メッセージ通番を用いた因果関係順序付けグループ通信プロトコル

坂元 紫穂子† 中村 章人††
立川 敬行† 滝沢 誠†

分散型応用システムでは，複数のエンティティ間でのグループ通信が必要となる．グループ通信では，グループ内の全エンティティがメッセージを受信するという原子性と，各エンティティがどのような順序でメッセージを受信するかという順序性を提供する必要がある．また，フォールトトレラントシステムの実現のために，同じ事象は，各エンティティで同じ順序に起こる必要がある．このような事象間の順序関係を，因果関係という．本論文では，グループ内の全エンティティに対して，メッセージ送信の因果関係の順で，メッセージを受信させるグループ通信プロトコルについて論じる．本プロトコルは，バッファオーバランによりメッセージの紛失が起こり得る高速通信網を利用する．また，指揮エンティティの存在しない完全分散型の制御方式に基づいている．受信メッセージの因果関係による順序付けは，メッセージの通番を用いて行うために，メッセージの紛失を検出し，復旧できる．

Causally Ordered Group Communication Protocol Using Sequence Numbers of Messages

Shihoko Sakamoto,† Akihito Nakamura,†† Takayuki Tachikawa† and Makoto Takizawa†

The distributed applications like groupware require group communication among multiple entities. In the group communication, it is important to discuss in what order each entity in the group can receive messages. In order to realize fault-tolerant systems, the same events have to occur in the same order in each entity. The ordered relation among the events is known as a causal order. This paper presents a reliable causally ordered group communication (CO) protocol which provides the same causal ordering of messages for all the entities in the group. In the CO protocol, the messages received are causally ordered by using the sequence numbers of the messages. The CO protocol is based on the fully distributed control scheme, i.e. no master controller, and uses high-speed networks where each entity may fail to receive messages due to the buffer overrun.

1. はじめに

分散型応用システムでは，複数の計算機上の応用エンティティ間での協調動作が必要となる．こうした各応用エンティティは，複数の応用エンティティにデータを転送する必要がある．エンティティの集合をグループとして，グループ内のエンティティ間で行われる通信をグループ通信とする．グループ通信を提供するプロトコルをグループ通信プロトコルとする．グループ通信プロトコルは，全エンティティがメッセージを受理したときのみ受信されたとする原子性と，各エンティティが一定の順序でメッセージを受信するという順序性を保障するサービスを提供する必要がある[18),20),21)]．あるメッセージが原子性と順序性を満足するとき，そのメッセージはグループ内で正しく受信されたとする．

メッセージがグループ内で正しく受信されたことの判断を行うための制御方式として，集中型と分散型がある．集中型制御では，ある一つのエンティティ（指揮者）が，複数のエンティティ（関係者）でのメッセージの正しい受信の判断を行う．Amoeba[6)]は，シーケンサと呼ばれる指揮エンティティがグループでの受信の制御を行う集中型である．ISIS[2)]とDelta-4[23)]は，

† 東京電機大学理工学部経営工学科
Department of Computers and Systems Engineering, Faculty of Science and Engineering, Tokyo Denki University
†† 電子技術総合研究所
Electrotechnical Laboratory

各メッセージの送信者がグループでの受信の制御を行う非集中型である．これらの方式では，各関係者は指揮者の判断を待たねばならず，指揮者の障害が全体障害を引き起こす場合がある．一方，分散型制御[17),18)]では，指揮者と関係者の区別がなく，各エンティティは受信したメッセージに基づいて正しい受信の判断を行う．本論文で提案するグループ通信プロトコルは，完全分散型である．

グループ通信の順序付けサービスとして，送信順序保存 (LO)，因果関係保存 (CO)，全順序 (TO) の三種がある．LO サービスでは，各エンティティから送信されたメッセージは，送信順に受信される[14),19)]．TO サービス[3)~6),8),10),17),18)]では，LO サービスの性質に加え，全宛先で，全メッセージが同一の順序で受信される．CO サービスでは，各エンティティは，メッセージの送信事象の生起順序[7),15)]でメッセージを受信する．

メッセージ間に因果関係がある分散型応用システムでは，CO サービスが必要となる．例えば，電子会議システムにおいて，あるエンティティ A が質問を送信し，B が質問の受信後にその回答を送信する場合を考える．質問と回答には因果関係がある．ここで，エンティティ C が質問の受信に失敗し，回答の受信後に質問が再送されたとする．C は，回答の後に質問を受信することになる．CO サービスは，必ず全エンティティが質問の後に回答を受信することを保障するものである．これに対し，TO サービスでは，全エンティティがメッセージを同じ順序で受信すればよく，全エンティティが回答，質問の順に受信する場合がある．

本論文では，応用エンティティのグループに対して，CO サービスを提供する CO プロトコルのデータ転送手続きを示す．CO サービスを提供するプロトコルに，ISIS の CBCAST[3)] がある．ISIS が高信頼な一対一型通信網を想定しているのに対し，本 CO プロトコルはメッセージの紛失が起こり得る多チャネル (MC) 放送型高速網[1)]を用いる．MC 網では，異なったエンティティからのメッセージは，各エンティティで同じ順序で受信できるとは限らない．このために，因果関係は保障されない．受信メッセージの因果順序付けには，メッセージの通番を用いる．ISIS 等で利用されているベクトル時刻[9)]では，メッセージの紛失を検出できない．しかし，本論文で示す方法では，メッセージの紛失を検出できるとともに，因果順序付けることができるものである．

