2003－HPC－95　(10)
2003／8／4

高バンド幅/耐故障性を持つクラスタ向けネットワークRI2Nの性能評価

三浦信一[†]　朴泰祐[††]
佐藤三久[††]　高橋大介[††]

コモディティネットワークであるEthernetを複数並列で用いることで，PCクラスタ向けに高バンド幅/耐故障性のあるネットワークを提供するRI2Nを提案し開発中である．RI2Nでは，並列リンクを正常時にはバンド幅向上に用い，障害発生時にはそれに対する代替リンクとして用いることで障害に対処する．今までの実装では耐故障性は実装できておらず，またGigabit Ethernetを使用した場合，性能が低いという問題点があった．今回の実装は耐故障性のための予備実装とGigabit Ethernetの性能向上を目的としている．本稿ではその実装及び現状での問題点を整理する．本システムを用いると，4本のFast Ethernetで最大約42MB/sec，2本のGigabit Ethernetで最大約200MB/secのバンド幅が得られた．

Performance evalusation of RI2N - Interconnection network system for clusters with wide-bandwidth and fault-tolerancy

Shin'ichi Miura,[†] Taisuke Boku ,[††] Mitsuhisa Sato[††]
and Daisuke Takahashi[††]

In this paper, we propose an interconnection network system named RI2N for clusters based on parallel links with commodity Ethernet which provides both wide-bandwidth and fault tolerancy. In RI2N, multiple links are used to enhance the total bandwidth in ordinary mode or redundant connection in failure mode. The former implementation lacked fault tolerance and did not take full advantage of Gigabit Ethernet. The purposes of our new implementation are pilot implementation of fault tolerance and improvement of the performance of Gigabit Ethernet. We describe the way of the implementation and the current problems. In current prototype implementation, it achieves the maximum bandwidth of 42MB/sec with four Fast Ethernet links and 200MB/sec with two Gigabit Ethernet links.

1. はじめに

現在，ハイパフォーマンスコンピューティング(HPC)の分野では大型計算機に代わるシステムとしてPCクラスタが多く用いられている．これらのシステムは近年のCPU処理能力の向上に伴いノード単体の性能は大幅に向上している．その反面，従来のMPPのような並列計算機専用ネットワークの場合と異なり，強力なエラー検出・回避機能やハードウェア自体の高信頼性，パケット集中に対するスイッチのタフネス等が低い場合が多い．この問題を解決するためにSAN:System Area Networkと呼ばれる高速なネットワークが開発・提供されている．特にMyrinet[1)]やInfiniband[2)]などは，クラスタに対して高速なネットワーク環境を提供する．しかしこれらのネットワークデバイスは高価であり対価格性能比がノード単体に比べて低い．このため，最も多く用いられてきたネットワークは，汎用ネットワークであるEthernetである．コモディティ製品であるEthernetは低価格化と高速化が進んでいるが，クラスタ向けの利用にはいまだにバンド幅・耐故障性ともに貧弱である．

我々はこれらの問題を解決するために，Ethernetを用い，安価に高バンド幅・対故障性を実現するRI2N:Redundant Interconnection with Inexpensive Networkを提案開発してきた[9)]．本稿では，問題であったGigabit Ethernetの処理能力に耐えるようRI2Nを改良し，また耐故障性の予備実装を行なう．その後，この実装によって得られる性能の評価を行なう．

† 筑波大学 大学院 理工学研究科
Graduate School of Science and Engineering, University of Tsukuba
†† 筑波大学 電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba

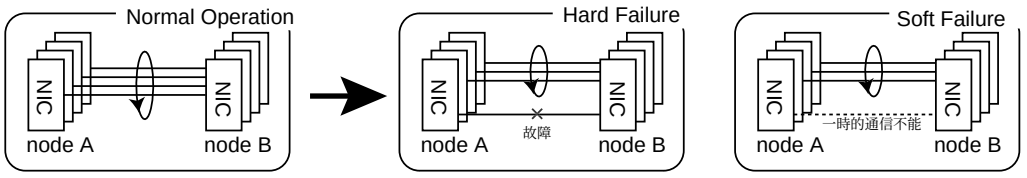

図 1 システムイメージ

2. RI2N: Redundant Interconnection with Inexpensive Network

クラスタのネットワークとして Ethernet を利用するときの問題点は，ハードウェアとソフトウェア (TCP/IP) の両方が汎用通信用に設計されていることである．このようなシステムでは NIC や Switch に急激な負荷が加わることを想定していない．そのため，NIC やスイッチに加わる負荷が大きいクラスタ環境では一時的なリンクの切断等が発生しやすい．

