明同步失败恢复.

QM 是 SSM 的核心部件,它根据 DSO 及应用数据模式的定义对 DSO 成员的应用数据模式(这里用的是关系模式)进行查询重写,将用户请求数据转换为数据库服务器可处理的查询,并根据数据库服务器和 DTSM 本身的状态进行事务性控制.

1.2 应用数据模式与数据同步对象

在 DTSM 模型中,SSM 的工作根据 DSO 和应用数据模式来进行,而 DSO 根据应用数据模式进行定义,下面给出应用数据模式的定义.

移动数据库同步后的数据应该是服务器数据集的一个行列子集(或派生子集),可表示为两个数据空间(数据模式和数据的集合)的映射关系*,因此数据同步过程就是完成数据空间的正确映射过程.关系型数据的映射关系的定义及维护可通过应用数据模式来定义.

*事实上,EMDBMS 管理的数据中可能有一部分是纯粹的本地数据,与 DBS 中的数据不存在映射关系,因此在数据同步过程中,DTSM 不处理这些数据,后面的讨论也不再考虑这部分数据.

© 中国科学院软件研究所 http://www.jos.org.cn

定义 **1**. 应用数据模式.

对不同数据空间 DV1,DV2 中的关系 R 和 R',应用数据模式来定义 R 和 R'之间进行同步时的变换函数 F.

在应用数据模式的基础上,立即可以定义数据同步对象.

定义 **2**. 数据同步对象.

系统设计者定义的数据依赖封闭[9]的有序缓存表的集合,对 DTSM 而言,它定义了一个原子数据同步系列,序列各个成员表要么全部完成,要么都不做.每个缓存表在数据模式映射表中有一个映射,定义了一个同步.各缓存表在同步对象中的顺序指示该表在 QM 中的处理顺序.

数据同步对象具有下面两个重要特点:

(1) 各成员表蕴含的数据(或语义)依赖形成函数依赖闭包,保证了使用 DSO 同步后的语义完整.至于数据冲突问题,本文不予讨论.

(2) DSO 的同步是原子的,保证消除失败可能引发的不一致.

需要指出的是,DSO 与数据同步粒度(如表级或元组级)是不同的.根据 DSO 的定义,它将具有语义依赖的表封装在一起.在数据传输时,可以选择表级或元组级等不同的同步粒度.

1.3 DSO设计规则

由于 DV1 是 DV2 上的子集,应用数据模式中的模式映射规则要提供足够的信息以实现双向映射.设计应用数据模式时需满足以下规则:

规则 0. 移动缓存表列对应唯一的数据源表列.缓存表作为数据源表的子集,应用模式必须满足该规则.如果缓存表列来源于不同的数据源表列,则定义 DBS 上的一个视图 V,令该缓存表列对应 V 中的某个列.

对于上载类的应用,应用数据模式要满足下面的规则 1 和规则 2.

规则 1. 对需要双向同步的应用,应用数据模式需要包含足够的数据源表的主码信息.

规则 2. 如果数据源表不是基本表,应用数据模式要求它在 DV2 中是可更新的.

规则 1 和规则 2 保证了上载元组的唯一性,以及对作为源表的视图的限制.具体的应用定义还需考虑下面两个规则.

规则 3. DSO 中全部缓存表所包含的语义依赖形成闭包.这是 DSO 定义的要求.

规则 4. DSO 形成的语义依赖闭包应尽可能小,即尽量降低 DSO 的规模.

这一规则主要为适应移动通信环境的易断接性,较小规模的 DSO 可以减少事务性同步的夭折次数,从而降低同步的通信代价.

1.4 基于DSO的查询重写

数据同步控制与当前的连接信息相关,因此同步前要进行查询重写,即 QM 将 SIB 中的 DSO 信息、MC 连接会话信息和请求数据相结合,得到新的 SQL 语句,完成数据空间的映射.

查询重写需要 DSOC 提供必要的应用数据模式信息,包括:

(1) 源表中与缓存表列相对应的各个投影列定义信息,需要标识投影列中的主码列.

(2) 与缓存表对应的各个数据源表.

(3) 映射的静态水平分片信息.新查询的 WHERE 子句,一般具有固定分片条件.

