社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－EVA－8　(5)
2004／3／18

# RHiNET-2 クラスタにおけるユニキャストを基にしたマルチキャストアルゴリズムの評価

鯉渕道紘[†]　大塚智宏[†]
渡邊幸之介[†]　天野英晴[†]

高性能 PC クラスタで用いられているシステムエリアネットワーク (SAN) は，多くの場合，拡張性，耐故障性を重視して様々なトポロジを取ることができる．しかし，任意のトポロジに適用することができるマルチキャストアルゴリズムの研究はこれまでほとんど行われていない．本稿では 64 台のホストにより構成された RHiNET-2 クラスタを用いてユニキャストを基にした各マルチキャストアルゴリズムの実機評価を示す．評価結果より，RHiNET-2 クラスタにおいて，バリア同期時間は訪問する目的地ホスト順を定めるマルチキャストアルゴリズムにより，最大 23%の性能差が生じることが分かった．また，1) 136 Byte (17 フリット長) のパケットによるバリア同期では，ユニキャストのホップ数の削減が最も重要であり，2) 552 Byte (69 フリット長, 512 Byte データ) のパケットによるマルチキャストではパケットのコンテンションの削減が最も重要であることが確認された．

# Performance Evaluation of Unicast-based Multicast Algorithms on RHiNET-2 Cluster

Michihiro Koibuchi,[†] Tomohiro Otsuka,[†] Konosuke Watanabe[†]
and Hideharu Amano[†]

System Area Networks (SANs) usually accept arbitrary topologies since connection flexibility and robustness are preferred over the uniformity of interconnections in high-performance PC clusters. However, a few unicast-based multicast algorithms for arbitrary topologies have been developed. In this paper, we evaluate their performance on a real PC cluster with 64 hosts called RHiNET-2. Execution results show that multicast algorithms, which determine the visiting order of destination hosts, give the impact of 23% of barrier synchronization latency. Then, shorter unicast hops are crucial to 136-Byte (17-flit length) packets multicasts, while decreasing packet contentions is crucial to 552-Byte (69-flit length, 512-Byte data) packets multicasts.

## 1. はじめに

PC クラスタにおいてパーソナルコンピュータ (PC) 間を接続するシステムエリアネットワーク (SAN) は，システムの性能向上の鍵の 1 つとなっている (Myrinet, InfiniBand)．SAN ではダイレクトメモリ通信を高速に行うために，従来の大規模並列計算機で用いられてきた相互結合網と同様に高バンド幅，低レイテンシであることが求められる．SAN はスイッチ群と大容量の point-to-point リンクを用いて構成されるため，パケットは複数のスイッチを経由して目的地に到達することになる．そのため，通信経路の設定が性能に大きく影響する．

SAN におけるルーティングアルゴリズムはバーチャルカットスルー方式 (VCT 方式) もしくはワームホール方式 (WH 方式) によりパケットを転送するため，デッドロックフリーであることが求められる．並列計算機の相互結合網と違い，SAN では任意のスイッチトポロジをサポートしていることが多い．したがって，ルーティングアルゴリズムはデッドロックフリーと経路保証を両立させることが難しい．

そのため，ユニキャスト (一対一) では，1) スパニングツリーの持つ非循環性と連結性を利用するルーティングアルゴリズム (Up*/Down*[1], Prefix[2])，もしくは 2) 循環を除去するために仮想チャネルを使用するルーティングアルゴリズム (構造化チャネル法[3]，DL[4]) などのシンプルな考え方に基づくものが提案されており，現在も盛んに議論がなされている．

一方，マルチキャスト (一対多) もバリア同期などで頻繁に使われる基礎的な操作であるため，効果的なマルチキャストアルゴリズムの開発が不可欠である．これまで，並列計算機の相互結合網向けに $k$-ary $n$-cube やメッシュを対象としたマルチキャストアルゴリズムは様々なものが提案されてきたが，SAN に適用できる手法は少ない．マルチキャストは 1) ユニキャストを基にした方法，2) ツリーを基にした方法，3) 経路を基にした方法，の 3 種類に分類できる[5]．

ユニキャストを基にしたマルチキャストは単純であり，マルチキャストするホスト数を $d$ とすると $\lceil log_2(d+1) \rceil$ ステップの手順が必要になる．一方，ツリーを基にした方法は，スイッチでパケットを複製，もしくは分割させ，各ホストに配信する方法である．この方法はホスト-スイッチ間のパケット転送数を減らすことができる利点がある．しかし，ツリーを基にした方法では，各スイッチにおいて各目的地ホストのメモリ空間，プロセス番号の設定を行う必要がある．さらに，図 1 に示したように，分岐したパケット間でフリットが同期して動く場合，特にワームホールルーティングでは，ユニキャストにおいてデッドロックフリーである経路群を用いても，マルチキャストパケット間でデッドロックが発生する恐れがある．例え

† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

ば，メッシュトポロジの一部を示した図 1 においてパケット A,B ともに，デッドロックフリールーティングである west-first turn モデルの経路を満たしている．しかし，パケット A,B はそれぞれスイッチ 1,0 においてブロックされているためデッドロックが発生している．このデッドロックの制御を不規則なトポロジで行うことは極めて難しい．一方，経路を基にした方法は，すべての目的地を通る経路でパケットを転送する．しかし，経路を基にする方法は SAN ではトポロジに制限がないため，経路が複雑になる問題がある．

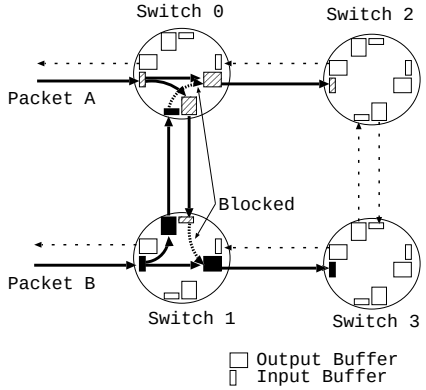



**図 1** 2 つのマルチキャストパケット間のデッドロック
Fig. 1 Two multicast packets in deadlock

そのため，本稿では SAN におけるユニキャストを基にしたマルチキャストについて焦点をあてる．これには，現在，1 つのスイッチに接続されているホスト群を 1 つのグループとしてグループ間転送とグループ内転送を階層的に行う手法や，Up*/Down*ルーティングの性質を利用した手法が提案されている[6]．しかし，実機での評価はほとんど行われていない．シミュレーションによる評価ではホスト内のパケット処理などを抽象化して高速化を行っている場合が多いため，実機の PC クラスタにおける各マルチキャストアルゴリズムの効果を正確に見積ることが難しい．

そこで，本稿では，64 台のホストで構成される RHiNET-2 クラスタ[7][8] における各マルチキャストアルゴリズムの性能を評価する．RHiNET-2 クラスタのネットワーク RHiNET-2 は 1) ユーザレベルダイレクトメモリ通信をハードワイヤードで実現したネットワークインタフェース RHiNET-2/NI，2) 8Gbps の光リンク，3) 64Gbps カットスルースイッチ RHiNET-2/SW，により構成される SAN である．RHiNET-2 は代表的な SAN である Myrinet, InfiniBand と同様にトポロジに制限がないため，様々なトポロジにおいてマルチキャストアルゴリズムを実装することができる．

以後，第 2 章では SAN における既存のユニキャストを基にしたマルチキャストアルゴリズムを示す．そして，第 3 章では RHiNET-2 クラスタについて述べ，第 4 章において RHiNET-2 クラスタを用いたマルチキャストアルゴリズムの評価結果を示す．最後に第 5 章においてまとめと今後の課題を述べる．

## 2. 既存のマルチキャストアルゴリズム

SAN においてユニキャストを基にしたマルチキャストアルゴリズムは，1) ユニキャストの平均ホップ数を削減すること，および 2) ユニキャスト間においてコンテンションフリー，もしくはコンテンションをできるだけ削減すること，が重要である．コンテンションとは，例えば図 2 のように複数のパケットが同時に 1 つの物理チャネル上に重なることをいい，どちらかのパケットにレイテンシが生じる．このコンテンションは経路を変更することで回避することができる．例えば，図 2 においてホスト 3 からホスト 17 へのパケットの経路を，スイッチ 3 を通る経路に変更することでホスト 7 からホスト 16 への経路とのコンテンションを防ぐことができる．しかし，多数のホスト間で，このように空間的に，もしくは時間的にコンテンションを避けるスケジューリングを行うことは難しい．特に，トポロジが特定されない場合，マルチキャストアルゴリズムがコンテンションフリーを保証することは極めて難しい．

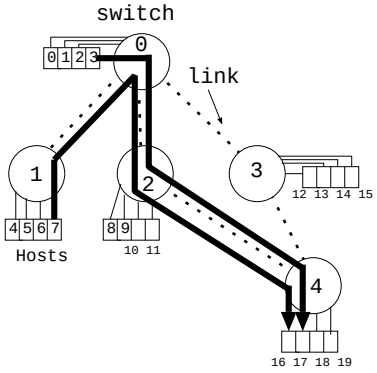



**図 2** コンテンションの例
Fig. 2 An example of contention

もっとも単純なマルチキャストアルゴリズムは，ホスト番号順にデータを配信 (訪問) する方法— Host-ID Order (HIO)—である．その他の単純なマルチキャストアルゴリズムとしてはランダムな順にホストを訪問する手法— Random Order (RO)—がある．RO は MPI の実装でしばしば用いられる[6] が，場合によっては効率の悪い訪問順になってしまう．例えば，図 2 において，ホスト 0 から，ホスト 1,2,...,11 およびホスト 16,17,...,19 にマルチキャストするとする．この場合，訪問順によってはホスト 4,5,6,7 からホスト 16,17,18,19 への最も経由スイッチ数の多いホスト間のユニキャスト通信が発生してしまう恐れがある．

