维普资讯 http://www.cqvip.com

移动分布式实时数据库中并发控制框架*)

雷向东　赵跃龙　陈松乔　袁晓莉
(中南大学 信息科学与工程学院　长沙 410083)

摘　要　在移动分布式数据库系统中采用三层结构，提出了 DMVOCC-DA-2PLV (Distributed Multiversion Optimistic Concurrency Control-Dynamic Adjustment of Serialization Order-Two-Phase Local Validation) 协议处理移动分布式实时事务。移动实时事务处理分两阶段进行。第一阶段在移动主机(MHs)上处理，并进行局部部分有效性检查性确认，使用向后有效性确认机制，与在服务器提交事务进行有效性确认。及早地检测数据冲突，节省了处理和通信资源。第二阶段在服务器处理，通过局部部分有效性确认的事务，提交到服务器进行局部最终有效性确认。协议消除了移动只读事务和移动更新事务的冲突，使用多版本动态调整串行次序技术，避免了不必要的事务重启动。如果移动只读事务所有读数据项通过局部部分向后有效性确认，则可提交，大大降低了移动只读事务的响应时间。在全局有效性确认中对分布更新事务进行检查，以保证分布串行性。通过模拟仿真，对 DMVOCC-DA-2PV 协议进行了性能测试，并与 DTO-2PC 和 DHP-2PL 进行了比较。实验结果表明 DMVOCC-DA-2PV 并发控制协议要优于其它协议。

关键词　移动分布式实时数据库系统，多版本乐观并发控制，多版本动态调整串行次序，部分有效性确认，提交，多版本数据广播

A Framework for Concurrency Control in Mobile Distributed Real-time Database Systems

LEI Xiang-Dong　ZHAO Yue-Long　CHEN Song-Qiao　YUAN Xiao-Li
(College of Information Science and Engineering, Central South University, Changsha 410083)

Abstract　Mobile distributed real-database systems (MDRTDBS) are viewed as three-tier architecture. This paper presents DMVOCC-DA-2PLV (Distributed Multiversion Optimistic Concurrency Control with Dynamic Adjustment using Two-Phase Local Validation) protocol for processing mobile distributed real-time transaction in mobile broadcast environments. Mobile real-time transaction processing is performed in two stages. In the first stage transaction processing is performed at MH(Mobile Host). At MHs all mobile transactions perform local partial backward validation of transactions. The local partial validation process is carried out against committed transactions at the server. Such an early data conflict detect detection feature can save processing and communication resources. In the second stage transaction processing is performed at server. Transactions that survive in local partial backward validation must be submitted to server for local final backward validation. The protocol can eliminate conflicts between mobile read-only and mobile update transactions, and resolve data conflicts flexibly using multiversion dynamic adjustment of serialization order to avoid unnecessary restarts of transactions. For a ready-only transaction at the MH, it can be committed locally if it passes all the local partial backward validation in the course of it execution. Respond time of mobile read-only transactions is greatly improved. In global validation distributed update transactions have to do check to ensure distributed serializability in all participants. The protocol presented is compared with DTO-2PC and D HP-2PL by simulation experiments. The results of experiment show that the new protocol proposed offers better performance in terms of miss rate, restart rate, commit rate and throughput.

Keywords　Mobile distributed real-time database systems, Muliversion optimistic concurrency control, Multiversion dynamic adjustment of serialization order, Partial validation, Commit, Broadcast disks

1　引言

由于资源的限制，特别是无线传输带宽限制，传统并发控制协议不适应移动计算环境中处理实时事务。传统并发控制协议分为悲观和乐观两类。悲观并发控制协议，如 HP2PL，事务请求数据封锁可能等待很长时间，实时事务可能延误截止时间。移动主机 MH(mobile host)为检测数据冲突必须与服务器持续同步。乐观并发控制协议，如 OCC-TI-WAIT50，重启动事务数量多，从而浪费处理和通讯资源和通信带宽[1]。传统的 2PC 提交协议是阻塞的非实时提交方法[2]，使用较多的通信信息，不适应移动计算环境。

