Vol.2017-HPC-162 No.11
2017/12/18

Oracle Storage Cloudの性能評価とGfarmファイルシステムへの組み込みの検討

北澤 昂大[1,a)]　建部 修見[2]

概要：大規模データの管理方法として，近年クラウドストレージが注目され，研究分野のみならず，一般企業や個人においても大量のデータを扱うことが増えてきた．大規模データの需要とともに，効率の良い利用が求められる．研究分野で活用されている，大規模データの処理や共有を効率的に行う分散ファイルシステムとして Gfarm がある．そこで本稿では，最初の試みとして Oracle Storage Cloud を用いクラウドストレージの性能評価と Gfarm への組み込みの検討を行った．性能評価においては，ネットワークがボトルネックのためにブロックサイズや多重アクセス，システムの設定などの条件次第で性能が大きく変わることを確認した．

1. はじめに

ストレージ技術の進歩により，一般企業や個人においても，大量のデータを扱うことが容易になって来ており，研究分野としての活用だけでなく，顧客情報の管理や処理などのマーケティング分野にも活用が進んでいる．ビッグデータへの需要は大きいものとなってきており，今後ますます，膨大な資源の共有や効率の良い利用方法が求められる．

大規模データの管理方法として，複数のストレージデバイスによってデータを管理する分散ファイルシステムを利用する方法がある．ひとつのストレージデバイスにおいて大規模なデータを管理することができないという物理的な問題の解決や，分散したデータの一元管理，また拡張性・可用性のために利用される．研究分野では，巨大なデータの管理・処理のために，Gfarm [1] などの分散ファイルシステムが採用されている．

また，大規模データの管理方法として，近年クラウドストレージが注目されている．クラウドストレージの特徴は，ネットワークを通して遠隔地のストレージにアクセスすることができ，それがサービスとして抽象化されユーザに提供されている点である．クラウドストレージの利点としては，「データを分散管理することで可用性がある」「データ容量に拡張性が有る」「インフラストラクチャーのサービスとして提供され，環境構築や保守管理の手間がかからない」などが挙げられ，個人でストレージを保守・管理するコストが減った．そして近年の通信技術の進歩に伴い，現在クラウドストレージは一般企業や個人で広く普及している．

しかし，クラウドストレージは，ローカルのストレージを利用するのと比べ，ネットワークが性能のボトルネックとなっている．加えてクラウドストレージのシステムの設定によっては性能が大きく変化し得る．データを単に書き込んだり読み込んだりするだけでなく，クラウドストレージにある大量のデータに処理を行いたい場合も同様の課題がある．また，接続のためにクラウドストレージが提供しているインターフェース（HTTP を用いた REST [2] API など）に合わせる必要があり，汎用性に課題がある．さらに，大規模データとなると費用面でのコストも問題にあげられる．そこで，ローカルのストレージとクラウドストレージの利点を持ちつつ，両者を一元管理し，かつ効率的に大規模データの処理を行いたい．

効率的な大規模データの並列分散処理を行う分散ファイルシステムとして，筆者らが開発を進めている Gfarm ファイルシステムがある．Gfarm ファイルシステムは，主に研究分野での大規模データ処理に利用されている分散ファイルシステムである．この Gfarm ファイルシステムのアーキテクチャの大きな特徴は，広域に分散する PC クラスタノードなどの CPU とローカルディスクの両方を利用し広域仮想ファイルシステムと並列分散処理を提供することにある．この Gfarm ファイルシステムにおけるアーキテクチャの利点を活用しつつ，ローカルのディスクだけではなくクラウドストレージの接続も統合することで，クラウド

[1] 筑波大学情報学群情報科学類
University of Tsukuba, School of Informatics, College of Information Science
[2] 筑波大学計算科学研究センター
University of Tsukuba, Center for Computational Sciences
[a)] kitazawa@hpcs.cs.tsukuba.ac.jp

