戦略的創造研究推進事業　CREST
研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」
研究課題「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」

研究終了報告書

研究期間　平成23年4月～平成29年3月

研究代表者：建部修見
（筑波大学計算科学研究センター教授）

§1 研究実施の概要

(1)実施概要

次世代 DNA シーケンサ、放射光、加速器などの実験装置、スパコンによる数値シミュレーションなどによりペタバイト規模を超える大規模データが生成されるようになった。この大規模データを解析することによる科学的手法はデータインテンシブサイエンスと呼ばれる。本研究課題は、大規模データ解析のためのシステムソフトウェアとして、ペタバイト規模のデータの高速アクセスを実現する分散ファイルシステム、計算ノード OS、大規模データ処理実行基盤の研究を実施した。それぞれの研究における実施概要を以下にまとめる。

・分散ファイルシステム

計算ノード数の増加に応じてスケーラブルにストレージ性能を増加させるため、計算ノード側にストレージを分散配置するスケーラブルなストレージアーキテクチャを前提に分散ファイルシステムの設計、実装を行った。既存のシステムでもこのアーキテクチャを前提にしたものはあるが、ポストペタスケールにおける数万計算ノードからの数百 TB/s の性能を実現するためには、メタデータの分散・冗長管理、ファイルデータの冗長化・分散アクセス、ローカルストレージの高性能化などの研究課題を克服する必要がある。これらの課題に対し、メタデータの分散・冗長管理は、ダイナミックソフトウェアトランザクションメモリ(DSTM)に基づくノンブロッキング分散トランザクションにより解決した。ファイルデータの冗長化・分散アクセスは、アクティブストレージによるストレージサーバ間 RAID により解決した。ローカルストレージはフラッシュデバイスを含む不揮発性メモリを想定したオブジェクトストレージを設計することにより解決した。これらの要素技術をベースに、スケーラブルな分散ファイルシステム PPFS の設計、評価を行った。これらの成果は国際会議、ジャーナルで発表した。オープンソースで開発し、既に実運用されている Gfarm ファイルシステムに取り込んで広く利用することで、新産業創出、今後のイノベーションにつなげる。

・計算ノード OS

研究の狙いは、分散ファイルシステムの性能を最大限に引き出すためのカーネルドライバおよびキャッシュ管理技術を構築することである。計算ノード OS が提供する各機構および全体システムの実装、性能評価、高性能化、高信頼化を行った。具体的には、キャッシュ管理機構とカーネルドライバを連携させ、性能測定実験を行った。そのキャッシュ管理機構はクライアント間でキャッシュを共有する協調キャッシュ機構であり、計算ノードのメモリ資源を有効に利用することを可能にする。また、そのカーネルドライバは InfiniBand RDMA を用いて高速にファイルデータを転送することができる。そのファイルデータ転送機構についても高性能化、高信頼化、実験による性能評価を進めた。次に、OS ノイズ(OS ジッタ)に関する実験をさらに進め、様々なパラメタを与えたときの科学技術計算アプリケーションの性能を測定した。また、分散ファイルシステムへの重複除外機構の導入についても研究を行った。クライアントノード上キャッシュに対する重複除外の適用についての研究を論文にまとめるとともに、I/O ノードが管理するファイルデータに対する重複除外機構を設計、実装した。これらの成果は国際会議、ジャーナルで発表し、Gfarm カーネルドライバ、Gfarm ファイルシステム用 RDMA モジュール、Linux 用ジッタ削減モジュール、FUSE 用ローカルアクセス最適化モジュールを公開した。

・大規模データ処理実行基盤

研究の狙いは、データインテンシブサイエンスのアプリケーションを効率的に実行するため、MPI-IO、大規模ワークフロー実行、MapReduce 処理、バッチキューイングシステムなどの実行環境を設計、実装することである。ワークフロー実行における並列性を保ちつつデータ移動を最小化するスケジューリング手法、効率的なノード内スケジューリング手法を提案した。バッチキューイングシステムについては、I/O インテンシブなジョブと CPU インテンシブなジョブが混在したジョブについてファイル局所性を考慮したジョブスケジューリング手法の設計を行った。これらの手法により、計算ノード側にストレージを分散配置する非均一アクセス性能を持つスケーラ

ブルなストレージアーキテクチャの性能を向上させることが可能となった。これらの成果は国際会議、ジャーナルで発表し、Pwrake ワークフローシステム、Gfarm MPI-IO プラグイン、Gfarm Hadoop プラグインを公開した。

(2)顕著な成果

＜優れた基礎研究としての成果＞
1. PPFS: a Scale-out Distributed File System for Post-Petascale Systems
概要:
ポストペタスケールにおいて要求される、数万計算ノードからの数百 TB/s の性能を実現するためのスケーラブルな分散ファイルシステム PPFS の設計と評価を行った。メタデータ管理では、ダイナミックソフトウェアトランザクションメモリ(DSTM)に基づくノンブロッキング分散トランザクションを用いるスケーラブルなメタデータサーバ PPMDS を用い、ファイルデータの管理では、不揮発性メモリを想定したオブジェクトストレージ PPOST を用いる。メタデータサーバとオブジェクトストレージ間の通信を最適化するため、バルクオブジェクト生成、オブジェクトプリフェッチングを提案した。評価において、メタデータ性能、ファイルデータアクセスのバンド幅ともにスケーラブルな性能を示した。さらに、現在最も高速と考えられる IndexFS に比べ同じサーバ数においてより高い性能であった。本成果は IEEE データサイエンスとシステムに関する国際会議 DSS 2016 で発表した。

2. RDMA-based Direct Transfer of File Data to Remote Page Cache
概要:
分散ファイルシステムのサーバクライアント間で InfiniBand RDMA により高速にファイルデータを転送する技術を開発した。その技術は、従来クライアントのカーネル内で行われていたデータコピー処理を除去し、サーバがクライアントのファイルデータ格納用メモリ領域に直接データを書き込むことを可能にする。国内外の類似研究の大半は RDMA でユーザ空間にデータを転送するものであった。本研究ではカーネル空間に転送する技術を構築した点に科学技術上のインパクトがある。本成果は IEEE クラスタコンピューティングに関する国際会議 Cluster 2015 で発表した。

3. Workflow Scheduling to Minimize Data Movement using Multi-constraint Graph Partitioning
概要:
データ移動を最小化し、並列実行可能なワークフローのスケジューリング問題を多制約グラフ問題として定式化を行った。既存研究では、入力ファイルの配置に基づくスケジューリングであったが、ワークフロー全体で最適化することにより大幅にデータ移動の削減が可能となった。ワークフローエンジンにそのスケジューリングアルゴリズムを組み込み、実問題において評価を行った結果、データ移動が大幅に削減可能となり、実行時間が短縮した。IEEE/ACM クラスタ、クラウド、グリッドコンピューティングに関する国際会議 CCGrid 2012 で発表した。

＜科学技術イノベーションに大きく寄与する成果＞
1. Gfarm ファイルシステム、MapReduce プラグイン、MPI-IO プラグイン、Pwrake ワークフローシステムのソフトウェアリリース
概要:
本 CREST で提案する分散ファイルシステムのアーキテクチャを実装した Gfarm ファイルシステム、およびファイルシステムの分散したデータを効率的に利用するための実行時システム、プラグインをリリースした。Gfarm ファイルシステムによりスケーラブルな分散ファイルシステムを実現することができる。MapReduce アプリケーション、MPI-IO アプリケーションはそれぞれのプラグインを用いることで、またワークフロー処理は Pwrake ワークフローシステムを利用することにより、ファイルシステムの分散したデータのデータ移動量を自動的に最小化し，効率的に実行するこ

とができる。これらのソフトウェアをオープンソースで公開することにより、幅広い産業での利用、これまでにはないビッグデータ処理による新産業の創出、今後のイノベーションへの手がかりとなる。

2. Gfarm カーネルドライバ、RDMA 通信モジュール、OS ジッタ削減モジュールのソフトウェアリリース
概要:
Gfarm ファイルシステムをマウントし、冗長なメモリコピーの削減、コンテキストスイッチを削減するためのカーネルドライバ、ネットワークアクセスを高速化するための RDMA 通信モジュール、アプリケーションの性能低下を抑える OS ジッタ削減モジュールをリリースした。Gfarm カーネルドライバにより、他ノードのバッファキャッシュにデータを効率的に転送可能となり、大きく性能が向上する。ビッグデータ処理を更に加速するソフトウェアであり、幅広い産業での利用が見込まれる。

3. 特定非営利活動法人つくば OSS 技術支援センターの設立
概要:
本 CREST 研究で開発したソフトウェアを社会に広く展開し、技術的なサポートの提供と技術情報を共有していくため、特定非営利活動法人を設立した。プロジェクトで開発したソフトウェアはプロジェクトの終了と同時に開発、メンテナンスが終了してしまうことが問題となるが、その問題を解決するための特定非営利法人である。企業などで、本 CREST 研究で開発したソフトウェアを安心して産業利用するためのサポート基盤となる。特定非営利活動法人により、成果を社会に広く還元し、企業利用の促進、新産業の創出をすすめていく。

§2 研究実施体制

(1)研究チームの体制について
①筑波大グループ
研究参加者

氏名	所属	役職	参加時期
建部 修見	筑波大学システム情報系	教授	H23.4～
川島 英之	同上	准教授	H25.5～
田中 昌宏	筑波大学計算科学研究センター	研究員	H23.4～
Mohamed Amin Jabri	同上	研究員	H24.12～
桑野 洋子	同上	研究員	H27.4～
平賀 弘平	筑波大学大学院システム情報工学研究科	M1～D3	H23.4～
大辻 弘貴	同上	M1～D3	H23.4～H28.3
大西 健太	同上	M1～2	H23.4～H25.3
鷹津 冬将	同上	M1～D3	H24.4～
李 燮鳴	同上	M1～D3	H24.4～
Silva Marilia Rodrigues de Meloe	同上	M1～2	H24.12～H26.3
龔 渝	同上	M1～2	H25.4～H27.3
古 江	同上	M1～2	H25.4～H28.3
蒋 立	同上	M1～2	H26.4～H28.3
神谷 孝明	同上	M1～2	H27.4～

堀尾 健太郎	同上	M1～2	H27.4～
三橋 龍也	同上	M1～2	H27.4～
馬 浩宇	同上	M1～2	H27.4～
岩井 厚樹	筑波大学情報学群情報科学類	B4	H28.4～
小林 淳司	同上	B4	H28.4～

研究項目
・分散ファイルシステム
・大規模データ処理実行基盤

②電通大グループ
研究参加者

氏名	所属	役職	参加時期
大山 恵弘	電気通信大学大学院情報理工学研究科	准教授	H23.4～H28.3
高橋 一志	同上	特任助教	H25.4～H28.3
石黒 駿	同上	M1～2	H23.4～H25.3
村上 じゅん	同上	M1～2	H23.4～H25.3
佐々木 慎	同上	M1～2	H25.4～H27.3
松宮 遼	電気通信大学情報理工学部，大学院情報理工学研究科	B4～M2	H25.6～H28.3
鈴木 光一朗	電気通信大学大学院情報理工学研究科	M2	H27.4～H28.3
鈴木 絵里香	電気通信大学電気通信学部	B4	H23.4～H24.3
鈴木 耀平	電気通信大学情報理工学部	B4	H27.4～H28.3

研究項目
・計算ノード OS
・キャッシュ管理機構およびファイルシステムのカーネルモジュールの性能評価，高性能化を進めるとともに，システム全体の評価，高性能化，高信頼化を行う。

(2)国内外の研究者や産業界等との連携によるネットワーク形成の状況について
同様の研究開発をすすめている米国アルゴンヌ国立研究所に本 CREST の RA を 3ヶ月間派遣し、情報交換を行うとともに、主に先方がすすめるストレージシステムの通信レイヤの研究開発を進めた。文部科学省と米国エネルギー省の合意による「現在および将来の HPC システムのシステムソフトウェアに関する共同研究」により、米国アルゴンヌ国立研究所と国際連携をすすめている。
開発した Gfarm ファイルシステムは、文部科学省がすすめる「京」を中核とする革新的ハイパフォーマンス・コンピューティング・インフラ(HPCI)の HPCI 共用ストレージの基本ソフトウェアとして運用されている。

§3 研究実施内容及び成果

3．1 分散ファイルシステム、大規模データ処理実行基盤(筑波大学 建部グループ)

本研究課題は、ポストペタスケール高性能計算において大きな課題となっているストレージ性能を

向上させることを目的としたストレージアーキテクチャ、ストレージシステムソフトウェア、計算ノードOS、ジョブスケジューリングの研究開発を行うものである。スーパコンピュータにおけるストレージ性能は、研究開始当初から CPU 性能、メモリ性能などにくらべ性能向上が見込めないため大きな問題となっていたが、ポストペタスケール高性能計算、とくにデータインテンシブサイエンスのための高性能計算においては、CPU 性能、メモリ性能、通信性能の性能向上に応じたストレージ性能の向上が求められ、その解決が大きな課題となっている。本研究は、この課題の解決を目指し CPU コア数の増加に対し、ストレージ性能を向上(スケールアウト)させるため、ストレージを計算ノード側へ分散配置する、スケールアウトするが非均一アクセス性能を持つストレージアーキテクチャを想定し、研究開発を行うものである。本研究で想定するスケールアウトするストレージアーキテクチャを図 1 に示す。

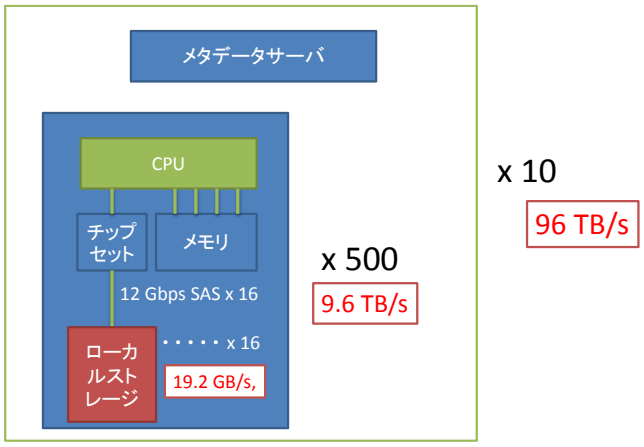

図 1. 想定するスケールアウトするストレージアーキテクチャ

ストレージは、計算ノードの高速なローカルストレージ、あるいは複数の計算ノードに対し直接接続されるストレージノードに持つ。ローカルストレージ数、あるいは直接接続されるストレージノード数は、計算ノード数の増加に応じて増加するため、ストレージ全体におけるバンド幅は計算ノード数に応じてスケーラブルに増加する。100 PFlops のスーパコンピュータについては、性能バランス的にストレージ性能として 100 TB/s ほどの性能が求められるが、このアーキテクチャでは例えば次のように実現することができると考えられる。まず各ストレージノードに 19.2 GB/s の物理性能を持つストレージをもたせる。研究開発当初は 12 Gbps の SAS を 16 本接続し 19.2 GB/s を達成することを想定したが、研究開始から 5 年を迎えた現在においては PCI Express、あるいは NVM Express で接続することにより同等あるいはそれ以上の物理性能が達成される。このストレージノードを 500 ノードにつきファイルシステムメタデータサーバを配置し、そのセットを 10 セット準備することにより、96 TB/s の物理性能を達成可能となる。このアーキテクチャにおいて、ストレージノードのストレージにはファイルデータを格納する。ファイルデータはストレージノードに分散配置されることとなるが、その配置場所やファイルデータの作成時刻、アクセス権限などメタデータをファイルシステムメタデータサーバが管理する。ファイルデータとメタデータを別管理とすることにより、ファイルデータに対する分散並列アクセスが可能となる。

研究開発当初、本ストレージアーキテクチャをもつストレージシステムとしては、MapReduce のためのストレージシステムとして設計された Google ファイルシステム、Hadoop 分散ファイルシステム、また研究代表者の先行研究である Gfarm ファイルシステムがあった。しかしながら、図 1 の 100 TB/s のストレージシステムを構成するためにはいくつか解決すべき課題がある。まず、メタデータサーバの性能、可用性の問題である。既存システムはいずれも単一のメタデータサーバを用いるものであるが、その場合計算ノードは数千ノードまでが性能限界となる。図 1 において計算ノードはローカルストレージを用いる場合で 5,000 ノード、複数の計算ノードでストレージノードをもつ構成の場合はその数倍となり、性能面で問題が生じる。可用性については、既存システムでも Gfarm ファイルシステムや若干構成が異なる Lustre ファイルシステムではスレーブメタデータサーバを用い、

マスターメタデータサーバの障害発生時にはスレーブメタデータサーバにフェイルオーバすることで可用性の向上を図っている。メタデータサーバの障害は、ストレージシステム全体の障害となってしまうため、可用性を向上させることは重要な課題となる。

