タプルスペース通信の複製管理方法

川口 昇† 佐藤文明† 水野忠則†

分散システムでは共有メモリを利用して複数のプロセスを並列に動作させることが可能である．しかし，共有メモリを管理するサーバに障害が生じるとそれを利用している全プロセスが停止してしまう問題がある．これを解決する方法の1つとして，ネットワーク上に共有メモリの複製を配置する方法が考えられる．本研究では共有メモリとしてLindaモデルのタプルスペース通信を対象とし，タプルスペース通信の特性を生かしたより効率的な複製管理方式を提案する．また我々はこの方式をUNIXシステム上に実装し，性能評価を行った．本論文では，実装の方針と性能評価について報告する．

A Method of Replication Control and Implementation of Tuple Space Communication

NOBORU KAWAGUCHI,† FUMIAKI SATO† and TADANORI MIZUNO†

On distributed system, it is possible for processes to run in parallel by using shared memory. But if the shared memory server once crashed, all the processes using it will stop its execution. As one of the solution of this we made the distributed computers to have the replicas of the shared memory. In this research we deal with the Tuple Space Communication of Linda as the shared memory, and we propose the method of the replication control using the characteristic of Tuple Space communication. And we implemented this method on UNIX system, and evaluated it. We will report the scheme of this and the evaluation.

1. はじめに

近年のネットワーク技術の向上により，複数の計算機をネットワークで接続した分散環境が一般的になってきた．分散システムでは，情報や資源を共有したり，負荷をネットワーク上の計算機に分散することで処理効率の向上をはかることができる．また，このような分散システムでは共有メモリを利用して複数のプロセスを並列に動作させることが可能である．共有メモリのモデルとしてはLinda[1)~3)] Cellula[4),5)]等，様々なモデルが提案されている．特にLindaは場を媒体とした通信を行うモデルで，並列プログラムを書くときによく用いられる．

ところがこのような共有メモリを利用して通信する場合，共有メモリを管理するサーバに障害が生じると，それを利用している全プロセスが停止してしまうという問題がある．そこでこれを解決する方法の1つとして，ネットワーク上に共有メモリの複製を配置する方法が考えられる．ただし，複製を配置することにより，複製間の一貫性を保証する複製管理が新たに必要となる．本研究ではLindaモデルのタプルスペース通信でタプルスペースの複製を配置したモデルを考え，この通信の特性を生かしたより効率的な複製管理方式を提案する．

Lindaの複製管理に関する研究として，プロセスごとにチェックポイントを設け，障害が生じたときにこれを参照してロールバックをする方法[6)]が提案されているが，ロールバック中はプロセスは停止したままである．そこで本研究では障害が生じてもプロセスが停止することなく処理を続行できる方法を提案する．

以下，2章では分散共有メモリシステムとしてLindaモデルのタプルスペース通信について述べる．3章では本研究で提案する複製管理方式について詳しく述べる．ここでは従来の方式とこれをタプルスペース通信に適用する際に必要な変更点について述べ，シミュレーションを用いて評価する．4章では提案方式を実装する際の方針について述べる．5章ではこれらをUNIXシステム上に実装し，通信部分の性能評価を行っている．6章はまとめである．

† 静岡大学情報学部
Faculty of Information, Shizuoka University

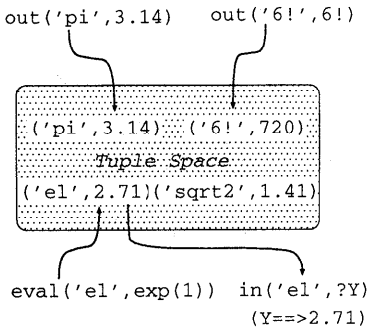


図1 Linda モデル
Fig. 1 Linda model.