本論文の 2 章では，グループ通信サービスのモデルを示す．3 章では，CO サービスを提供するためのデータ転送手続きを述べ，4 章では，評価を示す．

2. モデル

2.1 システムモデル

通信システムは，**図 1** に示す 3 階層から構成される．システム層のエンティティは，網層が提供するサービスを利用して，応用層に高信頼なグループ通信を提供する．システムエンティティ $E_1, \cdots, E_n$ の集合を群 $C=\langle E_1, \cdots, E_n \rangle$ とし，各 E_i は，網サービスを利用し，システムサービスアクセス点 (SAP) S_i を通じて応用エンティティ A_i にサービスを提供する ($i=1, \cdots, n$)．すなわち，$A_1, \cdots, A_n$ のグループが C を利用してグループ通信を行う．このとき，C は，$E_1, \cdots, E_n$ によって提供されるとする．高速網[1)]では，転送速度がエンティティの処理速度よりも速いために，バッファのオーバーランにより，エンティティがメッセージの受信に失敗する場合がある．

2.2 順序性

ある層の各エンティティ T_i が利用する下位層のサービスを，メッセージの系列であるログの集合[16)~18)]としてモデル化する．ここで，m 個のメッセージから成るログ L を $\langle p_1 \cdots p_m]$ と書き，$top(L)$ は先頭の p_1，$last(L)$ は最後尾の p_m を示すとする．各 T_i は，送信および受信したメッセージの履歴である送信ログ SL_i と受信ログ RL_i を持つ ($i=1, \cdots, n$)．以下，記号 p, q, r はメッセージを示すとする．

$s_i[p]$ と $r_i[p]$ は，各々 T_i における p の送信と受信の事象を表す．各 $r_i[p]$ には，p の送信事象が存在すると仮定する．事象間の先行関係→を以下のように定義する[7)]．

[定義] 任意の事象 e_1 と e_2 間で以下が成り立つならば，$e_1 \rightarrow e_2$ である．

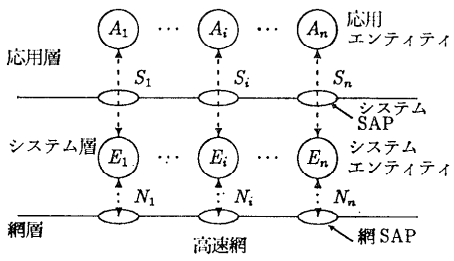


図 1 システムモデル
Fig. 1 System model.

（1）E_i において，e_1 が e_2 に先行する，

（2）ある E_i と E_j について，$e_1=s_i[p]$ かつ $e_2=r_j[p]$，または，

（3）$e_1 \rightarrow e_3$ かつ $e_3 \rightarrow e_2$ である事象 e_3 が存在する．

□

RL_i 内の p と q について，$r_i[p] \rightarrow r_i[q]$ ならば，p が q に先行する（$p \Rightarrow_{RL_i} q$）．SL_i 内の p と q について，$s_i[p] \rightarrow s_i[q]$ ならば，p が q に先行する（$p \Rightarrow_{SL_i} q$）．

[定義]

（1）RL_i が，$SL_1, \cdots, SL_n$ に含まれるすべてのメッセージを含んでいるとき，RL_i は情報保存である．

（2）E_i が各 E_j から受信した任意の p と q について，$p \Rightarrow_{SL_j} q$ ならば，$p \Rightarrow_{RL_i} q$ であるとき，RL_i は順序保存である．□

RL_i が情報保存ならば，T_i は各 T_j が送信した全メッセージを受信している．RL_i が順序保存ならば，T_i は，各 T_j ごとに送信順にメッセージを受信している．

次に，メッセージ間の因果関係[3]を定義する．

[定義] 任意の p と q について，$s_i[p] \rightarrow s_j[q]$ ならば，p は q に因果先行する（$p \prec q$）．□

$p \prec q$ は，q 以前に p が送信されたことを表す．$p \not\prec q$ でも $q \not\prec p$ でもないならば，p と q に先行関係はない（$p \| q$）．$p \preceq q$ は，$p \prec q$ または $p \| q$ を示す．

[定義] RL_i 内のすべての p と q について，$p \not\prec q$ の場合，$p \Rightarrow_{RL_i} q$ ならば，RL_i は因果関係保存である．□

因果関係保存であるとき，RL_i は因果関係順序付けされているとする．**図2**で，$RL_k=\langle gpq]$ は因果関係保存である．E_h が p の受信後に q を送信し，E_k は q より先に p を受信している．一方，E_k が p より先に q を受信したならば，$RL_k'=\langle gqp]$ となる．これは，順序保存ではあるが，因果関係保存ではない．

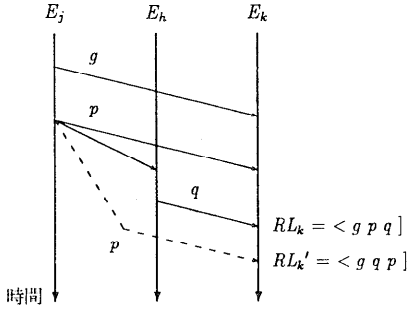


図 2 因果関係を保存した受信順序
Fig. 2 Causality-preserving receipt.

