Vol.2012-ARC-202 No.29
Vol.2012-HPC-137 No.29
2012/12/14

RDMAによる低オーバヘッドファイルアクセスと冗長記録

大辻 弘貴[1,3]　建部 修見[2,3]

概要：大容量のデータ処理にあたって，高性能な共有ファイルシステムが求められている．共有ファイルシステムにおいては，アプリケーションはネットワークを介してデータアクセスを行うが，この部分がボトルネックになると十分にシステムの性能を発揮することが出来ない．従って，高性能なネットワークを効率よく使用する必要がある．そこで，ここでは，Infiniband に備わる RDMA（Remote Direct Memory Access）を用い，低オーバヘッドのファイルアクセスを行う．また，大容量データの保管にあたり，ストレージ領域を節約するための冗長記録についても評価を行った．さらに，冗長記録されたデータをストレージノードの負荷分散に用いる方法についても提案し，それを評価した．

1. 序論

計算機で扱われるデータ量は増加し続けており，今やエクサバイトクラスに達しようとしている．特にデータインテンシブコンピューティングやe-サイエンスの発展は，その状況を強く後押ししている．そのような中，大容量データに対する高速なアクセス手段を確保することは不可欠である．アクセス性能が不十分であると，システム全体としての性能が損なわれる．これらの要求を満たすため，現在では，分散ファイルシステムが解決手段の一つとして用いられている．分散ファイルシステムは，ネットワークで相互に接続されたストレージノードから構成されており，データに対するアクセスもネットワークを介す場合がある．分散ファイルシステムは広域分散型と集中型に分類することが出来る．いずれの場合においてもファイル複製 [1] による高速化手法などが用いられる事がある．特に広域分散型では地理的に離れた拠点が含まれることがあり，これらに対するアクセスは大きな遅延を伴うことになり，性能も低下する．この種類の複製は，データを使用される場所の近くに移動し，アクセスを効率化することが目的である．筆者らはこれまでの研究において，大きな遅延を伴うネットワーク上で高速にファイルアクセスを行う手法について提案してきた．本稿では，前者の集中型のファイルシステムに重点を置いている．集中型の環境においては，ネットワークの遅延は数μ秒に留まり，ボトルネックとなる箇所が根本的に異なる．この場合，OS やシステムソフトウェアのレイヤが性能の劣化を引き起こす可能性が高まる．従って，広域環境をターゲットとした分散ファイルシステムとは異なり，それらの箇所に対する最適化を提案する．具体的には，Infiniband RDMA(Remote Direct Memory Access) の活用である．RDMA のリモートファイルアクセスに対する適用については筆者等による [2] において予備評価を掲載しており，現在普及しているストレージデバイスに対して，十分なアクセス性能を提供可能なことを示している．

一方，大容量データを扱う上ではアクセス性能だけでは無く，信頼性の確保も問題となる．前述のデータ複製はデータの損失に対する対策にもなるが，最低でも倍以上の記憶領域を要する．エクサバイトクラスのストレージシステムともなると，記憶装置がシステムの総コストに占める割合が増す可能性があることから，極力少ない追加容量にて高い信頼性を提供することが求められる．この解決策として，冗長符号が用いられる事がある．本稿では，上記の高速なアクセス手段に対し，オンザフライで符号化/復号化を行う機能を追加し，その予備評価を行った．また，この冗長記録を用い，複数クライアントから複数のストレージノードに対してアクセスする際に生じる負荷集中を平滑化する方法についても提案し，その評価を示す．