次の問題は、ファイルデータの冗長化の問題である。図 1 のストレージアーキテクチャでは、ストレージは計算ノード側に分散配置される。ストレージに障害が発生したとき、そこにしかデータがないとアクセスすることができなくなるだけではなく、最悪の場合はデータの喪失となってしまう。そのため、ファイルデータを複数のストレージノードに冗長配置することが必要となる。冗長配置については、Google ファイルシステム、Gfarm ファイルシステムなどではファイル複製を複数のストレージノードにもつ。ファイル複製は作成するだけストレージ容量が減ってしまう。例えば、Google ファイルシステムではファイル複製を 3 つ作成するが、その場合ストレージ容量は 1/3 となってしまう。Erasure coding を用いることで、ストレージ容量の減少を緩和することが可能であるが、符号化のオーバヘッドがあるため、Hadoop 分散ファイルシステムではまずファイル複製を作成し、あとで符号化によりストレージ容量の減少を緩和するなどの方法をとっている。そのため、性能を犠牲にしないで冗長化を行うことは重要な課題となる。

また、高速なストレージデバイスとして、フラッシュデバイスなどの不揮発性メモリの技術開発がなされている。これまでのストレージソフトウェアは HDD を前提として設計されているが、不揮発性メモリは HDD とは性能を向上させる手法が異なるため、ストレージソフトウェアの再設計が必要となる。

上記の課題は、分散ストレージシステムのソフトウェア研究開発における課題であるが、本分散ストレージシステムでは、性能をスケールアウトさせるため、ファイルデータを計算ノード側に分散配置する。そのため、ファイルデータの格納場所によりアクセス性能が異なる。つまり、メモリにおける非均一メモリアクセス（NUMA）と同様に、近いストレージと遠いストレージがあり、それぞれアクセス性能が異なり、均一ではない。計算ノードのローカルストレージ、あるいは直接接続されるストレージノードにアクセスする場合は、ストレージまでのネットワーク的距離が短く高速にアクセスできるだけではなく、全計算ノードが独立にアクセス可能なため、計算ノード数に対してスケーラブルなストレージアクセス性能を達成可能である。一方、計算ノードから遠いストレージに対するアクセスは、ネットワーク的距離が遠いだけではなく、全計算ノードがアクセスする場合はネットワークトラフィックが問題となり、必ずしもスケーラブルなストレージアクセス性能が実現するとは限らない。そのため、分散ストレージシステムを効率的にアクセスするためには、アクセスするストレージを適切に選ぶこと、および、なるべく近いストレージがアクセスできるようにタスクをスケジューリングすることが重要となる。新規ファイルの場合は、最も近いストレージを選ぶことによりアクセス局所性を向上させることが可能であるが、既存ファイルのアクセスの場合は、既存ファイルの格納場所に応じてタスクをなるべく近い計算ノードにスケジューリングすることが課題となる。

本研究グループでは、上記の研究課題に従い、分散ファイルシステム、大規模データ処理実行基盤について、以下の研究項目を設定した。まず、分散ファイルシステムについては、以下の研究項目を設定した。

1. メタデータの分散、冗長管理の設計
2. ファイルデータ冗長化・分散アクセスの設計
3. ローカルストレージの設計

大規模データ処理実行基盤については、本提案における分散ファイルシステムを高い性能でアクセスするためファイルアクセスの局所性を高めるタスクスケジューリングの研究開発を行った。特にデータインテンシブサイエンスにおいては、MPI-IO、ワークフロー実行、バッチキューイングシステム、MapReduce 処理が重要となる。そのため、以下の研究項目を設定した。

1. MPI-IO、ワークフロー実行、バッチキューイングシステム、MapReduce 処理の設計

上記の設計は研究期間前半の 2～3 年で行い、その後、プロトタイプ実装と性能評価、高性能化、ファイルシステム全体の評価、高性能化、高信頼化を行った。

以下、それぞれの研究課題について実施内容及び成果を記載する。

分散ファイルステム

1. メタデータの分散、冗長管理の設計

メタデータの可用性を高めるための冗長管理においては、冗長に保持するデータの同期方式によりどのような障害について耐障害性をもつかが決まる。H23 年度では、まず、それらの関係を明らかにし、それぞれの同期方式について定量的に性能を計測した。ログの書き込みが大きなオーバヘッドとなり得るが、その書き込みを RAM ディスクにすることにより、同期方式の違いによる性能差はほとんど見られなかった。研究成果は H23 年 7 月の情報処理学会 HPC 研究会において発表した。

H24 年度は、メタデータの分散管理の設計を行った。目標はスケールアウト性を確保するため、シェアードナッシング型の分散キーバリューストア（KVS）における実装が可能な設計である。ファイルシステムのメタデータで分散管理が難しいところはディレクトリ構造の管理である。既存のファイルシステムでは、ディレクトリ内のエントリ追加（ファイル作成）などの操作において、一貫性を保つため逐次化が行われるため性能を向上させることが困難であった。問題は同一ディレクトリ内のエントリの並列追加、ディレクトリの移動、消去などのファイルシステムのディレクトリ操作を、いかに一貫性を保ちつつ行うことができるかである。この問題に対し、まず、ディレクトリをシェアードナッシング型の分散 KVS に格納し、ダイナミックソフトウェアトランザクションメモリ（DSTM）に基づくノンブロッキング分散トランザクションを設計することにより解決を図った。初期性能評価においてメタデータサーバ数を増加させることにより IOPS が増加することが示された。本成果は H24 年 8 月に情報処理学会 HPC 研究会において発表した。

H25 年度は、前年度の設計を元にプロトタイプ実装を行い 15 サーバで 270,000 IOPS を達成した。H23 年に CMU で研究開発された GIGA+は 32 サーバで 98,000 IOPS を達成していたが、それを凌ぐ性能であった。さらに、評価で判明したいくつかの問題における解決を図った。一つ目の問題は、ノンブロッキング分散トランザクションを用いていることによるオーバヘッドと逐次化である。分散トランザクションを実現するため、現在の値がどのような状態にあるのかを示す別のキーバリュー（状態キーバリュー）を利用しているが、参照だけ行う場合についてもこの状態を表す状態キーバリューを参照する必要があり、参照についてもオーバヘッドがかかる。さらに、この状態キーバリューは単一サーバに置かれるため、この参照が逐次処理となってしまい、性能をスケールアウトさせることができない。この問題を解決するために、更新されないメタデータへのアクセスに対し、状態キーバリューを参照しないで直接参照可能な形に変え、参照オーバヘッドを減らすことを提案した。さらに、この二種類のデータが共存可能なシステム設計を行った。二つ目の問題は、サーバ処理をブロック型のスレッドで行っていたことによるデッドロックの可能性である。この実装では、サーバのスレッドプールを使い尽くしてしまうと、これ以上サーバ処理を行うことができず、資源不足によるデッドロックに陥ってしまう。この問題を解決するために、全てのサーバ処理をノンブロック型で再設計した。ノンブロック型処理のために、全ての RPC を非同期呼び出しと結果受取用コールバック関数で再設計した。

その後、より大規模でのメタデータサーバの性能評価を行うため、シミュレーションによる評価を行った。シミュレーションにあたり、楽観的プロトコルによる並列離散シミュレーションのためのフレームワークである CODES/ROSS を用いて、設計したメタデータサーバのモデルを設計、構築した。シミュレーションにより、実システムによる性能評価結果とほぼ同様の結果を得ることができた。評価環境のパラメタを変更することにより、メタデータサーバ性能を制限しているボトルネックが判明した。本結果は H27 年 12 月の情報処理学会 HPC 研究会において発表した。

2. ファイルデータ冗長化・分散アクセスの設計

まず、ストレージノードをまたがった冗長符号格納方式、遠隔メモリアクセス（RDMA）に基づく高速なファイルアクセスについて研究を行った。冗長符号格納では、erasure coding における Reed-Solomon 符号、Cauchy Reed-Solomon 符号、ARS 符号、Blaum-Roth 符号、Liber8Tion 符号、Good Cauchy Reed-Solomon 符号をプロトタイプ実装により性能比較を行った。符号化と書込、読込と復号をオーバラップさせることにより、符号化、復号のオーバヘッドを減らすことが可能であり、評価環境においては 5%以下に抑えられた。本成果の一部は H24 年 8 月の情報処理学会 HPC 研究会において発表した。RDMA に基づく高速なファイルアクセスについては、InfiniBand の RDMA

を利用する通信ライブラリの設計と Verbs API を用いたプロトタイプ実装を行った。

その後、冗長符号化のオーバヘッドを更に削減するための研究開発を行った。H24 年度の研究成果により、アクセスと符号化のオーバラップを行うことで符号化のオーバヘッドを減らすことが可能であることを示したが、冗長符号では元データと冗長にするための符号データの両方を格納することが必要であり、符号データ分のデータを追加で書き込まなければならない。この問題を根本的に解決するため、ストレージに冗長符号計算とデータ転送をオフロードしたアクティブストレージを提案した。

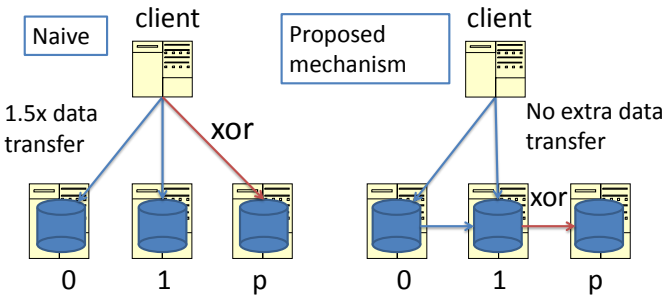

図 2. 2D1P の RAID-4 のアクティブストレージによるデータ転送と冗長コード計算

アクティブストレージでは、冗長符号に関わる計算と通信をストレージが行うため、書込を行うクライアント側のデータ転送量、冗長符号計算のオーバヘッドはない。2D1P(2 データディスク、1 パリティディスク)の RAID-4 の場合、図 2 に示すように通常の方法だとクライアントからの転送量が 1.5 倍となり、また冗長符号計算としての XOR 計算が必要となってしまう。提案方式のアクティブスストレージの場合は、クライアントからのデータ転送量は変わらず、またクライアント側での XOR 計算は必要なくなる。その代わり、ストレージ 0 はデータをストレージ 1 に転送し、ストレージ 1 で XOR 計算を行い、ストレージ p に転送する。本提案手法を InfiniBand 4xFDR を用いた PC クラスタで初期評価を行ったところ、ネットワーク物理ピーク性能となる 5.4GB/s の書込性能を達成することができた。冗長化を行わない RAID-0 と同様の性能を達成し、ゼロオーバヘッドの RAID-4 を実現した。本成果は、H26 年 9 月に IEEE クラスタコンピューティングに関する国際会議(Cluster)でポスター発表を行った。さらに、本方式を RAID-5 に拡張した。図 3 にそのデータ転送パターンを示す。

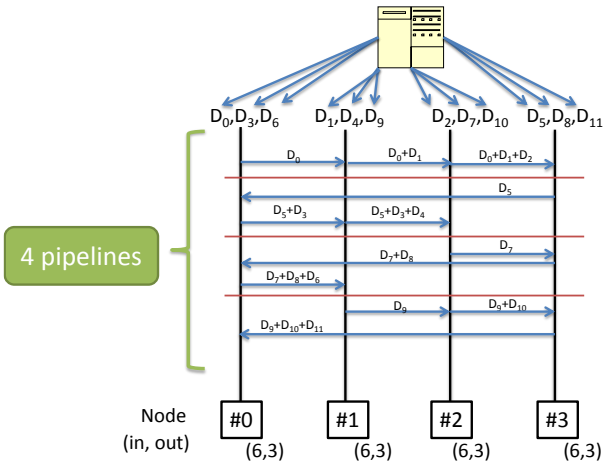

図 3. RAID-5 におけるアクティブストレージのデータ転送パターン

RAID-5 の場合は、3 ブロックずつに異なるストレージサーバにパリティブロックを持つため、図 3 のように 4 つの独立したパイプラインを構成することとなる。ストレージサーバ間のデータ転送を別ネットワークで行った 3D1P のアクティブストレージによる RAID-5 の性能を図 4 に示す。4 クライアント以上からのアクセスにおいて、3D の RAID-0 の性能を凌ぐ性能を示した。

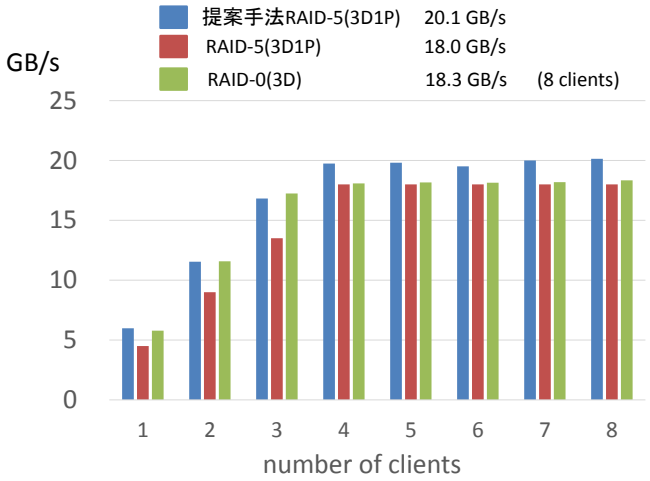

図 4. アクティブストレージ RAID-5 の性能評価

本成果は H27 年 12 月に IEEE データサイエンスとデータインテンシブシステムに関する国際会議（DSDIS）で発表した。

3. ローカルストレージの設計

まず、不揮発性メモリ等、次期ストレージデバイスにおける効率的なファイルシステムを目標として、ログ構造化ファイルシステムをベースに設計を行った。データインテンシブコンピューティングにおいて重要な操作となる並列書き込みを効率的に行うため、書き込み、更新操作は連続アクセスとなるように設計を行った。初期性能評価においては、物理的な性能に迫る性能を示し、現在利用されているファイルシステムに比べ高い性能を示した。本成果は H24 年 5 月の SACSIS2012 のポスター発表および H24 年 8 月の情報処理学会 HPC 研究会で発表した。

H25 年度からは、不揮発性メモリ用の新しいインタフェースとして提案されている OpenNVM を用いたオブジェクトストレージの設計を行った。OpenNVM はこれまでのブロックデバイスのインタフェースに加え、スパースアドレス、不可分バッチ書込、キーバリューアクセスなどのインタフェースを持っている。スパースアドレスを用い、比較的大きなリージョンサイズを持つ図 5 に示すような固定長リージョンによる設計を行った。

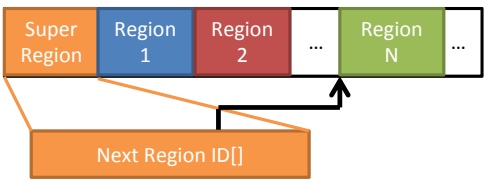

図 5. 固定長リージョンによるオブジェクトストレージの設計

スパースアドレスにより、リージョンサイズを大きくしても、実際に利用したブロックしか物理的容量は消費されないため容量の無駄はない。この設計に基づき、プロトタイプ実装を行った結果、16 スレッドのクライアントからのファイル生成において 186,000 OPS を達成した。固定長リージョンとすることによりファイル生成に関する最適化が可能となる。最適化では、N リージョンを予約するバルクリザベーションと、不可分バッチ書込による N ブロックを一括で初期化するバルク初期化を提案した。プロトタイプシステムによる性能評価では、16 スレッドのクライアントからのファイル生成において、128 リージョンを予約することにより 1.4 倍の性能向上、32 リージョンのバルク初期化を併用することにより更に 2.8 倍の性能向上を達成し、747,000 OPS を達成した。既存システムで最も性能が高い DirectFS は 61,300 OPS であり、12 倍以上の性能向上を達成した。本成果は H27 年 5 月にグリ

ーン、パーベイシブ、クラウドコンピューティングに関する国際会議（GPC）において口頭発表した。また今回設計したオブジェクトストアとはインタフェースが異なるものの、ストレージとしてほぼ同様の機能を持つ他のキーバリューストアとの性能比較を行った。その結果、RocksDB に対し 7 倍、NVMKV に対し 4.3 倍高速であった。これらの成果をまとめた論文は Journal of Information Processing に掲載された。

H28 年度は、これまで研究開発を行ったメタデータサーバ、ローカルストレージをベースに分散ファイルシステムの設計を進め、評価を行った。まず、ローカルストレージを遠隔クライアントからアクセス可能にするためのサーバ設計を行った。このサーバ設計にあたり、クライアントが自ノードの場合は、サーバを経由せず直接ローカルストレージをアクセスする。これによりローカルアクセスの性能向上を図った。図 6 に 512 バイトブロックでアクセスした場合の I/O バンド幅を示す。このベンチマークではそれぞれのサーバノードに 16 クライアントを起動した。サーバを経由しないで直接ローカルストレージをアクセスすることにより 1.9 倍の性能向上を実現した。

