Vol.2009-HPC-121 No.31
2009/8/6

クラスタ間高速ファイル転送方式の提案と評価

鈴木克典[†1]　建部修見[†1]

ネットワークの広帯域化に伴い広域環境における大規模データ共有が現実のものとなってきた．一方で，大規模データ処理のために PC クラスタのそれぞれのローカルディスクが用いられるようになった．その結果，多数のローカルディスク間の効率的な並列ファイル転送が必要となっている．本研究では，並列ファイル転送を効率的に行うためのスケジューリング手法を提案する．ファイル複製の適切な選択，送信元クラスタ内での動的な複製作成により効率的な並列ファイル転送を可能とする．

提案手法を実際の広域ネットワーク環境で評価した結果，1 ノードに偏って格納されている 50 GB のデータを 17 分 21 秒で転送することができた．これは同一の環境で 1 ノード対 1 ノードでファイル転送を行った場合の 86.1%の転送時間である．このとき，平均スループット 49.0 MB/sec，最高スループット 380 MB/sec の性能であった．

A Proposal of Efficient File Transfer Between Clusters

Katsunori Suzuki[†1] and Osamu Tatebe[†1]

Large-scale data sharing becomes possible due to the improvement of wide area network bandwidth. For large-scale data processing, local disks of compute nodes should be exploited to improve disk I/O performance, thus efficient and parallel file transfer mechanism is required between compute nodes of two PC clusters. This paper proposes a scheduling algorithm for parallel file transfer, exploiting file replica selection and dynamic replica creation within the source cluster nodes.

The proposed file transfer takes 17 minutes and 21 seconds to transfer 50 GB of data stored at one compute node, which reduces 13.9% of transfer time compared with a simple file transfer. It achieved 49.0 MB/s of the file transfer bandwidth in average and 380 MB/s in maximum.

1. はじめに

ネットワークが広帯域化するに伴い，広域な環境下で大規模なデータを共有する要求が高まっている．科学技術分野では実験やシミュレーションによりデータを生成し，その分析を必要とする．それらは実験設備，データを格納するためのストレージや分析に要する処理能力などの理由から地理的に離れた場所で行われることが多い．実際に欧州原子核機構 (CERN) では膨大なデータの処理のためにヨーロッパ規模，世界規模のデータ・コンピューティンググリッドの利用が計画されている．[2)]

一方，近年では科学技術計算などの大規模計算において，コストパフォーマンスの理由から PC クラスタにおける並列計算が一般的に用いられている．PC クラスタで大規模データ処理を行う場合，ディスク I/O に要求される性能を達成するため，NFS のような共有ファイルシステムではなくローカルなディスクを効率的に利用することが重要になる．

現在では分散した環境下でデータを転送するプロトコルとして，GridFTP[3)] が用いられている．GridFTP は単一ファイルの転送プロトコルであり，単一ストレージ間でのファイル転送を行う．しかし，以上で述べたようなローカルディスクを活用する場面では，データが複数のストレージに分散される．このように転送元，転送先がそれぞれ複数のストレージからなる場合はファイル転送に関するスケジューリングが必要となる．単純にそれぞれが転送を行った場合，輻輳が発生し性能が低下する可能性がある．そのため，それぞれのノードからの転送データ量制御が必要になる．

そこで我々は，文献[9)] において，広帯域ネットワークで接続された PC クラスタ間で多対多のファイル転送を効率的に行うためのファイル転送タスクスケジューリング手法について提案し，シミュレータによる評価を行った．提案手法では，基本的に各ノードが各ローカルディスクに分散配置されているファイルの転送を並列に行うが，全体として性能が上がるように各ノードからの転送速度，転送データ量，ファイル転送順序，転送ファイルの割り当て，転送するファイル複製の選択などを制御する．また，提案手法では特定のディスクにファイルが偏って格納されている場合，実行時に送信元クラスタ内で動的にファイルの複製を作成し，ファイル転送時間の短縮化を図る．評価の結果，適切なファイル複製を選択することができていること，動的に複製を作成することでファイル転送時間が短縮可能であるこ

†1 筑波大学システム情報工学研究科
University of Tsukuba, Graduates School of System and Information Engineering

Vol.2009-HPC-121 No.31
2009/8/6

とを確認した．しかし，文献[9]では，動的な複製作成のアルゴリズムの計算量が多いこと，特定のファイル配置においては十分な性能が出せていないことなどの問題があった．

そこで，本稿では動的なファイル複製作成アルゴリズムを改良し，より高速で，より効率的なアルゴリズムを作成した．さらに，実環境上でファイル転送実験を行い，スケジューリングアルゴリズムの評価を行った．