現在のコンピュータでは性能向上に加え耐故障性といった面からも，マルチストリームを多用する傾向がある．代表例としてあげられるのが HDD の冗長手段として一般的に用いられている RAID[8] (Redundant Array with Inexpensive Disk) システムである．RAID では，複数の HDD を同時にひとつの HDD と見立てて活用することにより，速度の向上と耐故障性を同時に提供している．我々はこの考えを基本に，クラスタ用ネットワークとしての特徴を加味し，高バンド幅と耐故障性を実現するネットワーク構築手段を提供することを考えている．このシステムを RI2N (Redundant Interconnection with Inexpensive Network) と呼ぶ．

RI2N では，RAID での複数 HDD に相当するものとして，NIC を複数束ね (以後，これを MP-NIC: Multiple-Port-NIC と呼ぶ)，高バンド幅と耐故障性を提供する (図 1)．最終的にはその環境をドライバレベルまたはプロトコルレベルで提供することで，低遅延を実現することを目指す．

過去にも，高バンド幅化には trunk 技術[3]，また耐故障性には MPI などの API レベルでの checkpoint/recover[4)5)6)7)] 等の手法が提案・実用化されてきたが，これらを同時に実現できたものはなかった．RI2N ではこれらの問題をネットワークの API・プロトコルレベルによって同時に解消することを目標にする．

3. 現在までの実装

現在までの RI2N の実装内容について整理する[9]．

3.1 実装レベル

我々の最終的な開発目標は RI2N をドライバレベルで提供することであるが，プロトタイプの実装は，

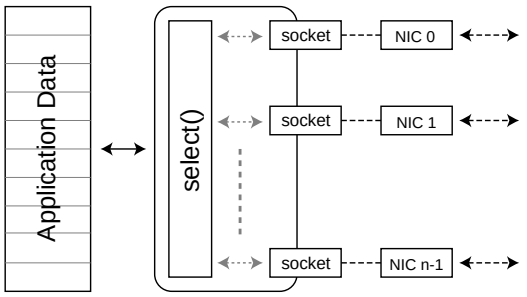

図 2 データ送受信イメージ図

ユーザレベルの実装となっている．内部では既存の TCP/IP のプロトコルと API が基本となっているので，Multi-socket, Multi-thread 等を用いて並列ストリームを提供する．これにより開発の容易さと高い移植性を持つが，既存の single-stream 用プロトコルや API が基本になるため，現在の RI2N はこの性能に大きく左右され，システムレベルでの処理の機能に重複が生じる．ユーザへの機能提供は従来の TCP/IP 通信からの移植を容易にするため，TCP/IP と互換性のある API を用意している．

3.2 データ送受信手法

データの送受信には，過去の評価結果[9] により，Select システムコールを用いて NIC の状態を常に監視し，各 NIC を用いて send/recv をする手法 (図 2) を選択している．Select システムコールの利用は CPU に高い負荷を与える可能性があるが，CPU と Network の処理能力が広がっている昨今では，トータルの能力が向上する見込みが高い．

3.3 RI2N パケットの構成

RI2N では複数の NIC を用いて送信するために，アプリケーションから渡されたデータを分割する．この分割されたデータ (chunk) にヘッダを付加したものが，RI2N のパケットとなる．このヘッダには受信側で正しくパケットを再構成するためのシーケンス番号等の情報が含まれている．今回の実装では送受信の効率化のため RI2N パケット長を固定としている．