そこで，同一スイッチに接続されている目的地ホストを 1 グループにし，各グループの 1 ホストにパケットを転送した後，グループ内で転送する手法—Switch-based Order (SO) と Switch-based Hierarchical-Order (SHO)—が提案されている[6]．例えば図 2 において同様のマルチキャストが発生したとする．この場合，まず，ホスト 0 からホスト 8 にパケットを送り，次にホスト 0, 8 からホスト 4, 16 へそれぞれパケットを送る．そして，最後にグループ内のホスト間で通信することにより，コンテンションフリーでマルチキャストを実行することができる．

しかし，SO，もしくは SHO アルゴリズムを用いた場合，グループ内の通信ではコンテンションフリーである一方，グループ間の通信ではユニキャストの順番によりコンテンションが頻繁に起こる可能性がある．そこで，さらにこのコンテンションを減らすために，ルーティングアルゴリズムに特化した手法も提案されている[6]．

**図 3** RHiNET-2 クラスタ
Fig. 3 RHiNET-2 cluster

## 3. RHiNET-2 クラスタ

本章では，マルチキャストアルゴリズムの評価に用いた RHiNET-2 クラスタについて述べる．

### 3.1 RHiNET-2 クラスタの構成

RHiNET-2 は新情報処理開発機構 (RWCP)，日立 (株)，慶應義塾大学により，分散配置されている PC を用いた並列分散環境の構築を目的として開発されたネットワークである．

16 スイッチ，64 台のホストで構成される RHiNET-2 クラスタを図 3 に示す．各ホストの PCI バス (64bit/66MHz) にはネットワークインタフェース RHiNET-2/NI が装着されている．また，スイッチ，ホストは 8Gbps の光リンク (2m および 5m) により相互接続されている．表 1 にホストの仕様を示す．

**表 1** ホストの仕様
Table 1 Specification of host

| | |
|---|---|
| CPU | Intel Pentium III 933MHz × 2 (SMP) |
| Chipset | Serverworks ServerSet III HE-SL |
| Memory | PC133 SDRAM 1GByte |
| PCI | 64bit/66MHz |
| OS | RedHat Linux 7.2 (kernel 2.4.18) |

#### 3.1.1 ネットワークインタフェース RHiNET-2/NI

ネットワークインタフェース RHiNET-2/NI はネットワークコントローラチップ Martini[7], 256 MByte SDRAM，および光インタフェースを持ち，汎用の 64bit/66MHz PCI バスを持つ PC に装着する．コントローラチップ Martini はユーザレベルゼロコピー通信，アドレス変換機構，メモリ保護等をハードワイヤードロジックで実装した ASIC チップである．Martini は大きく分けて，2 種類の基本通信命令—リモート DMA 転送と PIO による転送—を提供する．前者は高バンド幅を実現するためのもので，PUSH (リモートライト) と PULL (リモートリード) の 2 種類の通信を提供する．後者は，低レイテンシを実現できるため PCI バスを用いる場合に小さいサイズのデータ転送に適している．パケットはデータ転送単位である 8 Byte のフリットに細分化して転送される．また，ヘッダとテイルは計 40 Byte (5 フリット) である．

#### 3.1.2 スイッチ RHiNET-2/SW

スイッチ RHiNET-2/SW[9] は 8 個の入出力ポートを持ち，8 Gbps の光リンクでホストや他のスイッチと接続される[10]．ただし，現在，より安定した環境を構築するために光リンクの周波数を 800MHz から 600MHz に落としている．そのため，現在はリンクの最大転送容量は 6 Gbps となっている．よって，RHiNET-2/SW は本来 64Gbps のスループットを持っているが，現在は 48Gbps のスループットで稼働している．また，各ポートは 16 本の仮想チャネルを提供し，Go & Stop フローコントロールを採用している．各仮想チャネルは 4KByte のバッファを持っているため，200 m のリンク長をサポートする．

RHiNET-2 の光リンクモジュールは $10^{-20}$ オーダの極めて低い bit-error-rate (BER) を持ち，さらに，各フリットに ECC を付加することでエラー検出，訂正を行っている．そのため，RHiNET-2 では信頼性のある通信がハードウェアレベルで保証されている．RHiNET-2 クラスタの詳細および性能評価は[8] に示されている．