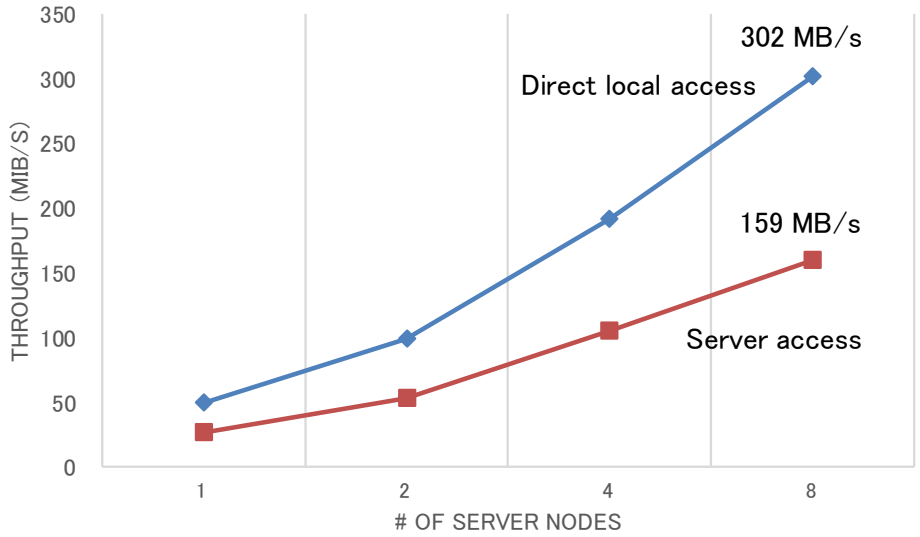

図 6. ストレージサーバ数を増加させたときの 512 バイトブロックアクセス性能

分散ファイルシステムの設計において、ファイルを作成する場合は、まずローカルストレージの利用を試み、利用できない場合は jump consistent hash によりリモートストレージを選択し、オブジェクトを作成し、そのエントリをメタデータサーバに登録する。ただし、この設計の場合、ファイルを作成するたびにストレージにオブジェクトを作成するためファイル作成遅延が増えてしまう。この問題を解決するために、バルクオブジェクト作成、オブジェクトプリフェッチングを提案し、ファイル作成のたびにオブジェクトを作成するのではなく、予め複数のオブジェクトを作成することによりファイル作成遅延を隠蔽する性能改善を行った。図 7 にメタデータサーバ数を変更したときのファイル生成性能を示す。このグラフは一秒間に生成したファイル数を示している。

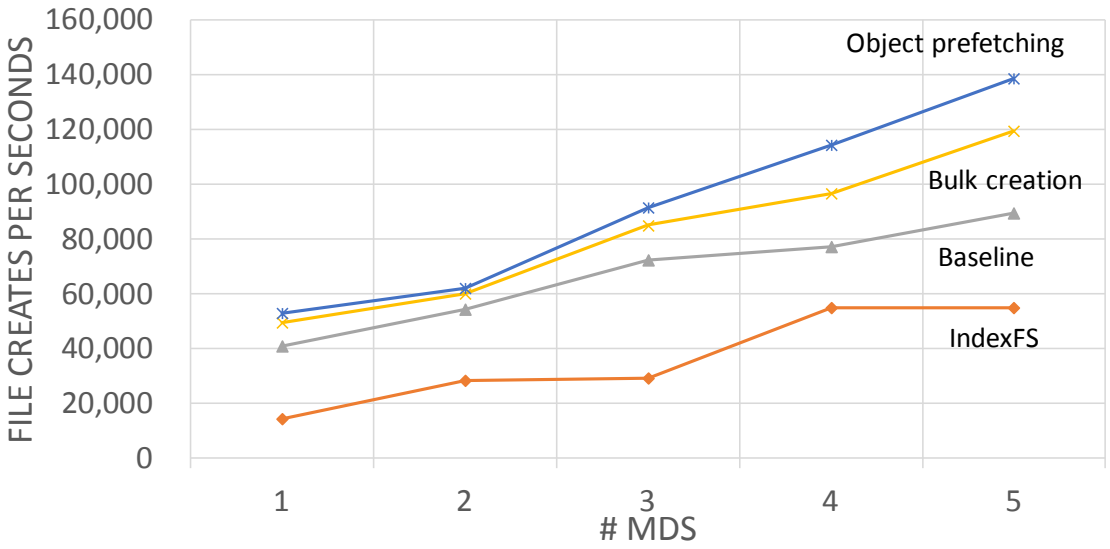

図 7.　メタデータサーバ数を変更したときのファイル生成性能

IndexFS は米 CMU の研究者が SC2014 で発表したファイルシステムである。128 メタデータサーバで秒間 842,000 ファイルを作成したと報告され、現時点で最も高速と思われるファイルシステムである。評価では 128 ノードまで準備ができず 5 ノードまででであるが、IndexFS と性能を比較したところ、最適化をしていない baseline の性能でも、同じサーバ数において IndexFS の性能を上回っていた。さらに、バルクオブジェクト生成、オブジェクトプリフェッチングの最適化を行うことにより、5 ノードで毎秒 139,000 ファイルを生成する性能を示した。メタデータサーバ数を増やすことで性能が向上しており、性能向上がこのまま続くとすると 36 ノードで毎秒 1,000,000 ファイルの生成が可能となる。本成果は H28 年 12 月に IEEE データサイエンスとシステムに関する国際会議（DSS）で発表した。

大規模データ処理実行基盤

1.　MPI-IO、ワークフロー実行、バッチキューイングシステム、MapReduce 処理の設計

本研究提案で研究開発する分散ファイルシステムは、全体としてのファイルアクセス性能はスケールアウトするが、ファイルアクセス性能が非均一となる。そのため、効率的に利用するためには、データアクセスについての局所性を利用し、データ移動を最小化することが重要となる。そのため、データアクセスの局所化を行い、データ移動を最小化するためのプロセススケジューリングに関する研究を行った。

大規模ワークフロー実行は、タスク間のデータ依存によりタスクグラフが構成される。タスクグラフの枝はデータの依存関係を表し、データ移動を最小化するためには、エッジカットを最小にするグラフ分割を考えることになる。ただし、並列実行を目的とする場合は、並列に実行できるタスクを分割する必要があるため、単純なグラフ分割問題に帰着することはできない。目的を達成するためには、多制約タスク分割問題に帰着することが必要であった。図 8 に一例を示す。ワークフローのタスクグラフを単純にグラフ分割すると、データ移動は最小となるが、図 8 の左図のように実行の前半部は 2 ノードで並列実行し、後半を別の 2 ノードで並列実行するというような結果が得られる。これに対し、並列実行可能な各ステップについてグラフ分割を行うように問題を多制約グラフ分割に帰着することにより、図 8 の右図のように 4 ノードで並列実行しつつデータ移動を最小化することができる。

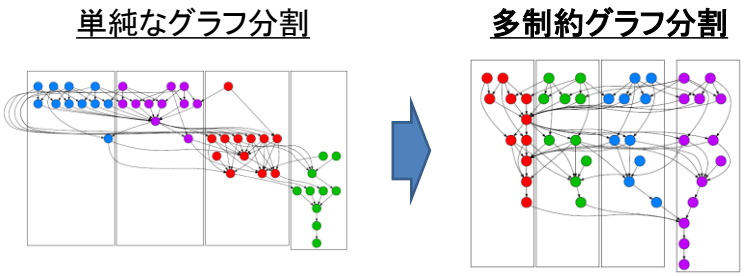

図 8.　多制約グラフ分割による並列性を保ちデータ移動を最小化するスケジューリング

開発しているPwrakeワークフローシステムに、この多制約タスク分割によるタスクスケジューリングを組み込み、性能評価を行った。その結果、データ転送量、ワークフロー実行時間を短縮させられることを確認した。従来研究では、入力ファイルの配置に基づいてスケジューリングを行うが、ワークフロー全体でデータ移動を最小化することにより、大幅なデータ移動の削減が可能となった。この成果は H24 年 6 月に IEEE/ACM クラスタ、クラウド、グリッドコンピューティングに関する国際会議（CCGrid）で発表した。

また、より大規模なワークフローについての実行を可能とするための検討を行った。具体的には、100 万コア、1 億タスクを想定した。この規模のワークフローの場合、ワークフローのタスクグラフをワークフローエンジンのサーバのメモリに収めることができなくなり問題となる。この問題を解決するため、階層的なワークフロー実行エンジンの設計を行った。第二階層のサーバにタスクグラフを分散配置することにより、100 サーバほどの利用で、目的を達成できることを示した。本成果は H24 年 8 月に情報処理学会 HPC 研究会で発表した。

H25 年度は、同一ノード内での複数のジョブ実行に関して、バッファキャッシュの再利用率を向上させ、ワークフローのジョブ実行時間を短縮するスケジューリング手法の提案を行った。既存手法において再利用率を向上させることは可能であるが、実際のワークフローに適用すると末尾タスクの実行において十分な並列性が得られず、逆にワークフロー全体の実行時間が延びてしまう。この末尾タスク問題を解決するために、二つのスケジューリング手法を提案した。一つ目は、末尾タスク問題が生じる直前まで従来手法である LIFO を用い、その後末尾タスク問題を生じないように Highest Rank First (HRF)に切り替える手法（LIFO+HRF）である。HRF は、バッファキャッシュを有効に再利用することができないが、一方で並列性を増やす方向で実行を進めるため末尾タスク問題を回避することが可能である。LIFO+HRF では、その両方式を組み合わせ、ほとんどの実行を LIFO で行うことでバッファキャッシュを有効活用し、かつ末尾タスク問題も解決することができる。二つ目は、ワークフローのそれぞれのステップにおいて、アプリケーションの性質が異なる可能性に着目し、その異なる性質のアプリケーションをなるべく同時に実行することにより、ノード全体における資源の有効活用を目指す手法である。たとえば、計算インテンシブなジョブとディスクインテンシブなジョブを同時に実行することにより、ディスクアクセスの競合を回避することが可能である。この手法では、ワークフローの各ステップにおけるジョブを、1 秒あたりに起動可能なジョブ数の割合に応じて実行する。この割合で実行することにより、各ステップのジョブをほぼ同時に完了することが可能となる。さらに、この手法だけだと末尾タスク問題が起こる可能性があるため、その問題が生ずる直前で HRF に切り替える（Rank Overlap+HRF）。この二手法を、天文データ解析のワークフロー Montage を用いてより詳細に評価した。LIFO+HRF、Rank Overlap+HRF の両手法は問題となっていたバッファキャッシュの有効利用が可能であり、最大既存手法と比べ 1.9 倍の高速化を達成した。さらに、末尾タスク問題も解決しており、96 コアのケースでは 12%の高速化を達成した。ノード当たりで処理するデータ量が増えるに従い、バッファキャッシュの有効利用の効果は高くなり、またコア数が増えるに従い末尾タスク問題の影響は大きくなる。今後のより大規模なデータ解析を高速化する重要なスケジューリング手法であることが分かった。本成果は H26 年 9 月に IEEE クラスタコンピューティングに関する国際会議（Cluster）において発表した。

H28 年度は、信頼性を向上させるため耐障害性の向上に関する研究を行った。これまでも、途中で中断したワークフローは、再実行することにより再開することが可能であったが、この再実行を

自動化する試みである。障害については、ワークフロータスクの一時的な障害、ノード障害、タスクのエラーを考慮する。Pwrakeワークフローシステムのマスターの障害については、マスターノードは1 ノードであるため、ワーカノードに比べ障害の可能性が極めて低く、手動で再実行することとする。タスクの一時的障害については、失敗したタスクを再実行する。他のノードで再実行しても異常終了する場合は、そのタスクにエラーがあると判断し、ワークフローを中断する。また、あるノードの通信の接続が切れる、あるいはそのノードで複数のタスクが異常終了した場合は、そのノードに障害が発生したと判断し、そのノードを切り離してワークフローを引き続き実行する。タスクを意図的に異常終了させた場合でも正しくワークフローを引き続き実行できることを確認した。また、ワークフローの入出力ファイルは Gfarm ファイルシステムに格納しているが、Gfarm ファイルシステムのサーバ障害についても評価を行った。Gfarm ファイルシステムのサーバ障害に対応するため、Gfarm ファイルシステム中のファイルに対し、ファイル複製を作成する設定にする必要がある。複製を作成した場合でもワークフローの実行時間が数%の増加でおさまること、また、Gfarm のサーバを意図的に異常終了させても正しくワークフローを引き続き実行できることを確認した。本成果は、H28 年 11 月に SC2016 併設のクラウド、グリッド、スーパコンピュータにおけるメニータスクコンピューティングに関する国際ワークショップ（MTAGS）で発表した。

バッチキューイングシステムの研究では、従来のバッチキューイングシステムの入力ファイルに対し、それぞれのジョブにおける入力ファイルを指定する方法を提案し、その情報を用いてファイルを格納しているノードにジョブを割り当てるスケジューラの設計を行った。さらに、複数ジョブから同一ファイルが参照される場合、自動的にファイル複製を作成する手法を提案した。これらの手法について、プロトタイプ実装を行い、評価を行った。20 プロセスがそれぞれ異なるファイルを読み込むワークロードでは、10 倍近くの性能向上を達成した。同一ファイルを 40 プロセスでアクセスするワークフローでは、67%の性能向上を達成した。

このスケジューラは I/O インテンシブなジョブを想定したスケジューリング手法であったが、CPU インテンシブなジョブについても扱うため、スケジューリング手法の拡張を行った。それぞれのジョブにおける入力ファイルからアクセス局所性の指標を求め、CPU の負荷と併せてパラメタ β によりノード割り当てを行う手法の設計を行った。バッチキューイングシステム Torque を元にしたプロトタイプシステムの実装を行い、BLAST ベンチマークで評価を行った。BLAST ベンチマークでは I/O インテンシブなジョブと CPU インテンシブなジョブの混合ベンチマークであるが、β の値を変えることにより効率的なスケジューリングを行うことが可能となることを示した。本成果は H27 年 5 月にグリーン、パーベイシブ、クラウドコンピューティングに関する国際会議（GPC）で発表し、IEEE Systems Journal への掲載が決定している。

H28 年度は、スケジューリング手法の改良を行った。これまでの手法は、CPU 負荷とアクセス局所性の指標を、パラメタ β を用い統合していたが、最適なパラメタ β の決定が難しかった。一方で、CPU 負荷については、動的な CPU 負荷を指標にするより、利用コア数に相当するスロット数で割り当てることで十分である。このことを利用し、提案手法の方針を見直した。具体的には、CPU 負荷については考慮せず、スロット数だけを考慮する。また、それぞれのジョブについて CPU インテンシブであるか、I/O インテンシブであるかを示す指標として RDR（Remote Degradation Ratio）を定義する。RDR はそれぞれのジョブに対し定義され、全てのアクセスがローカルストレージであった時のジョブの実行時間を L、全てのアクセスがリモートストレージであった時のジョブの実行時間を R としたとき、(R-L)/L で定義する。この指標は、リモートアクセスによるジョブの実行時間の延びの程度を示し、つまり I/O インテンシブな度合いを示している。改善したスケジューリング手法では、スケジューリングのスコアとして RDR にリモートにあるファイルサイズの割合をかけたものを用いる。スコアが大きい場合は、実行時間の延びが大きいことを示し、スコアが小さくなるようにスケジューリングを行う。この手法の評価を行い、これまでの手法より性能を向上させられることを示した。本成果は、H28 年 11 月に SC2016 併設のクラウドにおけるデータインテンシブコンピューティングに関する国際ワークショップ（DataCloud）で発表した。

★開発したソフトウェア
○公開済みのもの

・Gfarm ファイルシステム
Gfarm ファイルシステムは、本 CREST におけるスケールアウトする非均一アクセス性能を持つストレージアーキテクチャを想定したファイルシステムである。動作テストは Linux でなされ、多くの稼働実績がある。ダウンロード数は 18,000 を超えている。分散ファイルシステムであるため、計算機システムを利用する全てのユーザが利用可能であるが、特に、データインテンシブコンピューティング、ビッグデータ処理を行うユーザ、NFS では性能的に限界となっているユーザ、より高性能、高信頼性を求めるユーザ、また地理的に離れた複数拠点に複製を保持しディザスタリカバリを行いたいユーザなどが考えられる。H28 年度の成果により、InfiniBand の RDMA を用いた高速アクセスの成熟度を高め、Gfarm ファイルシステムに組み込んだ。