4. 拡　　張

今までの RI2N の実装[9] およびその予備評価では，

高バンド幅化機能のみの実装となっていた．その結果では，Fast Ethernet(100Base-TX) を同時に 4port 用いると約 40MB/sec の性能を得ることができた．しかし，GbE(Gigabit Ethernet,1000Base-T) を同時に 2port 使用した結果では満足した結果を得ることができなかった．また，耐故障性についてはアプローチの紹介にとどまり実装はできていなかった．今回は，この 2 点についてチューニングおよび拡張を行なう．

4.1 Gigabit Ethernet のチューニング

これまでの実装では，RI2N パケット構成・再構成とその送受信の流れでは，ユーザから渡されたデータをすべてパケットの形に処理した後に通信を開始していた．そのために，通信処理に伴う CPU の処理時間が隠蔽されず，パケットの構成・再構成の時間がオーバヘッドとして現れる問題点があった．Fast Ethernet では，RI2N の全体処理時間に占める通信時間の割合が大きいため，この問題は現れなかった．しかし，GbE の通信能力は Fast Ethernet にくらべ極端に大きく，パケットの構成・再構成が占める時間が浮き彫りになってしまった．このため GbE を用いた結果では満足できる結果が得られなかった．

この問題を解消するために，通信処理と並行して RI2N のパケットを構成・再構成するように変更を行なった．これによって，パケットの処理時間を通信処理時間で隠蔽できる．

4.2 耐故障性実装

耐故障実装について，我々は 2 つのアプローチを提案した．1 つ目は**バッファリング アプローチ**である．これは送受信双方で一定サイズのバッファ(ウインドウ) を用意し，リンクの failure 発生時には，他のリンクで再送を行なう．2 つ目は **RAID5 アプローチ**である．送信時に常に冗長データを含め送信を行なうことで，故障の検知に伴う再送要求が減る．そのため，ネットワークの故障時でも極端な速度低下は発生しない．しかしコンスタントにオーバヘッドが発生するため，通常の通信時においては高バンド幅の確保が難しくなる．

以上 2 つを検討した結果，我々は以下の方針に沿って実装をすることにした．

(1) 通常時にオーバヘッドが少ない，バッファリングアプローチを実装する．
(2) failure 発生時に送信に失敗したパケットの再送コストは我慢する．
(3) 継続的な failure にはストリームを送信段階から failure した方で通信をしないようにすることでオーバヘッドを削減する．

現在この方針に従って実装を進めている．

現在実現できている機能は，混雑/故障 (ただし socket 自体は正常) によって処理能力が低下しているネットワークをなるべく使用せず，正常なネットワークを優先的に使用する．現状にデータの再送機能を組み込むことで，単純な耐故障が実現できる．これについては後に述べる．

表 1 実験環境

PC	DELL PowerEdge 1600SC 2 台 (Pentium4 Xeon 2.8GHz 2-way SMP)
NIC	Adaptec Quartet 64 (100base-TX Ethernet × 4 ports) Intel PRO/1000MT (1000base-T Ethernet × 2 ports)
Switch	PLANEX FMX-0248K (Layer2) (48 Port Fast Ethernet Switch) PLANEX FMG-24K (Layer2) (24 Port Gigabit Ethernet Switch)
OS	Linux 2.4.20

5. 性 能 評 価

現在の RI2N では RI2N パケットの長さ (chunk size) は固定としている．そのためこのパケット長が RI2N の性能をほぼ決定する．大きなパケット長を用いると，連続したデータ転送が行なえるため本来の Ethernet の持つ性能を十分に発揮できる．また管理用ヘッダ等のオーバヘッドの低減も期待できる．しかし一方では，障害発生時にデータ損失のサイズが大きくなるために再送のペナルティが大きくなる．これに加えてアプリケーションでのデータ送受信サイズが小さいときは，並列ストリームの効果が発揮できないばかりか，パケット長を固定にしているため無駄に大きなデータを送ってしまうことになり転送効率は極端に落ちる．パケット長を短くすると障害時における適応性や動的な負荷分散という点は良くなるが，ヘッダがペイロードと比較して大きくなるためにこれがオーバヘッドとなる．また，必然と send()/recv() のシステムコールの回数が多くなり CPU の処理能力を必要とする．評価では，まず基本となる Fast Ethernet の性能評価を行ない．次に，いままで速度向上が得られなかった GbE での評価を示す．