本稿では，2 章において既存のファイル共有手法，ファイル転送手法について取り上げ，3 章で本手法が対象とする多対多のファイル転送の問題設定について述べる．4 章ではその問題設定の元で提案する手法についてアルゴリズム，システム設計の観点から説明し，5 章では評価について議論する．

2. 関 連 研 究

現在，ファイル転送プロトコルとしては FTP が一般的に用いられている．また，グリッド環境においては，FTP を拡張した GridFTP，GridFTP のセッションを管理してファイル転送の信頼性を確保する RFT (Reliable File Transfer)[4] が用いられることが多い．これらは単一ストレージ間のファイル転送を行っており，大規模データを効率的に転送するための工夫がなされている．本研究では，これら単一ファイルを扱うファイル転送プロトコルを多ノード対多ノードで並列に実行したときに，広帯域の広域ネットワークを効率的に利用するための転送スケジューリングの提案を行う．

N 対 M のファイル転送に関しては以下のような研究がなされている．Allcock ら[1] は GridFTP に対しストライピング機能を実装し，評価を行っている．このストライピング機能は並列ファイルシステム間で高速なファイル転送を行うためのものであり，単一ファイルの複数のファイルサーバへのストライプ転送を実現する．一方で，本研究ではクラスタ内の複数のディスクに分散して格納されている複数ファイルの転送を扱っている事が大きな違いといえる．

Weigle ら[5] は N ノード対 M ノードのデータ転送について “Composite Endpoint Protocol : CEP” を提案している．CEP は多対多のファイル転送において転送元のクラスタに存在しているファイルを「どのノードが，どの程度」転送することが最も効率が良いかをスケジューリングするものである．しかしながら，本研究では，輻輳を避けること，ディスクのシークを避けることなど性能低下を避けるための制限を設けている．そのため本研究は CEP と比べてより厳しい条件でのスケジューリングを行っているといえる．

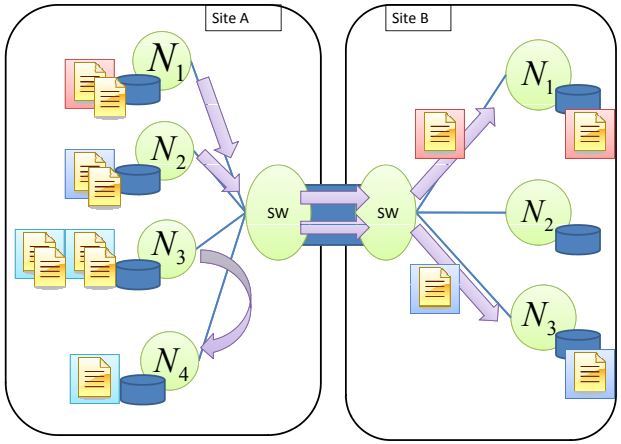


図 1 提案転送システムの概要

3. 問 題 設 定

本研究ではクラスタ内の複数のディスクに格納されるファイル群を効率的に他クラスタに転送するための機構について考察する．ここで，想定する問題設定についてまとめる．

- 想定環境概要
  図 1 に想定する環境を示す．本研究ではすべてのノードがそれぞれローカルなディスクを持ち，ファイルはそれらのディスクに格納されるようなクラスタ環境を想定する．また，クラスタ内のネットワークは均一であるとする．ネットワーク環境として，クラスタ間全体のバンド幅は，特定のノード間のバンド幅よりも大きいとする．また，クラスタ内のノード間のバンド幅はクラスタ間のノード間のバンド幅よりも大きいと仮定する．
- ディスクのシーク頻発による速度低下
  あるハードディスクに対し同時に複数ファイルを読み書きを行った場合，シークが頻発しディスクアクセス時間が極端に長くなる．そのため，同時に同一のディスクからファイル転送を行うことは，シークの頻発を起こしスループットを低下させる可能性があ

る．この問題はファイル転送性能に対し大きな影響を及ぼすにも関わらず，適切なスケジューリングによって回避可能な問題であり，ファイル転送スケジューリングを行う上で第一に守るべき制限として設定する．

- ファイル配置の不均一性
  特定のディスクに対し偏ってファイルが格納されている可能性がある．このとき，ファイル群転送において一部のディスクからの読み出し速度が全体の律速となると考えられ対策が必要となる．本手法では，転送時に動的にファイル複製を作成することでこの問題を解決する．
- ファイル複製の存在
  あるファイルに対し複製が複数のディスクに存在することがあると仮定する．このとき適切なファイル複製を選択することにより効率的なファイル転送が可能である．