广播介质可以看作延迟时间很长的广播磁盘来建模。服务器周期性地广播数据，不同访问频度的数据以不同的周期广播。热数据广播周期短，冷数据广播周期长，从而扩展了从服务器到 MH 之间的传输带宽。在这种广播模式中，传输带宽是不对称的。从服务器到 MH 的传输容量远大于从 MH 到服务器，从 MH 到服务器传输带宽是非常有限的。在这种不对称的传输带宽环境中，并发控制协议不应卷入 MHs 在事务执行期间进行数据冲突检测时与其它 MH 和服务器持续同步[1]。由于移动只读事务的长运行周期，在单版本调度器下，长移动只读事务常常被移动更新事务阻塞。

本文提出了 DMVOCC-DA-2PLV (Distributed Multiversion Optimistic Concurrency Control-Dynamic Adjustment of Serialization Order-Two-Phase Local Validation) 协议处理移动分布式实时事务。只读事务在 MH 处理，读操作从不等待和失败。在典型数据库系统中，读操作比写操作多，这个特性对于实践来说至关重要。在 DMVOCC-DA-2PLV 协议中局部有效性确认分两个阶段进行：局部部分有效性检查和局部最终有效性确认。移动主机(MHs) 上所有移动事务执行局部部分向后有效性确认，与在服务器提交事务进行有效性确认。如果事务通过 MH 上局部部分有效性确认，提交到服务器进行局部最终有效性确认。及早地检测数据冲突，节省了

*)国家教育部博士点基金(20030533011)资助。雷向东　博士生，副教授，研究方向：移动数据库技术、并行数据库技术。

维普资讯 http://www.cqvip.com

处理和通信资源。

2 协议设计

2.1 移动实时事务的提交

发起分布式实时 T 的 MH 称为 H-MH(Home MH)。与 H-MH 相连的移动支持站 MSS(mobile support station)上的事务 T_∞为协调者,协调各子事务的执行。如果 H-MH 不能处理 T 所有子事务,抽出能处理的子事务 T_i,其余的子事务送给 T_∞。当 MH 迁移到新的无线单元时,T_∞也应迁移新的单元,并把有关的提交 T 的状态信息通知新的协调者。由于 T_∞的位置可能发生变化,每一个参与者都必须跟踪 T_∞位置的变化。T_∞把 $T-T_i$ 分配给相关的数据库服务器。数据库服务器收到子事务 T_j 后,把 T_j 分配到 MH 或固定主机 FH(Fixed Host)处理。T_∞等待所有参与者的处理结果。在 T_j 处理过程中,参与者在 T_j 截止时间前可以无条件地中止 T_j。在 T_j 执行结束后,参与者向 T_∞传送提交 T_j 决定。如果参与者不能完成 T_j,参与者向 T_∞传送夭折 T_j 决定。由于事务的提交需要全体一致地提交,只要有参与者回答夭折其子事务 T_i,T 回滚。如果到了 T 截止时间时,T_∞还没收到子事务的提交或夭折的任何回答,则 T 回滚。如果所有的参与者回答提交其子事务,则 T_∞提交 T。图 1 给出 移动分布式实时数据库三层结构。

