The Immediate Dependency Relation: An Optimal Way to Ensure Causal Group Communication

Saul Pomares Hernández[1], Jean Fanchon[2], Khalil Drira[2]
[1]National Institute of Astrofisics, Optics and Electronics (INAOE),
Luis Enrique Erro #1, 72840 Tonantzintla, Puebla, Mexico.
[2]Laboratory for Analisys and Architecture of Systems of CNRS,
7 Av. Colonel Roche, 31077 Toulouse Cedex, France
{spomares@inaoep.mx, fanchon@laas.fr, khalil@laas.fr}

Abstract

In this paper we present a study on the subject of the *Immediate Dependency Relation* (IDR), and we show how by extending the IDR relation, one can ensure a global causal delivery in group communication, including in the overlapping group case. The main objective of this paper is to show that the use of the *Immediate Dependency Relation* (IDR) obliterates the notion that causality is expensive to set up in distributed systems. The IDR minimizes considerably the amount of control information sent per message to ensure causal ordering, without imposing restrictions in interaction (e.g. network topology, redifussion servers, executions models etc.). In order to demonstrate the feasibility of its implementation, we present an optimal broadcast causal protocol inspired by the IDR relation. We show the optimality of the protocol in terms of the amount of control information timestamped per message.

1. Introduction

In Group Communication Systems (GCS), causal ordering protocols are an essential tool to exchange information. The use of causal ordering provides built-in message synchronization and reduces the non-determinism in a distributed computation. Causal ordering provides an equivalent of the FIFO property at a global group communication level, where messages sent sequentially to the same group of processes over a communication network are replaced by sending causally related messages. The concept of causal ordering is of considerable interest to the design of distributed systems and can be found in applications of several domains, such as distributed cooperative engineering, teleconferencing [RAV92], multimedia systems [BAL95], mobile computing [PRA95], resource allocating protocols [TOR99] and security domains [SCH90].

We performed a relative study concerning various existing causal protocols, particularly of those protocols which support multi-group environments. We found that their use in the applications previously mentioned is not suitable for two main reasons: they either present restrictions in the mode of interactions, or they introduce a large amount of control information. The amount of control information (CI) in order to guarantee causal ordering in a group g_1 with n members is $\theta(n)$ [BIR91] (broadcast case) in the worst case. In overlapping groups, in the worst case, the amount of CI is $\theta(g \bullet n)$ [BIR91], where g is the number of groups in the system. For large values of n and g, the bound of $\theta(n)$ and $\theta(g \bullet n)$ (according to the case) is prohibitively high.

The objective of this paper is to show that the use of the *Immediate Dependency Relation* (IDR) obliterates the notion that causality is expensive to set up in distributed systems. The IDR minimizes considerably the amount of CI sent per message, without introducing restrictions in interaction (i.e. all processes are equals). We present in this paper an optimal causal protocol inspired by the IDR relation. The protocol works in a broadcast environment. It is optimal in terms of the amount of CI timestamped per message. Although the bound of control information (*CI*) timestamped per message remains $\theta(n)$ in the worst case, we demonstrate that the probability that this will occur is minimal.

The paper proceeds as follows:
First, in Section 1 we present a brief backgroup on the causality principle. A state of art on related work is presented in Section 2. Next, we present in Section 3, an in-depth study of the IDR principle for the broadcast case and the IDR extention

to support the multi-group case. In Section 4, we present the minimal causal broadcast protocol. Conclusions are provided in Section 5

1.1 Background

Causal ordering delivery is based on the causal precedence relation defined by Lamport [LAM78]. This relation connects two messages represented by pairs (identifier of site, clock value) in the following way:

Definition* 1:* The causal relation, denoted by $\rightarrow$, is defined by the following three rules:
1. $\langle i, a\rangle \rightarrow \langle j, b\rangle$ *if* $i=j \wedge a < b$
2. $\langle i, a\rangle \rightarrow \langle j, b\rangle$ *if* $\langle i, a\rangle$ is the sending of a message and $\langle j, b\rangle$ is the delivery of that message.
3. $\langle i, a\rangle \rightarrow \langle j, b\rangle$ *if* $\exists\langle k, c\rangle \mid (\langle i, a\rangle \rightarrow \langle k, c\rangle \wedge \langle k, c\rangle \rightarrow \langle j, b\rangle)$
where *i, j,* and *k* are process identifiers, and *a, b,* and *c* are local clock values of *i, j,* and *k,* respectively.

Causal ordering delivery in group communication presents two cases: the broadcast case (one group) and the multi-group case that includes overlapping groups. The causal delivery for the broadcast case is defined as follows [BIR93]:

Definition 2 Causal broadcast delivery (one group):

$$\textbf{IF } send(m) \rightarrow send(m'), \textbf{ then } \forall k \in c : \\ delivery_k(m) \rightarrow delivery_k(m')$$

Causal broadcast delivery ensures that if the diffusion of a message *m* causally precedes the diffusion of a message *m*', in a group *c*, then the delivery of *m* causally precedes the delivery of *m*' for all participant p_k that belongs to *c*.

In the rest of the paper we simply note *send*(*m*) by *m*. Thus, by definition we have:

$$m \rightarrow m' \Leftrightarrow send(m) \rightarrow send(m')$$

The case of causal delivery in a multi-group environment is more common in group communication. It is defined as follows:

Definition 3 Causal multi-group delivery

$$\textbf{If } send_i(m,c) \rightarrow send_j(m',c'), \textbf{ then } \forall k \in c \bigcap c' : \\ delivery_k(m) \rightarrow delivery_k(m')$$

We can say, in a general way, that *causal multi-group delivery* guarantees that if the diffusion of a message (*m*, *c*) causally precedes the diffusion of a message (*m*', *c*'), where *c* and *c*' are the original diffusion groups of messages *m* and *m*', respectively, then the delivery of *m* causally precedes the delivery of *m*' for all participants p_k that belongs to the intersection of groups *c* and *c*' [MOS93].