4. 並列ファイル転送スケジューリング

前章で述べた想定環境において各ノードのディスク上のファイル群を効率的に転送するために解決すべき問題としては以下の項目を挙げることができる．

- ファイルの転送順序，転送タイミングの決定
  輻輳の発生による転送速度の低下を回避するために各ノードが転送を開始するタイミングをスケジューリングしなければならない．
- 同時転送ノード数とデータ転送量の制御
  同時転送ノード数と各ノードからの転送量を制御することで，ネットワークに流れるデータ量を制限し，輻輳の発生を抑制する．
- ファイル複製の選択
  適切なファイル複製の選択を行うことで特定のノードに負荷が偏らないよう考慮する．
- 動的なファイル複製作成のスケジューリング
  全体としての転送時間を短縮するようにクラスタ内のファイル複製作成と外部への転送のスケジューリングを行わなければならない．

本章ではこれらの問題を踏まえ，具体的に提案手法について述べる．
なお，第 4.1 節から第 4.2.2 節の詳細については，文献[9] を参照されたい．

4.1 モ デ ル

モデル化に際し，問題を簡単化するためにクラスタ間のバンド幅を抽象化した「コネクション」概念を導入する．コネクション集合 C は固定サイズのコネクション c の集合とし，コネクションのサイズとはバンド幅の大きさであるとする．つまり C とはクラスタ間のバンド幅を固定のサイズで分割したものの集合であるといえ，ノードに対して c を割り当てることはノードに対して c のサイズのバンド幅で転送を行う許可を与えることを意味する．これにより，同時に転送を行うノード数の制限と各ノードからのファイル転送速度の制限を表現することが可能となる．

ノードの集合を $N = \{N_1, N_2, \cdots, N_n\}$，ファイルの集合を $F = \{F_1, F_2, \cdots, F_m\}$ としたとき，解くべき問題はファイル複製の選択決定であるノード集合 N に対するファイル集合 F の組み合わせ問題とファイル転送順序のスケジューリングであるファイル集合 F とコネクション集合 C の割り当て問題である．

ここで，転送処理に関して複数のファイルを一括して一つのデータとして扱っても問題はない．また，同様に一つのファイルを分割し，別々のデータとして転送しても問題はない．この特徴を利用するとすべてのノードが複数のファイルを持つ状態を，すべてのノードが一つの転送データを持つ状態であると見ることができる．また，全てのノードはファイル転送のタスクに関して互いに影響を与えず，独立であるといえる．これらの特徴からファイル集合Fとコネクション集合 C の割り当て問題は，転送に関して独立であるノードの集合Nとコネクション集合 C の独立タスクのスケジューリング問題と言い換えることができる．

4.2 ファイル転送タスクスケジューリング

「ファイル集合 F をコネクション集合 C に割り当てるスケジューリング」は NP 完全な問題である「ノード集合 N の独立タスクのスケジューリング」に還元することが可能であることを前節で述べた．そこで本手法では以下のようなステップを踏むことで，複数存在するファイル複製の選択問題とファイル転送のスケジューリング問題を解く．

4.2.1 単純割り当て

ノード集合 N をコネクション集合 C にスケジューリングするためには

- どのノード上のファイル複製を転送するかの決定
- ノード集合 N のコネクション集合 C への独立タスクスケジューリング

を求めなければならない．

そこでこのステップではまずはじめに順序をつけたノード集合を先頭から順に探索し，探索ノード上にて初めて現れたファイル複製をその探索ノードからの転送ファイルとして決定する．その上でノードの大きさをそのノードが転送するファイルの総量として，リストスケジューリング[8] によりコネクション集合 C に割り当てる．

4.2.2 複製の修正選択

単純割り当てのステップではファイル複製の選択決定をノード間の負荷の分散を考慮せずに決定しているため，特定のノードに負荷が偏ってしまい良いスケジューリング結果とはいえない．そこでこのステップでは転送するファイル複製の選択をやり直し，転送ファイル量の多いノードから少ないノードへ割り当てを修正することで，徐々に負荷の平均化を行う．

4.2.3 複製の動的作成

ファイル複製の修正選択ステップを経てもなお転送ファイルが偏って割り当てられているノードに対してファイル複製作成のスケジューリングを行う．

文献[9] ではこのステップを各ノードがファイルを外部へ転送中に複製作成可能なノードと複製量を徐々に求めていくアルゴリズムを採用していた．しかし，これでは問題パターンにより非常に計算が複雑になるという問題点があり，そのため，本手法では全体のファイルサイズとその割り当ての偏りから大まかに複製作成量を見積もる方法をとっている．