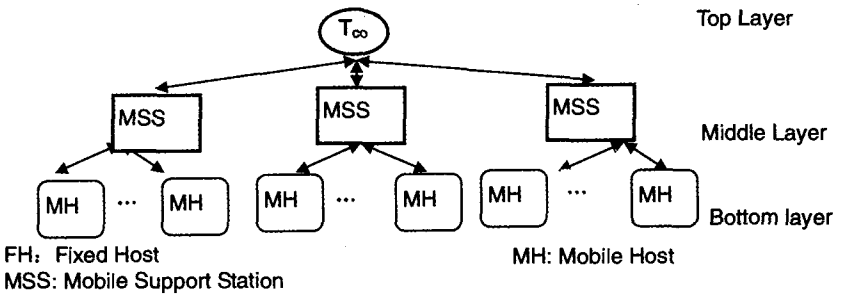

图 1 移动分布式实时数据库三层结构

2.2 多版本乐观并发控制机制

每个活跃事务 T_i 赋予一个时间戳 $TS(T_i)$,对于每个数据项 x 有一个版本序列$\langle x_1, x_2, \cdots, x_m\rangle$与之关联。每个版本 x_k 包含三个数据字段:content 表示 x_k 版本值、$WTS(x_k)$表示创建 x_k 版本的事务的时间戳、$RTS(x_k)$表示所有成功读取 x_k 版本的事务的最大时间戳。更新事务执行分为三个阶段:读阶段、有效性确认阶段和写阶段。在读阶段,假设更新事务 T_i 发出 read(x)或 write(x)操作。令 x_k 表示 x 的版本,其写时间戳小于 $TS(T_i)$的最大写时间戳。如果 T_i 发出 read(x),则返回 x_k 的值。其理由是一个事务读取在它之前的最近的版本。如果 T_i 发出 write(x)操作,且若 $TS(T_i)<RTS(x_k)$,则 T_i 重启动;否则创建 x 的一个新版本 x_i,同时将创建的新版本 x_i 存储 T_i 的私有工作空间中,在 T_i 结束之前对其他事务是不可见的。其理由是,如果 T_i 试图写入其它事务读取的版本,不允许该写操作成功。在有效性确认阶段,T_i 在 MH 上进行部分有效性确认。通过部分有效性确认的事务必须提交到服务器,进行最终有效性确认。在写阶段,若 T_i 通过最终有效性确认,则实际的更新就可写入数据库中。事务的有效性确认阶段和写阶段在临界区中,保证没有其他有效性确认事务使用相同的数据项。只读事务读最近提交的数据项版本,只经过读阶段和有效性确认阶段。

2.3 多版本动态调整串行次序

在移动计算环境中重启动一个事务开销较大。通过动态调整事务串行次序,避免不必要的事务重启动。考虑三并发事务 T_1,T_2,T_3,的执行历史片段 H。令 $r_i[x_k]$ 和 $w_i[x_i]$ 分别记为事务 T_i 读数据项 x 版本 x_k 和写数据项 x 版本 x_i,v_i和 c_i 分别记为 T_i 有效性确认和提交。

$H: r_1[x_0] r_2[x_0] w_1[x_1] r_3[y_0] r_2[y_0] w_3[y_3] w_1[y_1] v_1 v_2 c_1 c_2$

这个多版本历史片段 H 不是串行的。T_2 和 T_3 在 T_1 有效性确认时重启动。仔细分析执行历史片段 H,T_1 与 T_2 只有在数据项 x 上读-写冲突。因此,只要调整串行次序为 $T_2 \rightarrow T_1$,T_2 不需要重启动。T_1 与 T_3 有读-写和写-读两种冲突,T_3 不得不重启动。在多版本机制写-写操作对不再是冲突的,因为它们产生不同的版本。多版本动态调整串行次序在下列二种情况发生:

(1)如果进行有效性确认事务 T_v 与事务 T_i 有读-写冲突,即 $ReadSet(T_v) \cap WritSet(T_i) \neq \varnothing$,调整串行次序为 $T_v \rightarrow T_i$。T_i 的写不应影响 T_v 的读阶段。

(2)如果进行有效性确认事务 T_j 与事务 T_i 有写-读冲突,即 $WriteSet(T_v) \cap ReadSsset(T_i) \neq \varnothing$,调整串行次序为 $T_i \rightarrow T_v$。T_v 的写不应该影响 T_i 读阶段。

每个活跃事务 T_i 分配一个有效性确认间隔 $VI(T_i)=[lb, ub]$用于调整事务次序。如果事务 T_i 被串行在 T_j 之前,即 $T_i \rightarrow T_j$,则 T_j 的有效性确认间隔$[lb_j, ub_j]$和 T_j 的效性确认间隔$[lb_j, ub_j]$必须满足 $ub_i < lb_j$。每个事务 T_i 在开始执行时赋予有效性确认间隔为$[0, \infty]$。如果事务 T_i 的效性确认间隔为空,则 T_i 不可能再串行调整,必须重启动。