・Pwrake ワークフローシステム
Pwrake ワークフローシステムは、本 CREST における提案アーキテクチャを想定したワークフローシステムである。ワークフローは Rakefile により記述し、並列に実行可能なジョブは複数ノードで並列に実行される。ワークフローグラフを、本 CREST で提案した多制約タスク分割問題に帰着させ、自動的にその問題を解くことにより、データ移動が最小となるプロセススケジューリングを行う。ノード内では本 CREST での提案手法によりディスクキャッシュを有効利用し、末尾タスク問題を生じないようにスケジューリングする。動作テストは Linux でなされ、多くの稼働実績がある。データインテンシブアプリケーション、ビッグデータアプリケーションをはじめ、メールシステム、ビジネスインテリジェンスなど多くのアプリケーションで有用である。Rakefile でワークフローを記述するため、実行結果により処理が変わるようなダイナミックなワークフローなど、特に複雑なワークフロー処理を行いたい場合は唯一の候補となる。H28 年度の成果により、耐障害性が向上した。

★エクサスケールへの貢献度
エクサスケールのマシンでは、計算ノードのローカルストレージ、計算ノード側のストレージノード、並列ストレージ、グローバルストレージといった非均一アクセス性能を持つ階層的なストレージアーキテクチャとなっていくことが想定されている。本 CREST の研究成果はエクサスケールにおける階層的なストレージアーキテクチャにおけるシステムソフトウェアとして貢献する。またワークフローエンジンは需要が高まっているエクサスケールにおけるデータインテンシブアプリケーションの実行を支援し、エクサスケールマシンの応用範囲を広げる。

3．2 計算ノード OS（電気通信大学 大山グループ）

(1)研究実施内容及び成果

分散ファイルシステムの性能を最大限に引き出すための計算ノード OS の基盤技術を開発した。まず、分散ファイルシステムのアクセス性能をスケールアウトさせるための、カーネル機構によるローカルストレージへのアクセスを高速化する技術の研究開発を行った。また、クライアントキャッシュ技術および異なるノード間でデータ一貫性を保持するためのキャッシュ管理技術の研究開発を行った。さらに、ファイルシステムに高い負荷がかけられることにより計算ノード OS に生じる OS ジッタを実験的に評価し、その OS ジッタを削減する方法を構築した。カーネルドライバ、キャッシュ機構、OS ジッタ削減のそれぞれについて、具体的な研究内容を以下で述べる。

まず、分散ファイルシステムのクライアントインタフェースとして標準的に用いられている FUSE の機能や性能について調査を行った。一般にファイルシステムは、ファイルアクセスを高速化するために、一度読み込んだファイルデータをメモリ上にキャッシュしている。Linux ではこのキャッシュはページキャッシュと呼ばれる。一般に、FUSE を利用する場合、kernel_cache オプションをつけてファイルシステムをマウントすると、ページキャッシュを使用するのでファイルアクセスが高速化する。しかし、Gfarm ファイルシステムをマウントする際にこのオプションをつけることについては、2 つの問題があった。第一の問題は、POSIX コンシステンシはもとより close-to-open コンシステンシも保証されないことである。close-to-open コンシステンシとは、ファイルをクローズした後に、他のプロセスがそのファイルをオープンしたときに、そのクローズ前にそのファイルに対して行われた更新が観測できることを保証するセマンティクスである。複数のユーザが異なるマウントポイントに Gfarm ファイル

システムをマウントしている場合、キャッシュはマウントポイントごとに存在する。加えて、そのキャッシュが最新であるかどうかに関わらずキャッシュアクセスが行われるため、close-to-open コンシステンシを保証できない。第二の問題は、メモリが無駄に消費されることである。同一ファイルのキャッシュであっても、マウントポイントごとにキャッシュが作られるため、その分だけ無駄にメモリが消費される。その結果、Gfarm ではデフォルトでは kernel_cache オプションをつけずにファイルシステムをマウントするようになっている。この場合、ファイルをオープンするたびにキャッシュが破棄されるため、ページキャッシュが有効に利用されない。そこで、close-to-open コンシステンシを保証しながらページキャッシュを有効に活用できるようにする機構を提案した。提案機構では、複数のマウントポイントから Gfarm ファイルシステム上のファイルにアクセスしたときに、そのファイルのキャッシュを共有する。同一のファイルに対して複数のキャッシュを生成しないことにより、close-to-open コンシステンシが保証され、メモリ使用量を削減できる。提案機構は、ある端末が Gfarm のクライアントとして InfiniBand のような高速なネットワークを通じて Gfarm ファイルシステムにアクセスできるようになっており、そのクライアントに複数のユーザがリモートログインして利用するといった状況で特に有用となる。

次に、FUSE を用いて Gfarm ファイルシステムをマウントする場合に、ローカルストレージへのアクセスにおいて実行される冗長な処理の問題を解決する研究を行った。FUSE は、カーネルモジュールとユーザレベルデーモンで構成される。アプリケーションプロセスからファイルシステムへのアクセスにより、ユーザレベルデーモンへ要求が出される際に、コンテキストスイッチやメモリコピーが生じる。また、アプリケーションプロセスが read システムコールを発行してファイルのデータを要求すると、デーモンはファイルサーバへデータを取りにいく。もしクライアントと同一のノードのファイルサーバに目的のデータがある場合には、デーモンはローカルストレージにアクセスする。目的のデータがローカルストレージにある場合でも、Gfarm ファイルシステムを利用するプロセスは、必ずデーモンを介してデータへアクセスしなければならない。上述の通り、カーネルモジュールとユーザレベルデーモンの通信にはコンテキストスイッチとメモリコピーが伴う。ゆえに、プロセスがローカルストレージに直接アクセスする場合よりも I/O 性能は下がる。そこで、カーネルモジュールからローカルストレージへ直接アクセスすることを可能にする機構を提案した。提案機構では、デーモンがローカルストレージへアクセスする場合には、カーネルモジュールからデーモンへ要求を出すことはせずに、カーネルモジュールがローカルストレージに直接要求を出す。提案機構によってコンテキストスイッチやメモリコピーを削減することで、ファイルアクセスが高速化することが確認できている。図 9 は IOZone ベンチマークを用いて測定された、ローカルストレージに存在するファイルへのアクセス性能（バンド幅）である。全ての読み書きパターンにおいて、提案機構は元々のファイルアクセス性能に近い性能を達成していることがわかる。

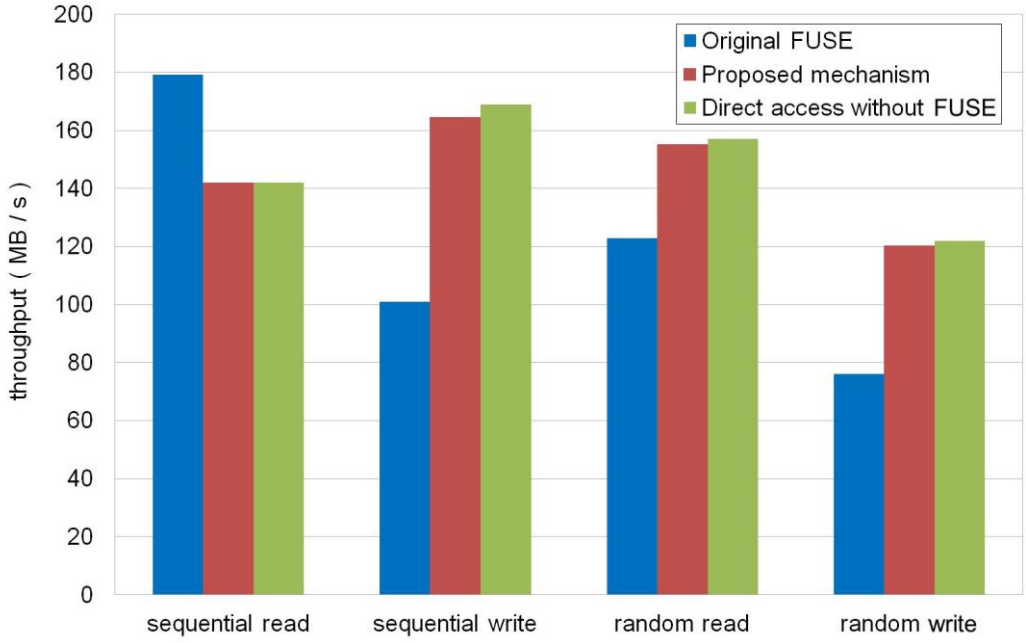

図 9. IOZone ベンチマークを用いて測定された，ローカルストレージに存在するファイルへのアクセスの性能

分散ファイルシステムのクライアントインタフェース（カーネルドライバ）が I/O サーバから高速にデータを取得するための技術についての研究も行った。高性能計算環境では InfiniBand と呼ばれる高速通信ネットワークデバイスが利用されることが多い。InfiniBand は RDMA (Remote Direct Memory Access)と呼ばれる高速通信機能を提供している。RDMA は、InfiniBand で結合されているあるノードが別のノードのメモリを直接読み書きすることを可能にする。RDMA を用いると、Ethernet などによるパケット転送を用いた場合に比べて、データを高速に転送することができる。InfiniBand 上でも、IPoIB (IP over InfiniBand) という仕組みを用いれば、TCP/IP のプロトコルで通信を行うことはできる。しかし、TCP/IP の利用には、TCP/IP のプロトコルスタックの処理のオーバヘッドなどが加わるという問題がある。分散ファイルシステムのクライアントインタフェースの高速化のために従来から採用されている方法は、その分散ファイルシステム専用のカーネルドライバを利用するというものである。カーネルドライバを用いた場合には、ユーザレベルのクライアントインタフェース（例えば FUSE ベースのもの）を用いた場合に比べて、プロセス間コンテキストスイッチ、カーネル-ユーザ間コンテキストスイッチ、ファイルデータのメモリコピーなどを少なくすることができる。本プロジェクトでも、Gfarm 専用のカーネルドライバを開発し、それによって、元々あった FUSE ベースのクライアントインタフェースよりも高速なファイルアクセスを実現している。しかし、カーネルドライバを導入してもなお、カーネル内に無駄なメモリコピーが残っていた。そしてそれは、InfiniBand の導入によって削減できることがわかった。我々は InfiniBand RDMA を用いて、サーバからクライアントのページキャッシュ領域に直接ファイルデータをゼロコピーで転送する機構を開発し、それを Gfarm に実装した。ページキャッシュ領域は OS がディスクなどから読み出したファイルデータを置くメモリ領域である。提案機構では、ファイルデータが最終的に配置されることになるクライアントのカーネル内メモリ領域に、サーバ側にあるファイルデータが直接書き込まれる。通常のカーネルドライバでは、クライアントの OS カーネル内で動作するカーネルドライバが、サーバから一旦ファイルデータを通信バッファに受け取った後に、それを OS がページキャッシュ領域にコピーする。提案機構では、このメモリコピーが除去されている。メモリコピーなどが除去されることにより、クライアントノードにおいて貴重である CPU 資源の消費も削減することができる。クライアントがサーバから自身のページキャッシュに InfiniBand RDMA によって直接ファイルデータを受け取る手法を実装し、実験を通じて有効性を示したのは、我々の知る限り、これが世界で初めての研究である。提案機構の有効性を実験により評価した。IOR ベンチマークによって基本性能を測定したところ、シーケンシャル

読み出しの性能が最大 78.4%向上することがわかった(図 10)。また、実際の科学技術アプリケーションの性能向上を測定する実験も行った。用いたアプリケーションは、天文画像を合成する Montage、すばる望遠鏡で撮影された画像を処理する SDFRED、ゲノム解析を行う NGS Analyzer-MINI である。実験の結果、ほぼすべての実行において、FUSE ベースのクライアントインタフェースや、元のカーネルドライバをしのいで、提案機構が最も高速な実行を実現できることがわかった。

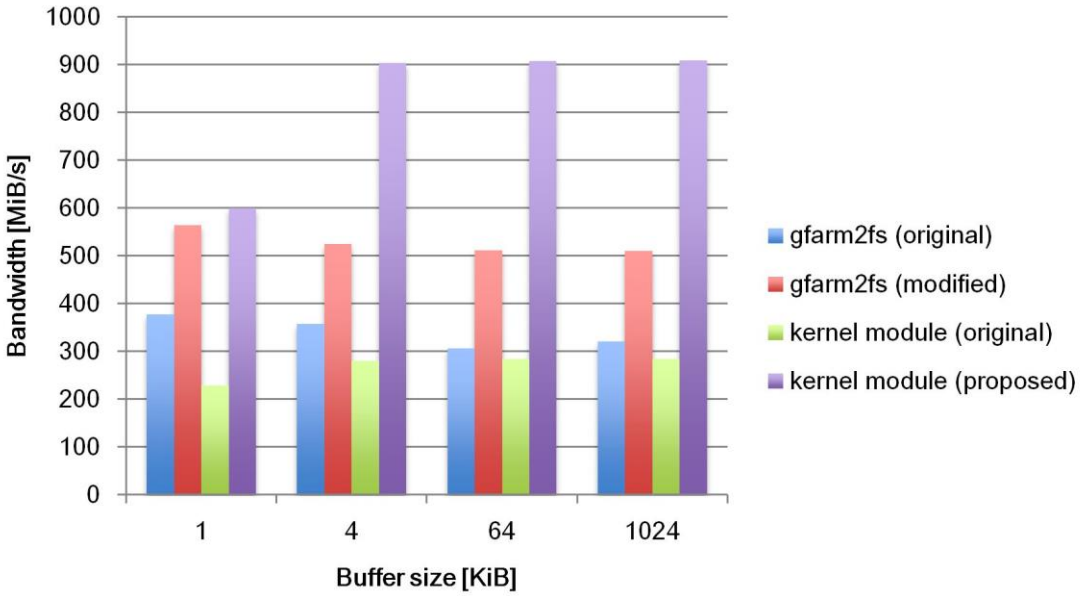

図 10. IOR sequential read ベンチマークを用いて測定した、分散ファイルシステムのサーバからクライアントへのファイル転送性能(カーネルドライバの使用の有無や InfiniBand RDMA の使用の有無を変えて測定)

この RDMA を最大限に活用したカーネルドライバ(クライアントインタフェース)および I/O サーバは、Gfarm を他のソフトウェアと組み合わせて利用する用途においても有用である。我々は、その用途の 1 つとして、仮想マシンのための準仮想化(para-virtual、 PV)ドライバを開発し、有効性を評価した。データインテンシブサイエンスの分野では Hadoop が広く用いられている。Hadoop の利用にあたっては、Hadoop がインストールされた仮想マシンクラスタがよく用いられる。ここで、Hadoop の実行の高速化のためには、Hadoop が動作する仮想マシンのディスク I/O を高速化することが必要である。また、Hadoop の実行をスケールアウトさせるためには、大量の仮想マシンイメージを格納することができる大容量のクラスタファイルシステムが必要となる。そこで、上で述べた InfiniBand RDMA を最大限活用してデータを転送する Gfarm を拡張し、RDMA を用いたスケーラブルな仮想マシンイメージ格納用クラスタファイルシステムを構築した。提案システムでは、PV ドライバにより、仮想マシン上でファイルの読み出しや書き込みが行われる度に、Gfarm のサーバとの間で InfiniBand RDMA によってデータが転送される。RDMA のゼロコピーデータ転送により、クライアントノードの CPU に負荷をほとんどかけることなく、高速にデータを転送できる。我々は仮想マシンモニタである QEMU/KVM を拡張して、Gfarm ファイルシステムに格納された仮想マシンイメージに対して仮想マシンが読み書きを行えるようにした。Gfarm と仮想マシンモニタを RDMA で接続したものと、TCP/IP で接続したものの両方を実装し、性能を比較した。その結果、シーケンシャル読み出しでもシーケンシャル書き込みでも、RDMA を用いた実装が最も高速であることを確認した。

キャッシュ機構についての研究を以下に述べる。キャッシュ機構の実現においては、クライアントノード上の貴重なメモリ資源をできるだけ節約しながらも、高性能を得られるようにすることが重要である。そこで我々は重複除外と呼ばれる技術に着目し、冗長なファイルデータを重複除外によって

除去し、メモリ消費量やストレージフットプリントを縮小したり、冗長なファイルデータへのアクセス回数を減らしたりする技術を開発した。具体的には、分散ファイルシステムのクライアントノードキャッシュ機構に重複除外を適用する技術の研究や、重複除外処理を GPU によって並列化する技術の研究や、分散ファイルシステムのサーバ側の処理に重複除外を適用する技術の研究を行った。重複除外機構はこれまでバックアップシステムで用いられることが多かったが、ファイルアクセス性能を向上させる可能性があることから、高性能計算やデータインテンシブ計算の分野でも活用されることが期待されている。しかし、実際に高性能計算などで用いられるファイルシステムに対して重複除外機構を実装し、現実的なアプリケーションや標準的なベンチマークによってその効果を実験的に評価した研究は少ない。