2. 関連研究

本稿の対象はネットワークを介したデータアクセスと冗長記録に関するものであり，その関連研究は多くのシス

[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

Vol.2012-ARC-202 No.29
Vol.2012-HPC-137 No.29
2012/12/14

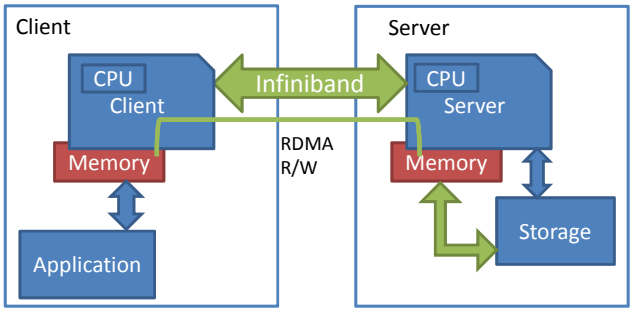


図 1 システムの構成

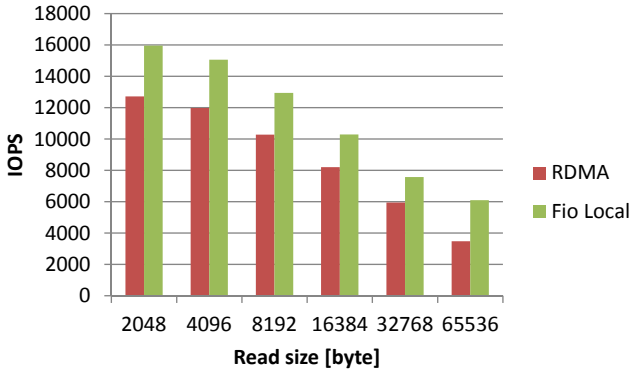


図 2 リモートアクセスとローカルアクセスの IOPS 比較 [2] より引用

テムや製品の実装に存在する．前者の遠隔アクセスについては，Gfarm[3] ファイルシステムや NFS[4]，Lustre[5]，PVFS[6] などがある．いずれのシステムもイーサネット経由でのファイルアクセスをサポートしており，特に NFS は NFS over RDMA[7] として実装が試みられており，Lustre は標準機能で RDMA によるファイルアクセスをサポートしている．また，PVFS over RDMA[8] という実装例もある．本稿におけるファイルアクセスについても，基本的な実装や目的はこれらの関連研究と同様であるが，冗長記録などの要素を取り入れている点で異なる．

冗長記録を用いた例としては，Microsoft Azure[9] のストレージなどがある．これは，広域に分散したストレージシステムについて，Easure Coding により冗長記録を作成し，信頼性の確保やレイテンシの低減を行っているものである．本稿においては，RDMA によるアクセスと冗長記録を組み合わせ，非常に少ないオーバヘッドにて，高信頼かつ高速なファイルシステムの実現を目指すための方法を述べる．

3. Infiniband RDMA による遠隔ファイルアクセス

3.1 概要

Infiniband RDMA を活用した遠隔ファイルアクセスについては，筆者らがこれまでに行っており [2] 本稿においてはその概要を示す．RDMA とは遠隔メモリアクセスを指し，ネットワークを介してリモートコンピュータのメモリ読み書きを行う物である．これは，対象ホストの CPU を介さずに行なえる上，イーサネットでは OS が行う場合がある輻輳制御やプロトコルの処理について，ハードウェア (HCA) が行うため，非常に低オーバヘッドとなる．結果として，イーサネットでは最低でも 100 μ s 以上のレイテンシとなるにも関わらず，RDMA を活用した場合には数 μ s のレイテンシに短縮する事が可能である．

Infiniband RDMA の利用方法は 3 種類に大分することが出来る．1 つは IPoIB(IP over Infiniband) であり，通常の IP(Internet Protocol) を Infiniband 上に流す物である．この方法では通常のネットワークインターフェースとして使える反面，プロトコルの処理などは OS が行うことになり，レイテンシは Ethernet とほぼ同等となり，各種オーバヘッドについても RDMA の恩恵を受けることが出来ない．2 つめは SDP(Socket Direct Procol) であり，IPoIB と比べると，輻輳制御などの機能を HCA(Host Channel Adapter) に任せるため，OS が重複して処理する必要がなくなり，より低レイテンシ・低負荷となる 3 つめは，Infiniband が提供する機能を直接利用する方法であり，Verbs API と呼ばれるインターフェースを利用する．この方法はネットワークの性能を最大限に引き出す事ができるものの，IPoIB や SDP と異なり，通常のソケットインターフェースが利用出来ないため，実装変更などのコストを要する．しかしながら，SDP と比較すると Verbs API には性能面で大きなアドバンテージがあり，ネットワークアクセスの性能がクリティカルな場面においては，コストをかけて実装する意義がある．