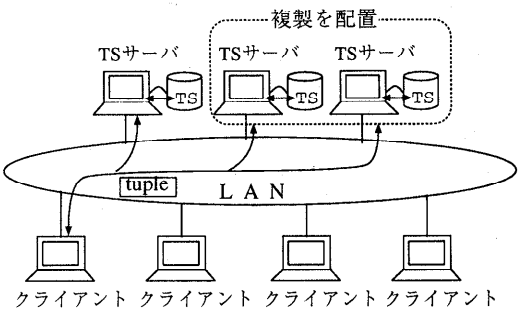


図2 複製付き TS 通信を用いた共有メモリシステム
Fig. 2 Replicated shared memory system using the TS communication.

2. 分散共有メモリシステム

2.1 TS 通 信

本研究では分散共有メモリとして Linda モデルのタプルスペース通信を扱う．Linda モデルは図1のように，タプルスペース（以下，TS と略記）と呼ばれる共有メモリ空間と，TS に対してデータ（タプルと呼ぶ）のやりとりを行うための4種の基本命令を提供する．基本命令は out()，eval()（タプルを TS に置く），in()（タプルを TS から取り出す），rd()（タプルを TS から読み込む）の4つで，プログラマは，この基本命令を使って並列実行の単位を生成し，プログラムを並列に実行できる．

このような TS 通信は，書き込み頻度が読み込み頻度より高い，タプルはファイルのように更新されることがないという2つの特性を持つ．前者の特性は書き込み処理が，out() での書き込みと，in() での削除の両方でなされていることからも分かる．後者の特性は，TS 通信が，タプルを置いて取り去るという通信形態をとるため，書き込もうとしているタプルが TS 中にすでに存在する場合はそのタプルを更新せずに別の場所に書き込まれるためである．これは TS 中に同一タプルが複数存在することを意味する．

2.2 複製付き TS 通信

従来の TS 通信では共有メモリである TS に障害が生じた場合にその影響が全プロセス（クライアント）に及ぶという問題があった．そこで本研究では TS の複製をネットワーク上に配置した図2に示されるモデルを対象とする．このようなモデルを利用した TS 通信を以後「複製付き TS 通信」と呼ぶ．

3. 提案する複製管理方式

3.1 定数合意方式

2.1 節で TS 通信では書き込み頻度が読み込み頻度よりも高いことを述べたが，本研究ではこの特性を生かして複製管理に定数合意方式[7]（図3）を採用する．この方式は書き込み定数 W，読み込み定数 R をそれぞれ設け，各定数の数だけ読み込みおよび書き込みを行う方式である．このとき，

$$W + R > N \quad (1)$$

を満たすように定数 W と R を設定すると，読み込み，書き込みがともに行われるものが少なくとも1つは存在する．したがって，読み込み時に最新のデータが得られ，一貫性が保証される．このときデータが最新かどうかは複製ごとに付加されたバージョン番号で判断する．

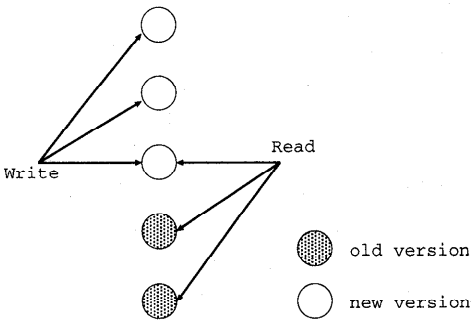


図3 定数合意方式
Fig. 3 Voting.

定数合意方式の利点は読み込み頻度，書き込み頻度に合わせて各定数を設定できる点で，書き込み頻度が高い TS 通信では書き込み定数を小さくすることで全体の処理時間を短縮できる．また本来この方式はファイルの複製に対して用いられる方法であるため，これを TS 通信に適用するにはいくつかの変更が必要になる．

3.2 改 良 点

3.2.1 バージョン番号

2.1 節で述べたように，タプルは更新されることが

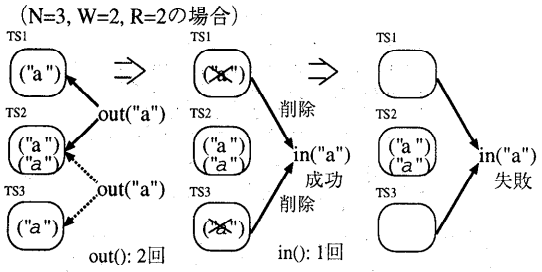


図4 削除操作での定数合意方式の問題点
Fig. 4 Problem of Delete operation.