まず、クライアントノードのキャッシュに重複除外を適用する方法について研究を行った。データインテンシブサイエンスの分野における多くのアプリケーションは巨大なファイルの読み書きを実行する。巨大なデータはページキャッシュに載せられるため、多くのメモリを消費する。クライアントにキャッシュを作成する仕組みとしてまず考えられるのは、ファイルデータ全体を保存するという方法である。しかし、この方法には問題点が 2 つある。1 つはキャッシュの保持に必要なストレージ容量がファイルサイズに等しく大きいことである。もう1つは、ファイルデータが更新される度に、保持しているキャッシュデータを更新もしくは破棄しなければならず、ファイルの一部が頻繁に更新されるような場合においてはキャッシュの効果が発揮されないことである。そこで、Gfarm においてクライアントがファイルデータを可変長のブロック（チャンク）単位でキャッシュとして自ノードのメモリに保存し、それらを後のアクセスで利用するキャッシュ機構を提案した。提案機構の構成を図 11 に示す。ファイルの分割方法としては CDC (content-defined chunking)を採用した。CDC はファイルをその中身に基づいて可変長のチャンクに分割する方式である。提案機構の利点は、冗長性がある複数のデータを 1 つにまとめられる点である。すなわち、データの中身が同一であるような複数のファイル断片に対して、同一のチャンクをキャッシュとして利用できるようになる。提案機構により、キャッシュによるローカルストレージの消費量を抑えるとともに、サーバ-クライアント間のデータ転送量を減らすことができる。さらに、ファイルの中身に基づいてチャンクに分割するので、ファイルが更新された場合におけるキャッシュの更新を最小限に抑えることができる。また、過去にアクセスしたチャンクと中身が同一であるチャンクを含むファイルを転送する際にも、キャッシュが利用できる。提案機構はファイル読み出しを高速化するが、さらに、ファイル書き込みに付随するオーバヘッドも最小化している。そのための 1 つの方法は、チャンク管理表の非同期更新である、もう 1 つの方法は、ファイルデータの操作とハッシュの計算の並列実行である。我々は IOR ベンチマークを用いて提案機構の性能を評価した。また、気象予測のための科学技術アプリケーションのデータを対象に、重複除外によるデータ削減率を評価した。その結果、ファイルデータのランダム読み出しでは、最大 9.2 倍の高速化が達成された。データ削減率については、平均チャンクサイズを様々に変化させて測定した結果、どの場合も 30%以上の削減率が実現できた。この研究の貢献は 2 つである。第一の貢献は、CDC に基づいてファイルをチャンクに分割し、そのキャッシュをクライアントに配置する分散ファイルシステムのためのキャッシュ機構の設計と実装を示したことである。我々の知る限り、CDC に基づくキャッシュ機構をクライアントキャッシュに適用した研究は存在しない。第二の貢献は、提案機構によってどの程度性能が向上するのかを、広く用いられているベンチマークを用いた実験により明らかにしたことである。

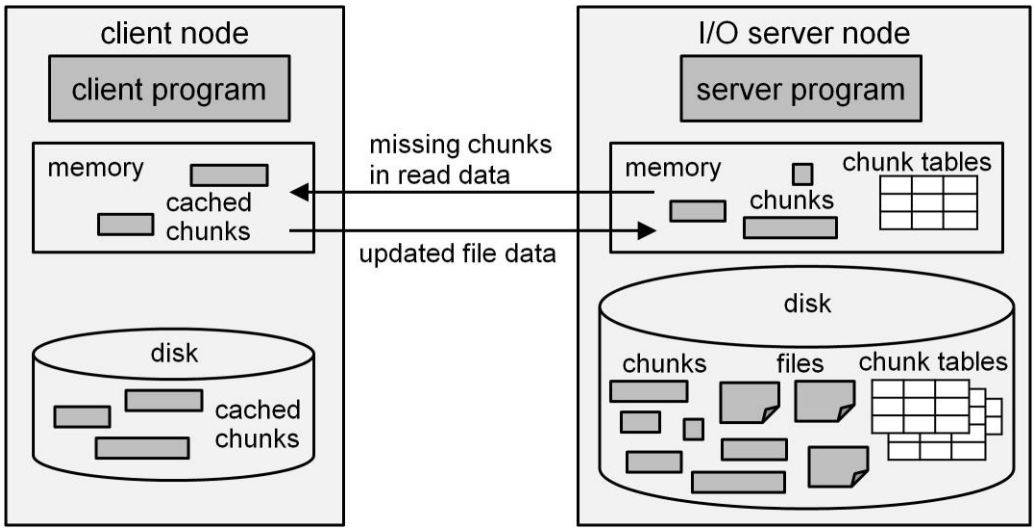

図 11. 重複除外を用いたクライアントキャッシュ機構の構成

次に、CPU と GPU を組み合わせることにより、CDC を高速化する手法を提案した。CDC に基づく重複除外機構の一つの課題は、CDC の処理すなわちファイルなどのデータをチャンクに分割する処理のオーバヘッドである。その処理が CPU サイクルやメモリを消費するとともに、その処理によってファイル操作のクリティカルパスが伸び、ファイル操作により多くの時間がかかるようになる。提案手法は 2 つの手法に分かれている。1 つは静的な負荷分散によるものであり、もう 1 つは動的な負荷分散によるものである。また、並列化に関しても 3 種類の手法を実装して比較した。第一の並列化手法は、CPU のみを用いて並列化するものである。第二の並列化手法は、GPU のみを用いて並列化するものである。第三の並列化手法は、CPU と GPU を組み合わせて並列化するものである。我々は Gfarm に提案手法を実装した。実験の結果、大きいサイズのファイルの読み出しでは、CPU と GPU の両方を用いるほうが、片方だけを用いるよりも高速であることがわかった。動的な負荷分散を用いた手法では、ワークスチールと呼ばれる技術を用いることによって、CPU を有効に活用できるようになることがわかった。

さらに、サーバ側に重複除外キャッシュ機構を導入し、その効果を測定する研究を行った。本研究グループの主な研究テーマはデータインテンシブサイエンスのための計算ノード OS である。そのため、まずはクライアントノード上重複除外キャッシュ機構の実装と評価に取り組んだ。ただ、その過程において、I/O サーバに重複除外キャッシュ機構を導入することの効果も評価し、クライアント側の機構との比較を行う必要があると判断した。クライアント、I/O サーバ、メタデータサーバのどのノードにどのようなファイルデータやキャッシュを配置し、どのようにそれらを管理するのが良いのかという問題に答えるのは単純ではなく、様々な実験データを元に議論すべきものである。そこで我々は分散ファイルシステム Gfarm のサーバ側の処理を重複除外機構で拡張したシステム ifarm を実装し、その性能を評価した。ifarm はインライン重複除外と呼ばれる方式を採用している。各 I/O サーバはローカルストレージから読み出したファイルデータをチャンクに分割する。また、各 I/O サーバはチャンクの情報を管理するためのチャンク表を管理する。I/O サーバがファイルに読み出しや書き込みを行う際には、適切なチャンクに対してその操作を適用する。また、チャンク表の適切な部分を参照、更新する。ifarm では、チャンク分割処理とクライアントアプリケーションの実行を非同期的に（並列に）実行できるようになっており、これによってアプリケーションの実行を高速化することができる。この非同期実行の導入後も、依然として、Gfarm が元々採用していた close-to-open コンシステンシは保証される。ifarm ではバージョン管理メモリを用いて、ファイルデータの更新が競合した場合でも、データの完全性を保証している。バージョン管理システムの Subversion で採用されている方法と同じく、チェックアウト、コミット、アップデートの 3 つの機能を用いて、データの完全性を保ちながら、データの並行な読み出しと書き込みを可能にしている。ifarm では、チャンクの GC も実装されている。一般に、重複除外をサポートするファイルシステムにおいては、ファイルの更新

やファイルの削除により、チャンクがどのファイルからも参照されなくなる（ごみになる）ことがある。ifarm では各チャンクに参照カウントを持たせている。チャンクを新たに参照する際には参照カウントをインクリメントし、ファイルの更新や削除によりチャンクを参照しなくなった際には参照カウントをデクリメントする。参照カウントがゼロになったとき、I/O サーバはチャンクをストレージデバイスから削除する。我々は IOR ベンチマークを用いて ifarm 利用時のファイル読み込みとファイル書き込みのスループットを測定した。その結果、チャンクの重複率が高いデータを用いる際に、ファイル読み込みスループットが元の Gfarm よりも向上することが確認された。また、IOR のファイル書き込み性能に関しては、元の Gfarm と ifarm との間に大きな差は出ないという結果が得られた。

分散ファイルシステム Gfarm に重複除外機構を適用する我々の一連の研究により、重複除外機構の適用による効果についての一定の知見を世に示すことができたと考えている。クライアントキャッシュに適用したときの性能向上、サーバに適用したときの性能向上、CPU のマルチコアや GPU を用いて並列化を行ったときの重複除外の高速化の度合いについて、定量的な実験結果を提示した。エクサスケールのファイルシステムやストレージを設計、実装したり、性能の予測や解析をしたりする際には、それらの実験結果を一つの指針として利用することができる。

キャッシュ機構に関しては、RDMA を用いた協調キャッシングについても研究を行った。我々は RDMA をクライアント−サーバ間のファイルデータの転送に用いるだけでなく、クライアント間のファイルデータの転送に用いることについても探求した。具体的には、InfiniBand で結合されたクラスタ上で動作するクライアントノード同士が分散ファイルシステムのキャッシュを共有する方法について研究を行った。これは協調キャッシング（cooperative caching）と呼ばれる方法であり、方法自体は 20 年以上前に提案されている。しかし、当時の計算機や通信デバイスの仕様は現在とは大きく違っており、また、InfiniBand RDMA も当時は存在しなかった。現代的な高性能計算の計算機環境において、協調キャッシングがどの程度有効であるかについては、全く明らかではないと言って良い。そこで我々は実際に Gfarm に協調キャッシング機構を実装し、実験により評価するという研究を行った。分散ファイルシステムに協調キャッシングを導入する場合、各クライアントがファイルデータのキャッシュをローカルノードに保持することになる。各クライアントはファイルデータを読みだす際に、元々の I/O サーバに加えて、別のクライアントもデータ提供元の候補として利用することになる。クライアントは他のクライアントにキャッシュの要求を出せるようにし、また、要求を受け付けることができるように、Gfarm のコードを拡張した。さらに、要求を受けたデータがローカルノードに存在する場合には、InfiniBand RDMA によって、あるクライアントのメモリから別のクライアントのメモリへゼロコピーで直接ファイルデータを転送できるようにした。高性能を志向した分散ファイルシステムに協調キャッシングを導入するにあたり、考慮すべきポイントは複数ある。まず、どのファイルのどの部分のキャッシュがどのクライアントノードに存在するかという情報を管理する必要がある。複数の管理の方法が考えられる。まず、分散ハッシュテーブルなどを用いて情報を完全に分散させて管理する方法がある。しかしこの方法は、スパコンのような高性能計算環境では、多くの通信のホップによりアクセスレイテンシが増加するという問題がある。もう一つは、分散ファイルシステム全体に対して単一の中央サーバを立て、そのサーバがすべてのキャッシュ情報をできるだけ精度良く把握するというものである。この方法は、キャッシュ情報の問い合わせや更新が頻繁になったときにサーバがボトルネックになるという欠点はあるものの、情報を高い精度で管理できるとともに、キャッシュ情報を得るまでのレイテンシも小さいことが予想されるという利点がある。よって、我々はまずこの方法を Gfarm に実装した。キャッシュ情報を管理する中央サーバはメタデータサーバと統合することとした。クライアントはファイルデータをサーバや他のクライアントから受信したときには、その情報を中央サーバに伝える。中央サーバは各クライアントから来る情報を統合し、各ファイルに対して、どのクライアントがどのくらいの回数のキャッシュ要求を出して、どのくらいのキャッシュデータが実際に転送されたかの統計情報を維持する。クライアントが中央サーバに、キャッシュを持っていそうなクライアントについての問い合わせを行った際には、この統計情報を元に、キャッシュを持っている可能性が一番高いクライアントを答える。この方法を Gfarm に実装し、評価を行った。上述した Montage、SDFRED、 NGS Analyzer−MINI の 3 つのアプリケーションを用いた実験の結果、協調キャッシングによって Montage において最大 5.8%の性能向上が確認できた（図 12）。全アプリケーション実行の全クライアントにおけるキャッシュのヒット率のうち最高だったものは 97.3%であり、平均ヒット率も常に

60%を超えていた。

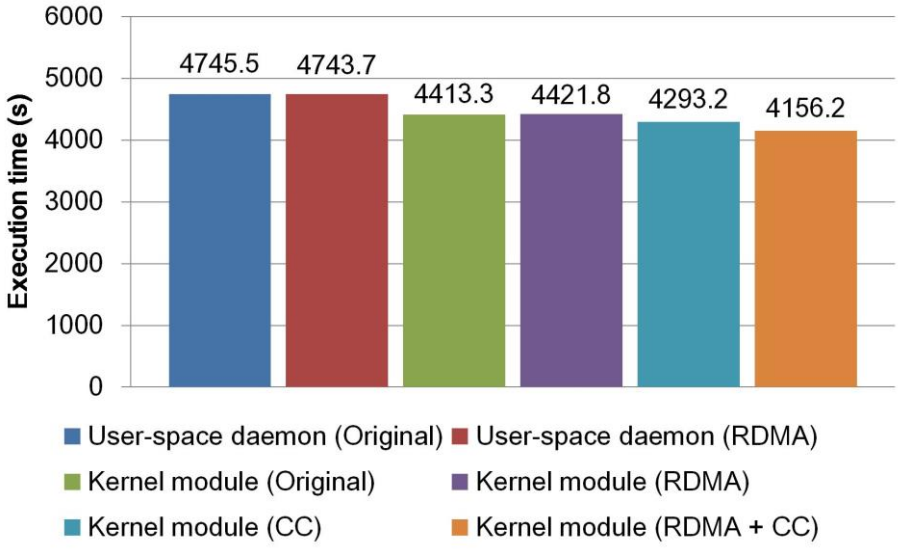

図 12. RDMA による高速ファイルデータ転送機構および協調キャッシング機構に関する設定を変化させて測定した、Montage ワークフローの実行時間

キャッシュの場所を管理する別の方法についても評価を行った。評価した方法は、ヒントベース協調キャッシングと呼ばれるものである。この方法では、クライアント各自が、自身の持つローカルな情報を元にキャッシュの場所を推測する。この方法では、あくまでキャッシュの場所に関するヒントのみを与える。この方法には、中央サーバに負荷が集中するという問題がない。ただし、ローカルな情報のみからキャッシュの場所を推測するため、キャッシュがある場所を予測する精度が高いとは限らない。我々は、キャッシュ管理のためのサーバを Gfarm の I/O サーバに配置してこの方法を実現する設計を検討した。通常、I/O サーバにはファイルの読み出しと書き込みに関する全ての要求が送られる。よって、I/O サーバはどのクライアントがどのファイルのどの部分にアクセスしたのかを詳しく把握することができる。これが I/O サーバにキャッシュ管理のためのサーバを配置する利点となる。各 I/O サーバノードに、キャッシュされたファイルデータのブロックについての情報を管理するための LRU リストを持たせ、そのリストを元にキャッシュを管理する。この設計に基づいて Gfarm のシミュレータを実装し、アプリケーションとして Montage を利用して評価を行った。その結果、提案方法によって高いキャッシュヒット率が実現できることがわかった。ノードによっては、90%以上のページが再利用できたことがわかった。

キャッシュを持つノードをベイズ的アプローチによって推測する方法についての研究も行った。この方法では、アプリケーションがどのデータをどのようなパターンで読んでいるかをベイズ統計の手法を用いて解析し、そのアプリケーションが動作しているノードと比べて似たデータを読んでいるノードを推定する。この方法においては、各ノードが、ある確率分布を計算する。この確率分布の距離をノード間の類似度として用いる。確率分布同士の距離を算出するには Kullback-Leibler 情報量（KL 情報量、KL 距離）を用いる。我々は、似たファイル読み出しパターンを持つノード同士の KL 距離が近くなるかどうかを確認する実験を行った。様々なパターンでファイルデータを読むノードを作成し、実際に KL 距離を求めた。その結果、似たデータを似たパターンで読んでいるノード同士の KL 距離は、全ての場合においてではないものの、近くなることがあることを確認した。

計算ノード OS 上で走るアプリケーションの性能を低下させるよく知られた要因として、OS ジッタがある。OS ジッタとは、アプリケーションの性能に影響を与えるような OS カーネルの処理やデーモンの処理である。高性能計算の分野においては長い間、OS ジッタの削減が高性能の実現のために重要だと言われ続けている。そこで我々は現代的なハードウェアと OS 上の実行おいて高性能科学技術計算のアプリケーションに対する OS ジッタの影響を測定するとともに、その OS ジッタによる