RDMA による遠隔ファイルアクセスは**図 1** に示す形で行われる．データのやりとりはメモリに対して直接読み書きが行われる．ローカルストレージからメモリへの転送に関しても，基本的に DMA が利用されており，全体が CPU by pass となっている．

3.2 評価

[2] より，RDMA を用いた遠隔ファイルアクセスの評価例を引用する（**図 2**）．

これは，リモートのノードに接続した高速なフラッシュメモリ (Fusion-io ioDrive) に対して，Infiniband QDR を介してネットワークアクセスを行った際の評価である．比較としてローカルアクセスの場合の性能を掲載している．評価の軸は IOPS であり，各サイズのストライドアクセスを 1 秒間に何回行なえたかを示している．遠隔アクセスにおいても，ストレージの性能をそれほど損なうこと無く利用出来ることが分かる．以降においては，これまでに示した方法にて遠隔アクセスを行うことを前提としている．

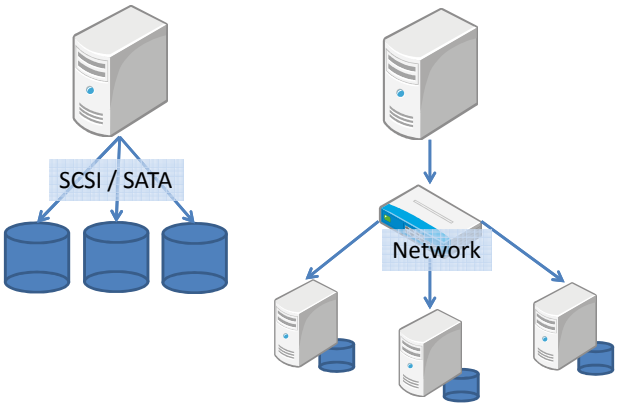
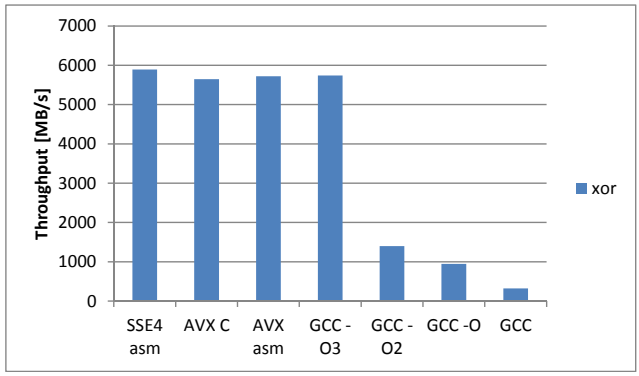


図 3　RAID(左) とネットワーク経由の記録 (右)



j

図 4　xor 演算のスループット

4. 冗長記録の適用

4.1 概要

冗長記録とは RAID などに見られるように，複数ある記憶装置のうちいくつかが故障した場合においても，記録されたデータをアクセス可能にするための方法である．単純な方法としては，データを複数のストライプに分割し，それらの排他論理和を別途保存する方式が上げられる．これは RAID-5 などで用いられており，数回の xor 演算で冗長記録の作成や復旧を行うことが可能である．本稿においては，これと同様の形で冗長記録を生成することを前提とし，各種の評価を行う．ただし，RAID とは異なり，それぞれのデータの保存先は同一ノードではなく，異なるリモートのノードに記録を行う．この様子を**図 3** に示す．

4.2 xor 演算のスループット

冗長記録の生成に関して，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．図**図 4** は，Intel 製 CPU(Sandy Bridge) において，各条件下でメモリ上のデータに対して xor 演算を行った際のスループットを示している．それぞれ，GCC による最適化やアセンブラによる実装の評価を行っている．アセンブラに関しては，Linux Kernel に含まれるソフトウェア RAID のためのコードを用いてる．また，スレッド数はいずれも 1 である．

最も高速であったのは SSE4 をアセンブラで用いた場合である．AVX 命令は Sandy Bridge 以降の Intel 製 CPU に搭載されているものであるが，現時点では整数演算は 128bit 幅となっており，SSE4 より若干性能が低い結果となった．

SSE4 の場合では，スループットは 6GB/s 近くになっており，Infiniband QDR 以上の速度かつ Infiniband FDR の実効速度に近い値となっている．従って，xor 演算を 1 回用いるような冗長記録に関しては，この点がオーバヘッドにならずに済むことが予想される．