ないため，タプルにはバージョンが存在しない．よって定数合意方式で用いられていたバージョン番号は不要となる．

また，TS 通信では in() を実現するために削除操作が必要になる．定数合意方式の条件を満たすには，削除操作は書き込み操作のときと同一のタプルの組に対してなされる必要がある．たとえば，**図 4** のように out() が 2 回実行された後，1 回目の in() を図のように実行すると 2 回目の in() でタプルの読み込みに失敗してしまう．これは 1 回の out() で書き込まれたタプルの組を削除しなかったためである．つまりタプルの削除は out() を 1 回分取り消すような形で実現されなければならない．そこでこれらのタプルの組をうまく削除できるように out() でタプルを書き込むときに，各タプルに「他の複製の位置を示す識別子」付加し，削除の際にこれを参照するようにした．

3.2.2 非同期処理

分散システムで共有領域にアクセスするときは他からのアクセスがないように共有領域にロックを掛ける必要がある．しかし今回のように共有領域が複数ある場合でも，通常は全複製にロックを掛ける必要があり，全ロックが解除されるまでの待ち時間が無駄になる．そこで本研究では複製にロックを掛けずに非同期に処理することにする．

非同期方式は処理時間の短縮という面では優れているが，ロックを掛けないためメッセージの到着順序が複製ごとに異なる場合があり，一貫性が保証されなくなる．そこで，**図 5** のように待ち行列中のメッセージを Birman のアルゴリズム[8),9)]を用いて並べ換え，到着するメッセージ順序がすべての複製で同一になるようにしてこれを解決する．

3.3 シミュレーションによる評価

従来の同期方式と提案する非同期方式の応答時間を比較するため，本研究ではシミュレーションで評価を行った．シミュレーションでは通信時間を 0 としているため，図 5 の 2PC アクセス部のメッセージ交換に

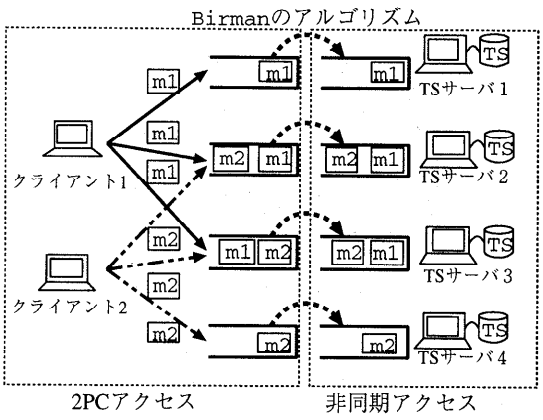


図5 非同期方式
Fig. 5 Asynchronous method.

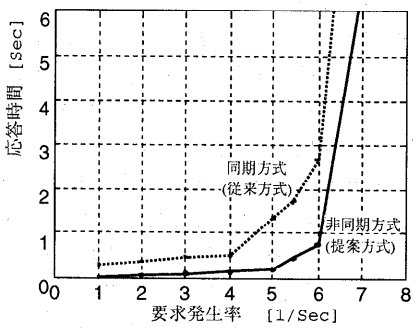


図6 シミュレーション結果
Fig. 6 Result of the simulation.

よる遅延は評価に含まれない．このため，非同期方式の評価は図 5 の非同期アクセス部のみの評価となっている．

図 6 はクライアント数を 5，TS サーバ数を 3，要求の平均処理時間を 10 ms としたときの要求発生率と応答時間の関係をグラフにしたものである．横軸の要求発生率はクライアント個々の要求発生率を示しており，縦軸の応答時間はクライアントが要求を出してから，TS サーバの応答を受信して処理を終了するまでの時間を示している．