In the rest of the paper we note $send_i(m,c)$ by (m,c), without further representing the site sender *i*, except if it is needed for clarification. Thus, we have by definition:

$$(m,c) \rightarrow (m',c') \Leftrightarrow send(m,c) \rightarrow send(m',c')$$

A bit of history

The causal diffusion of messages is usually ensured by sending control information (CI) attached to each message. Among one of the pioneer protocols that apply causal diffusion, we can cite the causal protocol of ISIS [BIR87B]. In the first

version of ISIS, a message transports the history to the messages that causally precede it. Let's suppose that a message *m* arrives to a recipient, and that all the messages in its past addressed to that recipient have already been received. In this case, *m* will also be delivered to the recipient. Otherwise, the appropriate messages on hold will be removed from the list of preceding message joint to *m*, and they will then be delivered to the recipient. After that, *m* will be delivered. In this case, a complex mechanism is required to prevent the unlimited growth of control information.

In the late 1980's Mattern and Fidge introduced a new mechanism, called *vector time clocks* [FID88, MAT89]. In vector time clocks, contrary to the first version of ISIS, each message *m* transports control information composed of ordered sets of type (site origin, vector time clock). When a recipient receives *m*, he uses the control information associated to it to determine if a causal predecessor of *m* must still be delivered to the recipient. If this is not the case, then *m* will be delivered to the recipient. Otherwise, *m* will be placed in a recipient buffer until all the causal predecessors addressed to that recipient have been delivered.

2. Related Work

We distinguish two main approaches to resolve the problem regarding the amount of control information needed to ensure causal group communication:

1. Partial causal ordering
2. Complete causal ordering
    i. Symmetric
    ii. Asymmetric

2.1 Partial causal ordering

Partial causal ordering consists in bounding in advance the amount of control information transmitted [HEL00, TOR99]. Its main drawback is that it no longer completely preserves causality. This approach can only provide an exact ordering under favorable circumstances. It is usefully in systems where ordering messages only impacts their execution, but does so not necessarily with precision. Such works include, for example, resource allocating protocols [TOR99].

2.2 Complete causal ordering

The objective of complete causal ordering is to reduce the size of the causal information transmitted, without losing causal dependencies. We can classify two categories of the associated protocols: symmetric protocols and asymmetric protocols, which we will describe next.

2.2.1 Causal symmetric protocols

Symmetry is generally defined as the:
"Regularity and harmony in the ordering of parts of a whole, or in the arrangement of concurrent elements to give the impression of unity" [HAC96]

We use the term symmetry in the sense that all participants share the same role and the same degree of responsibility in the system. Furthermore, a freedom of interaction exists between them. Consequently, the symmetric category does not make any assumptions regarding the group structure nor the topological network structure. This category is concerned with identifying the necessary conditions to ensure a causal delivery of messages, and/or with arranging optimal coding to represent and transmit this information

One of the first protocols in this category is the works of Peterson [PET89]. Peterson introduced the *context graph*. The context graph was designed to represent the causality between messages in a conversation protocol in a broadcast environment. This graph is directly acyclical; its vertexes correspond to the total set of messages, and the arcs represent the causal relationship between these messages. The reduced graph (Fig. 1 b) shows that if causal ordering of messages is ensured between every pair of immediate causal predecessor and successor messages, then causal ordering among all

messages will be automatically ensured due to the transitivity of the preceding relation. We use the graph in this paper only to depict the causal relationship that exists between a sub-set of messages. Nonetheless, no protocol presented in this article pretends neither to support nor utilise these graphs for their implementation.

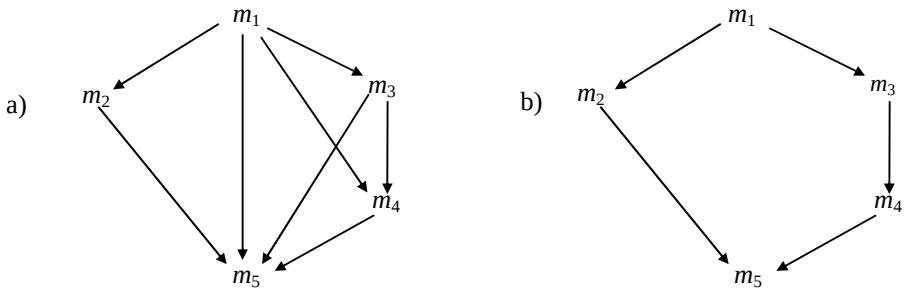

Fig. 1 Context graphs a) Display of the context relation between messages
b) Display of the direct dependency relation between messages

This category includes the multicast protocol of Kshemkalyani and Singhal [KSH96]. Their causal protocol codes information onto each message regarding all previous causal messages that are not yet guaranteed to be delivered in a causal manner by the protocol, or that are not known to have already been delivered. They refer to these conditions as p*ropagations constrains* (Definition 4). The propagation constraints (PC) specify the conditions on the information propagation to enforce causal ordering [KSH96].

Definition 4. The information $d \in m_{i,a}.Dests$ concerning a message $m_{i,a}$ sent to d must propagate along all causal paths, starting at event (i, a) up to, and only up to, the earliest events (j, b) on any such path, such that either:

PC 1. $delivery_d(m_{i,a}) \overrightarrow{=} (j, b)$, i.e. it is known that $m_{i,a}$ has been delivered or

PC 2. $\exists\ (k, c) \mid (i, a) \rightarrow (k, c) \rightarrow (j, b) \wedge d \in m_{k,c}.Dests$ i.e. it is guaranteed that it will be delivered in causal order.

The first PC means that the message $m_{i,a}$ is known to have been delivered in causal order at event (j, b) to destination $d = j$. Therefore, in any future multicast message to destination d, the message does not need to carry information concerning $d \in m_{i,a}.Dests$. The second PC means that there exists an event (k, c) that would ensure the causal delivery of the present message (j, b) with respect to (i, a). If (k, c) is causally delivered with respect to (i,a), the information about $d \in m_{i,a}.Dests$ does not need to be sent.