2.4 有效性确认

乐观并发控制机制有效性确认有两种方法:向后有效性确认和向前有效性确认。在向后有效确认中,进行有效性确认的事务执行与所有已提交事务的冲突检测。在向前有效确认中,进行有效性确认的事务与所有并发运行且尚处于读阶段的事务进行有效性确认。在 DMVOCC-DA-2PLV 协议中,事务处理分两阶段:第一阶段在 MHs 上处理,进行局部部分有效性确认;第二阶段在服务器上处理。如果所有移动事务直接提交到服务器进行有效性确认,对于没有通过有效性确认的移动事务,MHs 需等待服务器的通知很长时间,才知道这些事务需重启动。此策略会导致移动事务处理不可容忍地延时。在 MHs 进行局部部分有效性确认,MHs 能及时确定事务由于数据冲突需重启动。如此早地检测数据冲突,节省了处理和通信资源。MHs 上没有完整的和最新冲突事务的数据视图。例如,MHs 不知道在当前广播周期开始后提交到服务器的某些冲突事务的提交信息。MHs 可能自愿或不自愿与移动网络断开,造成 MHs 上的数据可能过时,错过接收服务器广播的有效性确认信息。因此,如果事务通过局部部分有效性确认,必须提交到服务器进行局部最终有效性确认。在广播周期开始时服务器广播在上一个广播周期提交的事务

维普资讯 http://www.cqvip.com

有效性确认信息。MHs 通过接收提交事务的有效性确认信息对 MHs 上的事务进行局部部分有效性确认。在 MHs 上进行部分有效性确认，MHs 及时检测到数据需冲突，从而确定那些移动事务需重启动。由于 MHs 上没有完整的和最新冲突事务的数据视图，如果移动事务通过局部部分有效性确认，必须提交到服务器进行局部最终有效性确认。

在 MHs 使用向后有效性确认。MHs 上所有移动事务，包括移动只读事务和移动更新事务，与在服务器上一个广播周期提交事务进行有效性确认。提交事务要串行在有效性确认的移动事务之前。因此，数据冲突检测是检查进行部分有效性确认的移动事务读集合与提交事务的写集合是否相交。

在服务器进行有效性确认的事务有移动事务和服务器事务。向前有效确认提供灵活的数据冲突解决方法，可选择有效性确认事务或冲突事务重启动，甚至可以通过强迫有效性确认事务在有效性确认阶段等待来避免一些事务的取消。因此，在服务器使用向前有效性确认。有效性确认事务串行在所有并发运行且尚处于读阶段的事务之前。

全局有效性确认需要检查分布数据项，这是因为分布数据项可能在有效性确认事务最后操作和其他并发事务有效性确认阶段之间被改变。为了保证分布串行性，对所有更新事务进行检查，确定所读写的发布数据项是否改变。如果更新事务所读写的发布数据项改变了，更新事务必须重启动。

3 DMVOCC-DA-2PLV 并发控制协议

3.1 在 MHs 上事务处理

3.1.1 事务读阶段处理

MHs 处理移动事务的读写请求和调整有效性确认间隔。当移动事务进行读或写请求时，它的有效性确认间隔被调整，以反映移动事务与提交事务之间的串行关系。令 x_k 表示 x 的版本，其写时间戳小于 $T_S(T_i)$ 的最大写时间戳。移动事务 T_i 发出读数据项 x，选择读其版本 x_k，$VI_{lb}(T_i)$ 被调整，即置 $VI(T_i)=VI(T_i)\cap[WTS(x_k),\infty]$，同时将 x_k 版本放入 T_i 的读集 ReadSet(T_i)中。移动事务 T_i 发出写数据项 x，如果 $TS(T_i)<RTS(x_k)$，则 T_i 重启动；否则创建 x 的一个新版本 x_i，VI_{lb}（T_i）被调整，即置 $VI(T_i)=VI(T_i)\cap[WTS(x_k),\infty]\cap[RTS(x_k),\infty]$，并放入 T_i 写集 WriteSet(T_i)中。