このときの書き込み頻度と読み込み頻度の割合は 1:1 とし，書き込み定数と読み込み定数はどちらも複製数の過半数の 2 とする．

また，クライアント数，TS サーバ数を変化させた場合でも図 6 と似た傾向のグラフが得られ，提案方式の優位性を示すことができた．

4. 複製付き TS 通信の設計

この章では複製付き TS 通信を実装する際の方針について述べる．

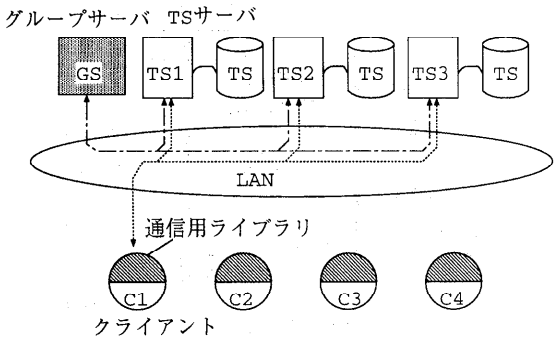


図7 システム構成
Fig. 7 Structure of the system.

4.1 システム構成

システム構成は図7のようになる．

このシステムはTSサーバ，クライアントプロセス，グループサーバからなる．

TSサーバはTSで送受信されるタプルの管理を行うサーバで，タプルの管理の他に後述のクライアントプロセスから出された要求をキューに貯める働きもする．本システムはTSを複数配置したモデルを対象とするため，TSの数だけTSサーバを用意する．TSサーバは「要求キュー」,「実行キュー」の2つのキューを持っている．要求キューはクライアントからの要求が入るキューである．システムはこのキューの各要求に付随するタイムスタンプを付け換え，実行可能状態にする．実行可能状態になった要求は実行キューに入れられタイムスタンプ順にソートされる．またこの処理とは並行に，システムは実行キューの先頭から要求を実行していく．

クライアントプロセスは実際にTS通信を利用するユーザによって生成されるプロセスである．ユーザは複製の存在を気にせずにTS通信を行えるように，複製管理アルゴリズムは通信用ライブラリに含める．

提案する複製管理アルゴリズムはクライアントとTSサーバ間のメッセージ交換で実現されるため，TSサーバにも複製管理アルゴリズムを実装する必要がある．このときクライアントプロセスは複数のTSサーバと通信する必要がある．そこでこれらの通信を容易にするため，クライアント–TSサーバ間の通信はグループ通信で実現する．すなわち，複数のTSサーバで1つのグループを構成し，クライアントはTSサーバ個々ではなく，このグループに対して通信を行うようにする．またグループ通信を用いることにより，障害時の回復処理がグループの離脱処理，登録処理という形で比較的容易に実現できる．本システムではグループ通信を実現するために，グループのメンバ情報（以後「グループ情報」と呼ぶ）等を管理するグループサーバを新たに配置する．

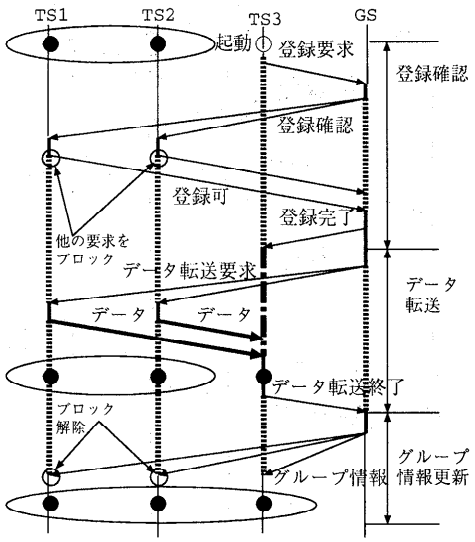


図8 グループ登録処理
Fig. 8 Process of entering the group.