Another example of a protocol that falls into this category is the protocol proposed by Birman [BIR91], which is based on vector time clocks. The author, in his protocol, proposes to compress the vector time clocks by sending only the vector positions that have changed between the diffusion of the message in question and the position of the current vectors. The main disadvantage of this approach is the unnecessary use of additional memory.

The work of Prakash and Raynal [PRA97] belongs in this category as well. Their work, as Peterson's work, uses the property of direct dependency to reduce the control information. This work extends the direct dependency property so as to support multicast environments. The Prakash's work does not use nor maintain context graphs to ensure causal ordering. To ensure causal ordering, a message m needs to carry information about only those messages m' on which its delivery is directly dependent upon. This approach is oriented to resolve problems in mobile agent applications. We will present and explain in detail the immediate dependency relation in Section 3.

The characteristics of symmetric protocols can be summarized by the following four points:

- In the worst case, the bound of the size of the control information remains at the maximum value of $\theta(n)$, $\theta(g \bullet n)$ or $\theta(n^2)$, as the case may be [CHA91, SCH94, KSH98B].
- All participants interact without an intermediary.
- All participants have the same role.
- The protocols may be used in real-time applications.

2.2.2 Causal Asymmetric protocols

The asymmetric category is composed of solutions that assume either a certain network topology, a particular group structure [BAL97], and/or execution models [ROD95].

The main drawback of protocols that take into account a particular network topology is that they can only be used in applications where the topology is fairly static, or when it can be calculated at the moment of compilation or configuration. These protocols turn out to be completely inadequate in the cooperative applications we consider, since it is impossible to know in advance when, where, or who (user, host, participant, agent, etc.) will participate.

An example of a protocol that assumes a network topology is the work done by Rodrigues and Verisismo [ROD95], who use causal separators as barriers to filter the information concerning the messages addressed to all the members of a causal zone (set of participants delimited by one or several causal separators). A causal zone never sends control information outside of its boundaries. Communication between participants who belong to different networks takes place through the use of predetermined re-diffusion servers, as shown in Fig 2a.

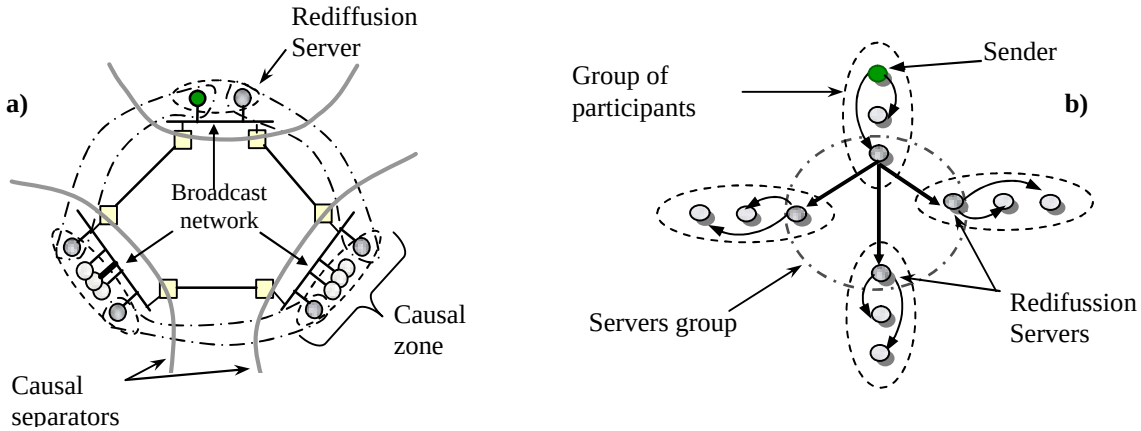

Fig. 2 a) Causal separators, b) Daisy Architecture

An example of a protocol that assumes a certain group structure is proposed by Baldoni et al [BAL97] who divide the participants into logical local groups, and who use causal servers to disseminate messages across these groups. By doing so, the amount of control information is reduced to the local group size. For example, in order to transmit a broadcat message to all existing participant in Fig. 2b, first the message must be transmitted in a local manner to the group to which it belongs; and then, the participant who functions as the local causal server rediffuses the message among the causal servers group (one server per group), which in turn, will rediffuse the message in a internal manner (Fig 2b).

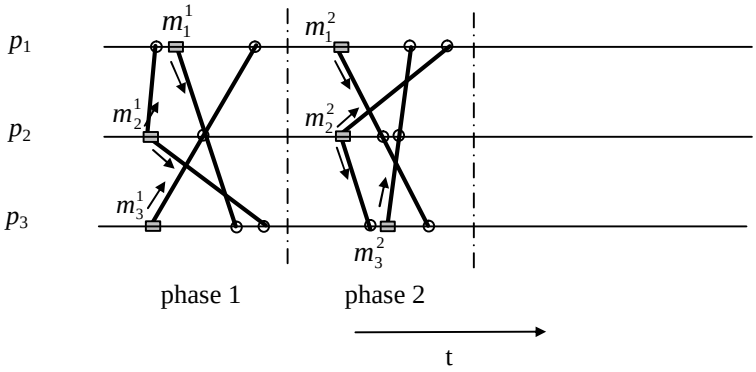

Fig. 3 Low Cost Approach

A last example of protocols based on particular execution models is the work done by Mostefaoui et al [MOS93], named *low cost approach* (Fig 3.). In this work, the diffusion of messages occurs in phases. In each phase, all participants diffuse one, and only one, message. A new phase begins only when all messages in the previous phase have been delivered. In other words, the execution is synchronous [AWE85]. In this scheme, the use of bandwidth is not optimal due to the fact that all participants must transmit one message per phase, even when they do not have data to tranmsit.

The characteristics of asymmetric protocols can be summarized by the following two points:

- Usually the protocols introduce an additional delay in the causal delivery of messages, due to either the use of re-diffusion servers, or to the chosen execution model.
- Complex organization and synchronisation structures are used.