性能低下を最小化するための方法を構築した。科学技術計算アプリケーションと、多くの I/O 処理を実行するプログラムを同時に動かして性能を測定した。その結果、I/O 処理に伴い、大量のメモリがファイルのキャッシュに消費されることがわかった。そして、メモリが不足した際に、アプリケーションやファイルキャッシュからメモリを回収するためのデーモン（kswapd）が起動することがわかった。大量の I/O が高い頻度で要求される状況においては、このデーモンは頻繁に起動される。このデーモンの処理や、他のデーモン（migration デーモンなど）の処理が、多くの CPU 時間を消費することがわかった。我々の実験によると、気象予測アプリケーション WRF の性能が、I/O 処理を実行するシングルスレッドのプログラムの存在によって、数十%も低下した。これは、分散ファイルシステムのサーバの高い負荷によって科学技術アプリケーションの性能が大きく低下することを示唆しており、データインテンシブサイエンスのアプリケーションの高速実行のためには、解決する必要がある問題である。

そこで本研究では、OS カーネルのページ回収機構を拡張し、ページ回収用デーモンが起動する前に専用のカーネルモジュールが一度に大量のページを回収することにより、デーモンのオーバヘッドを最小化する機構を開発した。この機構は、ページ回収を開始する空きメモリ閾値を、元のページ回収用デーモンが用いている閾値よりも低いものにしているため、デーモンよりも先に実行を開始することができる。また、回収するページ数を非常に大きい数に設定できるため、この機構自体が動作する回数を少なくすることができる。この機構には、上で述べたように、空きメモリ閾値と回収ページ数の 2 つのパラメタを持つが、両方のパラメタともに、ユーザが環境に合わせてカスタマイズすることができる。なお、提案機構は、メモリがページキャッシュのために大量に消費されている状況で起動するようになっており、キャッシュではない通常のメモリオブジェクトを配置するためにメモリが大量に消費されている状況では、起動しないようになっている。メモリオブジェクトにメモリが使用されている場合には、OS が大量のページを回収することは、性能を上げる面からは逆効果になる可能性が高い。現在がどちらの状況にあるかについての判断は、ディスクアクセスの頻度を監視することによってなされる。単位時間あたりのディスク読み出しの回数が閾値を超えていたら、大量のメモリがページキャッシュに消費されているとみなしている。ページ回収のデーモンについては、カーネルが動作調整用のパラメタを、少ないながらも提供している。そのパラメタの調整によって、OS ジッタはある程度小さくすることができるが、提案機構と比べてどちらが有効であるかは全く自明ではなく、実験によって明らかにする必要があった。また、元々提供されているパラメタを用いてアプリケーションが高性能を出せるようにするには、パラメタの調整が難しい可能性もある。適切ではないパラメタを与えると、OS ジッタがあまり減らないことが予想される。それについても実験によって明らかにする必要があった。我々は気象予測アプリケーション、および、大量のファイル I/O を実行するプログラムを同時に実行した。そして、提案機構の動作の有無によって、そのアプリケーションの性能がどう変化するかを調査した。また、元々カーネルが提供しているパラメタを調整し、調整によって得られる性能と、提案機構によって得られた性能を比較した。その結果、提案機構を用いた場合に最高の性能が出ることがわかった。大量のファイルアクセスを実行するプロセスが存在する状況で気象予測アプリケーションである WRF を実行したときの，WRF の各ステップの実行時間を計測した結果を図 13 に示す。この結果より，元々の実行時間が OS ジッタによって大幅に伸び，また，提案機構によってその伸びが軽減されていることがわかる。また、さらなる実験を行い、カーネルが提供するパラメタの調整も強力ではあるが、そもそも調整に熟練を要する可能性が高いとともに、提案機構を超える性能を出すことが難しいことを示した。カーネルが提供するパラメタでは、ページ回収を開始する閾値と、一度に回収されるページ数が連動して変わるようになっており、片方だけを独立して変化させることができない。提案機構では独立して変化させることができる。これにより、提案機構では、閾値となる空きメモリ量を小さい値にしたまま、回収ページ数を非常に大きいものにすることができるようになっている。

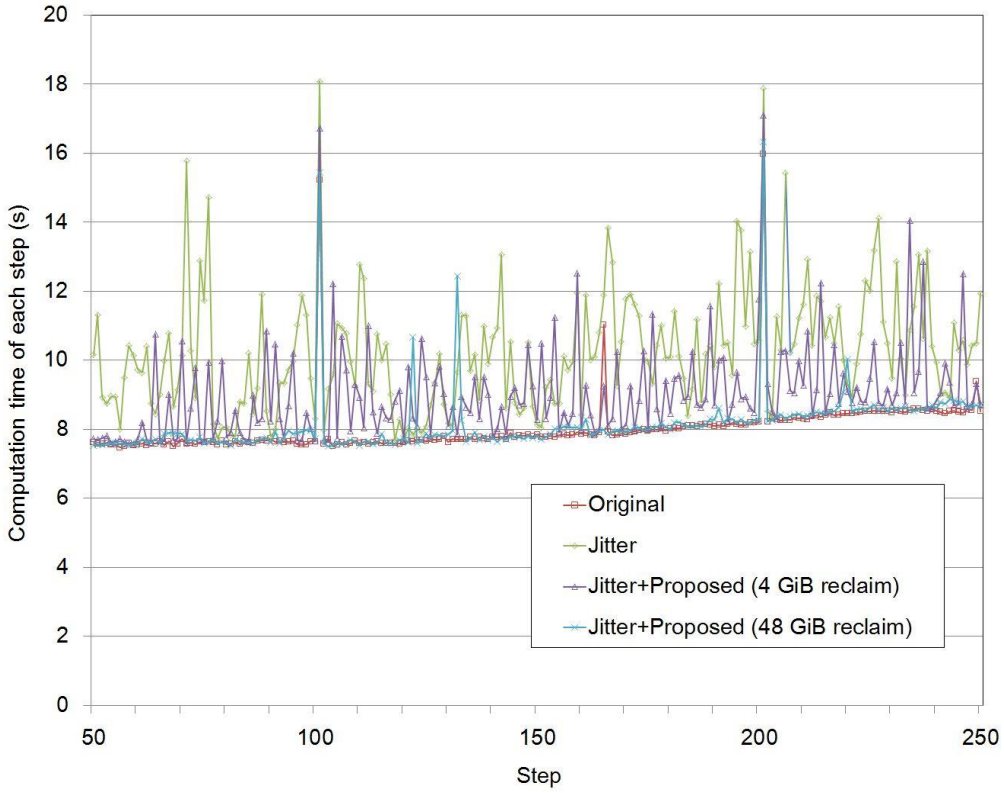

図 13. OS ジッタを生じさせるプロセスや提案機構が存在する設定と存在しない設定の下で測定した，気象予測アプリケーション WRF の各ステップの実行時間

我々は、分散ファイルシステムの性能を単に向上させることのみならず、高性能計算環境上でのデータインテンシブサイエンスの遂行において分散ファイルシステムを利用する様々なケースを想定し、使いやすいインタフェースを提供する研究も行っている。その一つとして、GPU コードから分散ファイルシステムにアクセスするためのインタフェースを設計、実装、評価する研究を行っている。そのインタフェースは分散ファイルシステムへのアクセスの処理の記述に関するプログラマの負担を減らすことができる。さらに、そのインタフェースを実装するライブラリには、そのインタフェースを経由することによるオーバヘッドを縮小する機構が入っている。具体的には、ファイルのオープン、リード、ライト、クローズといった API を GPU コードに対して提供し、ホスト側のコードを記述せずとも、GPU コードが分散ファイルにアクセスできるようにする。オーバヘッドを縮小する機構としては、まず、GPU のデバイスメモリなどにファイルデータのキャッシュを配置するものがある。また、GPUDirect RDMA を用いて、あるノードの GPU のデバイスメモリを他ノードが直接読み書きするものがある。予備実験を行った結果、提案するインタフェースを利用してもファイル読み出しの性能が低下せず、GPUDirect RDMA の効果によって逆に性能が上がる場合があることを確認している。

★開発したソフトウェア
○公開済みのもの
・Gfarm カーネルドライバ
Gfarm ファイルシステムをクライアントノード上でマウントする機能を持っているクライアントインタフェースである。従来から、Gfarm2fs という、マウントのための FUSE ベースのクライアントインタフェースは存在していた。このカーネルドライバは FUSE ベースのものと異なり、ユーザ空間ではなくカーネル空間で処理を行うため高速であるという特徴がある。Gfarm のクライアントマシン上で実行可能である。潜在的なユーザは Gfarm の利用において高いファイルアクセス性能を得たい人々である。

・RDMA ファイル転送モジュール

サーバノードとクライアントノードとの間が InfiniBand で結合されている場合に、InfiniBand の RDMA 機能を用いて高速にファイルデータを転送する機能を持っているソフトウェアモジュールである。このモジュールは、クライアントノードのカーネル内でメモリコピーを行うことなく、ファイルデータをファイル読み出しバッファに直接転送することを可能にする。このモジュールには、極めて高いファイルデータ転送スループットを実現できるという特徴がある。Gfarm のクライアントマシン上で実行可能である。潜在的なユーザは InfiniBand で結合されたクラスタ上での Gfarm の利用において極めて高いファイルアクセス性能を得たい人々である。

・OS ジッタ削減モジュール
高負荷のファイル I/O 処理が実行され、大量のメモリページがファイルのキャッシュに消費される状況において、メモリページの回収処理などが原因で発生する OS ジッタを最小化する機能を持つソフトウェアモジュールである。このモジュールが動作しておらず、カーネルパラメタのチューニングもしていない場合には、科学技術計算アプリケーションの性能が大きく下がることがわかっている。このモジュールには、その OS ジッタによる性能低下を非常に小さくするという特徴がある。Linux が動作するマシン上で実行可能である。潜在的なユーザは、高負荷のファイル I/O が行われる可能性がある環境で科学技術計算アプリケーションを高速に実行したい人たちである。

・ローカルアクセス最適化モジュール
FUSE ベースのクライアントインタフェースを利用して Gfarm ファイルシステムをマウントしている際、アクセスしようとしているファイルがクライアントのローカルストレージに存在する場合には、FUSE のユーザ空間デーモンを経由せず、カーネル内の処理のみでファイルを操作する機能を提供するソフトウェアモジュールである。このモジュールには、ローカルストレージ内のファイルに対するアクセスを高速化させるという特徴がある。Gfarm のクライアントマシン上で実行可能である。潜在的なユーザは Gfarm の利用において高いファイルアクセス性能を得たい人々である。

★エクサスケールへの貢献度
高スループットのファイル I/O の提供を支援するという形でエクサスケールの実現に貢献する。

§4 成果発表等

(1)原著論文発表 (国内(和文)誌 2 件、国際 (欧文) 誌 18 件)

1. Kenji Kobayashi, Shunsuke Mikami, Hiroki Kimura, Osamu Tatebe, “The Gfarm File System on Compute Clouds”, Proceedings of 1st International Workshop on Data Intensive Computing in the Clouds, pp.1034-1041, 2011 (DOI: 10.1109/IPDPS.2011.255)
2. Shunsuke Mikami, Kazuki Ohta and Osamu Tatebe, “Using the Gfarm File System as a POSIX compatible storage platform for Hadoop MapReduce applications”, Proceedings of 12th IEEE/ACM International Conference on Grid Computing (Grid), pp.181-189, 2011 (DOI: 10.1109/Grid.2011.31)
3. 大辻弘貴, 建部修見, アクセスパターンと回線遅延を考慮した遠隔ファイルアクセスの最適化, 論文誌コンピューティングシステム(ACS), 情報処理学会, No.4, Vol.4, pp.122-134, 2011
4. 木村浩希, 建部修見, MPI-IO/Gfarm:分散ファイルシステム Gfarm のための MPI-IO の実装と評価, 情報処理学会論文誌, No.52, Vol.12, pp.3239-3250, 2011
5. Masahiro Tanaka and Osamu Tatebe, “Workflow Scheduling to Minimize Data Movement using Multi-constraint Graph Partitioning”, Proceedings of IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.65-72, 2012 (DOI: 10.1109/CCGrid.2012.134)
6. Shun Ishiguro, Jun Murakami, Yoshihiro Oyama and Osamu Tatebe, “Optimizing Local File Accesses for FUSE-Based Distributed Storage”, Proceedings of the International Workshop on Data-Intensive Scalable Computing Systems (DISCS), pp.760-765, 2012. (DOI:

10.1109/SC.Companion.2012.104)
7. Yoshihiro Oyama, Shun Ishiguro, Jun Murakami, Shin Sasaki, Ryo Matsumiya, Osamu Tatebe, “Reduction of Operating System Jitter Caused by Page Reclaim”, In Proceedings of the 4th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS 2014), 8 pages, 2014. (DOI: 10.1145/2612262.2612270)
8. Masahiro Tanaka, Osamu Tatebe, “Disk Cache-Aware Task Scheduling For Data-Intensive and Many-Task Workflow”, Proceedings of IEEE International Conference on Cluster Computing (Cluster), pp.167-175, 2014 (DOI: 10.1109/CLUSTER.2014.6968774)
9. Hiroki Ohtsuji, Osamu Tatebe, “Network-based Data Processing Architecture for Reliable and High-performance Distributed Storage System”, Euro-Par 2015: Parallel Processing Workshops, Lecture Notes in Computer Science, Vol. 9523, pp.16-26, 2015 (DOI: 10.1007/978-3-319-27308-2_2)
10. Shin Sasaki, Kazushi Takahashi, Yoshihiro Oyama, Osamu Tatebe, “RDMA-based Direct Transfer of File Data to Remote Page Cache,” Proceedings of 2015 IEEE International Conference on Cluster Computing (Cluster), pages 214-225, 2015. (DOI: 10.1109/CLUSTER.2015.40)
11. Hiroki Ohtsuji, Osamu Tatebe, “Server-side Efficient Parity Generation for Cluster-wide RAID System”, Proceedings of 7th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp.444-447, 2015 (DOI: 10.1109/CloudCom.2015.25) (short paper)
12. Hiroki Ohtsuji and Osamu Tatebe, “Active-Storage Mechanism for Cluster-wide RAID System”, Proceedings of IEEE International Conference on Data Science and Data Intensive Systems (DSDIS), pp.25-32, 2015 (DOI: 10.1109/DSDIS.2015.101)
13. Shin Sasaki, Ryo Matsumiya, Kazushi Takahashi, Yoshihiro Oyama, Osamu Tatebe, “RDMA-based Cooperative Caching for a Distributed File System,” Proceedings of the 21st IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp.344-353, 2015 (DOI: 10.1109/ICPADS.2015.51)
14. Yoshihiro Oyama, Jun Murakami, Shun Ishiguro, Osamu Tatebe, “Implementation of a Deduplication Cache Mechanism using Content-Defined Chunking”, International Journal of High Performance Computing and Networking, Inderscience, Vol. 9, Issue 3, pp.190-205, 2016 (DOI: http://dx.doi.org/10.1504/IJHPCN.2016.076251)
15. Xieming Li and Osamu Tatebe, “Data-Aware Task Dispatching for Batch Queuing System”, IEEE Systems Journal (DOI: 10.1109/JSYST.2015.2471850) (to appear)
16. Yoshihiro Oyama, Shun Ishiguro, Jun Murakami, Shin Sasaki, Ryo Matsumiya, Osamu Tatebe, “Experimental Analysis of Operating System Jitter Caused by Page Reclaim”, The Journal of Supercomputing, Vol. 72, Issue 5, pp.1946-1972, May 2016 (DOI: 10.1007/s11227-016-1703-1)
17. Fuyumasa Takatsu, Kohei Hiraga, Osamu Tatebe, ″Design of object storage using OpenNVM for high-performance distributed file system″, Journal of Information Processing, Vol. 24, No. 5, pp.824-833, 2016 (DOI: http://doi.org/10.2197/ipsjjip.24.824)
18. Xieming Li, Osamu Tatebe, ″Improved Data-Aware Task Dispatching for Batch Queuing Systems″, Proceedings of the Seventh International Workshop on Data-Intensive Computing in the Clouds (DataCloud), 2016 (to appear)
19. Masahiro Tanaka, Osamu Tatebe, ″Fault Tolerance of Pwrake Workflow System Supported by Gfarm File System″, Proceedings of 9th Workshop on Many-Task Computing on Clouds, Grids, and Supercomputers (MTAGS), 2016 (to appear)
20. Fuyumasa Takatsu, Kohei Hiraga, Osamu Tatebe, ″PPFS: a Scale-out Distributed File System for Post-Petascale Systems″, Proceedings of IEEE International Conference on Data Science Systems (DSS), 2016 (to appear)