(4) 映射的动态水平分片信息.一般与当前连接信息和映射的主码有关.

(5) UserSession 中的连接会话信息.

QM 进行下载的查询重写方法如下:

Procedure Select_Reconstructor

Input: (1)~(5)应用数据模式信息和连接会话信息

Output: 新的查询

BEGIN

(a) 由(1),(2)得到查询的 SELECT 和 FROM 子句;

© 中国科学院软件研究所 http://www.jos.org.cn

(b) 由(3)得到映射的静态水平分片,生成部分 WHERE 子句;

(c) 解析(4)中动态水平分片条件的动态参数,根据(5)中的会话信息将动态水平分片条件重写为静态水平分片, 生成剩余部分 WHERE 子句;

d) 返回新查询;

END

其中查询重写根据分片条件确定 DV2 的一个行列子集.Select_Reconstructor 的难点在于(c)中结合(4),(5),因此模式设计时将(3)和(4)分别定义,为简化处理.

上载中的 INSERT,UPDATE 和 DELETE 等操作与 SELECT 的查询重写基本相同,需注意的是,生成的 WHERE 子句中包含完整的主码列.

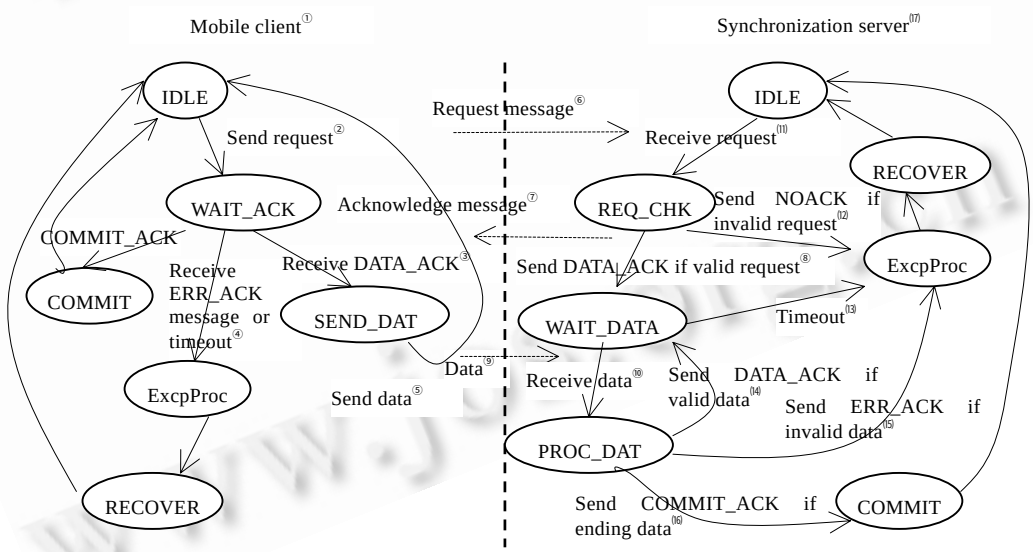
如果数据同步过程基于时标下载,则需要修改源表的模式并在查询重写时修改时标属性或增加时标限定谓词.下一节将介绍 DTSM 如何利用 DSO 解决同步的问题.

2 数据同步算法

DTSM 的数据同步过程是它接收 MC 的同步请求到完成(或终止)的一个工作循环.DTSM 通过同步有限状态自动机 SFSM(synchronization finite state machine)来控制它.

2.1 同步有限状态自动机SFSM

移动数据库系统可能发生各种故障,系统正常运行时,也可能进入其他状态.Lite/2 对数据上载使用图 3 的 SFSM,它包括 MC 和同步服务器的状态.MC 的状态变化由 MC 上的 EMDBMS 控制,而同步服务器的状态则主要由 RecvProc 和 QM 来控制.



①移动客户机,②发 Request,③收 DATA_ACK④收 ERR_ACK 或超时,⑤发数据,⑥请求消息,⑦应答消息,⑧正确发 DATA_ACK,⑨数据,⑩收数据,⑪收 Request,⑫无效发 NOACK,⑬超时,⑭数据有效发 DATA_ACK,⑮数据无效发 ERR_ACK,⑯数据结束发 COMMIT_ACK,⑰数据同步服务器.

