Causally Ordered Group Communication for Fault-Tolerance *

5Aa−2

Kenji Shima and Makoto Takizawa †
Tokyo Denki University ‡
e-mail{sima,taki}@takilab.k.dendai.ac.jp

1 Introduction

In distributed systems executing such applications as teleconferences and telemedicines, a group of multiple application processes communicate with each other. The application requires the processes in the group to receive reliably the messages in the causal order [4].

There are kinds of distributed systems. The first one is composed of two types of processes, i.e. clients and servers. There is another kind of distributed systems i.e. processes which autonomously compute and communicate with other processes. In these systems, a group of multiple autonomous processes are cooperated to achieve some objects. Here, it is required to achieve the *intra-group* communication [4] where the processes communicate with each other in the group.

The processes in the distributed system may suffer from kinds of faults. An approach towards making the system fault-tolerant is to replicate the processes in the system. In this paper, in order to support the fault-tolerant group communication, each process is replicated into a collection of multiple *replicas*, which is named a *cluster*. Our protocol supports the inter-cluster communication among the replicas in the clusters in order to tolerate the Byzantine faults of processes in the group. In section 2, we discuss the replication schemes. In section 3, we present the properties of the intra-group communication. In section 4, we discuss the inter-cluster communication in the group In section 5, we discuss how to support the causally ordered and fault-tolerant delivery of messages in the group.

2 Replication Schemes

We would like to consider how to replicate a process p_i into replicas $p_{i1}, \ldots, p_{il_i}$ $(l_i \geq 1)$. There are two kinds of approaches towards replicating p_i [1, 5]:

(1) *state-machine approach*, and
(2) *primary-backup approach.*

In the state-machine approach (*active replication*) [5], every replica p_{ij} is modeled as a deterministic finite state machine. That is, every p_{ij} does the same computation by receiving and sending the same messages $(j = 1, \ldots, l_i)$.

In the primary-backup approach (*passive replication*) [1], there is one *primary* replica p_{i1}. The other replicas $p_{i2}, \ldots, p_{il_i}$ are named *backup* ones. p_{i1} receives and sends messages and computes while no backup replica computes.

The state-machine approach implies more redundant processing and communication than the primary backup one because all the replicas do the same computation by sending and receiving the same messages. However, it requires less time-overhead for recovering from faults, and the computation can be taken over by the other replicas immediately if some replica is faulty. Moreover, the replicated processes might tolerate the Byzantine faults. Therefore, we would like to adopt the state-machine approach in the rest of paper.

3 Intra-Group Communication

A distributed application program is executed by the cooperation of multiple processes referred to as *group* $G = \{p_1, \ldots, p_n\}$ $(n \geq 2)$ communicating with each other by using the communication system. A state of processes is transited when an event occurs. There are three kinds of events: *sending*, *receipt*, and local events. Here, let $s_i(m)$ and $r_i(m)$ denote sending and receipt events of a message m in p_i, respectively.

Lamport [2] defines the *happened-before* relation $\rightarrow$ on the events as follows:

[Definition] For every pair of events e_1 and e_2, e_1 *precedes* e_2 $(e_1 \rightarrow e_2)$ iff

(1) e_1 happens before e_2 in p_i,
(2) $e_1 = s_i(m)$ and $e_2 = r_j(m)$, or
(3) for some event e_3, $e_1 \rightarrow e_3 \rightarrow e_2$. □

A causal precedence relation $\prec$ among messages and causally ordered delivery is defined as follows:

[Causal precedence] For every pair of messages m_1 and m_2, m_1 *causally precedes* m_2 $(m_1 \prec m_2)$ iff $s_i(m_1) \rightarrow s_j(m_2)$. □

[Causally ordered delivery] The communication system supports the causally ordered delivery of messages iff for every pair of messages m_1 and m_2, m_1 is delivered before m_2 if $m_1 \prec m_2$. □

The reliable delivery and the fault-tolerant delivery is defined as follows:

[Reliable delivery] The communication system *reliably* delivers a message m iff all the destination processes of m receive m. □

[Fault-tolerant delivery] The communication system *fault-tolerantly* delivers a message m if m is reliably delivered and every destination of m receives m in some *bounded* time after m is sent. □

We would like to discuss how to support the fault-tolerant delivery of messages in the presence of the process faults.

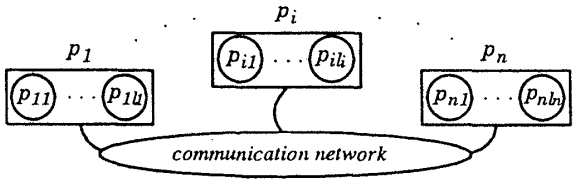

Figure 1: Group

4 Inter-Cluster Communication

In order to tolerate the process faults in the group, the processes are replicated to multiple replicas in the state-machine scheme. For a logical group composed of n processes $\langle p_1, \ldots, p_n \rangle$, a *group* G is composed of n clusters $c(p_1), \ldots, c(p_n)$. Each cluster $c(p_i)$ is a

*フォールトトレラント因果順序保存グループ通信
†島 健司、滝沢 誠
‡東京電機大学

collection of replicas $p_{i1}, \ldots, p_{il_i}$ ($l_i \geq 1$) of p_i ($i = 1, \ldots, n$) [Figure 1]. The replicas communicate with each other by using the communication system.

Some messages may be sent by faulty replicas in $c(p_i)$ to replicas in $c(p_j)$. The replica suffering from the Byzantine fault sends incorrect messages, no message or sends to incorrect destinations. The Byzantine fault of the replica can be detected by comparing the messages sent by the replica. Hence, the replicas in $c(p_j)$ have to receive messages from more than $2f_i$ replicas in $c(p_i)$ if at most f_i replicas are faulty.

