Vol.2013-ARC-207 No.10
Vol.2013-HPC-142 No.10
2013/12/16

# 分散ストレージシステムに対する低オーバヘッド冗長化書き込み手法の提案と評価

大辻 弘貴[1,3]　建部 修見[2,3]

**概要**：高信頼かつ高性能な分散ファイルシステムは，大規模データ処理の実現にあたって不可欠な存在である．分散ファイルシステムはネットワークを用いることでデータ共有を可能にしているが，同時にこの点はボトルネックともなりうる．したがって，高性能なネットワークを用いると同時にそれらを活用することは，今後のシステムに求められている重要な要素である．また，データの大容量化に伴い，効率的に信頼性を確保する手法も求められている．本稿においては，InfiniBand を用いたゼロコピーの通信ライブラリの開発と，それを用いた効率的な冗長記録の実装法および評価について述べる．この評価により，単純に冗長データを記録する場合と較べて大幅な性能向上を実現した．

## 1. 序論

分散ファイルシステムが取り扱うデータ量は数年以内にエクサバイトに到達すると見込まれている．特にビッグデータと呼ばれる領域やデータインテンシブコンピューティングの発展は，この傾向を後押ししている．このようなデータは，既存のシステムで処理した場合には多大な時間を要することから，高速なシステムの開発が求められている．大容量のデータを取り扱うためには，分散ファイルシステムが用いられることが多い．これは，ファイルという枠組みを維持しつつも，ネットワークによって共有を可能としているものである．しかしながら，ネットワークはコンピュータ内部で用いられている通信機構を比較して，一般的にボトルネックとなる可能性が高い．したがって，ネットワーク経由のデータアクセスを高速化することは，システム全体の性能向上を目指す上で，極めて重要である．

データアクセスの性能を高めると同時に，信頼性の確保も重要な課題である．巨大なシステムにおいては，常に何らかの故障が生じている可能性が高い．したがって，単一の故障によってデータが失われたり，サービスが中断されたりといった事がないよう，対策を講じる必要がある．このような要求に応じるため，従来はファイル複製 [1] が行われてきた．しかし，従前のようにデータ量は増加しているため，すべてのファイルを複製すると，元のデータ量の数倍の領域を必要とすることになり，効率が悪い．

そこで，本稿では冗長記録を取り入れることとした．冗長記録を用いると，複数の故障に対して 2 倍以下のデータ量でも対応できるため，複製と比較すると非常にストレージ消費量の観点からは効率が良い．しかしながら，最終的に記録されるデータ量は，単一の書き込みよりも増加することは避けられず，単純に冗長化したデータの書き込みを行う場合，ネットワークトラフィックの増大や輻輳を招き，性能低下要因となる可能性がある．本研究においては転送順序を効率化することにより，冗長データの書き込みに伴うネットワークトラフィック集中の回避を実現した．この転送方法は，ストレージノードをデータ処理や中継に用いるため，ネットワーク転送に伴うオーバヘッドの削減が鍵となる．本実装においては，InfiniBand が備える RDMA(Remote Direct Memory Access) を効果的に使用し，メモリコピー回数の最小化や低レイテンシ化に努めた．以降の章では冗長記録の方法およびこれらの効率的な書き込み手法について説明し，その実装予備評価を示す．

## 2. 関連研究

本研究はネットワークで接続されたストレージシステムを対象としており，多くの既存例が存在する．例えば，分散ストレージ・ファイルシステムの例としては，Gfarm[2] ファイルシステムや NFS[3]，Lustre[4]，PVFS[5] などがある．その中でも NFS は，NFS over RDMA[6] として，

[1] 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
[2] 筑波大学システム情報系
Faculty of Engineering, Information and Systems, University of Tsukuba
[3] 独立行政法人科学技術振興機構 CREST
JST CREST