3.1.2 局部部分有效性确认

在 MHs 进行部分有效性确认，使用向后有效性确认机制，以保证没有任何移动事务与上一个广播周期在服务器提交事务有数据冲突。服务器在广播周期开始时广播上一个广播周期在服务器事务有效性确认信息。事务有效性确认信息由 CommitSet，AbortSet，CT_ReadSet 和 CT_WriteSet 组成，其中 CommitSet 为通过服务器最终有效性确认并提交的移动事务集合，AbortSet 为未通过服务器最终有效性确认并夭折提移动事务集合，CT_ReadSe 为所有提交的事务读数据项集合，CT_WriteSe 为所有提交的事务写数据项集合。移动事务与提交事务进行部分有效性确认。如果移动事务部分有效性确认失败，移动事务不得不重启动。事务调度器按移动事务优先级调度。移动事务 T_i 对于已提交的事务 T_c 进行有效性确认，满足下面条件之一：(1)T_c 在 T_i 开始之前已经结束。(2)否则，ReadSet(T_i)∩WritSet(T_c)=∅，且 T_c 的写阶段在 T_i 开始其有效性确认阶段之前完成。

规则 1 如果进行部分有效性确认移动事务 T_i 与提交事务 T_c 有读-写冲突，即 ReadSet(T_i)∩WritSet(T_c)≠∅，调整串行次序为 $T_i\rightarrow T_c$，这意味着虽然 T_c 在 T_i 之前提交，T_i 的读应放在 T_c 写之前。调整 $VI(T_i)$ 使得 $VI_{ub}(T_i)<TS(T_c)$，即置 $VI(T_i)=VI(T_i)\cap[0,TS(T_c)]$。由于调整 $VI_{ub}(T_i)$，CT_WritSet 为所有提交事务的写集合，此规则可描述为：对于每个数据项 $x_k\in$ ReadSet(T_i)，如果 $x_j\in$ CT_WritSet，调整 $VI_{ub}(T_i)$ 到 min({WTS(x_k)|x_k in CT_WritSet})。x_k 标记已进行有效性确认，不需要再在服务器上进行最终有效性确认。因为在当前广播周期开始后提交的事务写数据项的时间戳大于 CT_WritSet 中所有数据项的写时间戳。

规则 2 如果进行部分有效性确认移动事务 T_i 与提交事务 T_c 有写-读冲突，即 WritSet(T_i)∩ReadSet(T_c)≠∅，调整串行次序为 $T_c\rightarrow T_i$，这意味着 T_c 的读不应影响 T_i 的写。调整 $VI(T_i)$ 使得 $VI_{lb}(T_i)>TS(T_c)$，即置 $VI(T_i)=VI(T_i)\cap[TS(T_c),\infty]$。由于调整 $VI_{lb}(T_i)$，所有提交事务的读集合为 CT_ReadtSet，此规则可描述为：对于每个数据项 $x_i\in$ WriteSet(T_i)，如果 $x_j\in$ CT_ReadSet，调整 $VI_{lb}(T_i)$ 到 max({RTS(x_j))|x_j in CT_ReadSet})。这是因为提交事务不可能读活跃事务所写的数据项。

移动只读事务如果所有读数据项通过部分向后有效性确认，则可提交，串行在当前广播周期前提交的事务之后，当前广播周期后提交的事务之前。移动更新事务如果通过部分向后有效性确认，必须提交到服务器进行最终有效性确认。因为移动更新事务串行在它到达有效性确认阶段之前提交的事务之后，所有活跃事务之前。

3.2 在服务器上事务处理

3.2.1 局部最终有效性确认

定义 1 假设事务 T_i 和 T_j 分别成功创建数据项 x 版本 x_i 和 x_j，则版本序列定义如下：

$x_i<<x_j\Leftrightarrow TS(T_i)<TS(T_j)$

提交到服务器的移动更新事务必须执行最终向后有效性确认，因为移动事务在 MHs 上执行部分有效性确认后，服务器上可能有新的事务提交。MHs 可能自愿或不自愿与移动网络断开，造成 MHs 上的数据过时。移动只读事务如果所有读数据项通过部分向后有效性确认，则不需要执行最终向后有效性确认，即可提交。