5.1 実 験 環 境

今回用いた実験の環境について，表 1 に示す．

接続は 2 台の PC 間にそれぞれの種類に応じた 1 台の Ethernet switch を挟み，MP-NIC の port 数に合わせた複数の Ethernet ケーブルを接続することで並列リンクを構成して用いる．実際の使用時には Network の経路の故障も考慮して switch も多重化する必要があるが，今回は機材の都合上 Fast Ethernet,GbE それぞれ一台用いた．Fast Ethernet を 4 port 同時に用いると理論最大性能は 50MB/sec になる．GbE の場合は 2 port の同時利用で理論最大性能は 250MB/sec となる．

5.2 Fast Ethernet

Fast Ethernet において port 数を 4 に固定し，

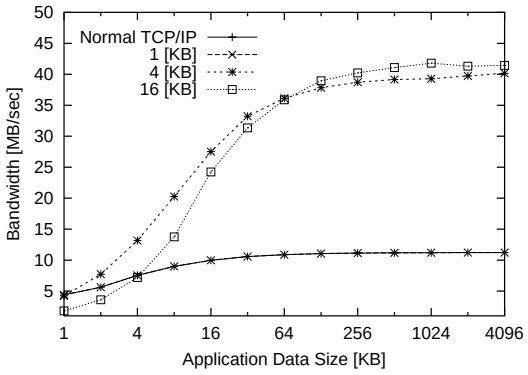

図 3 実験結果 Fast Ethernet (100Mbps) 4 Port 同時利用

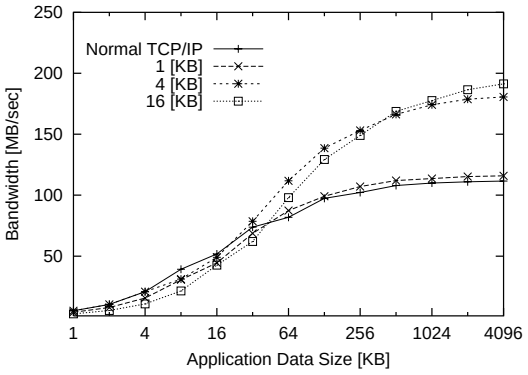

図 5 実験結果 Gigabit Ethernet (1000Mbps)

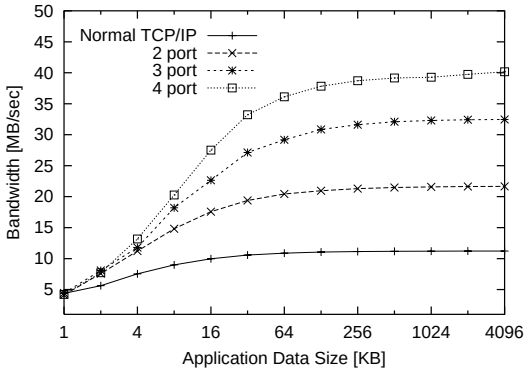

図 4 実験結果 Fast Ethernet (100Mbps) Port 数の変更による通信性能の変化

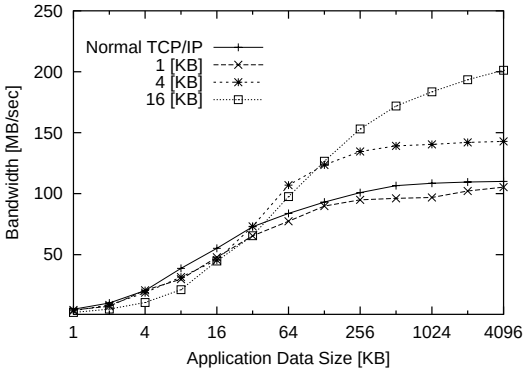

図 6 実験結果 Gigabit Ethernet (1000Mbps) Socket Buffer のサイズ変更による効果 (640KB)

chunk size を 1〜16KB に変化させた場合の評価結果を図 3 に，また chunk size=1KB で port 数を 2〜4 と変化させたときの評価結果を図 4 に示す．実測には 2 点間で ping-pong 通信を行ないその転送に基づく実効バンド幅を計測している．参考例として，標準的な TCP/IP の API を用いて計測を行なったデータ (Normal TCP/IP) も同一グラフ上に示している．