Fig.3 Data synchronization finite state machine

图 3 数据同步有限状态自动机 SFSM

为保证同步的事务性,SFSM 中引入了 3 个重要的事务性控制状态,即例外处理状态 ExcpProc,恢复状态 RECOVER 和提交状态 COMMIT.它们的作用在于:

(1) 例外处理状态 ExcpProc:收到无效输入后,根据上下文进行例外处理.

(2) 恢复状态 RECOVER:例外处理完成后,同步服务器暂存已完成的工作,修改同步事件日志.保留的消息

© 中国科学院软件研究所 http://www.jos.org.cn

队列可用于同步失败恢复.

(3) 提交状态 COMMIT:成功完成一次数据同步循环,释放所占用的资源,记入同步事件日志.

SFSM 中的各个状态给已出了较详细的说明,这里不再列举各状态定义.至于下载过程,需要在移动客户机端增加接收数据状态 RECV_DATA,需要说明的是,在 SFSM 工作过程中,将周期性发送状态确认信息,如果发现异常,则进行例外处理.

2.2 基于同步对象的事务性同步算法

下面给出的数据同步算法使用原子过程 BEGIN_SYNC(),COMMIT_SYNC 和 ABORT_SYNC()来控制同步服务器端的同步过程,并保证 DSO 的事务性语义.

Procedure Data_Synchronizor

Input:请求标识 ReqID,请求的数据同步对象名 SyncObjName

BEGIN

a) 根据输入参数获得第 1.4 节中 5)所需的信息,包括应用名 AppName 和 SyncObjName 信息;

b) BEGIN_SYNC( )启动同步事务;

c) 对 SyncObjName 中的每个成员表 DesTbName 执行下面处理:

d) 执行 RecoverSync(AppName, DesTbName)失败,则执行 ABORT_SYNC();

e) 根据请求的类型,执行第 1.4 节中的某个查询重写过程生成 SQL 语句 SQLStmt;

f) 执行 ProcSQL*(SQLStmt),将 SQLStmt 发送到数据库服务器执行;

g) 如果执行失败,则:

h) 执行例外处理过程 ExcpProc();

i) 执行 ABORT_SYNC()终止同步事务;

j) 将执行结果送入结果队列中;

k) 如果对应当前 DesTbName 的请求队列非空,转 e);

l) COMMIT_SYNC()提交同步事务;

END

数据同步过程在处理 DSO 的每一个成员缓存表时,先执行 RecoverSync()过程,检查该表上是否发生过同步失败,并且利用失败后保留的可用信息进行恢复.MC 的处理主要在执行 ExcpProc()而不存在消息队列的问题.

2.3 失败恢复

同步恢复与传统恢复有所区别.传统的数据库恢复是在数据库启动时进行检查(目前也有系统可以指定不同的启动模式,确定是否执行恢复),一旦要求进行恢复就执行数据库恢复例程.而本文讨论的数据同步过程的失败恢复是对 DSO 中的每个成员表执行 RecoverSync()过程,以利用消息队列中可用的数据,从而降低数据同步的代价.

Procedure RecoverSync

Input:应用名 AppName,待恢复的数据同步对象成员表名 DSOGrpTbName

BEGIN

a) 根据 AppName 和 DSOGrpTbName 检索日志队列;

b) 如果没有可用日志队列单元,则

c) 向响应队列中发送无恢复消息单元

d) 否则

e) 根据日志队列中可用数据生成恢复信息送入应答队列单元;

*ProcSQL 将对同步中的冲突进行消解,对无法消解的冲突可能中断同步或继续.本文不详细讨论.

© 中国科学院软件研究所 http://www.jos.org.cn

f) 将请求应答消息单元送入应答结果队列;

g) 将恢复信息结束单元送入请求的应答消息队列中;

END

DTSM 模型中的日志队列是 DTSM 进行同步恢复的基础,同步执行失败时,QM 要求恢复处理器 RecvProc 将请求队列中已上载的有用数据保留在日志队列中.当再次进行数据同步时,QM 要求 RecvProc 从日志队列中提取与当前请求 DSO 相关的日志恢复信息,并返回给 MC,MC 根据恢复信息可以不重发已保留在同步服务器上的数据.同步服务器向 MC 发送恢复日志消息,会导致消息量增长,但恢复信息相对于同步数据的规模可以很小,经过权衡可以减少数据同步的总代价.