ストレージとしての利点も持ち得る，大規模データをより容易に扱えるような仮想ファイルシステムを実現することを目指している．

このファイルシステムが実現すれば，拡張性の高さと管理コストの低さが利点のクラウドストレージと合わせ，大規模データの保守管理・また活用がより容易になる．さらには広域に分散した資源を仮想ファイルシステムにより，インターフェースによらずに一元管理しつつ，最適な分散処理を行うことができる．これは研究分野だけでなく，一般企業や個人でのビッグデータの活用という需要と合わせても，価値がある．

ただし先程述べたようにクラウドストレージはローカルのストレージを利用するのと比べ，ネットワークが性能のボトルネックとなっている．また，サービスとしてディスクやストレージシステムなどの環境がブラックボックスとなっている．システムの性能に影響する部分をサービスに委ねることとなり，アプリケーションの設計や資源の効率的な利用の面からは好ましくないと言える．そのため，パフォーマンス管理が必要になる．

そこで本論文では，分散ファイルシステム Gfarm において，クラウドストレージをサポートする広域仮想ファイルシステムの設計と実装について議論するために，クラウドストレージサービスの 1 つである Oracle Storage Cloud [3] に対して性能評価と，Gfarm への組み込みの検討を行った．性能評価を通して，以下を目的・貢献とする．

- クラウドストレージサービスとして隠蔽されたストレージの性能と，ネットワークがどの程度ボトルネックとなるかを明らかにする．
- アプリケーションやシステムに組み込み運用することを前提とした上で，性能が変化し得るクラウドストレージのアクセス・設定を調べる．また，Gfarm 上での運用を考えた際に性能や分散透明性を高めるため，クラウドストレージごとに最適な処理を行うことを目指す．
- ローカルストレージとともに利用することを考えた際に，クラウドストレージの最適な活用方法を検討する．
- クラウドストレージも多様化しているため，性能を示すことで，Gfarm によらず実運用検討時の採用の判断材料となる．

本論文は，5 章で構成される．第 2 章では，関連研究について述べる．第 3 章では，準備としてクラウドストレージについて述べる．第 4 章では，Oracle Storage Cloud の性能評価と考察について記述する．第 5 章では性能評価を受けて Gfarm の組み込みの検討を行う．そして第 6 章で，結論と今後の課題について述べる．

2. 関連研究

大規模データの効率的な管理や処理を行うための技術として，階層型ストレージ [4] [5] がある．階層型ストレージは，データの持つ価値に注目している．アクセス性能や価格，容量の異なるストレージを階層的に配置し，データをアクセス頻度や要件に合わせて適切な媒体に格納することによって，各媒体が持っているコストパフォーマンスやアクセス性能の利点を最大限に生かすようなストレージである．また，ストレージの自動階層化がある．ストレージ間のデータの移動は，あるポリシーに基づいて自動的に行われる仕組みで，ユーザーは階層化したストレージを意識すること無く統合されたストレージにアクセスが出来る．そのポリシーとして Information Lifecycle Management （ILM） [6] がある．大規模の情報を効率的に活用するために，時間経過による情報の価値の変遷に基づきデータの管理方法を決定する考え方である．Gfarm も大規模データの効率的な管理や処理を目的とし，多数のストレージを統合する機能を有すが，広域に分散したストレージのデータの共有仮想ファイルシステムとして機能している．そのストレージの対象として，アクセス性能や運用コストなど性質の異なるクラウドストレージの統合を新たに検討し，大規模データのより効率的な活用を目指している．