2.3 グループ通信サービス

[定義]

（1）多チャネル（MC）サービスとは，各 RL_i が順序保存であるサービスである．

（2）因果関係保存（CO）サービスとは，各 RL_i が情報保存かつ因果関係保存であるサービスである．□

MC サービスでは，各送信エンティティごとにメッセージを送信順に受信できるが，メッセージが紛失する場合があり，各受信ログは情報保存とは限らない．MC サービスは，計算機間を複数の高速通信チャネルで接続したシステムで提供される．一方，CO サービスを用いることにより，因果先行順 $\prec$ にメッセージを受信できる．本論文では，MC サービスを用いて，CO サービスを提供するプロトコルを考える．

2.4 分散型制御

群 $C=\langle E_1, \cdots, E_n\rangle$ 内での分散型制御における正しい受信の概念を示す．各 E_j が送信するメッセージには，E_j がそれまでに受信したメッセージの受信通知が含まれているとする．ここで，p と q は各々 E_k と E_j により送信されたメッセージとする．$s_k[p] \rightarrow r_i[p]$ かつ $s_k[p] \rightarrow r_j[p] \rightarrow s_j[q] \rightarrow r_i[q]$ ならば，q は，E_i で E_j について p を前確認するとする．

C 内で送信されたメッセージ p が，全エンティティで受信されたかどうかを，各 E_i が判断する基準として，以下の（1）～（3）の3段階がある．

（1）受理：E_i が p を受信する．

（2）前確認：E_i が，全エンティティが p を受理したことを知る．

（3）確認：E_i が，全エンティティが p を前確認したことを知る．

（2）で，E_i は E_j からのメッセージ q を受信することにより E_j が p 受理したことを知る．しかし，E_i は「他のエンティティ E_k が全エンティティで p が受信されたことを知っている」かはわからない．（3）では，全エンティティでこのことがわかる．ここで，全エンティティが p を原子的に受信したことになる．

3. 因果関係順序付けグループ通信プロトコル

本章では，MC サービスを利用し，群 $C=\langle E_1, \cdots, E_n\rangle$ に対して，CO サービスを提供する因果関係順序

付けグループ通信（CO）プロトコルを示す．

3.1 変　数

E_i により送信されたメッセージ p は以下の項目を持つ（**図 3**）．$p.F$ は，p 内の項目 F を示す．

- $p.CID$＝群 C の識別子．
- $p.SRC$＝送信エンティティ（E_i）．
- $p.SEQ$＝p の通番．
- $p.ACK_j$＝E_i が E_j から次に受信予定のメッセージの通番（$j=1,\cdots,n$）．
- $p.BUF$＝E_i が利用可能なバッファ数．

ここで，$p.ACK_j$ は，E_i が，$q.SEQ<p.ACK_j$ であるメッセージ q を，E_j から既に受信していることを示す．$p.ACK$ は $\langle p.ACK_1,\cdots,p.ACK_n\rangle$ を表す．各メッセージは，MC サービスを用いて，C 内の全エンティティに送信される．

各 E_i は，以下の変数を持つ（$j=1,\cdots,n$）．

- SEQ＝次に送信予定のメッセージの通番．
- REQ_j＝E_j から次に受信予定のメッセージの通番．
- AL_{kj}＝「E_j が，E_k から次に受信予定である」と E_i が知っているメッセージの通番（$k=1,\cdots,n$）．
- PAL_{kj}＝「E_j が，E_k からのメッセージで，次に前確認予定である」と E_i が知っているメッセージの通番（$k=1,\cdots,n$）．
- BUF_j＝E_j 内の利用可能なバッファ数．

ここで，$minAL_k$ と $minPAL_k$ は各々，$AL_{k1},\cdots,AL_{kn}$ と $PAL_{k1},\cdots,PAL_{kn}$ 内の最小値を示す（$k=1,\cdots,n$）．

本論文では，群 C は，群確立手続き[17),18)]により確立されているとする．群確立手続きにより，各 E_i は群内の全エンティティの SEQ と BUF の初期値を知るとともに，群 C の識別子 CID が定まる．

3.2 送 受 信

E_i は，システム SAP S_i から，応用エンティティ A_i のデータ送信要求を受け，以下のフロー条件が満足されるならば，送信手続きに従って，メッセージを送信する．ここで，W は最大ウィンドウ幅，H は最大メッセージ長である．各 E_i は，少なくとも $O(n)$ 個のメッセージを記憶できるだけのバッファを持たねばならない[11),15)]．

CID	SRC	SEQ	$ACK=\langle ACK_1,\cdots,ACK_n\rangle$	BUF	DATA

図 3 メッセージの形式
Fig. 3 Message format.