对于 MC,如果使用同步事件日志,在检查到 RECOVER 状态后的一种处理策略是禁止对缓存表的修改,直到完成一次数据同步,日志状态为 COMMIT.当然也可以选择其他的方法,这时同步服务器检查恢复队列中的数据有效性的算法会相应地复杂一些.

3 性能评估

数据同步过程的指标主要包括同步的执行时间和通信量.当 Lite/2 的 DTSM 检测到事务性同步故障时,要撤销未完成的 DSO 的事务.再次同步时,因重复传输数据而增加同步代价.另一方面,DTSM 基于队列的同步恢复机制将改善同步的通信负担,我们通过实验模拟分析不同情况下模型上载的效率和算法性能.至于下载,Lite/2 中使用全表同步,对单用户使用队列恢复意义不大,故评估重点考察的是上载的情况.

模拟平台的同步服务器为 PIII-450 的 PC 机,64M 内存,NT4.0 的操作系统,10Mbps 以太网卡.实验以另一台 PC 机模拟移动客户机,配有 10Mbps 的以太网卡,模拟平台使用下面的参数,见表 1.事实上,由于重点考察的是通信代价(以同步的通信数据总量为指标),因此较高性能的 PC 机对评估不会有太大影响.

实验用的数据模式是前面的保险业务的例子,数据集(Customer 和 CustInsu 组成的同步对象)的规模分别为 280K,560K 和 840K.

Table 1 Parameters list used in simulation

表 1 模拟实验参数表

Parameters①	Description②
QueLen③	Average length of log queue used by one DSO⑨
LifeTime④	Time of log queue to be maintained from being allocated⑩
CommBandW⑤	Bandwidth of communication of mobile client⑪
AbortTimes⑥	Abortion times of synchronization between two successful synchronizations⑫
MesgSize⑦	Total amount of messages sent during transactional synchronization⑬
UpdRatio⑧	Ratio of updated tuples to the whole mobile database between two synchronizations⑭

①参数名,②参数说明,③日志队列平均长度,④队列的保留周期,⑤通信的数据带宽,⑥同步的夭折次数,⑦事务的通信消息量,⑧EMDB 的更新率,⑨每个同步对象可使用的日志队列的平均长度,⑩日志队列的保留时间,从分配开始计算时间,⑪移动客户机的通信带宽,⑫两次成功同步间同步过程的夭折次数,⑬事务性同步过程中发送消息的总通信量,⑭两次同步过程之间发生的移动数据库的元组更新比例.

为模拟移动通信网络,可以调整 MC 模拟器的通信控制参数,主要包括数据包的尺寸(packsize)、数据包平均传输时间(avgtranstime)和发送的延迟周期(delay).用下面的公式来近似计算单位时间 T_0 内的平均通信带宽:

$$\text{CommBandW}=\text{PackSize}*T_0/(\text{AvgTransTime}+\text{Delay}).$$

实验主要测试不同参数对性能和通信代价的影响.选择了日志队列长度、同步夭折次数、缓存数据在同步夭折后的更新率以及待同步数据量,模拟过程给出下面的实验结果.

图 4~图 6 考察了选择不同的日志队列长度(QueLen)时,相邻成功同步间发生的同步事务夭折次数和完成的一次成功同步的总通信代价(MsgSize)之间的关系.假定中间无缓存更新.QueLen 的取值为 0(无队列),80(部分队列)和∞(完全队列).

图 6 中多次事务夭折后成功同步的总通信代价几乎保持水平,与一次成功同步的代价基本相当.而图 4 则恰好相反,在没有任何恢复日志的情况下,每次同步都要重发夭折前已经发送的数据,则数据量快速增长,曲线较陡,当同步的数据量较大时尤其明显,如图中的 Large DB 曲线.图 5 中使用了部分队列,与图 4 相比,同步的通

© 中国科学院软件研究所 http://www.jos.org.cn

信性能明显得到改善.但在同步数据较多时,如果事务夭折较多仍有一定的增长.