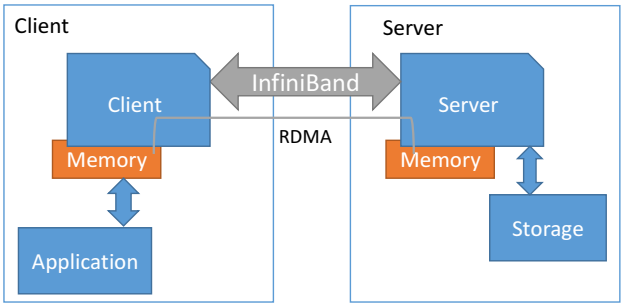



図 1　システムの構成 [11] より引用

RDMA を活用した実装も存在している．また，他のファイルシステムにおいても PVFS over RDMA[7] などの例がある．これらのファイルシステムは，信頼性やアクセス性能向上のために，ファイル複製機能を備えていることがある．ファイル複製を効率的に行う手法を取り入れたものとしては，Ceph[8] がある．Ceph はストレージノード間でデータ転送を工夫し，複製作成にあたる転送時間を短縮する試みがなされている．

冗長記録に関連したものとしては，HDFS RAID がネットワーク経由での冗長記録を行っているほか，Microsoft Azure[9] ストレージが，故障時の冗長記録の再構築を効率化する手法を取り入れている．

以上のように複製作成や冗長記録再構築の高速化・効率化に関する研究を示したが，本研究はそれらとは異なり，冗長データの書き込み時を対象としたものである．

## 3. 分散ファイルシステムに対するネットワークアクセス

### 3.1 概要

分散ファイルシステムはネットワークを用いて構築されているため，それらにアクセスするためにはネットワーク経由のデータアクセスが不可欠となる．本研究においては，高い性能を持ち，RDMA 機能を提供する InfiniBand を利用している．RDMA を活用したネットワークファイルアクセスについては，これまで筆者らが発表 [10] している．本節ではその概要のみを示す．

### 3.2 RDMA によるアクセス

RDMA の活用により，ネットワークを介してリモートコンピュータのメモリを直接読み書きすることが可能になる．この機能は InfiniBand のネットワークアダプタ (HCA - Host Channel Adapter) によりハードウェアで提供されており，CPU の関与を最低限にできるメリットがある．Ethernet などでソケットを用いて通信した場合，ソフトウェアが中心のネットワークスタックがボトルネックとなり，数十$\mu$秒のレイテンシとなるが，InfiniBand 上での RDMA ではそれを数$\mu$秒まで短縮できる．尚，InfiniBand は通常のネットワークインタフェースとして使えるように

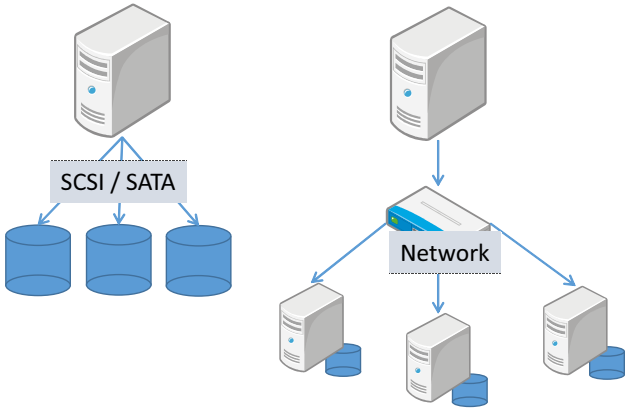



図 2　RAID(左) とネットワーク経由の記録 (右)　[11] より引用

する IPoIB (IP over IB) を提供しているが，本研究では性能を最大限引き出すために，Verbs API を利用している．これは，アダプタの持つ機能とほぼ 1 体 1 で対応した API 群であり，アドレスを指定したメモリ転送など原始的な命令を直接発行できる．一方で，一般的なネットワークとは使い方が大きく異なるため，転送方法などについては最適な形となるよう検討する必要がある．このネットワークを最終的にはファイルアクセスに利用する．これは**図 1** に示す構成となる．ローカルストレージに対するアクセスは基本的に DMA であり，ネットワーク転送も同様であることから，全体を通じて CPU バイパスとなっている．