グループ情報にはグループのメンバ数，各メンバのアドレス等が含まれ，クライアントはこの情報をもとにグループ通信を行う．通常，グループ情報はグループサーバと各TSサーバがそれぞれ保持する．グループの状態が変化した場合は，まずグループサーバのグループ情報が変更され，そのあと各TSサーバのグループ情報が変更される．このようにしてグループサーバ，TSサーバ上にはつねに最新のグループ情報が置かれている．

4.2 システムの動作

この節ではシステムの動作について説明する．ここではグループを構成するためのグループ登録処理について述べ，タプル操作命令の実行をout() 命令を例にとって説明する．また，サーバに障害が生じたときのグループ離脱処理についても説明する．そして最後にグループサーバに障害が生じたときの処理について述べる．

4.2.1 グループ登録処理

まずグループを構成するにはTSサーバが起動したときに図8に示すようなグループ登録処理が必要になる．

グループ登録処理は，登録確認処理，データ転送処理，グループ情報更新処理の3つから成っている．

最初の登録確認処理は新しいメンバをグループに登録してもよいかを既存のグループのメンバに確認する

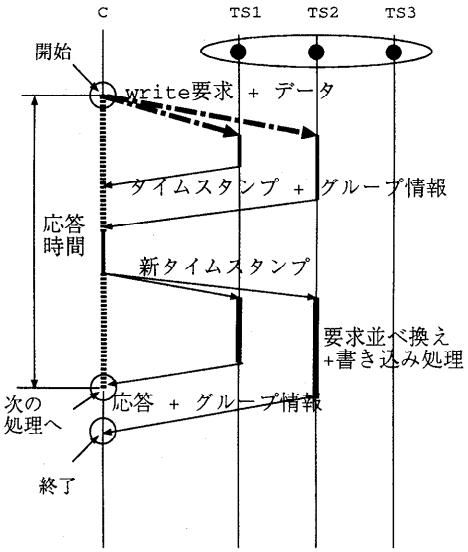


図9 out() 命令の実行
Fig. 9 Execution of out() instruction.

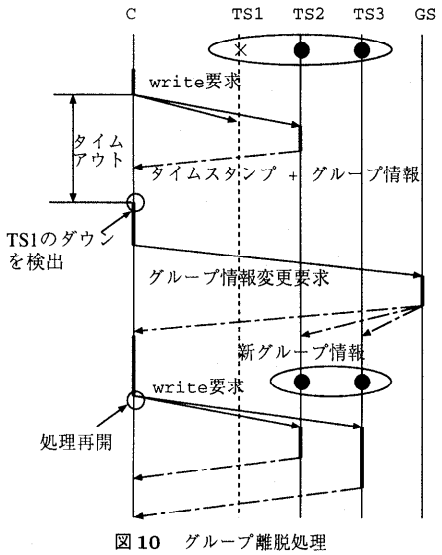


図10 グループ離脱処理
Fig. 10 Process of leaving from the group.

処理である．登録が可能な場合は，既存のメンバは現在処理中の処理を完了し，以後の要求をブロックする．

次のデータ転送処理は既存のメンバのTSデータを新メンバのTSにコピーする処理である．このとき複製間の一貫性を維持するためにデータ転送は読み込み定数の数だけ行う必要がある．

最後のグループ情報更新処理は新メンバをグループに加えた新たなグループ情報をグループのメンバに送信する処理である．また，この処理が終了すると同時に要求のブロックは解除される．

4.2.2 命令の実行

次に実際のタプル操作命令の実行を，**図9** に示される out() 命令を例にとって説明する．

クライアントで out() 命令が出されると，クライアントは write 要求と対象となるデータをグループのメンバの一部に送信する．送信すべきメンバの数は書き込み定数に従ってランダムに選択される．write 要求を受信したメンバは各自が保持する要求キューにこの要求を入れ，各メンバにローカルに用意されたタイムスタンプ（シーケンス番号）を送り返し，自分のタイムスタンプをインクリメントする．クライアントは要求を出した全メンバからタイムスタンプを受信すると，その中の最大値を選択し，これを再びグループのメンバに送信する．これを受信したメンバは，要求キューに貯められた要求に付随するタイムスタンプを，この新しいタイムスタンプに付け換えて実行キューに入れる．各メンバはこれらの処理と並行に，実行キューに入れられた要求を先頭から順に実行し，クライアントに応答を返す．このときクライアントは全メンバからの応答を待つ必要はなく，最初の応答が来た時点で次の処理に移ることができる．これは少なくとも1カ所で処理が完了すれば，残りの処理が失敗しても，後述するように失敗したサーバをグループのメンバから外すことで一貫性が保証できるからである．