3. The Immediate Dependency Relation

We have indicated in Section 2.2 that the control information overhead timestamped per message and in local storage for the broadcast case is $q(n)$ [SCH94], where n is the number of participants in the group. Likewise, the overhead information in the multi-group case is $q\,(g \bullet n)$ [BIR91], where g is the number of multi-recipient groups. In this section, we demonstrate that it is not always necessary to transmit all the control information (either $q(n)$ in the broadcast case, or $q(g \bullet n)$ in the multi-group case) attached to each message.

3.1 The Immediate Dependency Relation in a broadcast environment

The Immediate Dependency Relation (IDR) [PRA97] is the propagation threshold of the control information *CI*, regarding the messages sent in the causal past which must be transmitted to ensure a causal delivery. We denote it by $\downarrow$, and its formal definition is the following:

Definition 5 Immediate Dependency Relation $\downarrow$ (IDR):

$$m \downarrow m' \Leftrightarrow [\,(m \rightarrow m') \wedge \forall\, m'' \in M, \neg (m \rightarrow m'' \rightarrow m')]$$

Thus, a message m directly precedes a message m', iff no other message m'' belonging to M exists (M is the set of messages of the system), such that m'' belongs at the same time to the causal future of m, and to the causal past of m'.

This relationship is important since we show that if the delivery of messages respects the order of their diffusion for all pairs of messages in IDR, then the delivery will respect the causal delivery for all messages. This property is formalized as follows:

Proposition 1

$$\begin{aligned} &\text{If } \forall\, m,m' \in M,\ m \downarrow m' \Rightarrow \forall k \in c : \\ &\qquad delivery_k(m) \rightarrow delivery_k(m') \\ &\text{then } m \rightarrow m' \Rightarrow \forall k \in c : \\ &\qquad delivery_k(m) \rightarrow delivery_k(m') \end{aligned}$$

Causal information that includes the messages immediately preceding a given message is sufficient to ensure a causal delivery of such message.

We will next present an in-depth study of the IDR in the context of serial, as well as concurrent, messages.

3.1.1 Serial messages

Consider the diffusion of message m_4 ($send(m_4)$) such that $m_2 \downarrow m_3 \downarrow m_4$ (Fig. 4a). The only control information *CI* attached to m_4 in order to ensure a causal order relates to m_3, which is the only message in an immediate dependency relationship with m_4. The delivery of m_4 to participant p_k ($delivery_k(m_4)$) ensures that m_3 will be delivered to p_k in a causal order relative to m_4 ($m_3 \rightarrow m_4$). For all messages send(m') (in the example $m'=m_2$) in the causal past of m_3, such that $m' \downarrow m_3$, if m' is delivered in a causal order relative to m_3, then through the transitivity property, m' is guaranteed to be delivered in a causal order relative to m_4 ($m_2 \rightarrow m_4$). In other words, (Figure 4b), m_4 will be delivered once m_3 has been delivered; m_3 will be delivered once m_2 has been delivered, and so on.

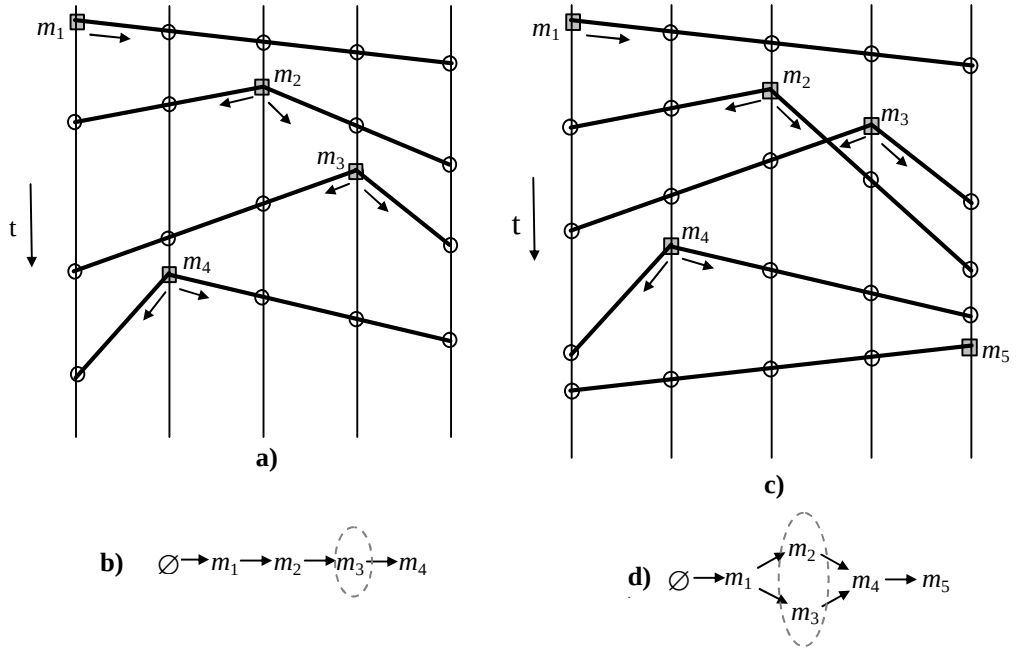

Fig. 4 a) Diagram of serial messages b) Serial precedence graph
c) Diagram of concurrent messages d) Concurrent precedence graph.