## 4. 冗長記録の手法

### 4.1 概要

冗長記録とは，元のデータに対して符号を付加し，記録の一部が消失または変化した際に復元できるようにするものである．RAID などが典型的な例として挙げられるが，これはあくまで 1 台のコンピュータ内でのデータの取扱であり，本研究ではネットワーク経由での冗長データ記録をターゲットとしている．**図 2** に示す図中の右側が本研究の対象構成である．

RAID では，記憶装置はそれぞれ単なるディスク等であることから，転送方法にそれほど工夫の余地はない．一方で，ネットワークを利用した冗長記録の場合，それぞれはコンピュータであるため，受け取ったデータを元に演算して別のノードへ転送するなど，多くの工夫の余地がある．本研究はこの利点を活用するものである．

### 4.2 データの格納方法

冗長記録にはいくつかの方法が存在する．典型的には，xor を用いた方法があり，本稿でも採用している．一般的にはあるデータを複数個に分割し，それらをビットごとに xor 演算し，その結果を別の場所に保存する手法がある．このように記録すると，任意の 1 台の記憶装置の故障に対応できる．しかし，1 台が故障すると単一記憶装置よりも

Vol.2013-ARC-207 No.10
Vol.2013-HPC-142 No.10
2013/12/16

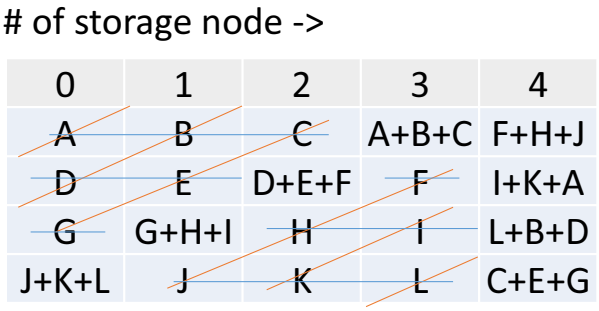



図 3　2-d xor の記録方法

可用性の低下した状態となるため実際の運用面では好ましくなく，2 台以上の故障に耐えられる方式が好ましい．ここでは，2-d xor と呼ばれる記憶方法を採用することにした．この記録方法を図 **3** に示す．ここではストレージノードは 5 台存在し，そのうち 1 台は冗長記録の保管専用である．A〜L は元のデータをあるブロック数で分割したものであり，最後の 1 台を除いた全てに分散して記録される．その際，他の横並びで保管されたデータをすべて xor したものを他の 1 台の記録するようにし，いずれのノードも 1 つ冗長記録を持つようにする．また，ストレージノードをまたいで斜め方向に生成した xor の結果を，冗長記録専用のノードに保管する．

このようにすることで，例えばノード 1 と 2 が故障した場合には，B を D と L，L+B+D のデータから復元，C を B と A+B+C から復元といったように，元のデータを復旧することができる．

この性質はいくつかのパターンに分けて容易に示せる．ノード数 n 台としたとき，末尾の冗長記録専用ノード (n-1) が故障した状態は，単純 xor の状態と同じであるから更に任意の 1 台の故障に対応できる．それ以外のケースについて，復旧が不可能な状況は横方向と斜め方向の xor のうち，いずれも復旧できない (2 個以上が故障している) 状態であるが，これは 2 台の故障で起こすことは出来ない (横と斜めで xor の取り方が必ず 1 つずれており，一方を復旧不可能な状態にすると一方が残る)．

本稿では，このデータ格納方法を対象として，書き込みの最適化手法について示す．

### 4.3　xor 演算のスループット

冗長記録の生成にあたり，排他論理和（以下 xor）演算はその中心である．従って，xor 演算の速度が十分に確保されている必要がある．この評価は第 137 回 HPC 研究会 [11] にて発表した通りである．実測で 1 コア 1 スレッドにおいても，約 6GB/s の xor 演算が可能であったことから，本研究における冗長記録については十分な性能が確保できると見込まれる．