クラスタ間でのデータ転送速度を 1 としたとき，クラスタ内のデータ転送速度を $h(h \geq 1)$ とする．**図 2** において，サイズ S_s のファイル a のうちサイズ B だけを他ノードに複製を作成するとき，B はクラスタ内では $X = \frac{B}{h}$ の時間で転送が可能である．複製を作成したのち，ファイル a の複製未作成部分のサイズは $A = S_s - B$ であるため，複製作成のための時間を含めた元々のノードからの外部へのファイル a の転送時間は $\frac{S_s - B}{1} + X$ となる．もしも，ファイル複製の作成先のノードがサイズ S_d のファイルを持っていた場合，複製作成先のノードからの外部へのファイル転送時間は $\frac{S_d + B}{1} + X$ となる．このとき，ファイル複製を作成することで，複製作成先のノードの転送時間が複製作成元のノードの転送時間を超えてはファイル複製を作成する事がかえって逆効果となってしまうため，以下の不等式を満たさなければならない．

$$\frac{S_s - B}{1} + X \geq \frac{S_d + B}{1} + X$$

これより，$B \leq \frac{S_s - S_d}{2}$ が求まり，このサイズ B までの範囲で複製作成をスケジューリングすればよい．

具体的には以下の手順を踏む．

(1) 最も転送時間のかかるコネクションを C_s，最も転送時間の短いコネクションを C_d とする．

(2) C_s に割り当てられているノード N_s に対して上記の方法でファイル複製の作成サイズを計算する．

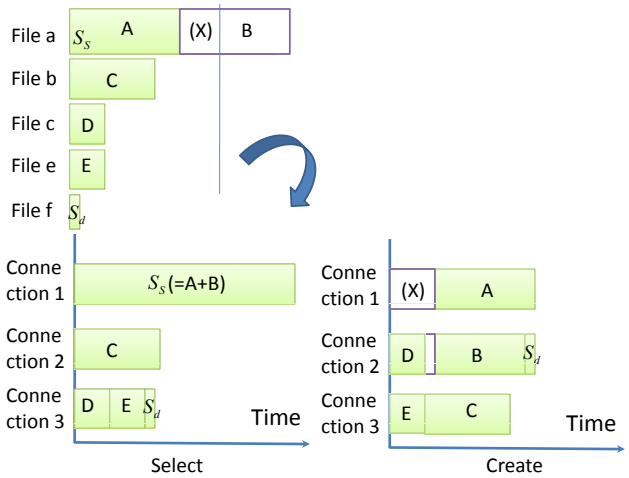


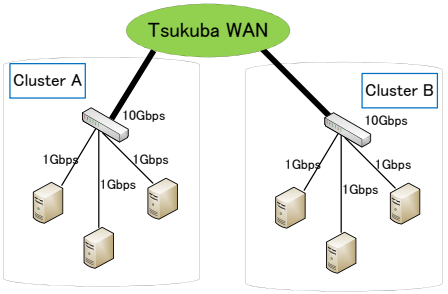
図 2 複製の動的作成ステップの概要

(3) C_s に割り当てられているノード以外のノード集合から複製作成先のノード N_d を得る．このとき，すでに複製の作成元及び複製の作成先となっているノードは再び選択されることはなく，該当ノードが存在しない場合は終了する．

(4) ファイル複製作成後，コネクションに対してノードの再割り当てを行う．このとき，N_s は C_s に，N_d は C_d に固定割り当てとし，さらに割り当てが行われていない残りのノードを複製作成のための転送時間も踏まえた上で改めてリストスケジュールによりコネクションに対して再割り当てを行う．

(5) もしも，(1) の状態よりも転送時間が長くなるのならば，(2),(3),(4) の更新処理を破棄した上で終了し，そうでないならば (1) へ．

このアルゴリズムではすべてのノードはファイル複製を作成し始める前にたかだか 1 回のみファイル複製を受け取り，さらに自分の受け取ったファイルの一部の複製を他のノードに作成する．このとき，複製作成後の転送時間は複製作成に必要な転送コストを含めても必ず複製作成前の転送時間よりも短くなるため，全体としての転送時間も複製を作成することで必ず短縮化することができる．



(a) ネットワーク環境

	Cluster A	Cluster B
Node_1	91.2 MB/sec	59.0 MB/sec
Node_2	93.9 MB/sec	59.8 MB/sec
Node_3	110.3 MB/sec	54.8 MB/sec
Node_4	111.3 MB/sec	54.9 MB/sec
Node_5	111.0 MB/sec	58.5 MB/sec
Node_6	109.3 MB/sec	58.2 MB/sec
Node_7	110.6 MB/sec	58.6 MB/sec