この実験では chunk size = 16KB の時に最大で約 42MB/sec の性能が得られた．このシステムでの理論最大性能が 50MB/sec であるのでピーク性能の約 84%となる．しかし図に示すように標準の TCP/IP の測定結果では 1 port あたり 11MB/sec の帯域が得られているため，この状態と比較すると最大で 95%程度の性能を提供している．

最大性能は chunk size=16KB の時であったが，アプリケーションデータ長が短い場合や故障時のリカバリー・コストを考えると chunk size=1KB の方が全体的な性能が良い．

5.3 Gigabit Ethernet

RI2N で GbE を 2port 同時に用いて chunk size を 1〜16KB に変化させた時の評価結果を図 5 に示す．最大性能は chunk size=16KB の時で約 191MB/sec となっている．

Fast Ethernet の結果と比較すると，性能を引き出すためには chunk size を 4KB より大きくしなくてはいけない．これは GbE の送信可能間隔に対して，システムコールの処理時間が長く，それに加えて RI2N のパケットの構成・再構成のオーバヘッドが加算されている為と考えられる．そのため，Fast Ethernet では chunk size が 1KB のときが最適であったが，GbE では 16KB と大きくなった．このためデータサイズが小さいときは並列転送のメリットが生かせず，その場合では結果的に chunk size が小さいほうが性能が良い．

Socket Buffer の設定

この結果ではリンク 2 本の GbE の最大性能が 250MB/sec であることを考えると 2port 使用時には 76%程度の性能しか出ていない．ただし，実際に標準の TCP/IP で計測したスループットである 111MB/sec を元に計算を行なうと 87%となる．しかしこれでも性能は低い．そこで TCP の Socket Buffer のサイズを変更した上で再度実験を試みた．実験の結果を図 6 に示す． Socket Buffer は 640KB に設定した．最大性能は chunk size が 16KB の時 200MB/sec となった．

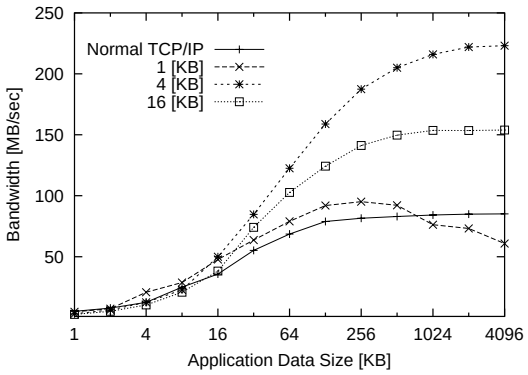

図 7 実験結果 Gigabit Ethernet (1000Mbps) JumboFrame を用いた通信性能 MTU=15000B

前述の結果と比較をすると，最大のパフォーマンスが若干ながら向上したのみであり，全体的なパフォーマンスに大差は無い．そのため TCP の Socket Buffer のサイズはこのシステムにはあまり影響を与えないと言える．Socket Buffer のサイズが性能に影響を与えない原因として，RI2N のパケットは chunk size が 1KB 以上とすでに大きい単位サイズで，send()/recv() が行なわれる．このように大きな単位の送受信では，データを普段から連続に送ることができるためにバッファの効果は少なくなると考えられる．

JumboFrame の使用

単純な Socket Buffer のサイズ変更では GbE では根本的な速度向上を得ることができなかった．我々はこの問題を Ethernet の MTU のサイズに問題があると考えた．そこで GbE の処理能力をファームウェアのレベルで底上げする試みとして，**JumboFrame** を用いた実験を行なった．通常 Ethernet では MTU の最大サイズは 1500B であるために，一度に大きなデータを処理できる GbE にはこれが大きな足かせとなっていることが分かっている．そこで JumboFrame では，この MTU のサイズを大きくすることで転送効率の上げる．この JumboFrame を用いた実験では機材の関係上ノード間を Ethernet Switch を用いず，クロスケーブルで接続をして計測を行なった．評価の結果を図 7 に示す． この結果によると，最大性能は chunk size = 4KB の時 225MB/sec となっている．