### 3.2 トポロジと固定ルーティング

RHiNET-2 におけるルーティングは，それぞれのスイッチにおいてパケットのヘッダフリットに記述されている目的地をインデックスにしてルーティングテーブルから出力ポートを得るテーブルルックアップ (分散) 方式の固定ルーティングである．また，出力仮想チャネル番号の増減は出力ポートと入力ポートの組をインデックスにしてルーティングテーブルから決定される．RHiNET-2/SW は 16 本の仮想チャネルを持つが，データ転送パケットが番号 0 から 7 までの仮想チャネルを使い，応答などのシステム制御パケットは番号 8 から 15 までの仮想チャネルを使う．しかし，両方のパケットとも各スイッチにおいて同一のルーティングテーブルを用いる．

そのため，ルーティングテーブルを変更することにより，RHiNET-2 クラスタは様々なトポロジ，ルーティングの組み合わせを取ることができる．

## 4. 評　　価

### 4.1 評 価 条 件

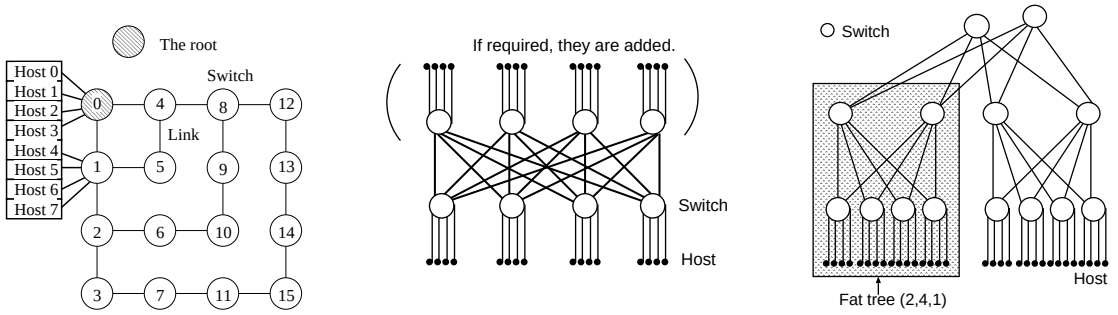
#### 4.1.1 トポロジとルーティングアルゴリズム

図 4 に示した 3 つのトポロジおよび 4×4 メッシュの計 4 種類のスイッチのトポロジを用いて評価した．各スイッチの 4 つのポートは異なるホストに接続し，残りの 4 つのポートは隣接スイッチに接続する，もしくは使用しない．RHiNET-2 クラスタは 16 台のスイッチで構成されているため，最大 64 ホストを持つ計算システムとなる．また，Fat ツリーおよび Myrinet Clos 網については上層のスイッチにホストを接続しない場合についても測定した．この場合，16 ホストを持つ計算システムとなる．また，スイッチ 0 に接続しているホストをホスト 0,1,2,3，スイッチ 1 に接続しているホストをホスト 4,5,6,7 というようにスイッチ番号順にホスト番号を割り当てた．

トポロジ A およびメッシュにおけるルーティングとしては，多数の仮想チャネルを持つ利点を生かし，仮想チャネルを必要としない Prefix ルーティング[2]，Up*/Down* ルーティング[1]，仮想チャネルを 2 本以上必要とする DL ルーティング[4]，多数の仮想チャネルが必要になる構造化チャネル法 (SBP)[3] を用いた．

Up*/Down* ルーティング，DL ルーティング，構造化チャネル法は，経路探索時に複数の最短経路を発見する場合がある．本実装では，経路を分散させるために，同一スイッチ間に複数経路が存在する場合，ホスト毎に異なる経路を割り当てた．また，仮想チャネルは Prefix ルーティングでは 1 本☆，Up*/Down*，DL ルーティングでは

☆ 仮想チャネル数とは，以後データ転送用パケットが使用する本数の




(a) トポロジ A　(b) Myrinet Clos 網　(c) Fat ツリー (2,4,2)

**図 4**　評価に用いたトポロジ
Fig. 4　Topologies considered in execution

2 本，構造化チャネル法ではそのトポロジの直径に従い，6 本用いた．また，Fat ツリーおよび Myrinet Clos 網ではデッドロックが発生することがないため，1 本の仮想チャネルを使う最短型ルーティングを用いた．