表 1 dd コマンドによるディスク書き込み性能

図 3 実験環境

4.2.4 ノード単位スケジューリングの生成

以上までのアルゴリズムによりコネクション集合に対するノード集合の割り当てスケジューリングを得ることができた．この割り当て情報から逆にノード集合に対するコネクション集合のスケジューリングを計算することが可能であり，これにより各ノードにおけるファイル転送スケジューリングを得ることができる．

ここでコネクション c はネットワークのバンド幅を抽象化したものであるとしたが，これは送信クラスタから見た場合，あるネットワークパイプの入力であると見ることができる．逆に宛先クラスタから見た場合にはネットワークパイプの出口であると見なすことができる．そのため，宛先クラスタ側においてもコネクションに対し空きディスク容量などを加味した上でノードを割り当てることで実際のノードからノードへのファイル転送をスケジューリングすることができる．

5. 評　　価

現在，システムとしては提案段階であるため，アルゴリズムについて C++言語で実装し，分散ファイルシステム Gfarm[7] のファイル転送機能を用いて実環境上で評価実験を行った．

実験には産業技術総合研究所の保有する PC クラスタ A の 7 ノードと筑波大学の保有する PC クラスタ B の 7 ノードを用いた．クラスタ A, B のノードはすべてグローバル IP アドレスを持ち任意に通信可能である．ネットワーク環境は**図 3(a)** のようになっている．すべてのノードが Gigabit Ethernet で接続され，スイッチ間は 10 Gigabit Ethernet で Tsukuba WAN 経由で接続されている．また，A, B のローカルディスク性能は**表 1** のようになっている．

表 2 クラスタ A,B のディスクからディスクへのファイル転送スループット

From - To	Throughput
Cluster A Node_1 - Cluster A Node_2	56.9 MB / sec
Cluster A Node_1 - Cluster B Node_2	35.1 MB / sec

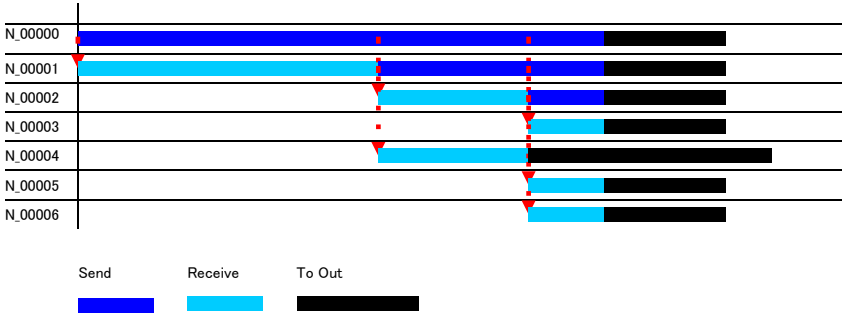


図 4 ファイル転送のスケジューリング結果

このクラスタ A, B の間でファイル転送を行い提案手法を評価する．すべてのデータはクラスタ A のうちの 1 つのノードのローカルディスクに格納され，クラスタ A から B へファイル転送を行う．転送するデータサイズは 50 GB とする．

5.1 スケジューリングと実測結果との比較

ファイル転送スケジュールを作るにあたって，クラスタ A 内でのファイル転送速度とクラスタ A,B 間のファイル転送速度を得る必要がある．これには Gfarm 上で複製を作成するコマンド gfrep で指定したホスト間でのファイル複製作成を実際に行い，その実行時間を測ることでディスクからディスクの転送速度を測った．**表 2** がその結果である．これよりクラスタ A 内ではクラスタ A,B 間よりも 1.6 倍の速度を出すことができることがわかる．

この速度情報を元に並列ファイル転送のスケジューリングを行った結果が**図 4** である．紺の線が同クラスタ内ノードへの送信，水色の線が同クラスタ内ノードからの受信，黒い線が外部クラスタへの送信を意味する．また，赤い点線矢印がクラスタ内での複製作成を表す．スケジューリングの結果から，Node_0 から各ノードへ複製を作成していく様子がわか