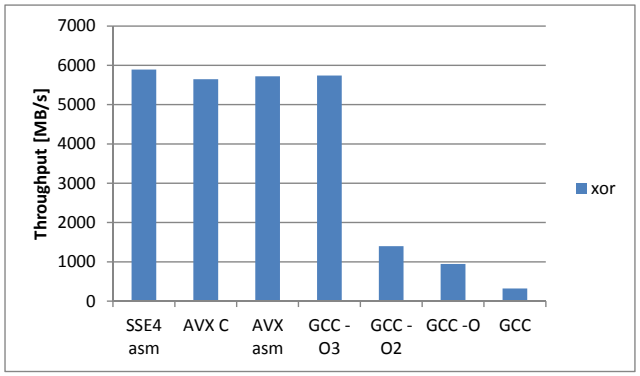



図 4　xor 演算のスループット [11] より引用

## 5.　冗長記録の最適化手法

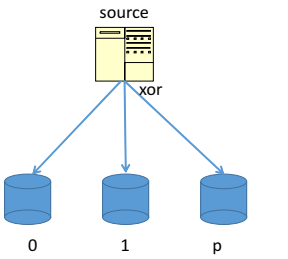
### 5.1　基本的な方針

本稿では前述のとおり，xor 演算によって冗長記録を作り出す．したがって，元のデータと xor 演算の結果を格納する際の性能が重要である．最初に，最も単純なケースについて考える．図 **5** はクライアントノード (上・黄色) がストレージノード (青) に書き込みを行う状況を示したものである．下にある表は，各ノードごとに発生するトラフィックを，1 を最大回線帯域として示したものである．

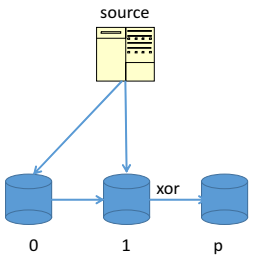
図中左のようにある 1 台のクライアントが 3 台のストレージノードに対して書き込みを行う例を考える．この場合，クライアントは元のデータを分割してストレージノード (青)0 番と 1 番に分散して書き込みを行い，この 2 つのデータを xor 演算したものを p のノードに書き込む．この書き込みにおいて，クライアントは元のデータの 1.5 倍のデータ量を送信しなければならない．したがって，書き込み速度は 2/3 に低下してしまう．

一方で，図中右のように転送順序を変更することを考える．この場合，クライアントはストレージノード 0 と 1 のみに元データを書き込む．そして，ストレージノード 0 は，受け取ったデータを保存しつつ隣のストレージノード 1 にデータをそのまま転送する．次に，ストレージノード 1 は 0 から受け取ったデータとクライアントから受け取ったデータについて xor 演算を行い，結果をノード p に転送する．このようにすると，クライアントから送信されるのは元データだけであり，冗長記録を行うことによる性能低下を回避できる．また，最終的に各ストレージノードに格納されるデータも全く同一の結果になる．

以上に示すアイディアが本提案の基本となっており，転送順序の変更や xor 演算を行うノードを移動することで，転送ボトルネックを最低限にすることを目標としている．前述のように，これはネットワークを利用した冗長書き込みであるから可能な手法である．




| | client | 0 | 1 | P |
|---|---|---|---|---|
| in | 0 | 1/3 | 1/3 | 1/3 |
| out | **1** | 0 | 0 | 0 |




| | client | 0 | 1 | P |
|---|---|---|---|---|
| in | 0 | 1/2 | 1/2 | 1/2 |
| out | 1 | 1/2 | 1/2 | 0 |

図 5　xor 計算と転送の最適化 (左：通常　右：通信量削減)

### 5.2　2-D xor への応用

前節で示した内容について，2-D xor への応用を考える．例えば，4 台のストレージノードと 1 台の冗長記録専用ノードが存在する場合，クライアントがすべてのデータを計算する単純な方法においては，データ量の増加分の逆数の性能 (64%) になり，スループットが低下する．

**図 6** は 2-D xor において転送量の削減を行った例で，1 台のクライアントから 5 台のストレージノード (うち 1 台は冗長記録専用) としてデータを記録する構成となっている（記録レイアウトは図 3 と同じである）．図のうち縦 5 本の線は，ストレージノードを表してる．矢印はノード間のデータ転送で，近くには対応するブロックが書かれている．図中最も下に書かれている数字は，それぞれのノードに何本の転送路があるかを示しており，これはトラフィック量に比例する．この例では，矢印 1 本あたりの回線帯域は，クライアントノードからは 12 本の線が出ており，その合計は回線帯域と等しいことから，1/12 である．また，xor 演算を行うブロック数についても記している．