[フロー条件] $minAL_i \le SEQ < minAL_i + min(W, minBUF/(H\times n))$．□

ここで，$enqueue(L,p)$ は，p をログ L の最後尾に追加する操作を表す．$broadcast(p)$ は，MC サービスにより，p を放送する操作である．

[送信手続き] {
$p.SEQ := SEQ$; $SEQ := SEQ+1$;
$p.ACK_j := REQ_j (j=1,\cdots,n)$;
$p.BUF :=$ E_i 内の使用可能なバッファサイズ;
$enqueue(SL_i,p)$; $broadcast(p)$;} □

p を，E_j が送信したメッセージとする．E_i が p を受信したとき，受理条件を満足するならば，受理手続きに従って，p は受理される．E_i は，E_j から受理したメッセージの受信ログ RRL_{ij} を持つ．

[受理条件] $p.SEQ=REQ_j$（ここで，$p.SRC=E_j$）．□

[受理手続き] {
$REQ_j := p.SEQ+1$;
$AL_{kj} := p.ACK_k (k=1,\cdots,n)$;
$BUF_j := p.BUF$; $enqueue(RRL_{ij},p)$;} □

3.3 障害の検出と復旧

MC サービスでは，エンティティがメッセージを紛失する場合がある．g を E_j が送信したメッセージとする．各 E_i は，以下の障害条件によって，g の紛失を検出する．

[障害条件]

（1）E_j から p を受信したとき，$REQ_j<p.SEQ$ ならば，$REQ_j\le g.SEQ<p.SEQ$ である g を紛失している．

（2）E_k から q を受信したとき，各 $j(\neq k)$ について $REQ_j<q.ACK_j$ ならば，$REQ_j\le g.SEQ<q.ACK_j$ で $g.SRC=E_j$ なるメッセージ g を紛失している．□

障害条件により，E_j からの g の紛失を検出したならば，E_i は，E_j に g の再送を要求する．

[再送手続き]

（1）E_j からの g の紛失を検出したならば，E_i は再送要求（RET）メッセージ r を E_j に送信する．ここで，$r.ACK_k := REQ_k (k=1,\cdots,n)$，$r.LSRC=E_j$，$r.LSEQ=p.SEQ$（障害条件（1）），または，$r.LSEQ=q.ACK_j$（障害条件（2））．

（2）E_j が E_i から r を受信したならば，E_j は $r.ACK_j<g.SEQ\le r.LSEQ$ である g を送信

する．□

図4（a）では，E_i が p を受信したとき，$REQ_j(=4)<p.SEQ(=5)$（障害条件（1））であるので，g の紛失を検出する．一方，図4（b）では，E_i は E_k からの q を受信したとき，$REQ_j(=4)<q.ACK_j(=5)$（障害条件（2））であるので，g の紛失を検出する．いずれの場合にも，E_i は E_j に $ACK_j=REQ_j(=4)$，$LSRC=E_j$，$LSEQ=5$ である RET メッセージを送信し，g の再送を要求する．メッセージ g を送信した E_j も，g の受信通知を一定時間内に全エンティティから受信しない場合には，g の再送を行う．

3.4 前確認手続き

p を，E_j が送信したメッセージとする．E_i が，$p.SEQ<q.ACK_j$ であるメッセージ q を E_k から受信することにより，E_k で p が受理されていることがわかる．各 AL_{jk} は，「$p.SEQ<AL_{jk}$ なる p は，既に E_k で受理されている」ことを示す $(k=1,\cdots,n)$．従って，$p.SEQ<minAL_j$ ならば，群内の全エンティティが既に p を受理していることがわかる．

[前確認条件]

$p.SEQ<minAL_j$（ただし，$p.SRC=E_j$）．□

前確認された，すなわち全エンティティで受信されたメッセージを，因果先行関係 $\prec$ により順序付ける方法について考える．

[定理 1] p を E_j から送信されたメッセージとする．また，q をメッセージとする．

（1）$p.SRC=q.SRC$ のとき，$p.SEQ<q.SEQ$ ならばかつそのときに限り，$p\prec q$ である．

（2）$p.SRC\neq q.SRC$ のとき，$p.SEQ<q.ACK_j$ ならばかつそのときに限り，$p\prec q$ である．

[証明]

（1）$p.SRC-q.SRC$ の場合を考える．まず，$p.SEQ<q.SEQ$ とする．$s_j[p]\rightarrow s_j[q]$ であるので，$p\prec q$ である．次に，$p\prec q$ と仮定する．データ転送手続きより，$p.SEQ<q.SEQ$ である．

（2）図2のように，E_j が E_h と E_k に p を送信したとする．ここで，E_h の p を受信後に q を送信し，E_k が p の後に q を受信する場合を考える．まず，$p\prec q$ と仮定する．$p\prec q$ より，$r_h[p]\rightarrow s_h[q]$ である．p を受理したとき，受理手続きより，E_h は REQ_j を一つ加算する．すなわち，$p.SEQ<REQ_j$．よって，$REQ_j\leq q.ACK_j$ より，$p.SEQ<q.ACK_j$ である．次に，$p.SEQ<q.ACK_j$ と仮定する．データ転送手続きより，E_h は p の受信前に q を送信する．つまり，$s_j[p]\rightarrow s_h[q]$．よって，$p\prec q$ である．□