る．まず Node_0 から Node_1 へ 25GB 程度のデータを転送している．さらに，Node_0 が Node_2 に，Node_1 が Node_4 に 12GB 程度の元々自分が持つファイルまたは，他のノードから受け取ったファイルを転送している．最後にはすべてのノードが同時にクラスタ B へファイル転送を行っている．クラスタ A,B の各ノード間の転送速度差や 10Gbps アップリンクの有効活用などの理由から，クラスタ間 1 ノード対 1 ノードのファイル転送よりも，このスケジュールに従ってクラスタ A の各ノードからクラスタ B の各ノードへファイル転送を行った方がより高速なファイル転送が実現できると考えられる．

実際のファイル転送には gfrep コマンドを利用した．しかし，本提案手法では一つのノードに格納されているファイルを分割し，複製を作成することで負荷の分散を考えるが，gfrep にはファイルの特定の部分のみの複製を作成する機能がない．そこで本実験では 50G の一つのファイルを転送するのではなく，スケジューリングにより計算されたサイズにファイルを分割して利用した．また，提案アルゴリズムでは「コネクション」を利用して割り当てを計算するが，実際のファイル転送においてはコネクションを実際の送信先ノードと対応付けを行わなければならない．今回はクラスタ B の各ノード（Node_0 6）に 1 対 1 で対応している．つまり，クラスタ A の Node_i はクラスタ B の Node_$i(i = 0 \cdots 6)$ にファイル転送を行う．

ファイル転送の結果が**図 5** である．各線分がそれぞれの処理に費やした時間である．図 5 の各線分の意味は図 4 と同様である．図 5 より，ほぼスケジューリング通りにファイル転送が行われているといえる．しかしながら，転送終了時間にばらつきがあるなど細部までは予想に従っているとはいえない．特にクラスタ A の Node_4 からクラスタ B の Node_4 へのファイル転送は非常に時間がかかっている．これは現在のスケジューリングがクラスタ内とクラスタ間のそれぞれ一つのノード間の速度差情報のみを用いるのみで，個々のノードのディスクアクセス速度や，トラフィック情報などが反映されていないためである．正確なスケジューリングには各ノード個別の情報をさらに用いるべきであるといえることが分かる．

5.2 1 対 1 のファイル転送と提案手法の比較

提案手法を 1 対 1 でファイル転送を行った場合と比較評価する．1 対 1 転送にはクラスタ A の Node_1 からクラスタ B の Node_1 を用い，提案手法と同様に gfrep コマンドで 50GB 分のファイルを転送した．結果を**図 6** に示す．1 対 1 の転送には 19 分 46 秒かかり，平均スループットは 42.2MB/sec であった．一方，提案手法は 17 分 21 秒かかり，平均スループットは 49.0MB/sec であった．以上の結果から，今回の実験ではクラスタ間でデータを転送する目的においては提案手法を用いることで 1 対 1 転送の 86.1%の時間で全データを