#### 4.1.2 測 定 項 目

RHiNET-2 クラスタではユニキャストを基にしたマルチキャストによりバリア同期を行うため，全ホストのバリア同期時間をパケットサイズが 17 フリット (ヘッダ，テイル計 5 フリット，データ 1 フリット, 残りはハードウェアパディング) の場合と 69 フリット (ヘッダ，テイル計 5 フリット，データ 64 フリット (512Byte データ)) の場合の 2 種類について測定した．データ転送にはパケット長が 17 フリットの場合 PIO を用い，69 フリットの場合はリモート DMA 転送を用いる．また，バリア同期時間は 1 つのマルチキャストについて 100,000 回実行した平均をとった．

評価したマルチキャストアルゴリズムは次の通りである．

- Random Order (RO)
- Switch-based Hierarchical-Order (SHO)
- Switch-based Contention Order (SCO)
- 1SHO-3RO
- Switch-based Host-ID Order (SHIO)
- Host-ID Order (HIO)

いずれのアルゴリズムも 64 ホストのバリア同期の場合，バリアの収集のためのブロードキャストに 6 ステップ，バリアの解放のためのブロードキャストに 6 ステップの計 12 ステップが通信に必要となる．

ここで各アルゴリズムを用いたバリア同期の実装を説明するためにホストの訪問順を示すリスト (*list*) を定義する．$list(0, 1, 2, \ldots, 15)$ は，ホスト 0 がバリアの収集を行うことを示す．そして，このリストにおいてバリア同期は，まず，ホスト後ろ半分にあたる 8, 9,. . ., 15 の 8 ホストがそれぞれホスト 0, 1,. . ., 7 にバリアの要求パケットを送信する．2 番目にホスト 5, 6, 7, 8 は，ホスト 0, 1, 2, 3 にパケットを送信し，3 番目にホスト 2,3 がホスト 0,1 に送信し，4 番目にホスト 1 がホスト 0 にパケットを送信することでバリアの収集が完了する．そして，次に逆順でバリアの開放を行う．ただし，ホスト 2 からホスト 0 への通信とホスト 3 からホスト 0 への通信のように依存関係のない通信はそれぞれ非同期に行うことができる．

簡単のため，以降の各アルゴリズムの説明ではスイッチ 0, 1, 2, 3 に接続している 16 ホスト (ホスト 0, 1, 2,..., 15) 間のバリア同期について述べる．

##### 4.1.2.1 RO, SHO, HIO

第 2 章で述べたように，RO では例えば *list*(0, 2, 9, 4, 13, 15, 7, 1, 14, 11, 3, 8, 6, 10, 12, 5) といったランダムな順番になる．また，HIO では $list(0, 1, 2, \ldots, 15)$ としてホスト番号順に訪問する．一方で SHO はグループ内の通信と，それ以外との通信を分けて行うため，例えば *list*(0, 8, 12, 4, 1, 9, 13, 5, 2, 10, 14, 6, 3, 11, 15, 7) のようになる．

##### 4.1.2.2 Switch-based Contention Order

SCO は SHO のパケットコンテンションの削減効果を調べるために評価した．SCO では SHO で発生するスイッチ間転送時のコンテンションを増やすために，ほぼ同時に 1 つのスイッチ間リンクに 4 つのパケットが通過するようにした．具体的には，SCO では，先の SHO のリストに対して $list(0, 1, 2, 3, 8, 9, 10, 11, 12, 13, 14, 15, 4, 5, 6, 7)$ という順となる．これにより，例えば，ホスト 5 からホスト 8 へパケットを送信している時に，ほぼ同時にホスト 6, 7, 8 からもホスト 9, 10, 11 へのパケットが生じるため，コンテンションが発生する．SHO と SCO を比較することで，スイッチ間リンクにおけるコンテンションが性能に与える影響が明らかになる．

##### 4.1.2.3 1SHO-3RO

1SHO-3RO は，SHO におけるグループ内の通信によるパケットホップ数の削減効果を調べるために評価した．1SHO-3RO では，SHO と同様の手順で各スイッチの 1 つの PC にマルチキャストを行うが，同一スイッチに接続されたホスト間のパケット転送を行う代わりにランダムに生成されたホスト対でパケット転送を行う．例えば，先の SHO のリストに対しては，$list(0, 8, 12, 4, 9, 15, 11, 10, 2, 14, 13, 6, 1, 7, 3, 5)$ となる．1SHO-3RO と SHO を比べることで，SHO の同一グループ内転送の効果が明らかになる．

##### 4.1.2.4 Switch-based Host-ID Order

SHIO では *list*(0, 4, 8, 12, 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15) のようにスイッチ内のホストをグループ化し，バリアの解放においてグループ間のデータ転送をした後，グループ内のデータ転送を行う．ただし，SHO と異なり，グループ間の転送はスイッチ番号順に行う．SHIO では，e-cube ルーティングを用いたメッシュにおいて HIO と異なり，コンテンションが発生しない (図 5 参照)．さらに，この場合，SHIO ではパケットのホップ数も高々2 ホップであるためほぼ理想的なブロードキャストが実現できる．図 5 において数字はグループ内パケット転送を省略した