本定理から，p と q の通番の比較により，$p\prec q$ かどうかを決定できる．

[定理 2] E_i が p と q を受信し，p は E_j により送信されたとする．$p\prec q$ で，各エンティティが $t\prec q$ である全メッセージ t を受信しているならば，E_i で，

（1）$p.SRC=q.SRC$ ならば，$p.ACK_h\leq p.ACK_h$ $(h=1,\cdots,n)$ であり，

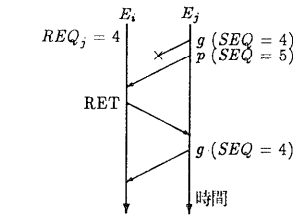
（2）$p.SRC\neq q.SRC$ ならば，$p.ACK_j<q.ACK_j$ かつ $p.ACK_h\leq q.ACK_h(h=1,\cdots,n,h\neq j)$ である．

[証明]

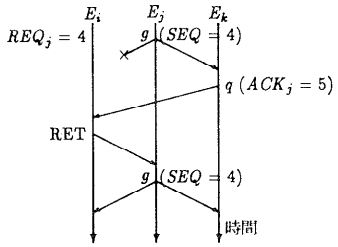
（1）$p.SRC=q.SRC$ とする．$p\prec q$ なので，$p\rightarrow_{SL_j} q$ である．明らかに，$p.ACK_h\leq q.ACK_h(h=1,\cdots,n)$ である．

（2）次に，$p.SRC\neq q.SRC(=E_k)$ とする．E_j が p を送信しているので，$p.ACK_j\leq p.SEQ$．定理1より $p.SEQ<q.ACK_j$．よって，$p.ACK_j<q.ACK_j$ である．

次に，ある $h(\neq j)$ について，$p.ACK_h>q.ACK_h$ の場合を考える．これは，メッセージは全エンティティで受信されているので，E_j



（a）障害条件（1）
（a）Failure condition (1)



（b）障害条件（2）
（b）Failure condition (2)

図4 障害の検出と復旧
Fig. 4 Failure detection and recovery.

が E_h から $a.SEQ=p.ACK_h-1$ であるメッセージ a を受信し，E_k が E_h から $b.SEQ=q.ACK_h-1$ であるようなメッセージ b を受信したことを示す（**図 5**）．$p.ACK_h>q.ACK_h$ の仮定より，$a.SEQ>b.SEQ$，すなわち，$b\prec a$．E_k が a の受信前に q を送信しているので，$b\prec q\prec a$ となる．一方，E_j では，a の受信後に p を送信しているので，$a\prec p$ となる．すなわち，$q\prec a\prec p$ となり，$p\prec q$ と矛盾する．よって，$p.ACK_h\leq q.ACK_h$．□

定理 1 が成立し，定理 2 の（1）または（2）が成り立たなければ，あるメッセージ t を紛失していることになる．

因果関係を保存するように，p をログ L に挿入する因果保存挿入（CPI）操作 $L\lhd p$ を以下に定義する．

[CPI 操作 $L\lhd p$] $L=<q_1,\cdots,q_m]$ は，因果関係保存とする．このとき，$L\lhd p$ を以下のように定義する．

（1）$m=0$ ならば，$L\lhd p=\langle p]$.

（2）$p\prec q_1$ ならば，$L\lhd p=\langle pq_1\cdots q_m]$.

（3）$q_m\preceq p$ ならば，$L\lhd p=\langle q_1\cdots q_m p]$.

（4）$q_i\preceq p\prec q_{i+1}(1\leq i\leq m-1)$ ならば，$L\lhd p=\langle q_1\cdots q_i p q_{i+1}\cdots q_m]$．□

各エンティティから受信したメッセージが，全宛先で受信されたことを確認した後に，メッセージを因果関係順に整列させる．各 E_i は，前確認されたメッセージから成り，因果関係保存した受信ログ PRL_i をもつ．p が RRL_{ij} 内で前確認されたならば，以下の前確認手続きによって，PRL_i に移される．$dequeue(L)$ は，p をログ L から取り除く操作である．

[前確認手続き]

For($j=1,\cdots,n$){

while($p=top(RRL_{ij})$ が前確認条件を充足する)

{

$p:=dequeue(RRL_{ij})$;

$PAL_{kj}:=p.ACK_k(k=1,\cdots,n)$;

$PRL_i\lhd p$;}} □

[例 1] 群 $C=\langle E_1,E_2,E_3\rangle$ を考える（**図 6**）．各 E_i の REQ_i の初期値を 1 とする．**表 1** に，図 6 内の各メッセージの項目 SEQ と $\langle ACK_1,ACK_2,ACK_3\rangle$ の値の変化を示す．E_2 は d を送信する前に，E_1 から c を受理（$c.SEQ=2$）し，E_3 から b を受理（$b.SEQ=1$）しているからである．d を受理したとき，$AL_{i2}:=d.ACK_i(i=1,2,3)$ となり，REQ と AL は以下のようになる．

$$REQ=\langle 3,2,2\rangle,\quad AL=\begin{bmatrix}2&3&2\\1&1&1\\1&2&1\end{bmatrix}.$$

$a.SEQ(=1)<minAL_1(=min\{AL_{11},AL_{12},AL_{13}\}=2)$ より，各 E_i で a が前確認される．これにより，各 E_i の PRL_i と RRL_{ij} は，図 6（a）となる．h が受理されたときの REQ と AL を以下に示す．

表 1 SEQ と ACK の値
Table 1 SEQ and ACK fields.