4.2.3 グループ離脱処理

次にTSサーバがダウンして要求に対する処理が失敗した場合について説明する．サーバのダウンは応答時間にタイムアウトを設けることで検出される．

たとえば，**図10** のようにTS1がダウンした場合は最初の応答でタイムアウトになり，TS1のダウンが検出される．そしてクライアントはTS1のダウンをグループサーバに通知する．グループサーバはこれを受信して，TS1をグループから外した新たなグループ情報をグループの他のメンバとクライアントに送る．そしてクライアントはこの新しいグループ情報を元に先ほどの処理を再開できる．もしその後TS1が回復した場合は，再びTS1の登録処理を行えばよい．

離脱処理を行った場合，グループのメンバ数，すなわち式(1)のNが減少する．しかしNが減少しても式(1)の条件は満されるため，一貫性のうえでは問題はない．

4.2.4 グループサーバの複製

ここではグループサーバに障害が生じた場合の処理について述べる．図8〜10より，クライアントやTSサーバがグループサーバにアクセスするのは，クライアントの処理開始時とグループのメンバが変更された

ときのみで，グループサーバへのアクセスはほとんどないことが分かる．よってグループサーバに障害が生じても，その間にグループのメンバが変化しなければ問題はない．

しかし，グループサーバの障害が頻繁に起こるのであれば，2台目のグループサーバを設置し，1台目のグループサーバとつねに同じ状態にしておけばよい．そして1台目のグループサーバに障害が生じたときのみ2台目のグループサーバを起動すればよい．この場合，グループサーバへのアクセスはほとんどないことから，グループサーバを2台設置したことによる性能劣化はほとんどみられないと予想される．本研究ではグループサーバを1台のみとし，このような処理は行わない．

5. 通信部分の評価

本研究ではこれらのシステムをUNIX上に実装し性能評価を行った．**図11**は従来の同期方式と，提案する非同期方式をUNIX上に実装したときの，それぞれの応答時間を測定したものである．ここでは通信部分についての評価を行い，TSへの書き込みや読み込みは，一定時間（要求の平均処理時間として以下で設定）プロセスを停止することで仮想的に表現した．

測定条件はシミュレーションに合わせて，クライアント数を1～10，TSサーバ数を3，要求の平均処理時間を10 msとした．シミュレーションのときと同様，横軸の要求発生率はクライアント個々の要求発生率を示しており，縦軸の応答時間はクライアントが要求を出してから，TSサーバの応答を受信して処理を終了するまでの時間を示している．また，書き込み頻度と読み込み頻度の割合を1:1とし，書き込み定数と読み込み定数をそれぞれ過半数の2とした．

このグラフから，クライアント数が増加するほど非同期方式の方が同期方式に比べてより少ない応答時間が得られることが分かる．

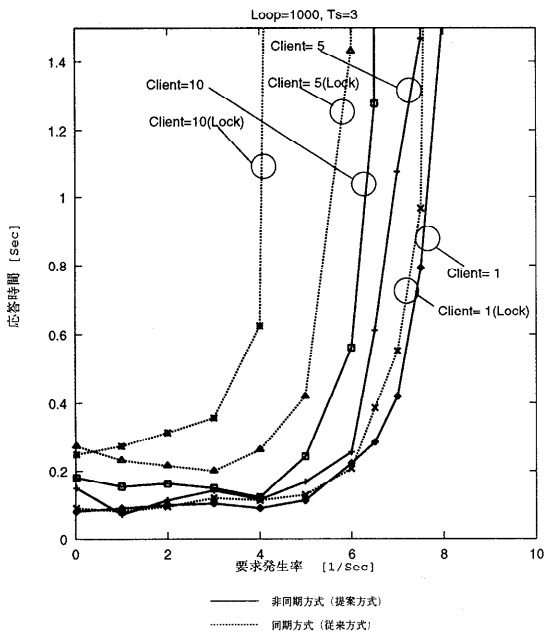


図11 要求発生率と応答時間の関係
Fig. 11 Relation between Ocuur rate and Respons time.