このように転送することで，クライアントからの送信データ量は元データと同じに保ったまま，冗長記録が行える．同時に，xor 演算を分散させる効果もあり，負荷集中によるボトルネックの回避が可能である．

尚これは一例であり，本転送方法の一般化及び定式化は現在取組中である．

## 6.　RDMA を活用した実装

### 6.1　概要

前章において述べた転送手法は，転送の中継や xor 演算などが多数存在することから，効率的にパイプライン処理を行う必要がある．本研究では，パイプライン処理のために RDMA を活用したネットワークライブラリを開発し，利用している．

### 6.2　ゼロコピー・パイプライン転送

RDMA では，直接メモリアドレスを指定して転送を行えることから，送受信者や処理のためのバッファを共有す

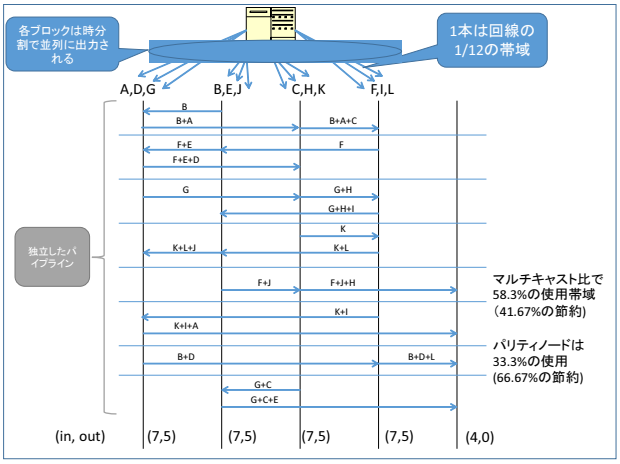



図 6　2-D xor の転送例

ることが可能になる．当然上書きや競合などの配慮はライブラリ側が行わなければならなくなるが，一般的なネットワークプログラミングでは不可避なカーネル・ユーザ空間相互や，アプリケーションとライブラリの間のメモリコピー削減できる．本実装においては，内部にリングバッファを用意し，これらはすべて InfiniBand HCA に register されている．この操作は pin-down とも呼ばれ，スワップアウトなどがされることのないようカーネルに通知すると共に，RDMA 可能な空間としてハードウェアに認識させるものである．この操作はオーバヘッドが大きいため，転送したい領域に対して随時行うことは効率的ではない．今回はパイプライン処理が目的であることから，一定領域の再利用がしやすいこともあり，ある連続領域を register し，そこをリングバッファとして使う方針を採用した．

パイプライン転送を行うにあたっては，2 つ以上のデータを必要とする処理を考慮する必要がある．例えば xor 演算は，各ノードでは 2 つのデータを元に行われる．したがって，前のノードが処理中のバッファを上書きすることのないよう配慮する実装を行った．各リングバッファには参照 tail と呼ばれるものを配置し，あるデータを必要とする処理は，処理が完了し次第この tail を前に進める．そして，RDMA 転送は，これら複数の tail のうちいずれも追い越すことの無いように行う．

### 6.3　片方向通信

RDMA は基本的に片方向通信であることから，読み書きされる側は特別な処理を必要としない．同時にこれは，データの到着などを受信側が感知できないということでもあり，対応が必要である．InfiniBand のリモートメモリ書き込みには，即値付きモードというものがあり，今回はこれを使用している．通常の書き込み要求であれば受信側は特にすべきことはないが，即値付きの場合は受信リクエストの領域を用意する必要がある．詳細は Verbs API 自体の解説になるため省略するが，パイプライン処理を行うため