メッセージ	SEQ	ACK
a	1	$\langle 1,1,1\rangle$
b	1	$\langle 2,1,1\rangle$
c	2	$\langle 2,1,1\rangle$
d	1	$\langle 3,1,2\rangle$
e	3	$\langle 3,2,2\rangle$
f	4	$\langle 4,2,2\rangle$
g	2	$\langle 4,2,2\rangle$
h	2	$\langle 5,3,2\rangle$

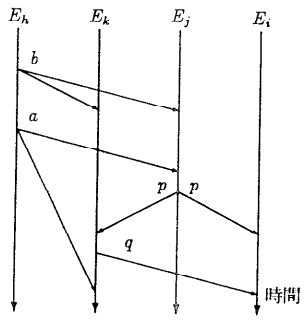


図 5 受信順序
Fig. 5 Receipt order.

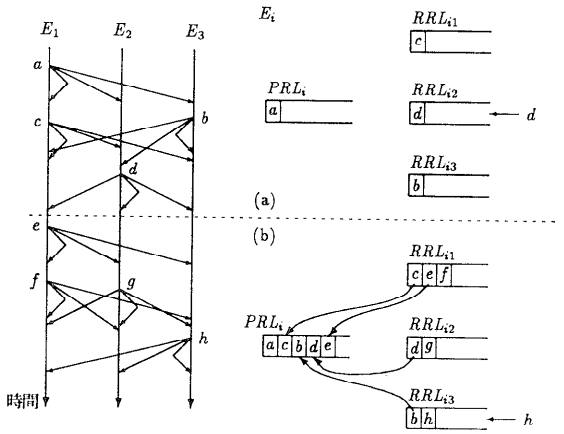


図 6 E_i での前確認と CPI 操作
Fig. 6 Pre-acknowledgment and CPI operation in E_i.

$$REQ=\langle 5,3,3\rangle,\quad AL=\begin{bmatrix}4&4&5\\2&2&3\\2&2&2\end{bmatrix}.$$

ここで，b,c,d,e は前確認され，PRL_i に移される．まず，c と e が，PRL_i の末尾に CPI 操作により追加される．a,c,e は，この順で E_1 から送信されるからである．次に，d が PRL_i に移される．$c.SEQ<d.ACK_1$ であり，定理1より，$c\prec d\prec e$ である．$d.SEQ<e.ACK_2$ であるので，$c\preceq b\preceq d$ より，b が c と d の間に挿入される．□

[補題 3] p と q をそれぞれ，$p.SRC=E_j$, $q.SRC=E_k$ であり，$p\prec q$ であるメッセージとする．q が E_i で前確認されたならば，p も E_i で前確認される．

[証明] E_i で q が前確認され，p が前確認されないと仮定する．q が前確認されることから，E_i は，$q.SEQ<g_h.ACK_k$ である g_h を各 E_k から受信している．つまり，E_i は，$s_i[p]\rightarrow s_j[g_j]$ である g_j を受理する．E_i が p の受信に失敗したとしても，g_j を受信したとき，障害条件から p の紛失を検出できる．よって，g_j は受理されず，q は前確認されない．E_i が p を受信したならば，p は前確認される．仮定と矛盾する．□

これは，E_i 内で因果関係 $\prec$ の順にメッセージが前確認されることを意味する．

3.5 確認手続き

E_i が，どのようにメッセージを確認するかについて考える．E_k からの q が前確認されたとき，各 $q.ACK_j$ は，PAL_{jk} に格納される $(j=1,\cdots,n)$．E_i は，E_k が $p.SEQ<PAL_{jk}$ である E_j からの p を前確認したことがわかる．このように，$minPAL_j$ は，$p.SEQ<minPAL_j$ である p が全エンティティで前確認されたことを意味する．よって，PRL_i 内の p が，以下の確認条件を満足すれば，p は E_i で確認され，ARL_i に移される．応用エンティティ A_i は，ARL_i の先頭から因果関係順にメッセージを取り出せる．

[確認条件] $p.SEQ<minPAL_j$（ここで，$p.SEQ=E_j$）．□

[確認手続き]

While($p(=top(PRL_i))$ が確認条件を充足する) {

　$p:=dequeue(PRL_i)$; $enqueue(ARL_i,p)$;} □

[例 2] 例1で，E_3 からの h が前確認されたならば，$PAL_{i3}:=h.ACK_i(i=1,2,3)$ となり，PAL は以下のようになる．

$$APL=\begin{bmatrix}4&4&5\\2&2&3\\2&2&2\end{bmatrix}.$$

ここで，$a.SEQ<c.SEQ<e.SEQ<minPAL_1(=4)$, $d.SEQ<minPAL_2(=2)$, $b.SEQ<minPAL_3(=2)$ より，a,b,c,d,e が確認される．$a\prec b\preceq c\prec d\prec e$（ただし，$b\|c$）である．□