3.1.1 Concurrent messages

The case of serial messages is not very common in distributed systems. Generally, distributed systems present the important characteristic of simultaneous execution, known as concurrency. Concurrency allows a distributed system to deal with numerous messages at the same time. The concurrency relation $\|$ is defined as the dual of the causal relationship $\rightarrow$; it is formally defined as follows:

Definition 6 Concurrence Relation $\|$

$$m \| m' \Leftrightarrow \neg (m \rightarrow m' \vee m' \rightarrow m)$$

Definition 6 indicates that a message m is concurrent to another message m' iff neither m causally precedes m', nor m' causally precedes m.
By using the IDR definition (Definition 5) and the definition of the Concurrence Relation (Definition 6), $\downarrow$ and $\|$ respectively, we can conclude that:

Proposition 2 The set of immediate predecessors to a message m are mutually concurrent.

$$(m' \downarrow m \wedge m'' \downarrow m) \Rightarrow m' \| m''$$

This property means that the sufficient and necessary control information *CI* to send with *m* only concerns the concurrent messages that have an immediate precedence relation with *m*. For example, let's consider a message m_4 such that $(m_2||m_3)\downarrow m_4$ (Figure 4c). The sufficient information to transmit with m_4 concerns the messages m_2 and m_3 ($CI = \{m_2, m_3\}$), in other words, the messages with an immediate precedence relation with m_4. The delivery of m_4 to participant p_k ($delivery_k(m_4)$) ensures that m_2 and m_3 will be delivered to p_k in a causal order relative to m_4 ($(m_2||m_3)\rightarrow m_4$). This means that m_4 will only be delivered once both m_2 and m_3 are already delivered.

3.2 Analysis of the IDR principle

The size of the control information attached to a message *m* depends, therefore, on the number of concurrent messages that have an IDR with *m*. In the best case, when we are dealing with a serial message, we note that $|CI| = 1$. Since the set of messages diffused by one participant cannot be mutually concurrent, the worse case turns out to be $|CI| = n$. Two conditions must occur for the worst case ($|CI| = n$) to take place (Fig. 5):

Condition 1- The degree of diffusion of messages must be equal to *n*, i.e. a parallel diffusion of *n* messages
Condition 2- Participant p_i presents an idle diffusion during a time interval $\Delta_i(n)$, where $\Delta_i(n)$is the reception time of *n* messages of condition 1.

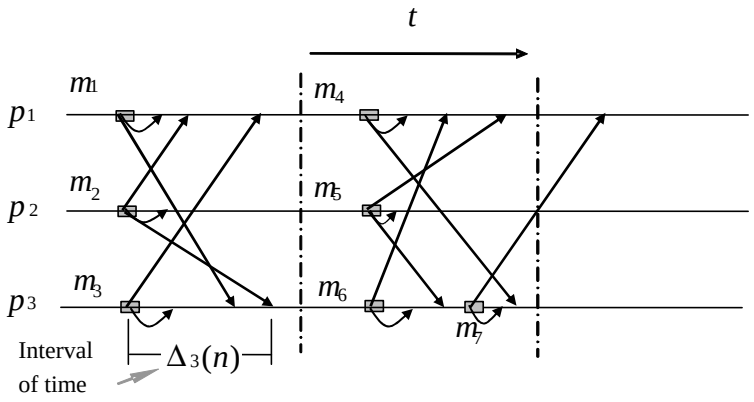

Fig. 5 Worst case diagram, when n=3

We consider the worst case to occur during the diffusion of each message at a local level, not at a global level. For example, as Fig. 5 shows, the worst case occurs during the diffusion of each message m_4, m_5, and m_6. This can be explained by the fact that if one considers that each participant automatically receives their diffusions, then participants p_1, p_2, and p_3 will receive *n* concurrent messages (Condition 1), and all of them will have remained inactive during the reception of these *n* messages (Condition 2). Therefore, during the next diffusion of e_4, e_5, and e_6 by p_1, p_2, and p_3 respectively, the worst case will result for each diffusion, meaning that $|CI| = n$,

What is the likelihood that the worst case will arise for a message *m*? In order to answer this question, we have used the Poisson law and Bernoulli's law. We have found that in the case of the example presented in Fig. 5, the likelihood that the worst case will occurs when n=3 is 0.0497. For values greater than n=3, the likelihood becomes negligeable; for further details, see [POM02B], [FAY85] and [JAC89].

3.3 The Immediate Dependency Relation in a Multi-Group environment

In Section 4.1, we showed in detail the theory regarding the Immediate Dependency Relation (IDR) for the diffusion of broadcast messages.

In this section, we will first define *Causal Inter-group Delivery*, which ensures a coherent delivery. We will then introduce an extension to the principle of immediate dependency, which treats the case of multicast multi-group diffusion. We call this extension the "Immediate Inter-group Dependency Relation" and we refer to it by its acronym IIGDR. In order to study

the diffusion of inter-group messages and particularly the new IIGDR property, we present the "Multi-group Message Diagram" as an aid to fully understand each property presented in this paper.

3.3.1 Immediate Inter-group Dependency Relation

In the same way that the IDR allows us to define the sufficient control information in a broadcast case, the *Immediate Inter-group Dependency Relation* (IIGDR) allows us to characterize the sufficient control information to ensure the causal delivery of messages in a multi-group case.

Definition 7 Immediate Inter-group Dependency Relation (IIGDR) $\uparrow$:

$$(m,c)\uparrow(m',c') \Leftrightarrow [((m,c) \rightarrow (m', c'))\wedge\forall(m'', c'')\in M, ((m,c)\rightarrow (m'', c'')\rightarrow (m', c') \Rightarrow c''\neq c \wedge c''\neq c')]$$

The definition presented above means that a message (m, c) has an immediate dependency relation with an message (m', c') if for all (m'', c'') that belong to the set of messages of system M, such that $(m, c)\rightarrow (m'', c'')\rightarrow (m', c')$, then group c'' is different than c and c' (where c, c' and c'' are the diffusion groups of messages m, m' and m'', respectively). We note that if only one group exists, the relation '$\uparrow$' coincides with the immediate dependency relation denoted as '$\downarrow$', which has been previously defined.

We show that if the delivery of messages respects the diffusion order of the messages in an IIGDR, then all the messages will be delivered in this order.

Proposition 3

$$\text{If } \forall\ (m,c),(m',c') \in M,\ (m,c) \uparrow (m',c') \Rightarrow \forall k \in c \bigcap c'\ delivery_k(m) \rightarrow delivery_k(m')$$

$$\text{then } \forall\ (m,c),(m',c') \in M,\ (m,c)\ \rightarrow\ (m',c') \Rightarrow \forall k \in c \bigcap c'\ delivery_k(m) \rightarrow delivery_k(m')$$

The attached information concerning the messages immediately preceding a given message is therefore sufficient to ensure the causal delivery of this message.