(2)その他の著作物(総説、書籍など)

(3)国際学会発表及び主要な国内学会発表

① 招待講演　　(国内会議 2 件、国際会議 1 件)

1. Osamu Tatebe, “File System for Post-Petascale Data-Intensive Computing”, International Supercomputing Conference, Hamburg, June 21, 2012
2. 建部修見, 不揮発性メモリ向けファイルシステムの設計, MemoryPlus Workshop, 東京, 2014 年 9 月 17 日
3. 大山恵弘, Linux のページ回収処理による高性能計算アプリケーションへの影響, MemoryPlus Workshop, 東京, 2014 年 9 月 17 日

② 口頭発表　　(国内会議 59 件、国際会議 31 件)

1. 大辻弘貴, 建部修見, アクセスパターンと回線遅延を考慮した遠隔ファイルアクセスの最適化, 先進的計算基盤システムシンポジウム SACSIS2011 論文集, pp.11-19, 秋葉原, 2011 年 5 月 25 日
2. 三上俊輔, 太田一樹, 建部修見, POSIX 準拠の広域分散ファイルシステム Gfarm 上での Hadoop MapReduce アプリケーション, 先進的計算基盤システムシンポジウム SACSIS2011 論文集, pp.181-188, 秋葉原, 2011 年 5 月 26 日
3. 平賀弘平, 建部修見, 分散ファイルシステムにおけるメタデータサーバの冗長化手法の検討, 情報処理学会研究報告, 2011-HPC-130(37), 7 pages, 情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 28 日
4. 木村浩希, 建部修見, MPI-IO/Gfarm におけるデータ配置を考慮したプロセススケジューリングの検討, 情報処理学会研究報告, 2011-HPC-130(33), 7 pages、情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 28 日
5. 大辻弘貴, 建部修見, Non-blocking RPC を用いた遠隔ファイルアクセスの最適化, 情報処理学会研究報告, 2011-HPC-130(30), 7 pages, 情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 28 日
6. 石黒駿, 村上じゅん, 大山恵弘, Gfarmのためのカーネルドライバへのキャッシュ機構導入の検討, 情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 28 日.
7. 村上じゅん, 石黒駿, 大山恵弘, 重複除外による Gfarm の性能向上に関する検討, 情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 28 日.
8. 田中昌宏, 建部修見, ワークフロー実行中のデータ移動を最小化するタスク配置方式, 情報処理学会研究報告, 2011-HPC-130(61), 8 pages, 情報処理学会 2011 年並列／分散／協調処理に関する『鹿児島』サマー・ワークショップ(SWoPP 鹿児島 2011), 鹿児島, 2011 年 7 月 29 日
9. 大辻弘貴, 建部修見, Non-blocking RPC を用いた遠隔ファイルアクセスの実装と性能評価, 情報処理学会研究報告, 2011-HPC-132(16) , 情報処理学会第 19 回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-19), 2011 年 11 月 28 日
10. 石黒駿, 村上じゅん, 大山恵弘, 広域分散ファイルシステム Gfarm におけるローカルストレージのファイルアクセスの高速化, 情報処理学会第 19 回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-19), 2011 年 11 月 28 日.

11. 村上じゅん, 石黒駿, 大山恵弘, Content-Defined Chunking を用いた重複除外キャッシュ機構による Gfarm の性能向上, 情報処理学会第 121 回システムソフトウェアとオペレーティング・システム研究発表会, 7 pages, 沖縄, 2012 年 5 月 8 日.
12. 田中昌宏, 建部修見, 大規模ワークフローに対応したPwrakeシステムの設計と実装, 情報処理学会第 135 回ハイパフォーマンスコンピューティング研究発表会, 2012-HPC-135(14), pp.1-7, 2012 年 8 月 1 日
13. 大西健太, 建部修見, 冗長符号を用いたサーバマシン間でのデータ分散手法の検討, 情報処理学会第 135 回ハイパフォーマンスコンピューティング研究発表会, 2012-HPC-135(22), pp.1-7, 2012 年 8 月 2 日
14. 鷹津冬将, 建部修見, 高速なストレージに適したログ構造化ファイルシステム MyLFS の設計と評価, 情報処理学会第 135 回ハイパフォーマンスコンピューティング研究発表会, 2012-HPC-135(27), pp.1-7, 2012 年 8 月 2 日
15. 平賀弘平, 建部修見, ノンブロッキングトランザクションに基づく分散ファイルシステムのための分散メタデータサーバの設計と実装, 情報処理学会第 135 回ハイパフォーマンスコンピューティング研究発表会, 2012-HPC-135(28), pp.1-9, 2012 年 8 月 2 日
16. 大辻弘貴, 建部修見, Infiniband を用いたファイルアクセスの高速化, 情報処理学会第 135 回ハイパフォーマンスコンピューティング研究発表会, 2012-HPC-135(29), pp.1-6, 2012 年 8 月 2 日
17. 石黒駿, 村上じゅん, 大山恵弘, 分散ファイルシステムを利用したアプリケーション実行時の I/O 競合の調査, 日本ソフトウェア科学会第 29 回大会, 11 pages, 東京, 2012 年 8 月 24 日.
18. 村上じゅん, 石黒駿, 大山恵弘, Content-Defined Chunking を用いた重複除外キャッシュ機構の実装と評価, 情報処理学会第 194 回 ARC・第 137 回 HPC 合同研究発表会(HOKKE-20), 7 pages, 北海道大学, 2012 年 12 月 13 日.
19. 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, 情報処理学会第 194 回 ARC・第 137 回 HPC 合同研究発表会(HOKKE-20), 2012-HPC-137(29), pp.1-6, 2012 年 12 月 14 日
20. 石黒駿, 村上じゅん, 大山恵弘, ページ回収に伴う OS ノイズの除去, 情報処理学会第 139 回ハイパフォーマンスコンピューティング研究発表会, 8 pages, 東京大学, 2013 年 5 月 29 日.
21. 鷹津冬将, 平賀弘平, 建部修見, 不揮発性デバイス向けの Object Storage の設計, 情報処理学会 2013 年並列／分散／協調処理に関する『北九州』サマー・ワークショップ(SWoPP 北九州 2013), 7 pages, 北九州, 2013 年 7 月 31 日.
22. Marilia Melo, Osamu Tatebe, “Fault Tolerance Design for Hadoop MapReduce on Gfarm Distributed Filesystem”, 情報処理学会 2013 年並列／分散／協調処理に関する『北九州』サマー・ワークショップ(SWoPP 北九州 2013), 5 pages, 北九州, 2013 年 7 月 31 日
23. 佐々木慎, 松宮遼, 高橋一志, 大山恵弘, Gfarm ファイルシステムへの Cooperative Caching の実装, 情報処理学会 2013 年並列／分散／協調処理に関する『北九州』サマー・ワークショップ(SWoPP 北九州 2013), 7 pages, 北九州, 2013 年 7 月 31 日.
24. 高橋一志, 佐々木慎, 松宮遼, 大山恵弘, ページキャッシュ最適化手法におけるベイズ的アプローチの検討, 情報処理学会 2013 年並列／分散／協調処理に関する『北九州』サマー・ワークショップ(SWoPP 北九州 2013), 9 pages, 北九州, 2013 年 7 月 31 日.
25. 李燮鳴, 建部修見, データ配置を考慮したタスクスケジューリング, 情報処理学会 2013 年並列／分散／協調処理に関する『北九州』サマー・ワークショップ(SWoPP 北九州 2013), 6 pages, 北九州, 2013 年 8 月 1 日.

26. 松宮遼, 平井成海, 佐々木慎, 高橋一志, 大山恵弘, CPU-GPU アーキテクチャを用いた Content-Defined Chunking の実装と評価, 第 25 回コンピュータシステム・シンポジウム (ComSys2013), pages 36-44, 東京, 2013 年 12 月 5 日.
27. Li Jiang, Hideyuki Kawashima, Osamu Tatebe, “Implementing Incremental Aggregate Computation on SciDB”, 情報処理学会システムソフトウェアとオペレーティング・システム(OS)研究発表会, 朱鷺メッセ, 2014 年 7 月 28 日
28. 松宮遼, 佐々木慎, 高橋一志, 大山恵弘, 並列分散ファイルシステムに対するオンライン重複除外機構の導入に向けて, 情報処理学会システムソフトウェアとオペレーティング・システム(OS)研究発表会, 8 pages, 新潟, 2014 年 7 月 28 日
29. 高橋一志, 佐々木慎, 松宮遼, 大山恵弘, 分散ファイルシステム Gfarm の協調キャッシュ機構の検討, 情報処理学会システムソフトウェアとオペレーティング・システム(OS)研究発表会, 8 pages, 新潟, 2014 年 7 月 28 日
30. 川島英之, 建部修見, 分析用データ処理系における効率的なデータ配送機構, 情報処理学会システムソフトウェアとオペレーティング・システム(OS)研究発表会, 朱鷺メッセ, 2014 年 7 月 29 日
31. 大辻弘貴, 建部修見, Cluster-wide RAID の実装と評価, 情報処理学会ハイパフォーマンスコンピューティング(HPC)研究発表会, 朱鷺メッセ, 2014 年 7 月 30 日
32. 佐々木慎, 松宮遼, 高橋一志, 大山恵弘. InfiniBand を利用した Cooperative Caching の実装と評価, 情報処理学会ハイパフォーマンスコンピューティング(HPC)研究発表会, 朱鷺メッセ, 2014 年 7 月 30 日
33. 鷹津冬将, 建部修見, WebGfarm: 分散ファイルシステム Gfarm の HTTP インタフェース, 情報処理学会ハイパフォーマンスコンピューティング(HPC)研究発表会, 朱鷺メッセ, 2014 年 7 月 30 日
34. 高橋一志, 佐々木慎, 大山恵弘. RDMA を用いた仮想マシンイメージストレージに向けて, 第 26 回コンピュータシステム・シンポジウム(ComSys2014), 芝浦工業大学, 2014 年 11 月 20 日
35. 大辻弘貴, 建部修見, Cluster-wide RAID 向けの集中型コントローラ, 情報処理学会ハイパフォーマンスコンピューティング(HPC)研究発表会, 小樽, 2014 年 12 月 2 日
36. 高橋一志, 佐々木慎, 大山恵弘, InfiniBand を用いた仮想マシンストレージ, 第 12 回 ディペンダブルシステムワークショップ(DSW 2014), 熱海, 2014 年 12 月 17 日.
37. 松宮遼, 佐々木慎, 高橋一志, 大山恵弘. ifarm: インライン重複除外機構を導入した並列分散ファイルシステム, 第 56 回プログラミング・シンポジウム, 伊東, 2015 年 1 月 11 日
38. Osamu Tatebe, “System Software for Post Petascale Data-Intensive Science”, Sapporo Summer HPC Seminar 2015, Sapporo, July 23, 2015
39. 川島英之, 建部修見, 縮退表現に基づくシーケンス演算処理の効率化, OS 研究発表会, SWoPP2015 別府, 2015 年 8 月 4 日
40. 三橋龍也, 川島英之, 建部修見, 環結合による類似検索の高性能化, HPC 研究発表会, SWoPP2015 別府, 2015 年 8 月 5 日
41. 堀尾健太郎, 川島英之, 建部修見, 暗号化データベースシステムにおける総和計算処理の並列化, HPC 研究発表会, SWoPP2015 別府, 2015 年 8 月 5 日
42. Li Jiang, Hideyuki Kawashima, Osamu Tatebe, “Applying Classification for Astronomical Data”, IPSJ SIGHPC, SWoPP2015 Beppu, Aug. 5, 2015
43. Mohamed Amin Jabri, Osamu Tatebe, “Design of concurrent B+Tree index for native KVS on non-volatile-memories”, IPSJ SIGHPC, SWoPP2015 Beppu, Aug. 5, 2015
44. 鈴木光一朗, 大山恵弘, SMB Multichannel の分散ファイルシステムへの適用

による並列 I/O 実現の検討, 2015 年並列／分散／協調処理に関する『別府』サマー・ワークショップ（SWoPP2015）, 6 pages, 別府, 2015 年 8 月 5 日.

45. 田中昌宏, 建部修見, 川島英之, 村田直郁, すばる HSC データ処理の性能調査と Gfarm/Pwrake の適用, HPC 研究発表会, SWoPP2015 別府, 2015 年 8 月 6 日

46. 神谷孝明, 川島英之, 建部修見, 並列ログ先行書き込みの評価, HPC 研究発表会, SWoPP2015 別府, 2015 年 8 月 6 日

47. 建部修見, Gfarm ファイルシステムの概要と実演, Gfarm ワークショップ, 神戸, 2015 年 9 月 4 日

48. 高橋一志, Gfarm カーネルドライバと RDMA による高速化, Gfarm ワークショップ 2015, 神戸, 2015 年 9 月 4 日.

49. 松宮遼, 髙橋一志, 大山恵弘, GPU コードから分散ファイルシステムにアクセスする試み, 日本ソフトウェア科学会第 32 回大会, 8 pages, 東京, 2015 年 9 月 11 日.

50. 桐井祐樹, 建部修見, Ross Robert, CODES/ROSS による分散ファイルシステムのための分散メタデータサーバ PPMDS の評価, 研究報告ハイパフォーマンスコンピューティング（HPC）, 2015-HPC-152(7), 9 pages, 札幌, 2015 年 12 月 16 日

51. Fuyumasa Takatsu, Kohei Hiraga and Osamu Tatebe, “Design of object storage using OpenNVM for high-performance distributed file system”, 2nd Annual Meeting on Advanced Computing System and Infrastructure (ACSI2016), Fukuoka, Jan. 18, 2016

52. Shin Sasaki, Kazushi Takahashi, Yoshihiro Oyama and Osamu Tatebe, “RDMA-based Direct Transfer of File Data to Remote Page Cache”, 2nd Annual Meeting on Advanced Computing System and Infrastructure (ACSI2016), Fukuoka, Jan. 20, 2016

53. 鷹津冬将, 平賀弘平, 建部修見, 高性能分散ファイルシステムのための分散メタデータサーバ PPMDS の評価, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2016-HPC-155(1), 11 pages, 松本, 2016 年 8 月 8 日

54. 田中昌宏, 建部修見, ワークフローシステム Pwrake における耐障害機能, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2016-HPC-155(11), 7 pages, 松本, 2016 年 8 月 8 日

55. Mohamed Amin Jabri, Osamu Tatebe, “Performance assessment of highly concurrent sorted linked list with good spatial locality (Unrefereed Workshop Manuscript)”, Vol. 2016-HPC-155(35), 4 pages, Matsumoto, Aug. 10, 2016

56. 建部修見, Gfarm ファイルシステムの分散メタデータサーバ設計, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2016-HPC-157(14), 7 pages, 那覇, 2016 年 12 月 22 日

57. 小林淳司, 建部修見, 並列離散イベントシミュレータを用いた分散メタデータサーバの性能評価, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2017-HPC-158(10), 9 pages, 熱海, 2017 年 3 月 8 日

58. 建部修見, 佐々木慎, 髙橋一志, 大山恵弘, Gfarm ファイルシステムにおける RDMA アクセスの設計, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2017-HPC-158(12), 6 pages, 熱海, 2017 年 3 月 8 日

59. 岩井厚樹, 建部修見, 田中昌宏, 並列ベンチマークのための同期複数タスク実行フレームワークの設計, 研究報告ハイパフォーマンスコンピューティング（HPC）, Vol. 2017-HPC-158(34), 8 pages, 熱海, 2017 年 3 月 10 日