ことを指す．なお，システム制御用パケットも同数の他の仮想チャネルを使用する．

場合におけるホスト 0 からのブロードキャスト—バリアの開放—の手順を示している．

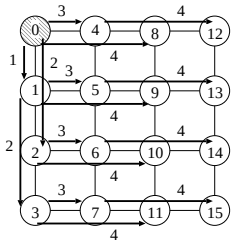


**図 5** 64 ホストのメッシュにおける SHIO マルチキャスト
Fig. 5 SHIO multicast on the mesh with 64 hosts

RO, SHO, SCO, 1SHO-3RO については，各々複数の訪問順を取ることが可能である．そこで，RO についてはランダムに 10 個のリストを生成し，その実行時間の平均をとった．また，SHO についてもスイッチグループの訪問順をランダムに 10 パターン決め，その実行時間の平均をとった．同様に，SCO, 1SHO-3RO は各 SHO のリストに対応させたリストを作成し，10 パターンの平均をとった．

### 4.2 実 行 結 果

表 2，3，4 に各トポロジ，デッドロックフリールーティングにおける 17 フリット長のパケットによるバリア同期の実行時間を示す．これらの表において SBP は構造化チャネル法を表し，表 4 において括弧内の数字はスイッチ数を表す．表 2，3 より，64 ホストのバリア同期において，RO は SHO に比べて 8.3%〜12%実行時間が大きくなっている．これは，SHO は 64 ホストのブロードキャストに必要な 6 ステップの内，同一スイッチのホスト間転送の 2 ステップは 1) コンテンションフリーであり，2) パケットを 1 ホップで送信できることに起因すると考えられる．

この 2 つの要因について詳細に検討を行うために，まず SCO と SHO を比較する．SCO は，同一スイッチ内のホストにマルチキャストした後，スイッチ間転送を行うため，SHO と手順が逆になる．そのため，SCO では SHO に比べてスイッチ間リンクに約 4 倍のコンテンションがかかる．表 2，3 より，SCO は SHO に比べ最大 5%実行時間が遅くなっているが，RO と SHO の差に比べると小さい．そのため，SHO の性能向上はコンテンションの発生が主たる原因ではないと考えられる．

また，表 2，3 より，1SHO-3RO と RO はほぼ同じ性能を示しており，SHO に比べて最大 12%実行時間が低下していることが分かる．1SHO-3RO と SHO ではまず，全スイッチの 1 つのホストにパケット転送を完了させる点は同じである．つまり，64 ホストのブロードキャストに必要な 6 ステップのうち，はじめの 4 ステップは同じである．その後，SHO は 1 ホップのパケット転送であることに対し，1SHO-3RO ではそのトポロジ，ルーティングにおける平均パケットホップ数のパケット転送となる．つまり，SHO と 1SHO-3RO との違いは最後の 2 ステップにおけるパケットのホップ数の差が大きいといえる．よって SHO と RO の性能差はパケットのホップ数によるものが最も大きいと考えられる．

ここで，パケットのホップ数がバリア同期時間に与える影響について図 6 に詳細を示す．図 6 は，経由スイッチ数を変化させた時の 2 ホスト間のバリア同期時間を示している．図 6 より，1 ホップ増える毎に約 0.7$\mu$sec 増えていることが分かる．このため，ブロードキャストにおいて，例えば 64 ホストの場合，6 ステップ，つまり，6 回のユニキャストが必要になることから，ホップ数がブロードキャストの性能に大きく影響するといえる．

**表 2** Topology A におけるバリア同期時間 ($\mu$sec)
Table 2 Execution time of barrier synchronization on Topology A ($\mu$sec)

| | Prefix | Up*/Down* | DL | SBP |
|---|---|---|---|---|
| RO | 52.39 | 50.53 | 47.86 | 47.83 |
| SHO | 47.31 | 45.09 | 43.97 | 43.98 |
| SCO | 49.59 | 45.79 | 44.33 | 44.37 |
| 1SHO-3RO | 53.02 | 49.98 | 48.05 | 48.04 |
| SHIO | 44.35 | 38.99 | 39.04 | 39.04 |
| HIO | 44.95 | 39.62 | 39.38 | 39.32 |

