文章编号:1007 – 6492(2000)04 – 0085 – 05

CSCW 中的一致性和局部一致性模型

周春辉,张 敏,张嘉一

(郑州工业大学电气信息工程学院,河南 郑州 450002)

摘 要:维护系统的一致性是 CSCW 系统的关键技术之一,旨在解决由于多用户同时操作可能产生的操作冲突,从而破坏系统的一致性. 首先讨论了操作冲突可能导致的三类系统不一致,简要介绍了 CSCW 系统中的并发控制方法,给出了基于不同分布式结构的整体一致性模型和局部一致性模型.最后,提出了一种树形结构来实现局部一致性模型.

关键词:协同工作;系统一致性;并发控制;一致性模型

中图分类号:TP 311.1 文献标识码:A

0 引言

计算机支持的协同工作(CSCW)主要目的是利用计算机、网络和多媒体等技术加强群体跨越时间,空间界限进行协作的能力. CSCW 系统可分为支持同步工作和支持异步工作两种.前者包括电子会议室和协同编辑系统等,后者则包括电子公告牌系统和电子邮件系统等.

在同步 CSCW 应用中,除应用共享系统只能支持多名用户串行操作外,大多数系统允许多人在各分布站点上并行地对共享数据进行操作.对于这类同步并行系统,维护共享数据的一致性是保证协作正常进行的基本要求,同时又是系统设计与实现中的一个难点. CSCW 系统作为一种分布式系统,它可以选择的分布式系统结构包括:完全集中式结构,完全复制式结构或二者的混合结构.完全集中式结构即客户/服务器结构,服务器处理所有客户的输入和显示输出事件;完全复制式结构在每个分布站点上都有一份相同的程序和数据拷贝,每个站点上可自主地根据用户的输入对本地数据拷贝进行处理,并将所作更新通知其他站点,以保证系统中共享数据的一致性;混合式结构则介于完全集中式结构与完全复制式结构之间,该结构通常根据需要将数据和程序在服务器和站点间进行较为灵活的分配.服务器主要完成用户登录,并发处理和共享数据的访问控制等,各站点都是先在本地对用户的操作进行处理和反馈,然后选择直接通知其他站点或是传送给服务器集中处理.

1 CSCW 中的破坏系统一致性问题

在同步并行 CSCW 的应用中,由于网络带宽和传输速率的限制,CSCW 系统为了提高系统的响应速度,特别是保证尽量快的本地响应速度,一般都是把共享对象复制到各用户本地端,即完全复制分布式结构.由于各用户操作的并行性和网络延迟的不确定性,多个用户处理过程不可避免地会产生操作冲突,从而造成三类破坏系统一致性的问题[1],即各用户端共享对象的不相同,这将影响协同工作的正常进行.下面以图 1 中的协同编辑过程为例,分析图中 O_1,O_2,O_3 分别在站点 N_1,N_2,N_3 上产生的 3 个不同操作,有以下 3 种情况.

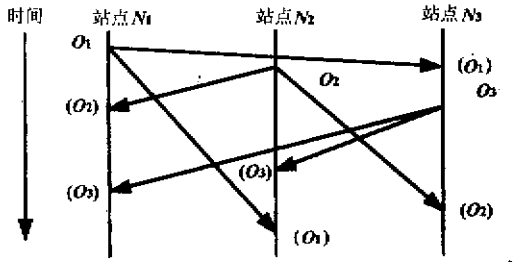

图 1 协同编辑示意图

收稿日期:2000 – 09 – 10;修订日期:2000 – 10 – 30

作者简介:周春辉(1978 –),女,河南省驻马店市人,郑州工业大学硕士研究生.

1.1 操作矛盾

如图1所示，O_1，O_2，O_3 在3个站点上的执行顺序各不相同，在站点 N_1 上的顺序为 O_1，O_2，O_3 在站点 N_2 上的顺序为 O_2，O_3，O_1 在站点 N_3 上的顺序则为 O_1，O_3，O_2，因此，除非3个操作的顺序是可以互换的，否则最终的操作结果及其在3个站点上的显示会各不相同. 在CSCW系统中，通常很难保证所有操作都可以互换，因而若不采用相应的并发控制策略，极易破坏系统的一致性. 比如在协同编辑系统中，假设共享文档的初始状态为“ABC”，对应于图1中的3个操作，O_1 是在位置1上插入字符“1”，O_2 是在位置0上插入字符“2”，O_3 是在位置0上插入字符“3”，最终的执行结果将为：站点 N_1 上“32A1BC”，站点 N_2 上“312ABC”，站点 N_3 上“23A1BC”.