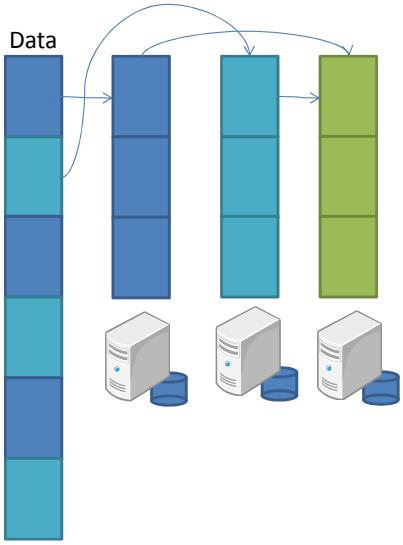


図 5　記録方法

4.3 冗長符号生成方法

前節で示したように，xor 演算を 1 回使用するような冗長符号生成においては，その演算がボトルネックになる可能性が低いことが示された．従って，ここでは最もシンプルな方法により冗長記録を行うこととした．具体的な方法としては，データをある一定のサイズで 2 つのストライプに分割して 2 台の記憶措置（異なるストレージノード）に置き，冗長符号を別のストレージノードに記録する．構成を**図 5** に示す．読み込みに関しては，2 つのストライプを読み込み結合するか，1 つのストライプと冗長記録により演算を行い，元のデータを復元する方法がある．

4.4 耐故障性

前節で述べた読み取り方法のうち，1 つのストライプと冗長記録を用いる方法がある事からも分かるように，3 つのうち 1 つの記録が損なわれてもデータを復元できることから，耐故障性を備えている．**図 6** は，正常時と故障時のアクセス方法を示す．正常時はストライピングされたデー

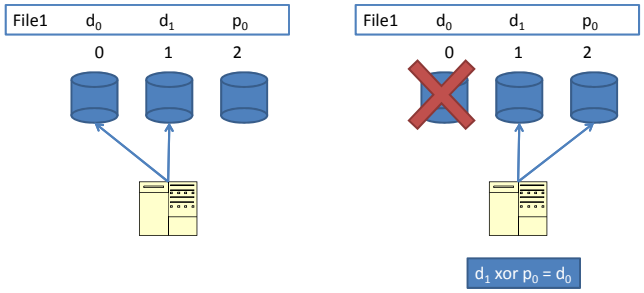


図 6 正常時のアクセス（左）と故障時（右）

タを読み込み，ブロックごとに結合する．故障時には，そのストレージノード以外のデータを読み込み，冗長記録から故障したノードのデータを復元する．

5. 負荷集中の緩和

5.1 提案の概要

前章では冗長記録と耐故障性について述べたが，ここでは復旧時に用いる方法を，ストレージノードの負荷集中を緩和するために用いる事を検討する複数のノードが，複数のストレージノードに対してアクセスを行っている場合，特定のストレージノードにアクセスが集中し，ボトルネックとなる事がある．これをホットスポットと呼ぶこととする．ホットスポットの発生を回避するためには，アクセス先のストレージノードを分散させる必要がある．複製を作成する分散ファイルシステムの場合，アクセス先の選択によりこれを実現することが出来る．しかしながら，前述のように複製を作成する方法では，少なくとも倍以上の記憶領域を必要とするため，記憶装置のコストを削減する上では不向きである．そこで，本来はデータの復旧のために存在している冗長記録を，負荷集中の回避に用いることを検討した．

5.2 負荷集中の発生

図 7 に示すように，ある 4 ノードのクライアントから 4 ノードのストレージノードにアクセスすることを想定する．この場合，1 番のストレージノードに対してアクセスが集中している．ストライピングされたデータは全て揃うまで使えないため，0 番や 2 番のストレージノードがどれだけ早くデータを返したとしても，スループットは 1 番のストレージノードに律速されてしまう．

ここでは，クライアント 0,1 がストレージノード 0,1 にアクセスしている中，クライアント 2,3 がストレージノード 2,3 にアクセスする場合と，ストレージノード 1,2 にアクセスする場合についての各クライアントにおけるスループットの比較を行った．**図 8** はこの性能を示しており，クライアント 2,3 が負荷集中を発生させると，約 33%の性能低下が見られた．