图 7~图 9 了考察只发生一次同步事务夭折时,使用不同的日志队列(分别对应图 4~图 6)对同步通信代价的影响.图中 X 轴为 EMDB 中 DSO 的缓存表的更新率(UpdRatio).图 7~图 9 在事务夭折前已经上载了大部分数据,因此总通信量比图 4~图 6 中进行较早事务夭折的结果偏高.各更新率都选择在上载基本相同的元组数时夭折事务.

图 7 中高更新率的通信代价在更新率为 0 时的基础上不断增加.当缓存表的数据规模较大时,数据更新导致更多的数据上载.图 8 和图 9 中使用日志队列可以减少对未更新但需要同步的数据的重传量,但如果图 8 中队列不够长,而更新又导致日志数据失效,则新同步过程的数据量较大,因而通信代价较高.图 7~图 9 的曲线变化有些相似,当事务夭折后的数据更新率(0.8)很高时,无论是否使用日志队列,通信代价都增长很多.但在较低更新率时,代价的优化还是较明显的.

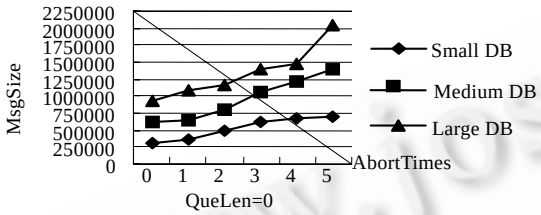


Fig.4
图 4

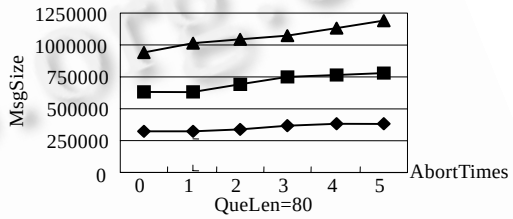


Fig.5
图 5

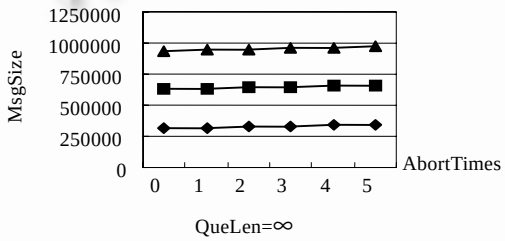


Fig.6
图 6

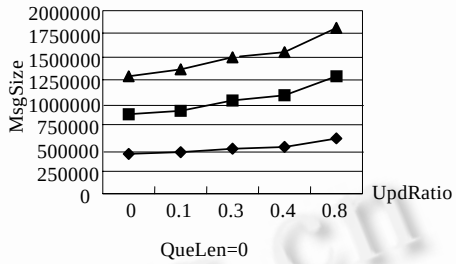


Fig.7
图 7

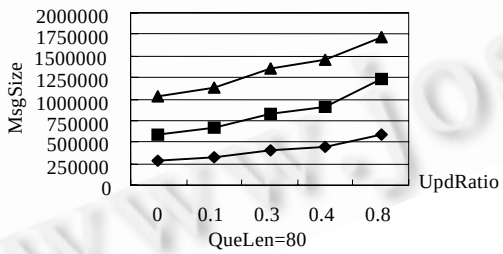


Fig.8
图 8

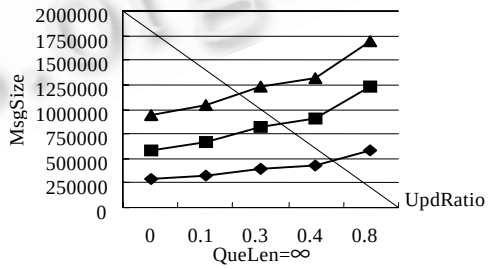


Fig.9
图 9

从性能评估中可以看出:(1) 基于 DSO 的事务性同步过程的通信代价的变化与事务夭折次数 AbortTimes 成反比,当使用日志队列时,同步恢复能明显降低事务性同步夭折所增加的代价;(2) 总的通信代价受到 MC 上的 EMDB 的更新率的影响,与 UpdRatio 成正比,同步恢复对低更新率时的性能改善相对明显,但当夭折后发生高比率更新时,日志队列带来的代价节约不如(1)中明显.