1.2 因果矛盾

在图1中，O_3 是当 O_1 抵达站点 N_3 以后才产生的，因此站点 N_3 上的用户在发出操作 O_3 时，可能已经看到了 O_1 的执行结果，即二者可能存在逻辑上的因果关系，其中 O_1 是“因”，O_3 是“果”. 但由于网络延迟，站点 N_2 上的用户却是先看到 O_3，后看到 O_1，对于该用户来说，O_3 是“因”，O_1 才是“果”，这就产生了因果关系上的不一致. 以网上闲谈为例，假设 O_1 是提出问题“现在几点了？”，O_3 是给出回答“8:50”. 对于站点 N_2 上的用户来说，由于是先看到答案，后看到问题，即使最终的显示内容与其他两站点完全一致，但由于操作过程的因果性未得到维护，仍会使其产生困惑.

1.3 预期矛盾

在图1中，O_2，O_3 是完全并行的两个独立操作，它们在产生时并未受到彼此的影响，但在站点 N_2 上，由于 O_2 先于 O_3 执行，因 O_2 的影响，O_3 执行前站点 N_2 的状态与站点 N_3 上产生时的状态已不相同，如果将 O_3 原样作用于站点 N_2，其执行结果可能与原来的预期结果有所不同. 仍以协同编辑为例，假设共享文档的初始状态为字符串“ABCDEF”，O_1 = insert(“11”,1)，O_2 = insert(“22”,3)，O_3 = remove(3,0)为3个不同节点上同时产生的并发操作，(记第一个字符的位置为“0”，位置标号相对于源串)分别表示在位置1,3之前插入字符串“11”，“22”以及删除“ABC”. 正确的结果是“1122DEF”，而按 O_1，O_2，O_3 的6种排列中的任何一种顺序执行，都不可能得到正确的结果.

2 CSCW中的并发控制策略

为了解决上述的一致性问题，人们提出了多种并发控制策略. 最早的协作应用使用较多的一般是令牌传递协议，在CSCW中称为floor控制[1]. 该算法假设只有一个floor，当某一成员需要操作时，必须拥有floor. 这种方法的缺点是阻碍了信息的自然流动，特别不适合高度并行的交互. 另一种情形[1]是系统不提供同步机制，它依靠用户对共享对象的协作感知，各用户按照社会协议和各自的角色分工等自觉地避免并发冲突，手工解决冲突带来的不一致. 这种方法在某些应用中，可以很好地工作，但由于缺乏必要的并发控制，用户手工维护一致性比较困难. 目前常用的方法一般基于这两者之间，根据不同的系统结构采取不同的方法，下面是一些典型的并发控制方法[2].

(1) 加锁法：加锁法是保证数据一致性的常用手段，它提供对共享对象的加锁和解锁操作. 每当用户欲修改共享对象之前，必须获得该对象的锁.

(2) 依赖探测法：它对每个操作加上时间戳，然后根据时间戳检查多个操作之间是否存在冲突.

(3) 操作转换法[1]：对于共享对象的本地操作立即执行，远程操作执行前对操作的各参数形式进行调整，以补偿由于执行其他操作而引起的共享对象的状态变化.

(4) 集中控制法：使用一个集中控制进程管理对共享对象的操作. 它接收所有用户的操作，按照一定的规则排序，然后广播给所有的用户按序执行，从而保证一致性.

(5) 可逆执行：为所有的操作定义全程时序，操作可以立即执行，但保留与操作有关的信息，以便在必要时取消该操作.

另外，这些方法在具体实现中还可以结合在一起使用，对不同的需求提供更好的服务.

3 整体一致模型

对于不同的分布式结构，解决一致性问题的方法可以有很大的不同. 基于完全复制式结构的CSCW的一致性模型称为整体一致模型，它是许多并发控制策略的基础. 模型中首先给出了操作间因果关系的形式定义.