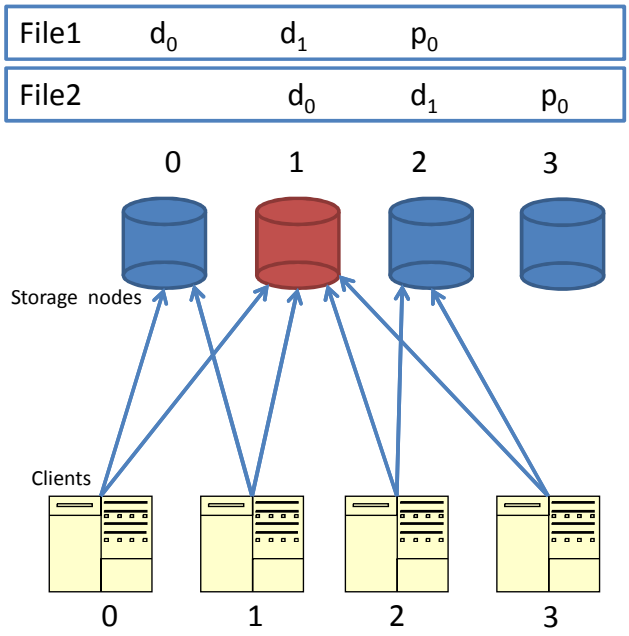


図 7 ストレージノード 1 に対する負荷集中の発生

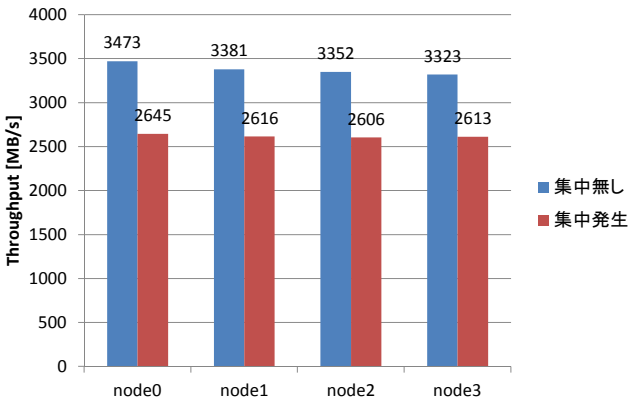


図 8 通常時と負荷集中発生時のスループット比較

5.3 冗長記録の利用による解決

負荷の集中したストレージノードにアクセスすると，前節で示したように性能の低下が引き起こされる．これを回避するために，冗長記録を利用してノードの負荷を低減する．状況は 5.2 に示すとおりで，4 台のクライアントが 4 台のストレージノードのデータにアクセスする．

クライアントノード 0,1 はファイル 1 にアクセスし，クライアントノート 2,3 はファイル 2 にアクセスする．通常通りにストライプにアクセスするだけでは，ストレージノード 1 に対して，アクセス負荷が集中する．この様子は図 7 に示すものである．上段の File は，そのデータがどのストレージノードに保管されているかを示している．d は各ストライプで，p は冗長記録である．一方，クライアントノート 2,3 のアクセス先のうち，1 つを冗長記録へと変