[補題 4] p と q を，E_i で前確認され，$p\prec q$ であるメッセージとする．このとき，q が確認されるならば，p も確認される．

[証明] $p.SRC=E_j$ かつ $q.SRC=E_k$ の場合を考える．補題3から，$p\prec q$ より，PRL_i 内で p は q に先行する（$p\rightarrow_{PRL_i}q$）．よって，確認手続きは，q より先に p に適用され，PRL_i は因果関係保存である．すなわち，$p\rightarrow_{ARL_i}q$．□

[定理 5] CO プロトコルは，MC サービスを利用して CO サービスを提供する．

[証明] メッセージ紛失がないならば，補題3と4より明らかである．あるメッセージ g が紛失したならば，障害条件により，g の紛失は検出され，再送手続きにより再送される．補題3より，$p\prec q$ であり，p が前確認されていないならば，q は前確認されない．補題4より，q は p の後に確認される．□

4. 評　価

本章では，群 $C=\langle E_1,\cdots,E_n\rangle$ に対する CO プロトコルの性能を考える．各 E_i が，メッセージを受信するごとに，メッセージの送信を行うと，C 内で送信されるメッセージ数が $O(n^2)$ となる．このために，各 E_j から少なくとも一つのメッセージを受信するか，または，メッセージを送信してから一定時間が経過した時に，メッセージを送信する．W をウィンドウ幅とする．メッセージ p は，p が受信されてから $2nW$ 個のメッセージを受信すると確認される．nW が前確認，さらに nW が確認のためである．よって，必要なバッファサイズは $O(n)$ である．図3に示すように，各メッセージは，n 個の受信通知 ACK を含むので，ヘッダ長は $O(n)$ である．

R を最も離れたエンティティ間の伝播遅延時間とする．p の受信通知を含むメッセージが各エンティティから並行して送信されるならば，p が受理されてから前確認されるまでの時間は R である．よって，p が受理されてから確認されるまでの時間は $2R$ である．

CO プロトコルは，SPARC 2 上の UNIX ユーザプ

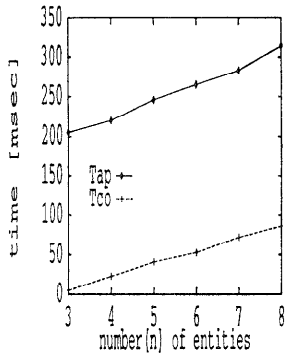


図 7 処理時間と遅延時間
Fig. 7 Processing time and delay time.

ロセスとして実装されている．Ethernet を MC 網として用いている．各ワークステーションごとに，CO エンティティと応用エンティティを一つずつ実装している．プログラムは C 言語で約 2500 文，実行形式の大きさは約 50 キロバイトである．メッセージ内のデータ長は 512 バイトである．ヘッダ長は 84 バイト ($n=8$) である．本評価では，各応用エンティティが一定間隔（100 ミリ秒）で連続してデータを送信している．**図 7** には，群 C 内のエンティティ数 n を変化させた場合の，各エンティティでメッセージ当りの処理時間 (T_{co}) と応用エンティティ間での遅延時間 (T_{ap}) を示す．この図は，各エンティティの処理負荷が $O(n)$ であることを示している．今後，スレッドを用いた実装等により，性能面の改良を行う必要がある．

5. ま と め

本論文では，高速な多チャネル（MC）網上の因果関係順序付けグループ通信（CO）プロトコルについて議論した．高速通信網では，各システムエンティティは，バッファのオーバランによりメッセージの受信ができない場合があり，再送により各エンティティ間で異なった順序でメッセージを受信する可能性がある．CO プロトコルを用いることにより，群内のすべての応用エンティティは因果関係順にメッセージを受信できる．群内でメッセージを非同期に送受信しながら，メッセージの原子的受信と因果関係順序での受信の判断を，各エンティティが分散して自分自身で行う．CO プロトコルでは，メッセージの紛失を検出するために通番を用いる一方，メッセージを因果関係順序付けるためにも用いる．最後に，CO プロトコルの評価を行い，プロトコルの処理負荷が $O(n)$ であることを，実装を通して示した．