两个操作 O_1，O_2 存在时间上的因果关系“$O_1 \longrightarrow O_2$”，则 O_1，O_2 满足下列条件之一(或任意组

合)：①它们均由同一站点发出，O_1 的产生先于 O_2；②它们由不同站点发出，如果 O_1 产生于站点 N_1，O_2 产生于站点 N_2，则 O_1 在站点 N_2 上的执行先于 O_2 的产生；③存在 O_3，使得 $O_1 \longrightarrow O_3$，且 $O_3 \longrightarrow O_2$.

两个操作 O_1，O_2 相互独立是指 O_1，O_2 不满足 $O_1 \longrightarrow O_2$，也不满足 $O_2 \longrightarrow O_1$. 表示为 $O_1 // O_2$.

整体一致性模型的定义：一个实时 CSCW 系统必须同时满足下面三个一致性条件，才是一致的：①数据一致性：执行完一组协同操作后，各站点上数据拷贝的内容应保持完全一致；②因果一致性：任意两个操作 O_1，O_2，若它们存在时间上的因果关系 $O_1 \longrightarrow O_2$，则在所有站点上，均是 $O_1 \longrightarrow O_2$；③预期一致性：任意两个操作 O_1，O_2，若它们相互独立 $O_1 // O_2$，则它们在任何站点上以任何顺序执行的结果都应与其在本地执行的预期结果一致.

在整体一致性模型中定义的一致性条件可总结为两个方面：操作结果一致性和操作过程一致性. 前者要求一组并发操作在各个站点执行完以后，所有站点上的数据拷贝的内容应完全相同，且与各用户所预期的结果相同. 后者要求若一组操作间存在时间上的因果关系时，在各站点上它们都应按这种先后关系执行.

整体一致性模型对于完全复制式结构来说，简化了系统的一致性维护算法. 但是在这类系统中，为了保证所有站点上数据拷贝的全局一致性，不可避免存在处理冗余. 随着协作人数的增加，并发操作常会急剧增长，性能趋于恶化，导致操作延迟增长，因此作为一种一致性模型，它还不够完善.

4　局部一致性模型

在 CSCW 系统中，对一致性的要求其实是为保证用户协同工作的正常进行而提出的，它应以协作者为中心，而非以系统为中心. 整体一致性模型是以系统为中心的，维护各共享数据拷贝的整体一致性. 然而对协作者而言，真正有意义的是协作过程中他所感知到的一致性，即系统对协作者可见部分的一致性——局部一致性.

对于 CSCW 中的局部一致性，很多研究工作已经展开，并在提高系统性能等方面取得了一些成果，但是效果并不是很理想. 为了进一步减少冗余，提高系统性能，本文提出一个基于树形结构的局部一致性模型，见图 2. 该模型放宽了整体一致性模型中的部分条件，使基于新模型的并发控制算法有更大的优化空间.