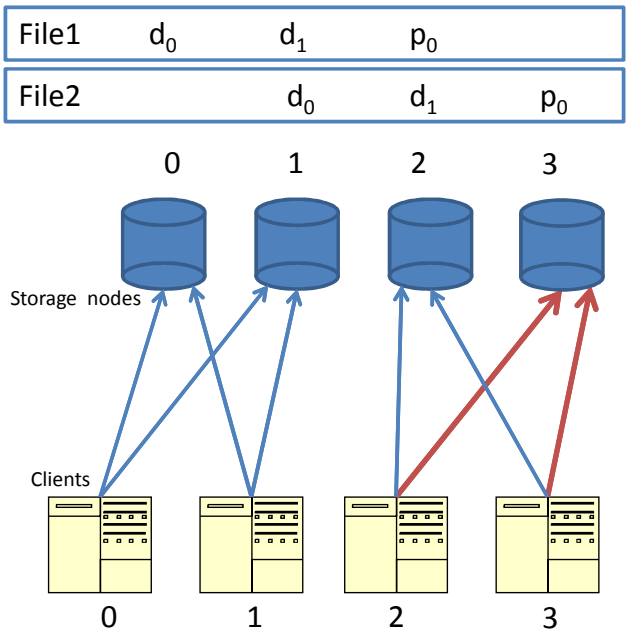


図 9 冗長記録の利用による負荷集中の回避

CPU	Intel(R) Xeon(R) CPU E5-2665 0 @ 2.40GHz 16cores = 8cores x2sockets
RAM	64GB
Infiniband HCA	Mellanox MT27500 FDR
Disk	Ram disk

図 10 評価環境

更することにより，ストレージノード 1 への負荷集中をなくすことが出来る（**図 9**）．この方法は，復号化に伴うコストと引き替えである事を留意しなければならない．この手法による性能の変化については次節に示す．

5.4 性能評価

5.3 に示した方法について，性能評価を行った．評価環境は，**図 10** に示す通りである．ネットワークは Infiniband FDR を用いている．ストレージは，ネットワークのバンド幅を上回るために，Ram disk を用いた．

図 11 はこの評価であり，横軸はクライアント番号，縦軸は各クライアントにおけるスループットである．Case1 はストレージノードに負荷集中が発生している状態で，Case2 は復号化を行いながらアクセスすることで負荷集中を軽減した場合の結果である．

クライアント 0,1 に関しては約 29%の性能向上が見られた．これは，ストレージノード 1 に対する負荷集中が解消

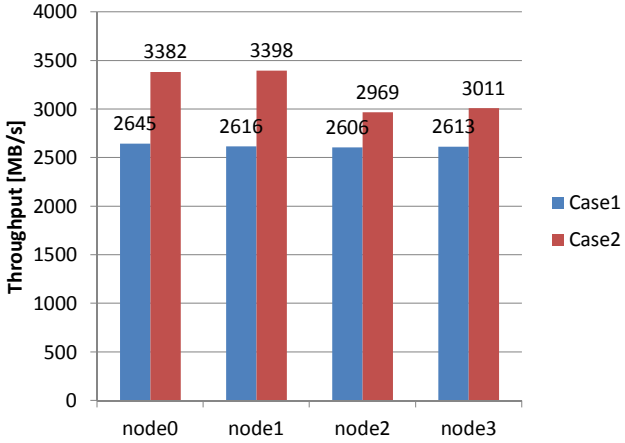


図 11 負荷集中の回避による性能向上

し，アクセス速度が向上した事によるものである．クライアント 2,3 に関しては，14%の性能向上が見られた．クライアント 0,1 と比較すると性能向上幅が少ないが，これは復号化処理によるオーバヘッドによるものである．xor 演算 1 回のスループットはこのスループットを上回っているが，今回の実装では演算と復号化のオーバラップがなされていないため，性能差があるものと考えられる．

クライアント全体を見た場合においても，スループットは向上しており，本手法が負荷集中に対して一定の効果を持つことが示された．

6. まとめ

本稿では，Infiniband RDMA を利用して遠隔ファイルアクセスを行う方法について延べ，それにより高い性能のリモートファイルアクセスが可能なことを示した．また，冗長記録に関しての基本的な性能評価を行うと共に，これを従前のリモートファイルアクセスに適用し，応用例とその可能性について示した．特に，取り上げた一つのケースについては，実測による評価を元に，冗長記録の応用により負荷集中の回避が可能である事を明らかにした．

今後の課題としては，ファイル書き込み時のトラフィック増加，ファイルの配置方法，アクセス時のストレージノード選択手法の検討，冗長記録方法の検討などが挙げられる．ファイルの配置方法は，一般的な複製については多く論じられており，また符号化についてもストレージ装置内に関しては文献も多い．しかしながら，本稿において提案した内容は複製ではなく，また記録先についても選択肢が多く，ネットワークを介しているなど多くの考慮すべき点があり，重要な課題であると認識している．

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 APPLICATIONS*, Vol. 23, pp. 187–200 (1999).

[2] 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).

[3] 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).

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

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

[6] 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).

[7] 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).

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

[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).