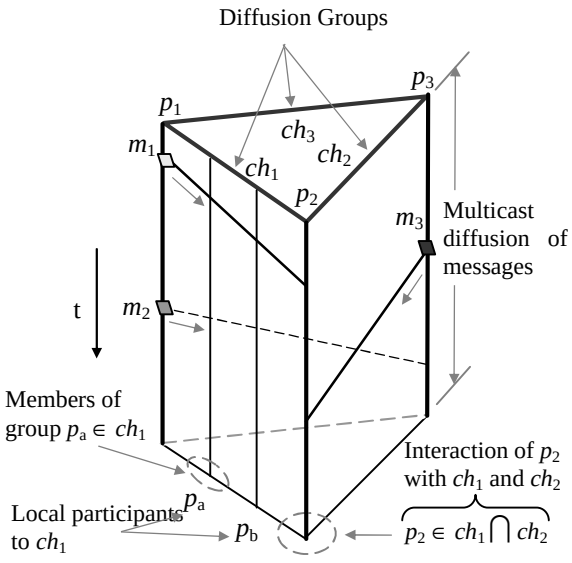

Fig. 6 Multi-Group Diagram

3.3.2 Multi-group Diagram

The three-dimensional multi-group diagram (Fig 6) may be used to represent numerous scenarios. Even if this diagram is not applicable at all times in a broad manner, it is useful to graphically illustrate the complex concept of causality in a multi-group environment.

This diagram was created so as to clearly represent:

- the interaction of each participant in the groups (according to the case)
- the participant members in the groups
- the diffusion and reception of messages sent to numerous recipients in the course of time

3.3.3 Messages with an Immediate Inter-group dependency

In order to better illustrate Definition 7, we present the following scenario as example.

Scenario: Consider a message m_4, such that $((m_2, ch_1)\|(m_3, ch_1))^{\uparrow}(m_4, ch_3)$, as shown in Figure 7. According to Definition 7, the messages which have an IIGDR with m_4 are messages m_2 and m_3. Therefore, the information that corresponds to these messages is sent as control information to m_4. This information is not taken into account for the delivery of m_4 by participant p_3 since $p_3 \notin ch_1 \bigcap ch_3$.

Participant p_3 only uses this information to update his system history file and for future diffusion of messages, as in the case of the diffusion of message m_5**.**

Let's now consider message m_5 (Figure 7). At the moment of diffusion of m_5, we apply Definition 7 to each message in the causal history of p_3. We find that the messages that have an IIGDR with m_5 are m_2, m_3 and m_4. Thus, as in the previous cases, the control information timestamped to message m_5 corresponds to the messages which have an IIGDR to m_5.

Message m_5 is delivered to p_2 ($p_2 \in ch_1 \bigcap ch_2$) only after messages m_2 and m_3 have been delivered. These messages, m_2 and m_3 are delivered to p_2 only after message m_1 has been delivered. This is ensured by the immediate dependency relation (IDR)

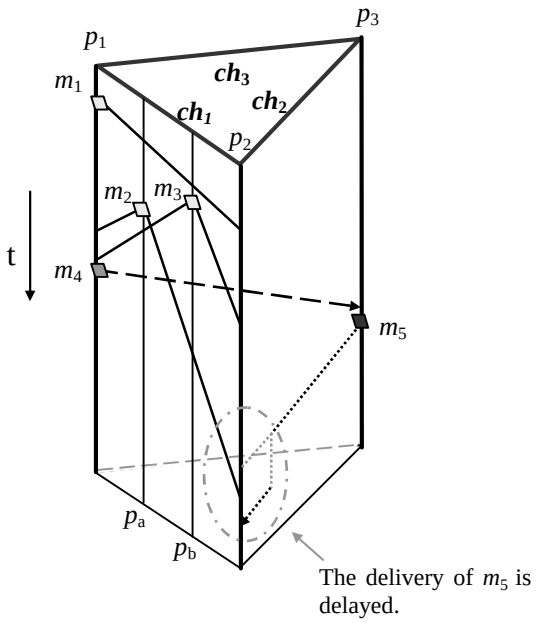

Fig. 7 Multi-group scenario

4. The Minimal Broadcast Causal Protocol (MBCP)

In order to reduce the amount of information, our broadcast causal protocol is based on the IDR property described in Section 3.1. Therefore, the only causal information timestamped to each message *m* corresponds to the messages with which it is linked to by an IDR. We demonstrate that our protocol, by using the IDR principle, is optimal in the sense that it transmits the minimal and necessary amount of control information to completely preserve the causal order.

4.1 Description of structures used in the protocol

To attain minimality, each participant manages in a local fashion the following information:

- $VT(p_k)$ is the vector timeclock. The size of the vector is equal to *n*. It is here that we keep track of the number of messages diffused by the participant.
- The structure of the control information CI_i is a set of entries $ci_{k,t_k} = (k, t_k)$. Each entry in CI_i denotes a message that is not ensured by participant p_i of being delivered in a causal order. The entry (k, t_k) represents a diffusion by participant p_k at a logical local timeclock $t_k = VT(p_k)[k]$.
- The structure of a message m_{i,t_i} is a quadruplet $m_{i,t_i} = (i, t_i, message, H_m)$, where *i* is the participant identifier, $t_i=VT(p_i)[i]$ is the logical local timeclock at node *i*, *message* is the message in question, and H_m contains the set of all entries that have an IDR with m_{i,t_i}.