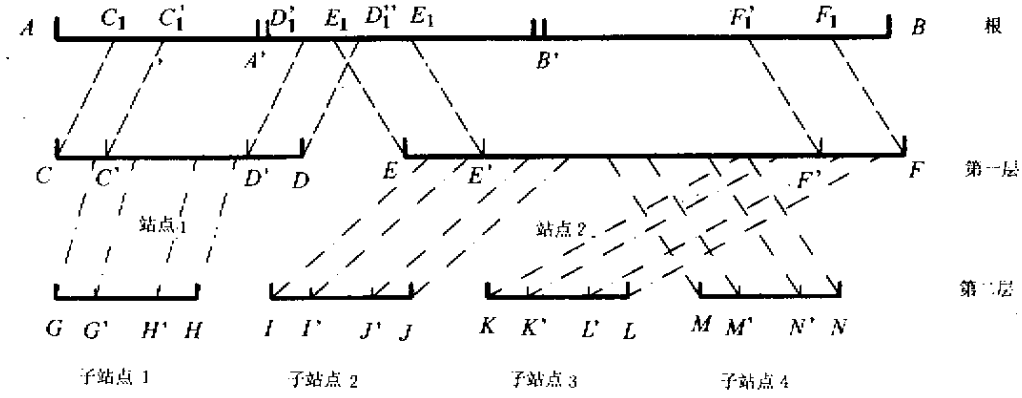

图 2　局部一致性模型

为了对并发控制进行优化，该模型采用了一种灵活的分布式数据存储方式——树形结构. 树的根是固定的，由网络管理员指定. 父站点和其子站点之间共享的数据称为局部共享数据. 在父站点上保存有一份局部共享数据的完整拷贝，称其为完全拷贝；在每个子站点中仅保存局部共享数据的某个子集，称其为局部拷贝；在父站点上与之相对应的数据域被称作局部区域；各站点上当前显示的数据范围称为当前显示区域. 因此可知，根上的完全拷贝即共享数据的完整内容；树的叶节点都是最终用户端(是子站点)；中间节点是其下级的父站点，是其上级的子站点. 所有的站点都可以作为用户端. 例如，在图 2 中，根和第一层上站点(站点 1 和站点 2)是父站点与子站点的关系，则区域 AB 表示父站点(根)上的完全拷贝；$A'B'$ 表示在根上的当前显示区域；区域 CD 表示

站点1上保存的局部共享数据子集,是局部拷贝(区域 EF 也是局部拷贝);根上与区域 CD 相对应的数据域 $C1D1$ 为局部区域;$A'B'$,$C'D'$,$E'F'$ 都是各相关站点的当前显示区域.上述存储结构具有较大的灵活性,各站点(根除外)局部拷贝的范围可根据需要加以伸缩变化,在图2中,局部拷贝 CD 最少可与站点1上的当前显示区域 $C'D'$ 重合,最多可与父站点上的完全拷贝 AB 重合.若所有子站点上的局部拷贝都与完全拷贝重合,则该结构就等同于一种完全复制式存储结构,因此完全复制式存储结构可以当作它的一个特例.

基于上述存储结构,我们给出下面的局部一致性模型.对用户而言,一个基于上述结构的CSCW系统若被认为是一致的,它必须同时满足下面3个条件.

(1) 部数据一致性:当一组协同操作执行完后,各客户端当前显示的数据(当前显示区域)应与其父站点上完全拷贝中的对应部分的数据完全一致.

(2) 局部因果一致性:在系统中,如果各兄弟站点的当前显示区域对应的父站点中的区域有重叠,该区域称为显示重叠区.发生在某显示重叠区中(包括各个站点之间的显示重叠区)的任意两个操作 O_1 和 O_2,若其操作对象属于局部共享数据,且存在时间上的因果关系,即 $O_1 \longrightarrow O_2$,则在该系统中任何当前显示区域包含此显示重叠区的站点上,应有 $O_1 \longrightarrow O_2$.

(3) 预期一致性:在系统中,发生在某显示重叠区中的任意两个操作 O_1,O_2,且它们相互独立 $O_1 // O_2$,若其操作对象属于局部共享数据,则它们在系统中任何执行这些操作的站点上以任何顺序执行的结果都应与其在本地执行的预期结果一致.

上述3个条件中,局部数据一致性和局部预期一致性可以保证子站点与父站点上局部共享数据的操作结果的一致性.由此可推论出,系统中的任意两个站点所包含的局部共享数据应随时保持一致.局部因果一致性则可保证对局部共享数据操作过程的一致性.

根据局部一致性模型的要求,本文制定了一个针对上述结构的的并发控制策略.在系统的并发控制中,为提高系统对本地操作的响应速度,并保证'系统中局部共享数据对用户可见部分的局部一致性',采用了下面的基本策略.

(1) 某子站点上发出操作,将该操作发送给其父站点;父站点接收到子站点发来的操作信息后,首先按接收顺序将其在本地执行,然后根据所操作的数据是否落在相应站点当前的局部区域中,有选择地将其发送给它的子站点.

(2) 某子站点接收到自己的父站点发来的操作信息后,直接按接收顺序将其在本地执行.

下面,我们根据局部共享一致性模型中的三方面要求,对这一策略进行分析.

4.1 局部数据一致性

为了保证共享数据的局部一致性,只需要满足下面两个条件即可:① 保证在某一时刻,每个子站点上的局部拷贝与父站点上对应的局部区域的内容完全一致;② 从该时刻起,所有作用于父站点上局部区域的操作都按相同顺序作用于子站点上对应的局部拷贝.

