# 一个新的因果序通信协议的研究与设计

The Research and Design of A New Causal Order Communication Protocol

唐文胜　史殿习

(湖南师范大学计算机教学部　长沙410006)(国防科大计算机学院　长沙410073)

**Abstract** In order to support the communication between the group member and support development of the distributed application, we propose a new causal order unicast and multicast mixed communication protocol based on the vector time. This protocol allows group member to send multicast message and unicast message, and assures delivery of these messagen in their causal predence order.

**Keywords** Causal relation, Group communication, Vector time

## 1 引言

对于由多个成员组成的分布式应用如CSCW应用等来说，为了完成一个或多个任务，多个成员通常组成一个或多个协作组。组成员为了完成其所承担的任务通常不仅要给其它成员发送组播消息，而且还要给某个或某些成员发送单播消息，并且，这两类消息相互穿插在组成员所处理的消息流中，形成了一种相互依赖的因果序关系，这种因果序关系对分析、推理和描绘组成员中各种事件之间的关系是非常有用的，有助于解决系统中各种各样的问题，如维护副本数据库相关状态的一致性，避免系统中死锁的出现等。

目前，现有的组通信支持系统如Isis、Transis、Totem以及Horus[1]等系统都没有对这种通信类型提供有效的支持。文[2,3,4]中基于时间向量的因果序协议只保证组播消息之间的因果序关系，而没有考虑单播消息的因果序；文[5,6]中基于二维时间矩阵的因果序协议只保证单播消息之间的因果序语义，而没有考虑组播消息之间的因果序语义；这对于基于组通信的分布式应用尤其是副本数据库应用来说是一个极大的限制。

为了有效地解决这一问题，支持分布式应用的开发，在对相关的因果序协议研究的基础上，我们提出了一个新的基于向量时间的因果序单播、组播混合通信协议CR_UMcast协议。该协议允许组成员既可以发送组播消息，又可以发送单播消息，并且保证按它们之间的因果优先顺序进行传递。

## 2 系统模型

我们假设一个分布式系统是一个由$n$个进程$p=\{a_1,a_2,\cdots,a_n\}$组成的分布式异步系统，即系统中对进程之间的消息传递延时没有限制，对进程的执行速度没有限制，且系统中没有一个同步时钟或一个全局时钟。系统中的进程通过消息传递进行通信，多个进程根据需要可以组成一个或多个组。

为了定义系统中的消息传递，我们引入事件的概念，将一个进程的执行过程看成是一个事件序列。系统中的事件遵从Lamport的之前发生(Happend-Before)关系[7]，用"→"表示该关系，其含义是一个事件潜在地影响另一个事件。一个进程内主要的事件包括：send(m)，recv(m)和dlvr(m)事件。我们将进程接收一条消息的事件recv(m)和传递一条消息的事件dlvr(m)看成两个不同的事件，一条消息可以立刻被接收，但其传递却可以被延迟，直到应用所要求的语义被满足。

## 3 因果序通信的定义

对于一个组来说，在系统运行过程中组成员的关系可能发生变化，如新成员的加入、原有组成员的离开或发生失效等，组成员关系的变化将对组成员之间的通信产生影响。为了避免这种情况的发生，我们假定系统中有专门的成员关系管理机制，负责对组中的成员关系进行管理，当组成员关系发生变化时，该机制保证组中所有正确的成员维护一致的成员关系，并保证所有正确的组成员保持一致的状态。为了给出因果序通信的含义，我们引入组视图的概念。

**唐文胜**　硕士，讲师，主要研究方向：软件工程，分布式系统，多媒体，CAI等。**史殿习**　博士，主要研究方向：分布式系统，软件工程等。

**定义1**(视图) 假设一个组 g，组 g 的一个视图是指由组 g 中当时处于正确的运行状态且彼此之间可以进行相互通信的成员进程组成的进程子集，而且该子集被其中的所有成员进程都认同。我们用 $v_i(g)$ 来表示组 g 的一个视图，每一个视图都有唯一的标识。