**表 3** メッシュにおけるバリア同期時間 ($\mu$sec)
Table 3 Execution time of barrier synchronization on the mesh ($\mu$sec)

| | Prefix | Up*/Down* | DL | SBP |
|---|---|---|---|---|
| RO | 51.49 | 45.77 | 45.62 | 45.61 |
| SHO | 46.20 | 42.19 | 42.10 | 42.10 |
| SCO | 48.52 | 42.09 | 42.05 | 42.07 |
| 1SHO-3RO | 51.53 | 45.92 | 45.97 | 45.96 |
| SHIO | 44.32 | 38.83 | 38.84 | 38.85 |
| HIO | 46.14 | 38.92 | 38.89 | 38.92 |

**表 4** 各トポロジにおける 16, 32 ホストのバリア同期時間 ($\mu$sec)
Table 4 Execution time of barrier synchronization with 16 or 32 hosts on each topology($\mu$sec)

| | Fat-Tree (6) | Myrinet Clos(8, 16 hosts) | Myrinet Clos(8, 32hosts) | Mesh (4×2) |
|---|---|---|---|---|
| RO | 27.68 | 27.62 | 33.60 | 35.00 |
| SHO | 26.07 | 25.89 | 31.90 | 32.59 |
| SCO | 25.90 | 25.89 | 31.91 | 32.55 |
| 1SHO-3RO | 28.32 | 28.10 | 33.32 | 35.06 |
| SHIO | 25.86 | 25.86 | 32.42 | 31.64 |
| HIO | 25.85 | 25.89 | 32.38 | 31.65 |

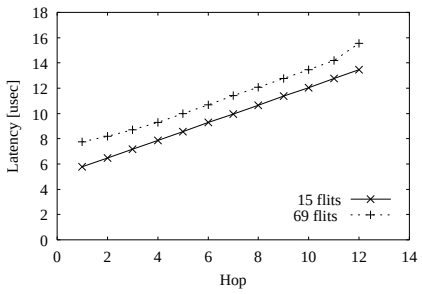



**図 6** ホップ数毎の 2 ホスト間のバリア同期時間 ($\mu$sec)
Fig. 6 Executiohn time of barrier synchronization on two hosts ($\mu$sec)

次にホスト，もしくはスイッチ ID 順にユニキャストした場合のブロードキャストについて比較を行う．RHiNET-2 クラスタではスイッチを 2 次元に配置させ，左上のスイッチから順にホストに ID を割当てたため，番号順にブロードキャストすることでホップ数を削減できるということが予測される．また，一般的に PC クラスタを構築する場合にも，このような ID の割り当ては，単純であるため，使用されることがあると考えられる．

表 2，3 より，SHIO と HIO は RO に比べて 10%から 23%の性能向上を達成していることがわかる．これは，HIO では隣接スイッチのホスト間通信が多く，パケットホップ数を抑えることができたためと考えられる．また，SHIO では，スイッチ間リンクのコンテンションが HIO に比べておさえられるためさらに高性能であるといえる．

次に，トポロジ，ルーティングアルゴリズム，ホスト

数がマルチキャストアルゴリズムに与える影響について検討する．トポロジとルーティングアルゴリズムの組み合わせにより，ユニキャストのホップ数が決定される．そのため，バリア同期時間は，これらに大きく左右されるといえる．具体的には，表 2，3 より各マルチキャストアルゴリズムにおいて構造化チャネル法，DL ルーティングが Up*/Down* ルーティング，Prefix ルーティングに比べて最大 16%性能向上していることが分かる．また，同様に，トポロジの性能が良い—直径が小さい，次数が大きい—ほど各マルチキャストアルゴリズムの性能が向上していることが分かる．しかし，表 2，3，4 より，各マルチキャストアルゴリズムの優劣はこれらにより大きく変わることはほとんどないといえる．

最後にパケットのデータサイズを 512 Byte にした場合の評価結果を表 5 に示す．表 5 において括弧内の数字はスイッチ数を表している．512 Byte データの場合，1 フリットが 8Byte であるため，パケット長はヘッダとテイルフリットを合わせて計 69 フリットとなる．表 5 において，Fat ツリーは比較のため図 4 の 6 スイッチを用いた場合 (2,4,1 Fat ツリー) と 14 スイッチを用いた場合 (2,4,2 Fat ツリー) ともに 16 ホスト間での実行時間を測定している．ただし，14 スイッチを用いた場合，最下層の 8 つのスイッチの 2 ホストずつを使用して 16 ホストとした．表 5 より，SHO, SHIO の性能が高いことが分かる．これは，512 Byte のデータ転送においては，スイッチ内転送を増やし，コンテンションを削減することが重要であることを示している．また，2 つの Fat ツリー の比較をすると，スイッチの階層数が増えることにより，14 スイッチの Fat ツリーの方が性能が悪いことが分かる．これはホップ数の差によるものと考えられる．表 3，4，5 より，パケット長が小さい場合はマルチキャストアルゴリズムにおけるパケットのホップ数が性能に大きく影響する一方，パケット長が大きい場合は，パケットの経路が分散していること、すなわち、コンテンションを削減することが性能に大きく影響するようになるといえる．