为满足上述条件,首先在系统初始化和新站点联入时应保证相应站点最初的阶段一致性.在以后的操作中只要满足上述策略,就能保证满足以上条件.

4.2 局部因果一致性

对于某显示重叠区域的操作 O_1,O_2,假设它们有因果关系 $O_1 \longrightarrow O_2$.则有以下两种可能:① 它们都由同一站点发出,且其发出顺序必定是先 O_1,后 O_2,则两操作在其父站点上的接收和执行顺序也是先 O_1,后 O_2;② 它们由不同站点发出,假设 O_1,O_2 分别产生于站点1,2,则 O_1 必先经过父站点转发至站点2并在其上执行,然后 O_2 才得以产生,因而也可以断定 O_1 必先于 O_2 抵达或产生于父站点.

由于父站点最终也按此顺序将其发送给那些局部区域包含此显示重叠区的子站点,因此,可保证所有站点都按先 O_1,后 O_2 的顺序将其作用于各自的本地拷贝,从而保证了局部因果一致性.

4.3 局部预期一致性

为维护操作的预期一致性,在系统中采用标注的方法[3].标注方法立足于共享数据本身,无论某一操作执行前发生了多少并发操作.通过对共享数据增加标注,把并发操作执行所引起的数据变化部分屏蔽起来,使该操作执行时的共享数据状态仍然与该操作产生前的数据状态一致.通过该方法,在可以确保在系统中任何执行这些操作的站点上以任何顺序执行的结果都应与其在本地执行的预期结果一致.具体方法如下:设操作 O 产生前瞬间的站点状态为 S,当 O 传到父站点执

行时,若有 O 的并发操作在父站点上已经先执行,则通过加标注的方法,隐藏由于先执行的并发操作所引起的局部共享数据的变化部分,使得父站点的站点状态仍是 S,然后执行 O,最后,去除所加的标注,恢复局部共享数据的隐藏部分.对于执行该操作的所有其它站点都按以上方法处理.

至此,通过上述策略,可保证各子站点上的局部拷贝满足局部一致性的要求.作为局部拷贝的一个子集,各站点当前显示区域的内容也必定满足局部一致模型中的 3 个条件.

5　结束语

本文讨论了 CSCW 系统中的一致性问题,简述了常用的并发控制策略,介绍了基于完全复制式结构整体一致性模型,分析了该模型的不足之处.在此基础上,给出了局部一致性的定义,对其维护共享数据一致性方面进行了讨论,提出了一种基于树形结构的局部一致性模型.这种模型的实验工作正在进行中.

参考文献:

[1] SUN C. A generic operation transformation scheme for consistency maintenance in real-time cooperation editing system[A]. PAYNE S C, PRINZ W. Proceedings of the International Conference on Supporting. Group Work (Group '97)[C]. Phoenix: Arizona, 1997. 425 – 434.

[2] 冯晨华. CSCW 系统中的并发控制机制的研究[J]. 计算机工程与应用, 1999(4): 21 – 23.

[3] 何鸿君,吴泉源,罗　莉. 协同编辑中维护操作意愿的文档标注方法[J]. 软件学报, 1999, 10(2): 425 – 434.

[4] 郑庆华,李人厚,王余蓝. 分布式多媒体协同工作系统[J]. 软件学报, 1998, 9(增刊): 25 – 28.

[5] 董轩明,徐光佑. 适用于实时协同编著系统的并发控制研究[J]. 小型微型计算机系统, 1996, 17(10): 1 – 6.

The Consistency of and Local Consistency Model for CSCW System

ZHOU Chun – hui, ZHANG　Min, ZHANG Jia – yi

(College of Electrical & Information Engineering, Zhengzhou University of Technology, Zhengzhou 450002, China)

Abstract: Keeping consistency of operating system is one of the key techniques of CSCW. It solves the problems caused by multi – users. First, this article discusses three kinds of inconsistency that may result from operating system conflict; then briefly introduces some control methods of concurrency in CSCW system and describes two models which are based on different distribution structure – integrated consistency model and local consistency model; and, finally, the article presents a tree structure for achieving local consistency model.

Key words: computer supported cooperative work; consistency of system; concurrency control; consistency model