4 小 结

同步服务器是移动数据库系统中的最关键部件.Lite/2 的设计同时考虑了缓存数据间可能存在依赖以及通

© 中国科学院软件研究所 http://www.jos.org.cn

信代价的优化问题.它的主要特点有:(1) 使用数据同步对象保证数据的语义完整并将事务性引入同步过程.(2) 给出一个基于数据同步对象的事务性同步模型 DTSM,它考虑了数据同步的两个问题,功能部件可采用多线程,更具有开放性和可伸缩性.(3) 支持失败恢复.通过使用日志队列,支持与传统恢复有所区别的失败恢复技术,降低同步的代价.本文所提到的同步技术的多个问题,如数据同步对象技术、移动数据库系统中的失败恢复等策略对于 Web 应用、中间件或其他支持多用户的系统都有借鉴意义.

致谢 本文许多观点是在与陈霞、丁治明等同学的讨论中完善的,对他们特别表示感谢.同时还要感谢课题组中其他成员的支持.

References:

[1] Barbara, D., Imielinski, T. Sleepers and workaholics: caching strategies in mobile environments. Journal of VLDB, 1995,4(4): 567~602.
[2] Acharya, S., Alonso, R., Franklin, M., *et al*. Broadcast disk: data management for asymmetric communication environment. In: Carey, M.J., Schneider D.A., eds. Proceedings of the ACM SIGMOD. Montreal: ACM Press, 1995.199~210.
[3] Gray, J., Helland, P., *et al*. The dangers of replication and a solution. In: Jagadish, H.V., Mumich, I.S., eds. Proceedings of ACM SIGMOD. Montreal: ACM Press 1996. 173~182.
[4] Terry, D.B., Petersen, K., Spreitzer, M. J. *et al*. The case for non-transparent replication: examples from bayou. Bulletin of the Technical Committee on Data Engineering, 1995,21(4):12~20.
[5] Sybase Corporation. Manual for SQL Anywhere Studio Version 7.0, 2000.
[6] Gray, J., Reuter, A. Transaction Processing: Concepts and Techniques. San Francisco, CA: Morgan Kaufman Publishers Inc., 1993. 333~347.
[7] Bernstein, P.A., Newcomer, E. Principles of Transaction Processing. San Francisco, CA: Morgan Kaufman Publishers Inc., 1997.
[8] IBM Corporation. MQSeries Planning Guide, 2000.
[9] Sa Shi-xun, Wang Shan. An Introduction to Database System. 3rd ed., Beijing: Higher Education Press, 2000. 183~187 (in Chinese).

附中文参考文献:

[9] 萨师煊,王珊.数据库系统概论(第 3 版).北京:高等教育出版社,2000.183~187.

Transactional Synchronization Based on Data Synchronization Object in Lite/2*

ZHANG Xiao[1], MENG Xiao-feng[2], WANG Shan [1,2]

[1](*Institute of Computing Technology, The Chinese Academy of Sciences, Beijing* 100080, *China*);

[2](*Institute of Data and Knowledge Engineering, Information School, Renmin University of China, Beijing* 100872, *China*)

E-mail: zhang_xiao99@263.net; {xfmeng,suang}@public.bta.net.cn

http://www.ict.ac.cn

Abstract: Data synchronization is one of the most important methods to conciliate conflicts and maintain the data consistency between mobile clients and servers in mobile database systems. The novel concept of data synchronization object is used in Lite/2, a prototype of the mobile database system, to define a transactional synchronization method and is able to maintain the semantic integrity. Synchronization recovery is an important technique combined with the queue mechanism in Lite/2 to optimize the communication costs. The experimental results show that the Lite/2 has good performance in both synchronization and communication.

Key words: data synchronization object; synchronization recovery; transactional synchronization; query rewriting

* Received February 8, 2001; accepted May 25, 2001

Supported by the National Natural Science Foundation of China under Grant No.60073014; the National High Technology Development 863 Program of China under Grant No.863-306-ZD12-12-1

© 中国科学院软件研究所 http://www.jos.org.cn