**定义2**(因果优先关系) 假设一个组 g，其当前视图为 $v_i(g)$，消息 m1和 m2是 $v_i(g)$ 中的成员在 $v(g)$ 中发送的两条消息，m1和 m2既可以是单播消息，也可以是组播消息；RevMsgSet(m1)和 RevMsgSet(m2)分别表示接收消息 m1和消息 m2的成员集合，且 RevMsgSet(m1)∩RevMsgSet(m2)≠φ。如果 send(m1)→send(m2)，则 ∀ a∈RevMsgSet(m1)∩RevMsgSet(m2)·dlvr(m1)→dlvr(m2)，我们称 m1因果优先于 m2传递。

## 4 向量时间

向量时钟是分布式系统中逻辑时钟的一种实现方式。向量时间可以精确地维护分布式系统中事件之间的因果优先关系。对于一个组 g 来说，组 g 中的每个成员 $a_i$ 都维护一个向量时间 $VT_i[1..n]$，其中，n 为组的大小，$VT_i[i]$是成员 $a_i$ 的局部逻辑时钟，$VTi[j](i\neq j)$ 表示成员 $a_i$ 所知道的成员 $a_j$ 的最新的逻辑时间，整个向量 $VT_i$ 构成了成员 $a_i$ 对全局的逻辑时间的视图。由于我们使用向量时间的主要目的是用其来标识系统中消息的因果关系的，因此，我们用 $VT_i[i]$来表示成员 $a_i$ 到目前为止所发送的消息的数量；$VT_i[j](i\neq j)$表示到目前为止成员 $a_i$ 所知道的成员 $a_j$ 发送的消息的序号；成员 $a_i$ 通过向量 $VT_i$ 来得知组 g 中所有成员所发送的消息的序号。同时，我们用 $VT_m$ 来表示消息 m 的向量时间。

对于一个组成员来说，决定消息 m1、m2的因果优先关系的方法是判断它们的时间向量 $VT_{m1}$ 和 $VT_{m2}$ 的大小。两个时间向量 $VT_1$ 和 $VT_2$ 的大小关系"≤"和"<"的定义如下：

(1) $VT_1\leqslant VT_2$ iff $\forall k\in 1..n: VT_1[k]\leqslant VT_2[k]$

(2) $VT_1<VT_2$ if $VT_1\leqslant VT_2$ and $\exists i: VT_1[i]<VT_2[i]$

## 5 协议设计

为了减少附加在消息上、用于决定消息之间因果优先关系的信息的数量，在 CR_UMcast 协议中假定单播消息是阻塞的，即一个成员发送一条单播消息后，一直处于等待状态，直到接收到接收者的应答为止。作出这样的假设完全是与实际应用相一致的，因为在实际应用中如在副本数据库应用中，一个成员在发送一条单播消息时，通常是等待该消息完成之后再继续执行。基于这一假设，我们便可以采用时间向量来同时解决组播消息与单播消息之间的因果序优先关系。

### 5.1 设计思想

设计 CR_UMcast 协议的难点主要包括如下两个方面：一是每个成员 $a_i$ 在发送组播消息 m 或发送单播消息 m 时如何递增 $a_i$ 所维护的向量时间 $VT_i$ 并将其附加在消息 m 上，以便接收者能够根据附加在 m 上的时间向量来决定消息 m 与其它消息之间的因果优先顺序；二是每个成员 $a_i$ 采用什么样的规则来判断何时将接收到的一条消息 m 传递给应用。

针对第一个问题，我们采取的方法是，每当成员 $a_i$ 发送一条组播消息 m 时，成员 $a_i$ 按递增其所维护的向量 $VT_i$，并将更新后的 $VT_i$ 附加在消息 m 上，而后将消息 m 组播给组中其它成员；而当发送一个单播消息 m 时，成员 $a_i$ 不对时间向量 $VT_i$ 进行修改，而是直接将其附加在消息 m 上，而后将其发送给接收者。