1. Initially,
i. $VT(p_i)[j] = 0 \ \forall \ j{:}1\ldots n.$
ii. $CI_i \leftarrow \varnothing$

2. For each diffusion of message *send*(*m*) to p_i
i. $VT(p_i)[i] = VT(p_i)[i] +1$
ii. $H_m \leftarrow CI_i$

3. $m= (i, t_i=VT(p_i)[i], message, H_m)$
i. **Diffusion *send*(*m*)**
ii. $CI_i \leftarrow \varnothing$

4. For each reception *receive(m)* à p_j , $i \neq j$,
$m=(k,t_k,message, H_m)$
To enforce a causal delivery of *m*

i. *Delivery condition*
if not ($t_k = VT(p_j)[k] +1$ **and** $t_l \leq VT(p_j)[l] \ \forall \ (l, t_l) \in H_e$)
then
wait
else
ii *delivery*(*m*)
iii. $VT(p_j)[k] = VT(p_j)[k] +1$
iv. **if** $\exists ci_{s,t'} \in CI_j \,|\, k = s$
then
$CI_j \leftarrow CI_j \setminus \{ ci_{s,t'} \}$
endif
$CI_j \leftarrow CI_j \bigcup \{(k, t_k)\}$
v. $CI_j \leftarrow CI_j \setminus H_m$
vi. **endif**

Table 1 The Minimal Broadcast Causal Protocol (MBCP)

Each structure in the broadcast protocol has a well-defined function. Each function is in correlation with what the information held in reserve in each of its structures represents. The information in the vector time *VT*( ) contains either a *complete view* or a *partial view* of the causal system. We consider it to have a complete view if at a given instant *t*, the board contains the information regarding the last message diffused by each participant. Otherwise, we consider it to have a partial view. It is by means of the structure *VT*( ) that we can guarantee a complete causal delivery. The process of bringing up to date the structure *VT*( ) is described in Section 2.2.

The information in *CI* has a partial copy of *VT*( ) at any given moment, meaning that:

$$\text{if } \exists (i, t) \mid (i, t) \in CI_k \text{ then } VT(p_k)[i] = t$$

The updating process of *CI* takes place at the moment of diffusion or reception of a message, as shown in Table 1 (lines 1.ii, 4.iv and 4.v). In line 1.ii, the diffusion of a new message *m* ensures, through the IDR property, the causal delivery of the messages contained in H_m. In line 4.iv, by means of the property of concurrent messages, we observe that the messages diffused by the same participant are not concurrent, and thererefore, we do not store in the *CI* the information concerning each participant's most recent message.

The information in structure H_m contains the messages which have an IDR with the message in question. Structure H_m is created at the moment of diffusion of a message (line 1.ii). The information in H_m is unique for each local diffusion message *e*, except in the case where $H_m=\varnothing$.

At this time we will progressively explain the implementation of the broadcast protocol, by using the example described in Figure 4c.

4.2 Scenario example

Example. Consider the group of participants $g = \{p_1, p_2, p_3, p_4, p_5\}$ and the diffusion of message m_4 to p_2. Prior to the delivery of m_4 to p_5, $CI_5 = \{(3,1),(4,1)\}$ and $VT(p_5) = \{10110\}$ must take place (Fig. 4c). This is deduced from the broadcast causal protocol (Table 1).

Diffusion of message m_4 to p_2

2.i The value of vector $VT(p_2)[2]$ is incremented by one. This ensures the sequential delivery of all messages *m* diffused by the same participant p_k.
2.ii CI_2 is assigned to H_{m4}, therefore $H_{m4} = \{(3,1),(4,1)\}$. The information in H_{m4} corresponds to the concurrent messages with an immediate relation to m_4.
3.i Diffusion of message m_4=(2,1,message, H_{m4}), *send*(m_4). We note that the protocol is intended for the use in multirecipient communication groups; however, it may be applied to communication supports that simulate multirecipient communication.
3.ii CI_2 is erased. All the messages in the causal past of p_2 that have an immediate dependency relation with m_4 are guaranteed to be delivered in a causal order after the message *delivery*(m_4).

Delivery of message m_4 to p_2

4.i Once each participant receives a message, he confirms that the message satisfies the delivery condition, which verifies that the reception message in question fulfills the causal property. In this case, message m_4 responds to the delivery condition, and therefore, is delivered to the application.
4.iii The value of vector $VT(p_5)[2]$ is incremented by one. This way, we maintain, in a local manner, a record regarding the messages emmited causally by each participant.
In other words, the information stored in $VT(p_k)$ is a register of the causal history of the group, as seen by participant p_k.

CI_5 is updated in the following manner:

4.iv If an entry of participant *k* already exists in CI_5, then it is updated by the last message received in p_k. In this case, there is no control information about p_2 in CI_5. Thus, only entry (2,1) is associated to CI_5. At this moment during the implementation of the protocol, $CI_5=\{(3,1),(4,1),(2,1)\}$
4.v All entries concerning the immediate predecessors of m_4, including H_{m4}, which also exist in CI_5 are erased from CI_5. On account of the transitivity property, the delivery of message m_4 guarrantees that all the immediate predecessors, including in H_{m4}, will be delivered in a causal order. In this example, m_4 arrives with $H_{m4} = \{(3,1),(4,1)\}$; thus, $CI_5 = CI_j \setminus H_{m4} = \{(2,1)\}$

5. Conclusions

In this article we have presented a detailed study of the immediate dependency relation and an extention to the IDR relation to support multi-group communication. We showed how through its use we can guarantee, in an optimal manner, causal

group communication for the basic causal broadcast case, as well as for the generic multi-group communication case. We demonstrated in this article that by using the IDR relation, it is possible to build optimal causal protocols with desirable characteristics for distributed systems, such as:

- *dynamic configuration* which allows it to adapt its composition in the course of time
- *symmetric organization* which considers all participants to be equal, thus, permitting them to interact without the need of mediators (servers) and
- *asynchronous diffusion* of events, providing participants the freedom to interact at the moment they desire.

In addition, we have presented an efficient broadcast causal protocol. It is optimal as far as the quantity of control information needed to be sent per message.

Bibliography

[AWE85] B. AWERBUCH. "Complexity of Network Synchronization," *Journal of the ACM*, Vol. 32, 4, pp. 801-823, 1985.