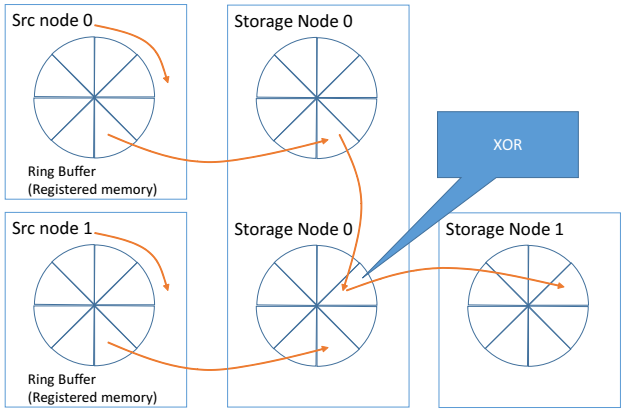



図 7　ゼロコピーパイプライン処理・転送

| | |
|---|---|
| CPU | Intel(R) Xeon(R) CPU E5-2665 x2 |
| RAM | 64GB |
| InfiniBand HCA | Mellanox MT27500 4x FDR (56Gbps) |

図 8　評価環境

には受信の把握が必要であり，受信した段階で処理および次のノードへの転送など行っている．

本ライブラリでは，基本的にポーリングにより各種転送命令の完了通知を回収している．Verbs API は基本的にノンブロッキングかつ非同期な API であることから，他に処理すべきことがないときにはポーリングを行うことで進行することとした．

### 6.4　予備評価

**図 9** は，この転送ライブラリを用いて，ノード間でデータをゼロコピーリレーした際のスループットを示している．評価環境は**図 8** に示す通りで，以後の評価についても同様である．

グラフ中 perftest は，InfiniBand のライブラリの一部として提供されている性能評価ツールによる実測最大性能であり，このライブラリが目指す転送性能である．オレンジの線は，2 台のノード間で対向した転送路を用意し，スループットを計測したものである．小さな転送サイズにおいては perftest よりも低いが，16KB ほどの転送ブロックではほぼ同等の性能を示している．グレーの線は 3 台のノード間でゼロコピーリレーを行った際のスループットである．対向 2 ノードよりは若干低下するものの，十分な性能を示していることが分かる．

## 7.　性能評価

### 7.1　条件

本提案の有効性を示すための性能評価を行うが，現在は単純な冗長記録について予備評価を行っている段階であり，本章ではその結果について示し，2D xor などの記録方式への応用を考察する．また，本評価ではローカルストレージへの書き込みは省略し，ストレージノードのメモリ