针对第二个问题，我们采用如下的规则来决定是否将一条消息传递给应用：

①当成员 $a_i$ 接收到来自成员 $a_j$ 的时间向量为 $VT_m$ 的组播消息 m 时，直到下面的条件成立，才将消息 m 传递给应用：

$$\forall k\in 1..n: VT_m[k]=VT_i[k]+1 \quad \text{if } a_k=a_j$$
$$VT_m[k]\leqslant VT_i[k] \quad \text{其它} \qquad (1)$$

②当成员 $a_i$ 接收到来自成员 $a_j$ 的时间向量为 $VT_m$ 的单播消息 m 时，直到下面的条件成立，才将消息 m 传递给应用：

$$\forall k\in 1..n: VT_m[k]=VT_i[k] \quad \text{if } a_k=a_j$$
$$VT_m[k]\leqslant VT_i[k] \quad \text{其它} \qquad (2)$$

在具体设计 CR_UMcast 协议时，我们将 CR_UMcast 协议分为两个层次：底层由可靠的单播通信协议 RUP 和可靠的组播通信协议 RMP 构成，负责消息的发送和接收，它们提供可靠的、保证 FIFO 顺序语义的通信语义，顶层由消息传递过程 CRUM_Delivery()组成，该过程使用上面给出的规则对消息进行判断，决定是否将接收的消息传递给应用。下面给出 CRUM_Delvery()组成的算法描述。

### 5.2 算法描述

对每个组成员 $a_i$ 来说，每当其接收到一条来自其它成员的消息 m 时，首先判断消息 m 的类型，而后进行相应的处理，并将经过接收过程处理过的消息传送给消息传递过程 CRUM_Delivery()。图1给出了过程 CRUM_Delivery()的算法描述。其中，变量 C_MsgDelayQ 是一个用于保存消息的延迟队列。每当过程 CRUM_Delivery()接收到底层传送的一条消息 m 时，将消息 m 保存到队列 C_MsgDelayQ 中；而后调用函数 GetMsgFromQ()依次取出队列中每一条消息

m，首先判断消息 m 的类型，如果消息 m 是组播消息，则使用公式(1)中的规则判断消息 m 的向量时间是否满足传递条件，如果满足传递条件，则将 m 传递给应用，并将消息 m 从队列 C_MsgDelayQ 中删除，而后对成员 $a_i$ 所维护的时间向量进行更新；如果消息 m 是单播消息，则使用公式(2)中的规则判断消息 m 的向量时间是否满足传递条件，如果满足传递条件，则将 m 传递给应用，并将消息 m 从队列 C_MsgDelay 中删除。重复上述过程，直到队列 C_MsgDelayQ 中没有满足传递条件的消息为止。

```
Procedure  CRUM_Delivery(m)
 C_MsgDelayQ ← C_MsgDelayQ ∪ {m};
 repeat
  m   ← GetMsgFromQ(C_MsgDelayQ);
  VT_m ← GetVTofMsg(m);
  a_j ← GetSenderID(m);
  Switch MsgType(m) of
   Case 'Mcast':
     If ∀k∈1..n:  VT_m[k] = VT_i[k]+1   if a_k=a_j
                  VT_m[k] ≤ VT_i[k]     otherwise
       dlvr(m);
       C_MsgDelayQ ← C_MsgDelayQ \ {m};
       ∀k∈1..n: VT_i[k] ← max(VT_i[k], VT_m[k]);
   endif
   Case 'Ucast':
    If ∀k∈1..n:  VT_m[k] = VT_i[k]   if a_k=a_j
                 VT_m[k] ≤ VT_i[k]   otherwise
    dlvr(m);
    C_MsgDelayQ ← C_MsgDelayQ \ {m};
    ∀k∈1..n: VT_i[k] ← max(VT_i[k], VT_m[k]);
   endif
 until C_MsgDelayQ not included deliverable msg
```