5 Fault-Tolerant Ordered Delivery

5.1 Assumptions

We make the following assumption.

(C1) At most f_i ($\leq l_i$) replicas are faulty at the same time in each cluster $c(p_i)$.

(C2) The communication system is *reliable* and *synchronous* and satisfy FIFO.

(C4) Each message m has a unique identifier. This is realized by using a process identifier denoted by $m.src$ and a *sequence number* denoted by $m.sn$, given by a process.

(C5) The replicas cannot change the identifier of the message.

5.2 Fault-tolerant delivery

We would like to discuss how the clusters in G communicate with each other. There are four ways for the replicas in the cluster $c(p_i)$ to send m to $c(p_j)$ in G:

(B) each replica p_{ik} in $c(p_i)$ sends m to all replicas $p_{j1}, \ldots, p_{jl_j}$ in $c(p_j)$ for $k = 1, \ldots, l_i$,

(SB) each p_{ik} sends m to a subset $I_j(p_{ik}) \subseteq c(p_j)$ for $k = 1, \ldots, l_i$,

(MB) each replica p_{ik} in a subset $S(p_i) \subseteq c(p_i)$ sends m to all replicas in $c(p_j)$, and

(MSB) each replica p_{ik} in a subset $T(p_i) \subseteq c(p_i)$ sends m to a subset $K_j(p_{ik}) \subseteq c(p_j)$.

Figure 2 shows the number of messages transmitted by each replica in $c(p_i)$ for l_i. In the SB method, each replica sends the smallest number of messages. Therefore, the SB method is the best one.

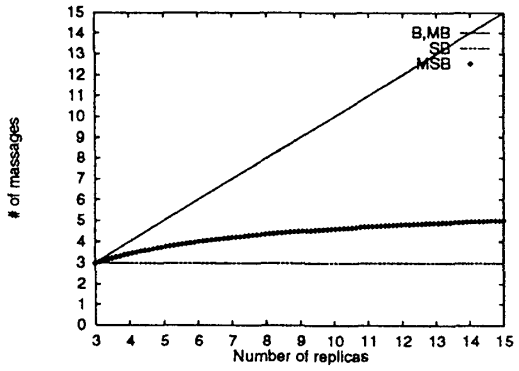

Figure 2: Number of messages sent by each replica

5.3 Ordered delivery

The replicas in a cluster $c(p_j)$ may receive messages in different orders due to the communication delay and the Byzantine fault of the replicas.

As presented in the papers [4], each message m sent by p_{ik} carries the confirmation field ack_{jh} which denotes the sequence number of the message which p_{ik} expects to receive next from p_{jh} ($h = 1, \ldots, l_i$, $j = 1, \ldots, n$). The messages can be causally ordered as follows:

[Causally oredering rule] For every pair of messages m_1 and m_2, m_1 *causally precedes* m_2 ($m_1 \prec m_2$) if

(1) $m_1.sn < m_2.sn$ if m_1 and m_2 are sent by the same cluster, and

(2) $m_1.sn < m_2.ack_{ik}$ if m_1 is sent by p_{ik} and m_2 is sent by the other replica. □

It is straightforward that the following property holds from the causally ordering rule.

[Proposition] On receipt of messages m_1 and m_2, each replica p_{ij} can decide whether $m_1 \prec m_2$, $m_2 \prec m_1$, or $m_1 || m_2$. □

It is noted that the faulty replica can change the confirmation fields ack in the messages. This means that the faulty replica may send the incorrect precedence information to other replicas.

Suppose that all the replicas in $c(p_i)$ send m_2 after receiving m_1. On receipt of m_2 from p_{ik} in $c(p_i)$, p_{jh} knows that $m_1 \prec m_2$ in p_{ik} (written as $m_1 \prec_{ik} m_2$).

[Replica perception] Each replica p_{jk} perceives that $m_1 \prec m_2$ if $| \{ p_{ih} \mid m_1 \prec_{ih} m_2 \} | \geq f_i + 1$. □

That, each replica in $c(p_j)$ decides "$m_1 \prec m_2$"if more than f_i replicas in $c(p_i)$ notify that m_1 causally precedes m_2, i.e. $m_1 \prec m_2$.

[Process perception] Each process p_j perceives that $m_1 \prec m_2$ if at least $f_j + 1$ operational replicas in $c(p_j)$ perceive that $m_1 \prec m_2$. □

[Theorem] The fault-tolerant delivery is satisfied by process perception rule. □

6 Concluding Remarks

This paper discusses how to support the fault-tolerant and causally ordered delivery of messages in the group in the presence of the Byzantine faults of processes. Therefore, we have discussed protocols for the inter-cluster communication in the group and shown the evaluation of them.

Reference

[1] Budhiraja, N., Marzullo, K., Schneider, B. F., and Toueg, S., "The Primary-Backup Approach," *Distributed Computing Systems, ACM Press*, 1994, pp.199-221.

[2] Lamport, L., "Time, Clocks, and the Ordering of Events in a Distributed System," *Comm. ACM*, Vol.21, No.7, 1978, pp.558–565.

[3] Lamport, L., Shostak, R., and Pease, M., "The Byzantine Generals Problem," *ACM Trans. Programming Languages and Systems*, Vol.4, No.3, 1982, pp.382–401.

[4] Nakamura, A. and Takizawa, M., "Causally Ordering Broadcast Protocol," *Proc. of the 14th IEEE ICDCS*, 1994, pp.48–55.

[5] Schneider, B. F., "Replication Management using the State-Machine Approach," *Distributed Computing Systems, ACM Press*, 1993, pp.169-197.