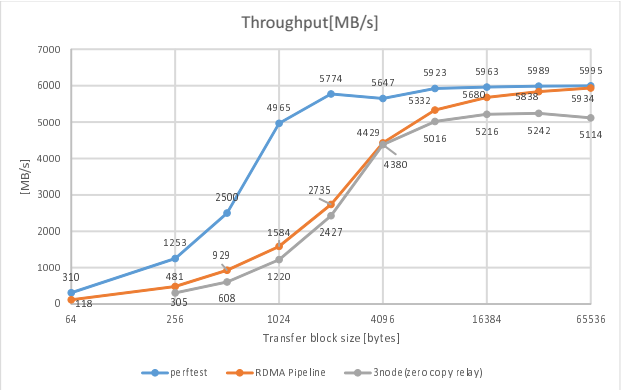



図 9　ゼロコピーパイプラインリレーのスループット

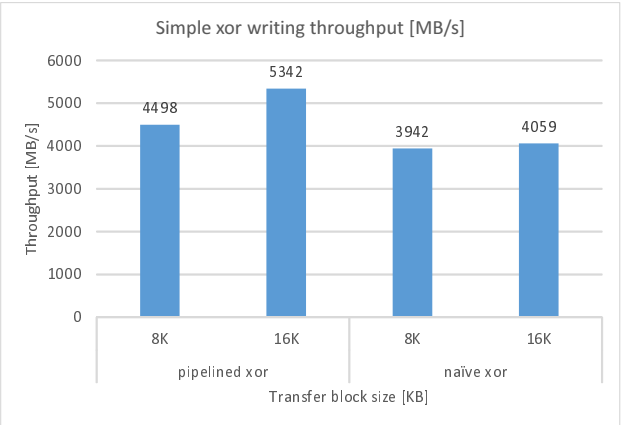



図 10　単純 xor 冗長記録の手法別スループット比較

にデータを書き込む段階まで行っている．

### 7.2　結果

**図 10** は，図 5 のそれぞれに該当する構成を，開発したネットワークライブラリにより構築してスループットを計測した結果である．グラフ中右側はナイーブな転送手法で，左側は最適化した転送手法による性能である．それぞれ，4KB と 8KB の転送ブロックサイズで性能を評価した．いずれの場合においても，最適化した転送手法による性能はナイーブな転送手法のものを上回っている．特に 16KB ブロックの転送においては 32%の高速化を達成しており，最適化の効果があることを示している．

2-D xor の転送最適化はこのような転送路の組み合わせであることから，この実装により実現できる可能性が高いことを示唆している．

## 8.　まとめ

本稿では，ネットワークを利用した冗長記録の手法について延べ，その転送を効率化する提案を行った．また，その実現に必要なネットワークライブラリの開発について紹介し，ゼロコピー転送などのメリットを引き出していることを示した．このライブラリを用いて予備評価を行った結果，提案手法の元になっているアイディアが実現可能であ

り，有効性もあることを明らかにした．

今後の課題としては，2D xor の転送最適化を任意のノード数でもできるよう一般化・定式化することと，ネットワークライブラリ自体の性能向上が挙げられる．また，2D xor を行う場合の性能評価も同様に重要である．

**謝辞** 本研究の一部は，JST CREST「ポストペタスケールデータインテンシブサイエンスのためのシステムソフトウェア」による．

## 参考文献

[1] Chervenak, A. L., Foster, I. T., Kesselman, C., Salisbury, C. and Tuecke, S.: The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets, *JOURNAL OF NETWORK AND COMPUTER APPLICATIONSSSS*, Vol. 23, pp. 187–200 (1999).

[2] Tatebe, O., Hiraga, K. and Sod, N.: New Generation Computing, Ohmsha, Ltd. and Springer, *Gfarm Grid File System*, Vol. 28, No. 3, pp. 257–275 (2010).

[3] Callaghan, B., Pawlowski, B. and Staubach, P.: NFS Version 3 Protocol Specification, *RFC 1813* (1995).

[4] Braam, P. J.: Lustre, *http://www.lustre.org/*.

[5] Carns, P. H., Iii, Ross, R. B. and Thakur, R.: Pvfs: a parallel file system for linux clusters, *In ALS 00: Proceedings of the 4th annual Linux Showcase and Conference* (2000).

[6] Callaghan, B., Lingutla-Raj, T., Chiu, A., Staubach, P. and Asad, O.: NFS over RDMA, *Proceedings of the ACM SIGCOMM workshop on Network-I/O convergence: experience, lessons, implications*, NICELI '03, New York, NY, USA, ACM, pp. 196–208 (2003).

[7] Wu, J., Wyckoff, P. and Panda, D.: PVFS over InfiniBand: Design and Performance Evaluation (2003).

[8] Weil, S. A., Brandt, S. A., Miller, E. L., Long, D. D. E. and Maltzahn, C.: Ceph: A Scalable, High-performance Distributed File System, *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, OSDI '06, Berkeley, CA, USA, USENIX Association, pp. 307–320 (2006).

[9] Huang, C., Simitci, H., Xu, Y., Ogus, A., Calder, B., Gopalan, P., Li, J. and Yekhanin, S.: Erasure coding in windows azure storage, *Proceedings of the 2012 USENIX conference on Annual Technical Conference*, USENIX ATC'12, Berkeley, CA, USA, USENIX Association, pp. 2–2 (2012).

[10] 大辻弘貴，建部修見：Infiniband を用いた遠隔ファイルアクセスの高速化，情報処理学会研究報告ハイパフォーマンスコンピューティング HPC135，p. 6 (2012).

[11] 大辻弘貴，建部修見：RDMA による低オーバヘッドファイルアクセスと冗長記録，情報処理学会研究報告ハイパフォーマンスコンピューティング HPC137，p. 7 (2012).