图1　消息的因果序传递过程

### 5.3　算法正确性证明

本节将证明协议 CR_UMcast 保证按消息之间的因果优先顺序传递消息，并且具有终止性，即不会永久地阻塞一条消息的传递。由于篇幅所限，在此只给出证明的思路。

**定理1**　假设一个组 g，其当前视图为 $V_i(g)$，消息 m1 和 m2是 $V_i(g)$中的成员在 $V_i(g)$中发送的两条消息，m1和 m2既可以是单播消息，也可以是组播消息；RevMsgSet(m1)和 RevMsgSet(m2)分别表示接收消息 m1和消息 m2的成员集合，且 RevMsgSet(m1)∩RevMsgSet(m2)≠ϕ，如果 send(m1)→send(m2)，则协议 CR_UMcast 保证 $\forall a_k \in$ RevMsgSet(m1)∩RevMsgSet(m2)：dlvr(m1)→dlvr(m2)。

分四种情况证明：(1)m1、m2都是单播消息；(2)m1、m2都是组播消息；(3)m1是单播消息、m2是组播消息；(4)m1是组播消息、m2是单播消息，在上述每一种情况中又分为两种情况，即①m1、m2是同一个发送者发送的；②m1、m2是不同的发送者发送的。

**定理2**　假设一个组 g，其当前视图为 $V_i(g)$，消息 m 是 $V_i(g)$中的成员在 $V_i(g)$中发送的一条消息，则协议 CR_UMcast 保证不会永久地延迟消息 m 的传递。

使用反证法分两种情况：(1)m 是单播消息；(2)m 是组播消息来证明定理2的正确性。

**小结**　与文[2,3,4,5,6]中的协议相比，CR_UMcast 协议具有如下两个优点：①允许组成员同时发送单播消息和组播消息，而且保证按它们之间的因果优先顺序进行传递；②采用向量时间来判断消息传递的条件，使得附加在消息上的信息相对要少，随着组大小的变化，附加在消息的消息量的变化为 O(n)；而文[5,6]中的因果序协议由于采用二维时间矩阵来对消息进行因果排序，随着组大小 n 的变化，附加在消息上用于决定消息的因果序的信息量的变化为 $O(n^2)$，因此，CR_UMcast 协议对于[5,6]中的协议来说是一个极大的改进。

在对上述因果序通信协议及相关协议如成员关系管理协议研究的基础上，我们设计实现了一个基于 CORBA 的组通信服务系统 GCS[8]，在 GCS 中设计实现了一个因果序单播、组播通信服务 UMCS[7]，并将该服务成功地应用到我们开发的一个过程管理系统 PDM[9]中，为 PDM 系统的开发提供了有效的支持，取得了令人满意的效果。

## 参 考 文 献


1　Renesse R V, et al. Horrs: a flexible group communications system. Communication of the ACM, 1996, 39(4): 76～83

2　Birman K P, Schiper A, Stephenson P. Lightweight Causal and Atomic Group Multicast. ACM Transactions on Computing Systems, 1991, 9(3): 272～314

3　Cheriton D R, Skeen D. Understanding the Limitation of Causally and Totally Ordered Communicaton. In: Proc. of 14th ACM Symposium on Operating Systems Principles. Asheville, North Carolina, 1993. 44～57

4　Raynal M, Singhal M. Logical Time: Capturing Causality in Distributed Systems. Computer, 1996, 34: 49～56

5　Raynal M, Schiper A, Touge S. The Causal order abstration and a simple way to implement it. Information Processing Letters, 1991, 39(6): 343～350

6　Schiper A, et al. A New Algorithm to Implement Causal Ordering. In: WDAG: Intl Workshop on Distributed Algorithms. Springer－Verlag, Berlin, 1989

7　Lamport L. Time, clocks, and the ordering of event in a distributed system. Communication of ACM, 1978, 21(7): 558～565

8　史殿习，吴泉源，等. 协同应用中组通信服务的研究与设计. 计算机辅助设计与图形学学报，2000，12(1)

9　史殿习，吴泉源，等. 一个基于组通信的设计过程管理系统的设计与实现. 计算机工程与科学，已录用