**表 5** 512 Byte データの集合通信時間 ($\mu$sec)
Table 5 Execution time of collective communication with 512 Byte data ($\mu$sec)

| | メッシュ(16) | Fat ツリー (14) | Fat ツリー (6) |
|---|---|---|---|
| RO | 65.60 | 46.81 | 37.86 |
| SHO | 56.21 | 39.06 | 35.03 |
| SCO | 63.08 | 40.07 | 35.73 |
| 1SHO-3RO | 65.31 | 41.05 | 37.65 |
| SHIO | 52.95 | 37.29 | 34.79 |
| HIO | 61.65 | 38.46 | 35.65 |

## 5. ま と め

64 ホストの RHiNET-2 クラスタにおけるユニキャストを基にした既存のマルチキャストアルゴリズムのバリア同期時間について調査した．評価結果より，RHiNET-2 クラスタにおいてバリア同期時間は，マルチキャストの訪問するホスト順を定めるアルゴリズムにより，最大 23%の性能差が生じることが分かった．また，1) 136 Byte (17 フリット長) のパケットによるバリア同期では，パケットのホップ数の削減が最も重要であり，2) 552 Byte (69 フリット長) のパケットによるマルチキャストでは，パケットのコンテンションの削減が最も重要であることが分かった．各マルチキャストアルゴリズムは 構造化チャネル法，DL ルーティングなどのパケットのホップ数が小さく，かつ，経路を分散することができるルーティングアルゴリズムを使用することによりバリア同期時間を最大 16%向上することができた．また，同様に，各マルチキャストアルゴリズムの性能は直径が小さく，かつ，次数が大きいトポロジを用いることで性能が向上することも確認された．

**謝辞** RHiNET-2 クラスタに関して貴重なご意見を下さった慶應義塾大学理工学部西 宏章助手、河野 賢一氏、上樂 明也氏、北村 聡氏に感謝致します。

## 参 考 文 献

1) M.D.Schroeder and al et.: Autonet: a high-speed, self-configuring local area network using point-to-point links, *IEEE Journal on Selected Areas in Communications*, Vol. 9, pp. 1318–1335 (1991).
2) J.Wu and L.Sheng: Deadlock-Free Routing in Irregular Networks Using Prefix Routing, *Proceedings of Parallel and Distributed Computing Systems*, pp. 424–430 (1999).
3) 堀江, 石畑, 池坂: 並列計算機 AP1000 における相互結合網のルーチング方式, 電子情報通信学会論文誌, Vol. J75-D-1, No. 8, pp. 600–606 (1992).
4) M.Koibuchi, A.Jouraku and H.Amano: Descending Layers Routing: A Deadlock-Free Deterministic Routing using Virtual Channels in System Area Networks with Irregular Topologies, *Proceedings of the International Conference on Parallel Processing*, pp. 527–536 (2003).
5) Libeskind-Hadas, R., Mazzoni, D. and Rajagopalan, R.: Tree-Based Multicasting in Wormhole-Routed Irregular Topologies, *Proceedings of the Merged 12th International Parallel Processing Symposium and the 9th Symposium on Parallel and Distributed Processing* (1998).
6) Kesavan, R. and Panda, D.: Efficient Multicast on Irregular Switch-Based Cut-Through Networks with Up-Down Routing, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, No. 8, pp. 808–828 (2001).
7) K. Watanabe, T. Otsuka, J. Tsuchiya, H. Harada, J. Yamamoto, H. Nishi, T. Kudoh and H. Amano: Performance Evaluation of RHiNET-2/NI: A Network Interface for Distributed Parallel Computing Systems, *Proceedings of International Symposium on Cluster Computing and the Grid*, pp. 318–325 (2003).
8) 大塚, 渡邊, 北村, 原田, 山本, 西, 工藤, 天野: 分散並列処理用ネットワーク RHiNET-2 の性能評価, 先進的計算基盤システムシンポジウム SACSIS, pp. 45–52 (2003).
9) S.Nishimura, T.Kudoh, H.Nishi, J., K.Harasawa, N.Matsudaira, S.Akutsu, K.Tasho and H.Amano: High-speed network switch RHiNET-2/SW and its implementation with optical interconnections, *Hot Intercoonect*, pp. 31–38 (2000).
10) 鯉渕, 渡邊, 河野, 上樂, 天野: RHiNET-2 クラスタを用いたルーティングアルゴリズムの実機評価, 電子情報通信学会技術研究報告 CPSY-2003-13, pp. 43–48 (2003).