规则 3 进行最终有效性确认移动事务 T_v 读数据项 x 版本 x_k，如果 x_k 没做已进行有效性确认标记，并存在版本 x_{k+1}，$x_k<<x_{k+1}$，说明 T_v 读版本 x_k 后有提交事务写了数据项 x 新版本 x_{k+1}，调整 $VI_{ub}(T_v)$ 到 $WTS(x_{k+1})$，即置 $VI(T_v)=VI(T_v)\cap[0,WTS(x_{k+1})]$。

规则 4 进行最终有效性确认移动事务 T_v 写数据项版本 x_v，提交事务不可能读 T_v 写的版本 x_v，调整 $VI_{lb}(T_v)$ 到数据项 x 版本中最大 $RTS(x)$，即置 $VI(T_v)=VI(T_v)\cap[\max(RTS(x)),\infty]$。

3.3 全局有效性确认

如果有效性确认事务 T_v 是更新事务，对于 ReadSet(T_v) 和 WriteSet(T_v) 中每个分布数据项 x_k，T_v 有效性确认间隔 $VI(T_v)$ 向前调整为 $VI(T_v)=VI(T_v)\cap[WTS(x_k),\infty]\cap[RTS(x_k),\infty]$，以保证分布串行性。

4 性能评价

通过仿真模拟测评本文提出的多版本乐观并发控制协议(MVOCC-DA-2PV)性能。测试的主要参数是事务延误截止时间率(miss rate)、重启动率(restart rate)和提交率(commit rate)。模拟的模型由服务器、移动主机和广播磁盘组成。广播磁盘传输数据项和控制信息，采用聚类磁盘组织[5]。数据项所有版本放在同一个磁盘上，每个数据项所有版本将相继

维普资讯 http://www.cqvip.com

广播。热数据的老版本位于最新版本之后，都放在快速磁盘上；冷数据所有版本则位于慢速磁盘上。每个数据项的所有版本以相同的频率广播。模拟程序用 C++编写，模拟实验参数如表 1 所示。

表 1 模拟系统参数设置

Parameter	Value
Mobile Host:	
Operation/Subtransaction	5～10
Proportion of read-only transactions	70%
Handoff probability	0.2%
Disconnection probability	5%
Disconnection duration	1～50 seconds
Slack factor	2.0～6.0
Number of wireless cell	5
Number of MHs	10
Think Time	1～7 seconds
Server:	
Database size	500 data items
Probability of hot items	30%
CPU scheduling	Earliest Deadline First

为了比较并发控制协议的性能，选用 DHP2PL(Distributed High Priority Two-Phase Locking)[3] 和 DTO-2PC (Distributed Timestamp Ordering combined with 2PC) [4]两协议作为基准协议。DHP2PL 和 DTO-2PC 是单版本协议，服务器只保存数据的最新版本，并周期地广播每一个数据项。