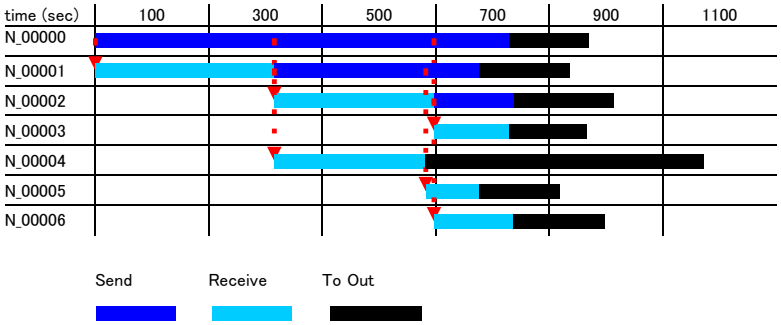


図 5 ファイル転送時間計測結果

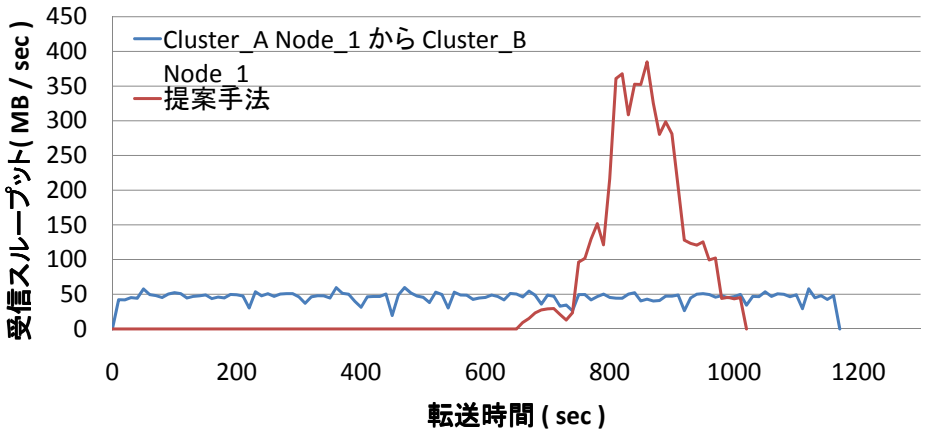


図 6 1 対 1 転送及び提案手法のクラスタ B における受信スループット

転送することができたといえる．

また，このとき提案手法のクラスタ B における受信スループットの各ノードごとの内訳を**図 7** に示す．各ノードとも多少のゆらぎがあるものの，同時に転送を行うノードが増えたとしても各ノードの受信スループットにはそれほどの変化は見られない．これより今回

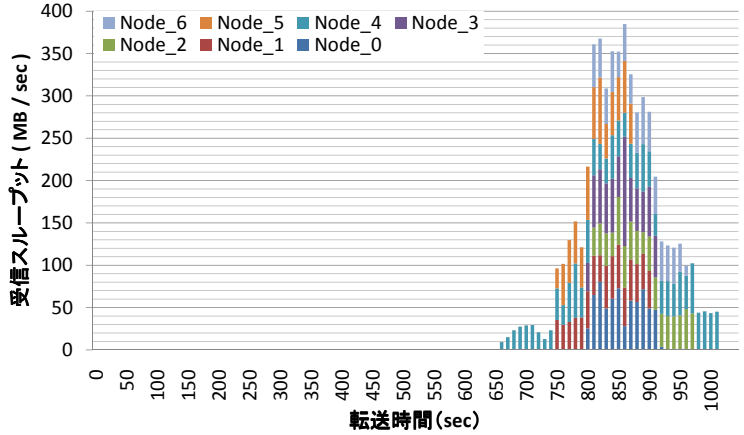


図 7 クラスタ B の各ノードにおける受信スループットとその合計値

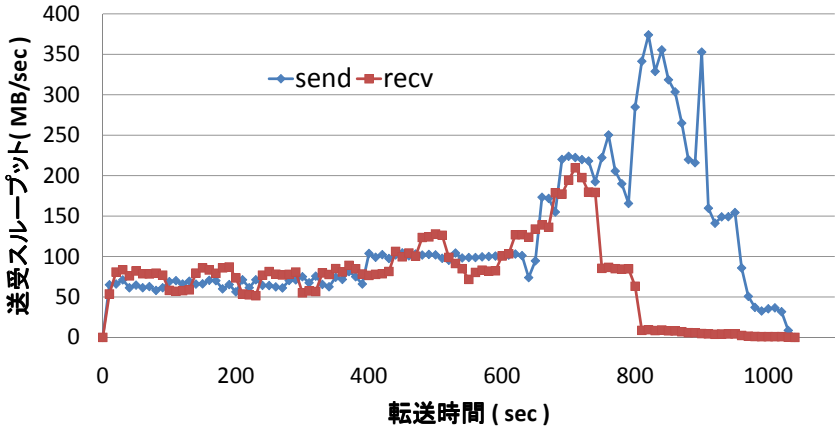


図 8 クラスタ A 内外の各ノードの送受信スループット合計値

の実験では Gigabit Ethernet で接続されたノード 7 台のみしか用いておらず，合わせても最高 7Gbps の帯域しか扱えていないことを加味したとしても，各ノードからの転送が衝突し，性能が低下することを回避できているといえる．この本提案手法の輻輳による転送速度低下を避ける特徴はさらに転送に用いるノード数を増加させたときにはさらに有効に働くと予想される．

図 8 は提案手法でファイル転送を行った時のクラスタ A 内外の転送スループットである．これは単位時間当たりに Node_$i(i = 0 \cdots 6)$ が送受信したデータ量の合計値である．初期状態ではクラスタ A の Node_1 のみしか転送ファイルを持っておらず，これを他のノードに転送し，転送ファイル保有ノードが増加するにつれ送受信スループットが増加する様子が分かる．開始 700 秒付近で受信スループットが下がり始め，800 秒付近でほぼ 0 になるのはクラスタ A 内での複製作成が終了し始め，各ノードがクラスタ B のノードへ転送を始めるからである．図 6，図 7，図 8 より，提案手法では最高で 380MB/sec 程度のスループットをディスクからディスクへの転送で実現しているといえる．

6. まとめと今後の課題

本稿では広帯域なネットワークで接続されるクラスタ間において，その広帯域ネットワークを有効に活用し高速にファイル転送を行う手法について提案し，評価した．提案手法は各ノードが並列にファイル転送を行うことを基本とし，さらに各ノードからの転送量制御，転送順序制御，動的な複製作成などにより効率的に転送を行う．