参 考 文 献

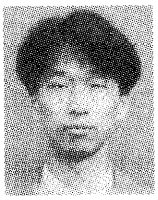
1) Abeysundara, B. W. and Kamal, A. E.: High-Speed Local Area Networks and Their Performance: A Survey, *ACM Comput. Surv.*, Vol. 23, No. 2, pp. 221-264 (1991).
2) Birman, K. P. and Joseph, T. A.: Reliable Communication in the Presence of Failures, *ACM Trans. Computer Systems*, Vol. 5, No. 1, pp. 47-76 (1987).
3) Birman, K. P., Schiper, A. and Stephenson, P.: Lightweight Causal and Atomic Group Multicast, *ACM Trans. Computer Systems*, Vol. 9, No. 3, pp. 272-314 (1991).
4) Chang, J. M. and Maxemchuk, N. F.: Reliable Broadcast Protocols, *ACM Trans. Computer Systems*, Vol. 2, No. 3, pp. 251-273 (1984).
5) Garcia-Molina, H. and Spauster, A.: Ordered and Reliable Multicast Communication, *ACM Trans. Computer Systems*, Vol. 9, No. 3, pp. 242-271 (1991).
6) Kaashoek, M. F. and Tanenbaum, A. S.: Group Communication in the Amoeba Distributed Operating System, *Proc. of IEEE ICDCS-11*, pp. 222-230 (1991).
7) Lamport, L.: Time, Clocks, and the Ordering of Events in a Distributed System, *Comm. ACM*, Vol. 21, No. 7, pp. 558-565 (1978).
8) Luan, S. W. and Gligor, V. D.: A Fault-Tolerant Protocol for Atomic Broadcast, *IEEE Trans. Parallel and Distributed Systems*, Vol. 1, No. 3, pp. 271-285 (1990).
9) Mattern, F.: Virtual Time and Global States of Distributed Systems, *Proc. of Parallel and Distributed Algorithms Conf.*, pp. 215-226 (1988).
10) Melliar-Smith, P. M., Moser, L. E. and Agrawala, V.: Broadcast Protocols for Distributed Systems, *IEEE Trans. Parallel and Distributed Systems*, Vol. 1, No. 1, pp. 17-25 (1990).
11) Nakamura, A. and Takizawa, M.: Reliable Broadcast Protocol for Selectively Ordering PDUs, *Proc. of IEEE ICDCS-11*, pp. 239-246 (1991).
12) Nakamura, A. and Takizawa, M.: Design of Reliable Broadcast Communication Protocol for Selectively Partially Ordered PDUs, *Proc. of the IEEE COMPSAC '91*, pp. 673-679 (1991).
13) Nakamura, A. and Takizawa, M.: Priority-Based Total and Semi-Total Ordering Broadcast Protocols, *Proc. of IEEE ICDCS-12*, pp. 178-185 (1992).
14) 中村章人，滝沢 誠：多チャネルシステム上の

送信順序保存放送通信プロトコル，情報処理学会論文誌，Vol. 34, No. 1, pp. 135-142 (1993).
15) Nakamura, A. and Takizawa, M.: Causally Ordering Broadcast Protocol, *Proc. of IEEE ICDCS-14*, pp. 48-55 (1994).
16) Raynal, M.: About Logical Clocks for Distributed Systems, *ACM Operating Systems Review*, Vol. 26, No. 1, pp. 41-48 (1992).
17) Takizawa, M.: Cluster Control Protocol for Highly Reliable Broadcast Communication, *Proc. of the IFIP Conf. on Distributed Processing*, pp. 431-445 (1987).
18) Takizawa, M.: Design of Highly Reliable Broadcast Communication Protocol, *Proc. of the IEEE COMPSAC '87*, pp. 731-740 (1987).
19) Takizawa, M. and Nakamura, A.: Partially Ordering Broadcast (PO) Protocol, *Proc. of the 9th IEEE INFOCOM 90*, pp. 357-364 (1990).
20) Takizawa, M. and Nakamura, A.: Reliable Broadcast Communication, *Proc. of IPSJ Int'l Conf. on Information Technology (Info Japan)*, pp. 325-332 (1990).
21) 滝沢 誠，中村章人：放送型通信アルゴリズム，情報処理，Vol. 34, No. 11, pp. 1341-1349 (1993).
22) Tanenbaum, A. S.: *Computer Networks* (2nd ed.), Prentice-Hall, Englewood Cliffs, NJ (1989).
23) Verissimo, P., Rodrigues, L. and Baptista, M.: AMp: A Highly Parallel Atomic Multicast Protocol, *ACM SIGCOMM '89*, pp. 83-93 (1989).

(平成6年7月11日受付)
(平成6年11月17日採録)

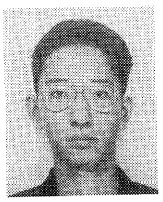
坂元紫穂子（学生会員）

1971年生．1991年東京電機大学理工学部経営工学科入学．現在，同大学同学部同学科4年次在学中．分散型システム，通信プロトコル等に興味を持つ．



中村 章人（正会員）

1966年生．1989年東京電機大学理工学部経営工学科卒業．1991年同大学院理工学研究科修士課程修了．1994年同大学院理工学研究科博士課程修了．現在，通商産業省工業技術院電子技術総合研究所研究員．工学博士．「OSIプロトコル技術解説」ソフトリサーチセンタ(共訳)．分散型システム，通信プロトコル等に興味を持つ．



立川 敬行（学生会員）

1971年生．1994年東京電機大学理工学部経営工学科卒業．現在，同大学院理工学研究科修士課程在学中．分散型システム，通信網，プロトコル等に興味を持つ．

滝沢 誠（正会員）

1950年生．1973年東北大学工学部応用物理学科卒業．1975年同大学院工学研究科応用物理学専攻修士課程修了．同年(財)日本情報処理開発協会入社．1986年東京電機大学理工学部経営工学科講師，1987年同助教授，1994年教授．工学博士．1989年9月より1年間ドイツ国立情報処理研究所（GMD）客員教授．1990年7月よりKeele大学(英国)客員教授．1991年よりマルチメディア通信と分散処理研究会幹事．分散型データベースシステム，通信網等の研究に従事．電子情報通信学会，ACM，IEEE各会員．「データベースシステム入門技術解説」ソフトリサーチセンタ，「OSIプロトコル技術解説」ソフトリサーチセンタ（共訳）．