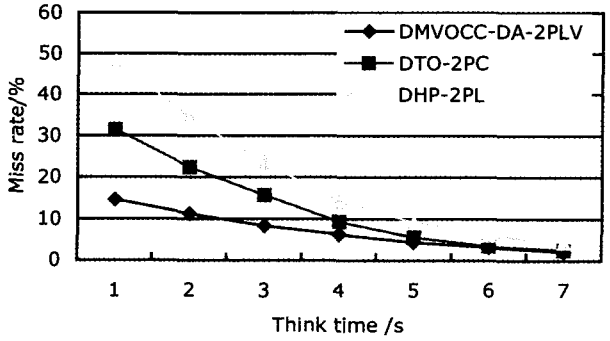

图 2 移动事务延误截止时间率

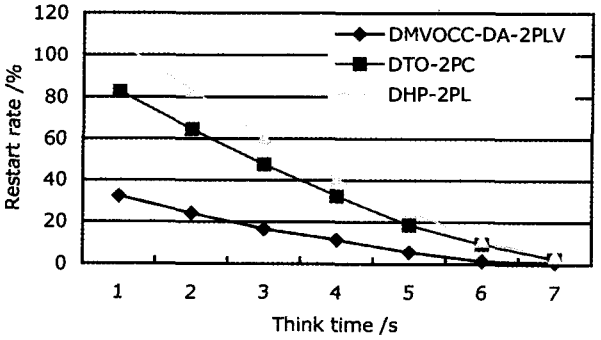

图 3 移动事务重启动率

图 2 显示在不同移动事务负荷对移动事务延误截止时间率的影响。DMVOCC-DA-2PLV 协议延误截止时间率比 DTO-2PC 协议和 DHP-2PL 协议都低，其原因是移动只读事务在 MHs 处理，移动事务在 MHs 部分有效性确认，及早地检测数据冲突，减少了移动事务延误截止时间率。另一个原因是多版本机制清除了移动只读事务和移动更新事务之间冲突，读请求从不失败且不必等待。图 3 显示不同移动事务负荷对移动事务重启动率的影响。传统 DHP-2PL 性能相对较差。这是由于多版本机制消除了只读事务和更新事务的冲突，降低了只读事务的响应时间。局部部分有效性确认有利于 MHs 较早检测数据冲突，通过多版本动态调整串行次序，避免任何不必要的事务重启动。因此，移动事务重启动数大大减少。图 4 显示不同移动事务负荷对移动事务提交率的影响。DMVOCC-DA-2PLV 协议性能要优于 DTO-2PC 协议和 DHP-2PL 协议，主要原因是多版本机制消除了移动只读事务和移动更新事务之间冲突，移动事务在 MHs 部分有效性确认，及早地检测数据冲突，通过多版本动态调整串行次序，避免不必要的事务重启动。

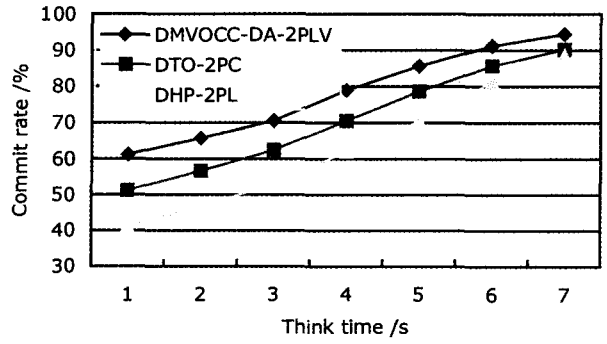

图 4 移动事务提交率

结束语 本文提出了移动计算环境中 DMVOCC-DA-2PLV 并发控制协议处理移动分布式实时事务。移动实时事务处理分两阶段进行。第一阶段在移动主机(MHs)上处理，并进行局部部分有效性检查性确认，使用向后有效性确认机制，与在服务器提交事务进行有效性确认。及早地检测数据冲突，节省了处理和通信资源。第二阶段在服务器处理，通过局部部分有效性确认的事务，提交到服务器进行局部最终有效性确认。协议消除了只读事务和更新事务的冲突，使用动态调整串行次序技术，避免了不必要的事务重启动。移动只读事务如果所有读数据项通过局部部分向后有效性确认，则可提交，大大降低了只读事务的响应时间。在全局有效性确认中对移动分布更新事务进行检查，以保证分布串行性。通过模拟仿真，对协议进行了性能测试。实验结果表明新协议要优于其它并发控制协议。

参 考 文 献

1 Lee V C S, Lam Kwok-Wa, Son Sang H. On transaction processing with partial validation and timestamp ordering in mobile broadcast environments [J]. IEEE Transaction on Computers, 2002, 51(10):1196～1211

2 Lam Kam-Yiu, Chan E, Leung Hei-Wing. Concurrency control strategies for ordered data broadcast in mobile computing systems[J]. Information Systems, 2004, 29(3):207～234

3 Lam Kam-Yiu, Kuo Tei-Wei, Tsang Wai-Hung. Concurrency Control in mobile distributed real-time database systems [J]. Information Systems, 2000, 25(4):261～286

4 Data A, Son S H. Limitations of priority cognizance in conflict resolution for firm real-time database systems [J]. IEEE Transaction on Computers, 2000, 49(5):483～501

5 Pitoura E, Chrysanthis P K. Multiversion data broadcast [J]. IEEE Trans Computers, 2002, 51(10):1334～1230