評価は分散ファイルシステム Gfarm のファイル転送機能を用い，実際に 2 クラスタ間でファイル転送実験を行った．結果として，50GB のデータを 17 分 21 秒で転送し，これは 1 ノード対 1 ノードの場合の 86.1%の時間であった．また，最もディスク I/O の遅いノードが 54.8MB/sec であるという環境のもと，平均スループット 49.0MB/sec，最高スループット 380MB/sec の性能を得た．

しかしながら，現在の実装では個々のノードごとのディスク I/O 速度やネットワークトラフィック状況，CPU 使用率などの情報を反映したスケジューリングを行っておらず，正確なスケジューリングであるとは言い難い．また，現在のスケジューリング方法でははじめに全スケジュールを静的に計算し，転送処理ではスケジュールに従ってファイル転送を行っている．この方法では，転送途中でネットワーク状況などが変わった場合に対応できず，再スケジューリングなどの処理が必要になってくる．

今後の課題として，単独のシステムとして実装を進める．提案システムとしてはマスターサーバを置かなくとも，各ノード上のデーモンがコミュニケーションを取り合うことで実現可能であると考えられるが，マスターが存在した方が良い場面もある．今後さらにシステム設計について詰めていく．また，本提案手法では同時に複数のノードが転送を行うため，

転送速度の制御がより重要である．そこで，この対策としては産業総合研究所で開発されている PSPacer[6] を利用することを検討している．これにより各ノードからの転送が正確に平滑化され，より効率的なファイル転送が実現可能であると考えられる．

本実験ではディスク I/O の速度が大きな影響を与えていた．現在のハードディスクではネットワークのスループットを超えることは難しく，ディスクがボトルネックとなる．しかしながら，SSD を使用した場合ではネットワークがボトルネックとなる場合が増え，これにより，本提案手法を適用可能な場合もより増大すると考えられる．そのため今後 SSD を用いた場合の対応を考えていく．

謝辞 本研究の一部は，文部科学省次世代 IT 基盤構築のための研究開発「e-サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発」における研究課題「研究コミュニティ形成のための資源連携技術に関する研究」（データ共有技術に関する研究）および文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」（公募研究 A02-17，課題番号 21013005）による。

参 考 文 献

1) William Allcock, John Bresnahan, Rajkumar Kettimuthu, and Michael Link. The globus striped gridftp framework and server. In *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, p. 54, Washington, DC, USA, 2005. IEEE Computer Society.
2) Flavia Donno and Maarten Litmaath. Data management in wlcg and egee. worldwide lhc computing grid. Technical Report CERN-IT-Note-2008-002, CERN, Geneva, Feb 2008.
3) T.Perelmutov I.Mandrichenko, W.Allcock. Gridftp v2 protocol description.
4) R.K. Madduri, C.S. Hood, and W.E. Allcock. Reliable file transfer in grid environments. In *LCN '02: Proceedings of the 27th Annual IEEE Conference on Local Computer Networks*, p. 0737, Washington, DC, USA, 2002. IEEE Computer Society.
5) E.Weigle and A.A. Chien. The composite endpoint protocol (cep): scalable endpoints for terabit flows. In *CCGRID '05: Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05) - Volume 2*, pp. 1126–1134, Washington, DC, USA, 2005. IEEE Computer Society.
6) 高野了成, 工藤知宏, 児玉祐悦. 精密な帯域共有とトラフィック隔離を実現するパケットスケジューリング方式 (計算機アーキテクチャ・ハイパフォーマンスコンピューティング・「ハイパフォーマンスコンピューティングとアーキテクチャの評価」に関する北海道ワークショップ (hokke-2009)). 情報処理学会研究報告，2009-HPC-119(9)，pp.67-72，Feb.2009., 2009/2/26-28.
7) 建部修見, 曽田哲之. 広域分散ファイルシステム gfarm v2 の実装と評価. 情報処理学会研究報告，2007-HPC-113， pp.7-12，Dec.2007, 20071207.
8) 須田礼仁. ヘテロ並列計算環境のためのタスクスケジューリング手法のサーベイ. 情報処理学会論文誌. コンピューティングシステム, Vol.47, No.18, pp. 92–114, 20061115.
9) 鈴木克典, 建部修見. 並列ファイル転送のためのスケジューリングアルゴリズム. 情報処理学会研究報告，2009-HPC-119(9)，pp.49-54，Feb.2009., 2009.