RI2N で JumboFrame を用いた評価では，既存の MTU=1500B の状態よりも良いパフォーマンスを得ることができた．しかしながら JumboFrame の利用には，これに対応する switch が高価で限られる問題がある．RI2N の目的は安価に高バンド幅と耐故障性を提供することにあり，JumboFrame を使用することは，その目的から外れる可能性がある．我々はこれを踏まえた上で JumboFrame を用いずとも性能を向上させる方法を考えなければいけない．

表 2 Round Trip Time

Fast Ethernet

Normal TCP/IP	2 port	3 port	4 port
88 μsec	432 μsec	440 μsec	462 μsec

Gigabit Ethernet

Normal TCP/IP	2 port
375 μsec	500 μsec

5.4 通信遅延時間

通信の遅延時間は HPC にとって重要なパラメータの 1 つである．計測の目的は，RI2N を用いた際の絶対的なオーバヘッドを計測することである．今回は 1KB データの Round Trip Time を通信遅延時間の代わりとして計測した．RI2N を用いた Round Trip Time と通常の TCP/IP を用いた結果も併せて表 2 に示す．chunk size は Fast Ethernet, GbE ともに 1KB としている．通信の遅延時間が重要になるのは，バリア同期などの 1KB 以下の比較的転送量の小さいデータの時である．そのときデータサイズか小さいため並列転送効果はほとんどなく，オーバヘッドのみが現れる．

標準的な TCP/IP を用いた結果よりも極端に性能が悪い要因として，chunk size が固定であることが挙げられる．このため，データサイズが小さくとも無駄にデータを送るため性能が低下する．これを回避するためにある程度柔軟に chunk size を変更できるようにしなければいけない．もう 1 つの要因は，RI2N がユーザレベルでの実装であるという点である．RI2N 中では Select システムコールを用いるため，select() とそれに付随する処理時間がオーバヘッドとなる．

Fast Etherne の結果は GbE のそれよりも遥かに小さい．今回用いている Fast Ethernet の NIC は特別であり基板上の ASIC チップによって TCP の基本的操作をハードウェア的に行なえる．そのためこのような差が出てしまったと考えられる．

6. 今後の実装

性能向上と耐故障性のためにさまざまな実装を試みている．現在の取組みについて述べる．

6.1 Multi-thread を用いた実装

我々は MP-NIC に対する並列ストリーム生成方法として Multi-thread を用いる方法と Select システムコールを用いる 2 種類の方法を提案していた．現在の実装では Multi-thread の問題点と過去の評価[9] により，Select システムコールを用いる手法を選択している．しかし，今後の耐故障性の実装にはより多くの CPU 資源を使う．特に GbE を用いたシステムではデータの処理能力がネットワークのそれに対して相対的に弱くなる．また現在の実装ではブロッキング通信のみがサポートされているが，一般的に HPC 分野では通信時間の隠蔽のために非ブロッキング通信が

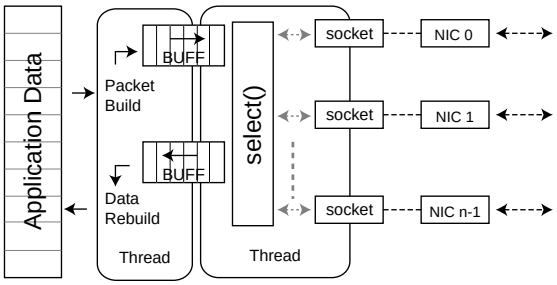

図 8 Multi-thread を用いた新しい実装イメージ

用いられる．そのため非ブロッキング通信のサポートは RI2N の絶対条件ではあるが，Multi-thread を用いない現在の実装ではこのサポートは難しい．そこで並列ストリームの形成に Multi-thread を用いるのではなく，データの送受信と送受信データの構成にのみ Multi-thread を用いる実装を現在進めている．これはデータバッファの構成・再構成の処理を送受信部から分離し，いわゆる producer-condurmer 関係の thread で処理を行なうものである (図 8)． データの送受信部を単一のスレッドがすべて管理することで，ネットワークの状態がすべての送受信で共有でき柔軟な対応も可能になる．