6. まとめ

本研究では従来のTS通信を拡張した複製付きTS通信を対象とし，TS通信の特性を生かした複製管理方式を提案した．

提案方式ではまず，複製管理に定数合意方式を用い，TS通信の特性に合わせて改良を加えた．また，応答時間を短縮するため処理を非同期にし，一貫性を保証するためBirmanのアルゴリズムを組み合わせた．そしてこれらをシミュレーションを用いて評価し，従来方式に比べてより短い応答時間を得られることを示した．

提案方式の実装においてはグループ通信の機構を用いてTSサーバの障害に容易に対処できるようにした．また提案方式と従来方式を実装して通信部分の性能評価を行い，クライアント数が増加した場合に提案方式の優位性を示すことができた．

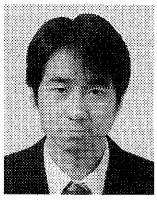
今後は複製の配置方法についても考察し改良を加えていく．

参考文献

1) Nicholas, C. and David, G.: How to Write Parallel Programs: A Guide to the Perplexed, *ACM Computing Surveys*, Vol.21, No.3 (1989).
2) 奥及 博，明石 修，村上健一郎，大海良治：NueLinda Interpreter in NueLinda—非均質システムNueLindaインタプリタの自己記述，情報処理学会論文誌，Vol.34, No.4, pp.568–579 (1993).
3) 木村康則，住元真司，細井 聡，小沢年弘，服部 彰：Linda処理系の試作について，信学研究報告，CPSY90-22 (1992).
4) 吉田紀彦，樽崎修二：場と一体化したプロセスの概念に基づく並列協調処理モデル Cellula，情報処理学会論文誌，Vol.31, No.7 (1990).
5) 吉田紀彦：協調計算モデル Cellula の超並列処理系の構想，情報処理学会プログラミング言語・基礎・実践研究会研究報告，pp.67–74 (1992).
6) 野里貴仁，杉本 明：分散Lindaのフォールトトレラント性，情報処理学会プログラミング言語・基礎・実践研究会研究報告，pp.75–82 (1992).

7) Gifford, D.K.: Weighted voting for replicated data, *Proc. 7th ACM SIGOPS Symp. Oper. Syst. Princip.* (1979).
8) Birman, K., Schiper, A. and Stephenson, P.: Lightweight Causal and Atomic Group Multicast, *ACM Trans. Computer Systems*, Vol.9, No.3 (1991).
9) 滝沢　誠，中村章人：放送通信アルゴリズム，情報処理，Vol.34, No.11 (1993).

(平成9年5月12日受付)
(平成9年9月10日採録)

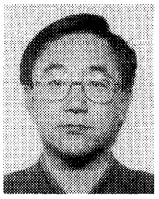


川口　昇（学生会員）

昭和47年生．平成8年静岡大学工学部情報知識工学科卒業．同年静岡大学大学院理工学研究科計算機工学専攻入学，現在に至る．主な研究分野は分散データベース．

佐藤　文明（正会員）

昭和37年生．昭和59年岩手大学工学部電気工学科卒業．昭和61年東北大学大学院修士課程修了．同年三菱電機（株）入社．平成7年静岡大学工学部助教授．現在静岡大学情報学部助教授．工学博士．分散処理システム，通信ソフトウェア開発環境，形式記述技法の研究に従事．電子情報通信学会，IEEE各会員．



水野　忠則（正会員）

昭和20年生．昭和43年名古屋工業大学経営工学科卒業．同年三菱電機（株）入社．平成5年静岡大学工学部知識情報工学科教授．現在情報学部情報科学科教授．工学博士．情報ネットワーク，プロトコル工学，モーバイルコンピューティングに関する研究に従事．著書としては，「プロトコル言語」（カットシステム），「MAP/TOPと生産システム」（オーム社），「分散システム入門」（近代科学社），「分散システム—コンセプトとデザイン」（電気書院）等がある．電子情報通信学会，IEEE各会員．