（国際）

1. Shunsuke Mikami, Kazuki Ohta, Osamu Tatebe, “Hadoop MapReduce on the Gfarm

file system", Korea-Japan e-Science and Cloud Symposium, Korea, Dec. 15, 2011
2. Osamu Tatebe, "I/O requirement of CCS applications", EOFS Exa-scale I/O Interface Workshop, Tokyo, May 22, 2012
3. Hiroki Ohtsuji, Osamu Tatebe, "High Throughput, Low latency and Reliable Remote File Access", PRAGMA24, Bangkok, Thailand, Mar. 21, 2013
4. Yoshihiro Oyama, Shun Ishiguro, Jun Murakami, Shin Sasaki, Ryo Matsumiya, Osamu Tatebe. "Reduction of Operating System Jitter Caused by Page Reclaim", the 4th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS 2014), Munich, Germany, Jun. 10, 2014.
5. Masahiro Tanaka, Osamu Tatebe, "Disk Cache-Aware Task Scheduling For Data-Intensive and Many-Task Workflow", IEEE International Conference on Cluster Computing (Cluster), Madrid, Spain, Sep. 24, 2014
6. Hiroki Ohtsuji, "Network-based Storage Architecture for Exa-scale Computing Systems", PRAGMA 27 Workshop, Bloomington, Oct. 15, 2014
7. Jiang Li, Hideyuki Kawashima, Osamu Tatebe, "Incremental Window Aggregates over Array Database", IEEE International Conference on Big Data (BigData), Washington D.C., Oct. 30, 2014
8. Shin Sasaki, Ryo Matsumiya, Kazushi Takahashi, Yoshihiro Oyama, "Cooperative Caching for Gfarm file system", In Proceedings of the International Workshop on Modern Science and Technology 2014 (IWMST '14), pages 350-357, Wuhan, China, Oct., 2014
9. Ryo Matsumiya, Shin Sasaki, Kazushi Takahashi, Yoshihiro Oyama, "Why don't parallel file systems for HPC use online deduplication mechanisms?", In Proceedings of the International Workshop on Modern Science and Technology 2014 (IWMST '14), pages 358-366, Wuhan, China, Oct., 2014
10. Osamu Tatebe, "System Software for Data-Intensive Science", ICT International Exchange Workshop 2014, Tsukuba, Nov. 1, 2014
11. Ryo Matsumiya, Shin Sasaki, Kazushi Takahashi, Yoshihiro Oyama, "ifarm: Implementing Inline Deduplication to a Distributed File System", In Work-in-progress session in the 9th Parallel Data Storage Workshop (PDSW 2014), New Orleans, USA, Nov. 16, 2014
12. Shin Sasaki, Ryo Matsumiya, Kazushi Takahashi, Yoshihiro Oyama, "Performance Improvement of Gfarm Using InfiniBand RDMA", In Work-in-progress session in the 9th Parallel Data Storage Workshop (PDSW 2014), New Orleans, USA, Nov. 16, 2014
13. Osamu Tatebe, "System Software for Post-Petascale Data-Intensive Science", 2014 ATIP Workshop: Japanese Research Toward Next-Generation Extreme Computing, New Orleans, Nov 17, 2014
14. Osamu Tatebe, "System software for Post-Petascale Data-Intensive Science", JST/CREST International Symposium on Post Petascale System Software (ISP2S2 2014), Kobe, Japan, Dec. 3, 2014
15. Yoshihiro Oyama, "An Efficient Caching Mechanism for Post-Petascale Distributed Storage", JST/CREST International Symposium on Post Petascale System Software (ISP2S2 2014), Kobe, Japan, Dec. 3, 2014
16. Masahiro Tanaka, "Workflow system for data-intensive many-task computing", JST/CREST International Symposium on Post Petascale System Software (ISP2S2 2014), Kobe, Japan, Dec. 4, 2014
17. Osamu Tatebe, "System Software for Post-Petascale Data-Intensive Science", 10th AEARU Workshop on Computer Science and Web Technology, Tsukuba, Feb.

27, 2015

18. Xieming Li and Osamu Tatebe, “Data-Aware Task Dispatching”, the 10th International Conference on Green, Pervasive and Cloud Computing (GPC 2015), May 4, 2015
19. Fuyumasa Takatsu, Kohei Hiraga, and Osamu Tatebe, “Design of object storage using OpenNVM for high-performance distributed file system”, the 10th International Conference on Green, Pervasive and Cloud Computing (GPC 2015), May 4, 2015
20. Yuki Kirii, Hiroki Ohtsuji, Kohei Hiraga, Osamu Tatebe, “Simulation of PPMDS: A Distributed Metadata Management System”, Summer of CODES Workshop, July 13, 2015
21. Hiroki Ohtsuji, Osamu Tatebe, “Network-based Data Processing Architecture for Reliable and High-performance Distributed Storage System”, 4th Workshop on Big Data Management in Clouds (BigDataCloud), Vienna, Aug. 24, 2015
22. Shin Sasaki, Kazushi Takahashi, Yoshihiro Oyama, Osamu Tatebe, “RDMA-based Direct Transfer of File Data to Remote Page Cache,” 2015 IEEE International Conference on Cluster Computing (IEEE Cluster 2015), Chicago, Sep., 2015.
23. Hiroki Ohtsuji, Osamu Tatebe, “Breaking the Trade-off between Performance and Reliability of Network Storage System”, PRAGMA29, Depok, Indonesia, Oct. 7, 2015 (The Best Technical Talk Award)
24. Hiroki Ohtsuji, Osamu Tatebe, “Server-side Efficient Parity Generation for Cluster-wide RAID System”, 7th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Vancouver, Dec. 3, 2015
25. Hiroki Ohtsuji and Osamu Tatebe, “Active-Storage Mechanism for Cluster-wide RAID System”, IEEE International Conference on Data Science and Data Intensive Systems (DSDIS), Sydney, Dec. 11, 2015
26. Shin Sasaki, Ryo Matsumiya, Kazushi Takahashi, Yoshihiro Oyama and Osamu Tatebe, “RDMA-based Cooperative Caching for a Distributed File System”, The 21st IEEE International Conference on Parallel and Distributed Systems, Melbourne, Dec. 17, 2015
27. Junji Kobayashi, Osamu Tatebe, “Simulation study of distributed metadata server”, 2nd Summer of CODES Workshop, Argonne, Jul. 12, 2016
28. Osamu Tatebe, “Data Integrity support for Silent Data Corruption in Gfarm File System”, Storage Developer Conference, Santa Clara, Sep. 19, 2016
29. Xieming Li, Osamu Tatebe, ″Improved Data-Aware Task Dispatching for Batch Queuing Systems″, the Seventh International Workshop on Data-Intensive Computing in the Clouds (DataCloud), Nov., 2016
30. Masahiro Tanaka, Osamu Tatebe, ″Fault Tolerance of Pwrake Workflow System Supported by Gfarm File System″, Proceedings of 9th Workshop on Many-Task Computing on Clouds, Grids, and Supercomputers (MTAGS), Nov., 2016
31. Fuyumasa Takatsu, Kohei Hiraga, Osamu Tatebe, ″PPFS: a Scale-out Distributed File System for Post-Petascale Systems″, Proceedings of IEEE International Conference on Data Science Systems (DSS), Dec., 2016

③ ポスター発表　　（国内会議 11 件、国際会議 18 件）

1. 鈴木絵里香, 石黒駿, 村上じゅん, 大山恵弘, Gfarm へのキャッシュ機構の導入, 日本ソフトウェア科学会第28回大会, 沖縄, 2011 年 9 月 28 日.
2. 石黒 駿, 大山 恵弘, Gfarm におけるクライアントキャッシュの効率化手法の提案,

第 23 回コンピュータシステム・シンポジウム(ComSys2011), 京都, 2011 年 11 月 30 日.

3. 鷹津冬将, 建部修見, ログ構造化ファイルシステム mylfs の設計と初期評価, 第 10 回先進的計算基盤シンポジウム(SACSIS2012), pp.22-23, 神戸, 2012 年 5 月 16 日
4. 石黒駿, 村上じゅん, 大山恵弘, FUSE の高速化手法を適用した Gfarm の並列アクセス性能, 第10回先進的計算基盤システムシンポジウム(SACSIS2012), pp.26-27, 神戸, 2012 年 5 月 16 日.
5. 村上じゅん, 石黒駿, 大山恵弘, Gfarm のための CDC による重複排除キャッシュ機構の実装と評価, 第 10 回先進的計算基盤システムシンポジウム(SACSIS2012), pp.30-31, 神戸, 2012 年 5 月 16 日.
6. 石黒駿, 村上じゅん, 大山恵弘, ページフレーム回収に伴う OS ノイズの除去, 2013 年ハイパフォーマンスコンピューティングと計算科学シンポジウム(HPCS2013), p.86, 東京, 2013 年 1 月 15 日.
7. 佐々木慎, 大山恵弘, Gfarm ファイルシステムへの Cooperative Caching の実装, 第 11 回先進的計算基盤シンポジウム(SACSIS2013), pp.109-110, 仙台, 2013 年 5 月 23 日.
8. 高橋秀明, 高橋一志, 大山恵弘, ページ回収による OS ノイズを削減するシステム, 日本ソフトウェア科学会第 31 回大会, デモ・ポスターセッション, 名古屋, 2014 年 9 月 9 日
9. 高橋一志, 佐々木慎, 大山恵弘, InfiniBand を用いた仮想マシンストレージ, 第 12 回 ディペンダブルシステムワークショップ (DSW 2014), 熱海, 2014 年 12 月 17 日
10. 高橋一志, 佐々木慎, 大山恵弘, InfiniBand RDMA 向けの新たな準仮想化ドライバの提案, Annual Meeting on Advanced Computing System and Infrastructure (ACSI) 2015, つくば, 2015 年 1 月 27 日
11. 松宮遼, 遠藤敏夫, 大山恵弘, 深化する記憶装置階層のための大規模データ処理基盤の提案, 第 57 回プログラミング・シンポジウム, 伊東, 2016 年 1 月 8 日.

(国際)

1. Hiroki Kimura, Osamu Tatebe, "MPI-IO/Gfarm: An Optimized Implementation of MPI-IO for the Gfarm File System", Proceedings of 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.610-611, 2011 (DOI: 10.1109/CCGrid.2011.82)
2. Masahiro Tanaka, Osamu Tatebe, "Challenge of Pwrake, a workflow system for post-petascale data-intensive sciences", PRAGMA 21 Workshop, Sapporo, Oct. 18, 2011
3. Jun Murakami, Shun Ishiguro, Yoshihiro Oyama, Osamu Tatebe, "A Cache Deduplication Mechanism for Gfarm by Content Defined Chunking", The 2nd AICS International Symposium, Kobe, Japan, Mar. 1, 2012.
4. Hiroki Ohtsuji, Osamu Tatebe, "High Throughput Remote File Access with Infiniband", PRAGMA22, Melbourne, Australia, Apr. 19, 2012
5. Hiroki Ohtsuji, Osamu Tatebe, "Remote File Access with Infiniband RDMA", PRAGMA23, Seoul, Korea, Oct. 10, 2012
6. Osamu Tatebe et al., "System Software for Post Petascale Data Intensive Science", ACM/IEEE SC12 research exhibition, Nov. 11-16, 2012
7. Hiroki Ohtsuji, Osamu Tatebe, "High Throughput, Low latency and Reliable Remote File Access", PRAGMA24, Bangkok, Thailand, Mar. 21, 2013
8. Ryo Matsumiya, Kazushi Takahashi, Yoshihiro Oyama, Osamu Tatebe, "Content-defined Chunking for CPU-GPU Heterogeneous Environments", The 12th USENIX Conference on File and Storage Technologies (FAST '14), Santa

Clala, USA, Feb. 18, 2014.

9. Hiroki Ohtsuji, Osamu Tatebe, “POSTER: Preliminary evaluation of optimized transfer method for cluster-wide RAID-4”, Proceedings of IEEE International Conference on Cluster Computing (Cluster), pp.284-285, Sep. 23, 2014 (DOI: 10.1109/CLUSTER.2014.6968775)
10. Yoshihiro Oyama, “An Efficient Caching Mechanism for Post-Petascale Distributed Storage”, In poster session of JST/CREST International Symposium on Post Petascale System Software (ISP2S2 2014), Kobe, Japan, Dec. 2, 2014
11. Osamu Tatebe, “System software for Post-Petascale Data-Intensive Science”, International Symposium on Post Petescale System Software (ISP2S2), Kobe, Dec. 2, 2014
12. Yoshihiro Oyama, “An Efficient Caching Mechanism for Post-Petascale Distributed Storage”, International Symposium on Post Petescale System Software (ISP2S2), Kobe, Dec. 2, 2014
13. Masahiro Tanaka, “Pwrake - a workflow system for data-intensive science”, International Symposium on Post Petescale System Software (ISP2S2), Kobe, Dec. 2, 2014
14. Fuyumasa Takatsu, Kohei Hiraga, and Osamu Tatebe, “Object Storage for OpenNVM flash primitives”, International Symposium on Post Petescale System Software (ISP2S2), Kobe, Dec. 2, 2014
15. Fuyumasa Takatsu, Kohei Hiraga, and Osamu Tatebe, “Object Storage for ioDrive”, 10th AEARU Workshop on Computer Science and Web Technology (AEARU-CSWT 2015), Tsukuba, Feb. 26, 2015.
16. Xieming Li and Osamu Tatebe, “Proposal of task dispatch exploiting data locality for batch queuing system”, 10th AEARU Workshop on Computer Science and Web Technology (AEARU-CSWT 2015), Tsukuba, Feb. 26, 2015
17. Hiroki Ohtsuji and Osamu Tatebe, “New Generation Distributed Storage System Architecture for Exascale Era”, 10th AEARU Workshop on Computer Science and Web Technology (AEARU-CSWT 2015), Tsukuba, Feb. 26, 2015
18. Osamu Tatebe, Tomoteru Yoshie, “Data Integrity for Silent Data Corruption in Gfarm File System”, 21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015), Okinawa, Apr. 13 and 14, 2015

(4)知財出願

なし

(5)受賞・報道等

①受賞

Li Xieming, “Design of Data-Aware Task Scheduler”, PRAGMA Student Workshop, Best Technical Award, October 16, 2013

Hiroki Ohtsuji, Osamu Tatebe, “Breaking the Trade-off between Performance and Reliability of Network Storage System”, PRAGMA29, Depok, Indonesia, Oct. 7, 2015 (The Best Technical Talk Award)

②マスコミ(新聞・TV等)報道(プレス発表をした場合にはその概要もお書き下さい。)

③その他

(6)成果展開事例

①実用化に向けての展開

＜公開可能なもの＞

- CREST における研究成果の普及とサポートを行うため、2013 年 4 月 26 日に特定非営利活動法人つくば OSS 技術支援センターを設立した。
- 開発したプログラム「Gfarm ファイルシステム」「Gfarm MPI-IO プラグイン」「Gfarm Hadoop プラグイン」について、HP（URL; http://sourceforge.net/projects/gfarm/）にて公開中。
- 開発したプログラム「Gfarm ファイルシステム」は文部科学省が進める「京」を中核とする革新的ハイパフォーマンス・コンピューティング・インフラ（HPCI）の HPCI 共用ストレージとして使われている。
- 開発したプログラム「Pwrake ワークフローシステム」を、HP（URL; http://github.com/masa16/Pwrake/）にて公開中。

②社会還元的な展開活動

- Gfarm シンポジウム、Gfarm ワークショップを開催し、本研究で開発した Gfarm ファイルシステム、Pwrake ワークフローシステムなどについて紹介した。
- ACM/IEEE スーパコンピューティングに関する国際会議 SC において研究展示を行った。

§５ 研究期間中の活動

５．１　主なワークショップ、シンポジウム、アウトリーチ等の活動

年月日	名称	場所	参加人数	概要
2013 年 9 月 19 日	特定非営利活動法人つくば OSS 技術支援センター設立記念シンポジウム	東京	30 人	研究成果発表、成果普及促進
2014 年 12 月 2 日～4 日	JST/CREST International Symposium on Post Petascale System Software	神戸	200 人	研究成果発表、共同研究促進
2014 年 12 月 16 日	Gfarm シンポジウム 2014	東京	32 人	研究成果発表、成果普及促進
2015 年 7 月 23 日	Sapporo Summer HPC Seminar	札幌	11 人	研究成果発表
2015 年 9 月 4 日	Gfarm ワークショップ 2015	神戸	17 人	研究成果発表、成果普及促進
2015 年 12 月 14 日	Gfarm シンポジウム 2015	東京	27 人	研究成果発表、成果普及促進
2016 年 10 月 21 日	Gfarm ワークショップ 2016	神戸	20 人	研究成果発表、成果普及促進

§６ 最後に

研究開発当初の目標はおおむね達成し、想定以上の成果を上げることができた。解決すべきポストペタスケールのストレージシステムの研究課題については、それぞれ解決方法を提案し、プロトタ

イプシステムによる実証、評価を行った。これらの成果は、国際会議、ジャーナルで発表するとともに、オープンソースで実運用システムとして開発を続けている Gfarm ファイルシステムに取り込み、性能向上を図った。ただし、Gfarm ファイルシステムの安定版への取り込みについては、コードの成熟度を上げ広範囲にわたるテストをクリアしなければならない。実運用システムでの利用においては、システム停止やデータ消失につながる不具合などがあってはならないためである。そのため、限られた研究期間では研究成果の取り込みがなかなか難しい。この辺りは、実用化に向けての支援が必要である。

CREST 研究は、研究の進め方について自由度が高く、極めて多くの進展があった。チーム内の全体ミーティングは毎月 1 度開催し、年に 1 度はハッカソンスタイルの研究合宿を開催した。以下の写真は 2015 年 6 月 3 日～5 日に開催した研究合宿で撮影した全体写真である。

ハッカソンスタイルの研究合宿における全体写真