[BAL95] R. BALDONI, Prakash R., Raynal M., Singhal M., "Efficient causally ordered communications for multimedia real-time applications", In Proceedings of the 4th International Symposium on High Performance Distributed computing, pp 140-147, Whasington, D.C., Aug. 1995.

[BAL97] R. BALDONI, R. FRIEDMAN, R. VAN RENESSE. "The Hierarchical Daisy Architecture for Causal Delivery," 17th IEEE Int'l Conference on Distributed Computing Systems, May 1997.

[BIR87B] K. BIRMAN, T. JOSEPH. "Reliable Communication in Presence of Failures," *ACM Trans. Compt. Syst.*, 5(1), pp.17-76, Feb. 1987.

[BIR91] K. BIRMAN, A. SCHIPER, P. STEPHENSON. "Lightweight Causal and Atomic Group Multicast," *ACM Trans. Compt. Syst*. Vol. 9, No. 3, pp. 272-314, Aug. 1991.

[BIR93] K. BIRMAN. "The Process Group Approach to Reliable Distributed Computing," *Communications of the ACM*, Vol. 36, No. 12, pp 36-53, 1993.

[CHA91] B. CHARRON-BOST. "Concerning the Size of Logical Clocks in Distributed Systems," *Inf Process Lett* 39, pp. 11-16 (1991).

[FAY85] G. FAYOLLE, P. FLAJOLET, P. JACQUET, "Analysis of a Stack Algorithm for Random Multiple Access Communication," Special Issue on Random-Access Communication, IEEE Transactions on Information Theory IT-31, pp. 244-254, 1985.

[FID88] C.A. FIDGE. "Timestamps in Message-Passing Systems that Preserve Partial Ordering," *Australian Computer Science Communications*, Vol 10, no. 1, pp. 56-66, 1988.

[HAC96] Dictionnaire HACHETTE Encyclopédique Illustré, ISBN 2.01.28.0476.4, 1996

[JAC89] P. JACQUET, "Contribution de l'Analyse d'Algorithmes a l'Évaluation de Protocolos de Communication," PhD thesis of l'Université de Paris-Sud Centre d'Orsay, Nov. 1989.

[HEL00] J. HELARY, G. MELIDEO, M RAYNAL. "Tracking Causality in Distributed Systems: a Suite of Efficient Protocols," in Proc. 7th Int'l Colloquium on Structural Information and Communication on Complexity, *Carleton Scientific*, June 2000.

[KSH96] A.D. KSHEMKALYANI, M. SINGHAL. "An Optimal Algorithm for Generalized Causal Message Ordering," Proc. 15th Annual ACM Symposium on Principles of Distributed Computing (PODC '96), pp. 87-87, ACM, May 1996.

[KSH98B] A.D. KSHEMKALYANI, M. SINGHAL. "Necessary and Sufficient Conditions on Information for Causal Message Ordering and their Optimal Implementation," *Distributed Computing* 11, pp. 91-111, 1998.

[LAM78] L. LAMPORT. "Time, Clocks and the Ordering of Events in Distributed Systems," *Communications ACM* 21(7), pp. 558-565, 1978.

[MAT89] F. MATTERN. "Virtual Time and Global States of Distributed Systems," *Parallel and Distributed Algorithms*, North-Holland, pp. 215-226, 1989.

[MOS93] A. MOSTEFAOUI, M. RAYNAL. "Causal Multicast in Overlapping Groups: Towards a Low Cost Approach," IEEE Workshop on Future Trends of Distributed Computer Systems, pp 136-142, Sept.1993.

[PET89] L. PETERSON, N. BUCHHOLZ, R. SCHLICHTING. "Preserving and Using Context Information in Interprocess Communication," *ACM Transaction on Computer Systems*, 7, pp. 217-246, 1989.

[POM02A] S. POMARES HERNANDEZ, K. DRIRA, J. FANCHON, M. DIAZ, “An Efficient Multi-Channel Distributed Coordination Protocol for Collaboration Engineering Activities,” IEEE Int. Conf. on Systems, Man and Cybernetics (SMC’2002), Hammamet (Tunisia), October 6-9, 2002.

[POM02B] S. POMARES HERNANDEZ, “Services De Coordination Et Protocoles De Diffusion Causale Pour Les Applications Coopératives Distribuées”, PhD thesis of National Institute Politechnique of Toulouse, France, Nov., 2002

[PRA95] R. PRAKASH, M. RAYNAL, M. SINGHAL, “ An effient causal ordering algorithm for mobile computing environments”, Technical report, Parallel Architectures Dept., Inst. Nat. de Rech. En Inf. et en Aut., France, 1995.

[PRA97] R. PRAKASH, M. RAYNAL, M. SINGHAL. “An Adaptive Causal Ordering Algorithm Suited to Mobile Computing Environments,” Journal of Parallel and Distributed Computing, pp 190-204, Mar. 1997.

[RAV92] Ravindran K., Prasad b., “Communication Structures and paradigms for distributed conferencing applications”, 12th IEEE Int. Conf. On Distributed Computing Systems, May 1992.

[ROD95] L. RODRIGUES, P. VERISSIMO. “Causal Separators and Topological Timestamping: an Approach to Support Causal Multicast in Large-Scale Systems,” Proc. 15th Int’l Conference on Distributed Computing Systems, Vancouver, British Columbia, Canada, May 1995.

[SCH90] F. Schneider, “Implementing fault-tolerant services unsing the state machine spproach: A tutorial”, ACM Compt. Surveys, vol 22, No. 4, pp 299-319, Dec. 1990 .

[SCH94] R. SCHWARZ, F. MATTERN. “Detecting Causal Relationships in Distributed Computations: in Search of the Holy Grail,” *Distributed Computing* 7, pp. 149-174, 1994.

[TOR99] F.J. TORRES-ROJAS, M. AHAMAD. “Plausible Clocks: Constant Size Logical Clocks for Distributed Systems,” *Distributed Computing*, 12(4), pp. 179-196, 1999.