Multi-thread の利用は SMP マシンでのみ効力を発揮するが，シングル CPU 構成のマシンでは thread の切替によるオーバヘッドが付加される欠点はある．しかしながら現在の HPC 向けクラスタは SMP 構成をとることが多く，また Hyper-threading 技術の普及によってシングル CPU 構成でも効率的な Multi-thread の利用が可能になっている．そのため今後は Multi-thread を用いた実装へと発展させる．

6.2 耐故障性の実装

現在の実装は NIC の状態を絶えず監視し，送受信が可能になった NIC より優先的に RI2N パケットを送受信する．この機能をネットワーク切断というアクシデントに対応するためには，すでに send() で TCP/IP バッファに格納された，またはネットワークの途中で消失した RI2N パケットを再送する必要がある．RI2N での send 側ではこの消失したパケットについては捕捉することは難しい．そこで recv 側で send 側で付加されたシーケンス番号をもとに，すでに正しく再構成できるシーケンス番号から一定数の離れた RI2N パケットを受信した時点で再送要求を出し，受信した send 側で該当するシーケンス番号のパケットを再送する実装を準備している．

7. 終わりに

本稿ではコモディティEthernet を用いた冗長/高バンド幅ネットワーク RI2N の概要と評価実験の結果および今後の実装について述べた．今後は，RI2N を MPICH 等に適用したうえで，ベンチマーキングを行ない，アプリケーションの特性及び各種条件に対する性能を評価する．また PVFS といった並列ファイルシステムへのアクセスなどに応用して，これらの特徴を踏まえた最適な API の用意を行なう．さらに，クラスタ管理システムなどのシステムとの連携について検討していく予定である．

謝　　辞

本研究を行なうにあたり，貴重な助言・提言を頂いた CREST「メガスケールクラスタ研究チーム」のメンバーに深く感謝します．本研究の一部は科学技術振興事業団「戦略的創造研究推進事業 (CREST) — 情報社会を支える新しい高性能情報処理技術 — 『超低電力技術によるディペンダブルメガスケールコンピューティング』」および文部科学省 科学研究費補助 (基盤研究 (C),12680327) による．

参 考 文 献

1) Myricom, INC. *http://www.myri.com/myrinet/* .
2) InfiniBand Trade Association. *http://www.infinibandta.org/* .
3) H. Tezuka, et.al., "PM: An Operating System Coordinated High Performance Communication Library", High Performance Computing and Networking, LNCS Vol.1225, pp. 708-717, April 1997.
4) E. Elnozahy, et.al., "A survey of rollback-recovery protocols in message-passing systems", Technical Report CMU-CS-96-181, Carnegie Mellon University, October 1996.
5) G. Fagg, a Dongarra. "FT-MPI:FaultTolerant MPI,Supporting Dynamic Applications in a Dynamic World" Euro PVM/MPIUser's Group Meeting 2000 ,Springer-Verilag. pp. 346-353. 2000.
6) George Bosilca, et.al., "MPICH-V: Toward a Scalable Fault Tolerant MPI for Volatile Nodes", ACM/IEEE ISC2002, 2002.
7) 高宮 安仁, 他, "ユーザー透過な耐故障製を実現する MPI へ向けて", JSPP2002 論文集, pp. 217-224, 2002.
8) Peter M. Chen, et.al., "RAID : High-Performance, Reliable Secondary Storage", *ACM Computing Surveys,* volume 26(2), pp. 145-185. 1994.
9) 三浦 信一, 他, "ユーザレベルでのマルチリンク利用による高バンド幅/耐故障性をもつクラスタ向け結合ネットワーク RI2N", 情報処理学会研究報告 2003-HPC-93, pp. 13-18.