3. 準備

準備として，クラウドストレージの概要と，実験に利用した Oracle Storage Cloud について述べる．

3.1 クラウドストレージ

Gfarm への統合を視野に入れて，現在広く普及しているクラウドストレージについて述べる．

一般に広く普及しているクラウドストレージとして，Amazon S3 [7], Google Cloud Storage [8], Oracle Storage Cloud, OpenStack Swift [9] などがある．ストレージの形式としては，オブジェクトストレージ，ファイルストレージ，ブロックストレージがあり，サービスによってその提供は様々だが，オブジェクトストレージとしての利用をサポートしたものが多く見られる．オブジェクトストレージのアクセスには，一般に REST API を利用する．REST API は，サービスによって多様であるが，事実上の業界標準として OpenStack Swift, Amazon S3, CDMI [10] が浸透している．Gfarm への統合を考えた時に，このようなサービスによって異なるインターフェースに対応しなければならない点がまず課題として挙げられる．また，クラウドストレージの性能もサービスによって異なることが予想され，それぞれのストレージシステムなどはブラックボックスとなっている．したがって効率の良いデータ利用・アプリケーションへの適用を考えた時に，クラウドストレージの性能を評価し管理していく必要がある．本論文では，まず最初の試みとして Oracle Storage Cloud に焦点を当てている．

3.2 Oracle Storage Cloud

Oracle Storage Cloud は，Oracle 社が提供するパブリッククラウドサービスの 1 つである．同サービスの別体系のストレージとして，アクセス頻度の低いデータを格納することを目的とした Oralce Storage Cloud Archive があるが，統合を視野に入れた際にアクセス性能の面から現実的でないため採用しない．よって本論文では Oralce Storage Cloud Archive に言及しない．Oracle Storage Cloud は，オブジェクトストレージとして提供され，接続インターフェースとして，OpenStack Swift に準拠した REST API を利用する方法と，NFSv4 を介してアクセスできる Oracle Storage Cloud Software Appliance [9] （以下 OSCSA と呼ぶ）というシステムを利用する方法がある．OSCSA は REST API によるオブジェクトストレージへのアクセスを隠蔽し，POSIX 準拠の NFS インターフェースを提供するシステムである．本論文の Oracle Storage Cloud の性能評価では，OSCSA を利用している．OSCSA が設定として提供している，性能に影響を与え得るものとして以下がある．

- OSCSA のインストール先サーバにアタッチされたローカルストレージに保持されるキャッシュ及びキャッシュサイズ．読み込み性能に大きく影響を与える．デフォルトは 300 GiB またはローカルストレージの残量に依存する．
- プリフェッチ機能．1 MiB 単位で指定ができる．キャッシュミスした際の読み込み性能に影響を与える．デフォルトでは無効になっている．
- Sync Policy．OSCSA のファイルシステムに対する同期，非同期書き込みを制御できる．POSIX Standard 設定（fsync , ODSYNC , OSYNC）と，Acynchronous 設定がある．デフォルトでは POSIX Standard 設定が指定されている．

したがって，上の要素も性能評価を行う際に考慮する．

4. 実験：Oracle Storage Cloud の性能評価

Oracle Storage Cloud の性能評価を行う．

4.1 実験環境

実験環境を**表 1** に示す．

表 1 実験環境

クラウドストレージの データセンタ	http://uscom-central-1. storage.oraclecloud.com
研究室のサーバから データセンタへの遅延	平均遅延 148.737 ms
研究室のサーバの OS	CentOS7
研究室のサーバの CPU	Intel（R） Xeon（R） CPU QuadCore E5620 @ 2.40GHz*2
研究室のサーバのメモリ	24 GB（4 GB*6）
研究室のサーバの HDD	500 GB （SAS HDD /2.5 インチ）
研究室のサーバのイーサネット	Broadcom 10 Gigabit
クラウドストレージへのアクセス方式	OSCSA による NFSv4 アクセス
OSCSA のインストールノードと NFS クライアントノード	同ノード

4.2 実験ツール

OSCSA を利用すると NFSv4 でのアクセスが可能なため，実験ツールとして fio [10] を利用した．fio は，ファイルシステム及びブロックデバイスの I/O 性能を測定するためのツールである．同じような I/O 性能の測定のために簡易的に使われるツールとして，LINUX の dd コマンドがあるが，fio は，より細かい条件を指定できるため採用した．ブロックサイズの指定や，マルチスレッドによる並列処理，1 スレッドにおける非同期 I/O の連続発行数を指定する iodepth オプションなど，様々な条件で測定を行うことが出来る．

4.3 性能評価の目的と方法

Gfarm に組み込むことを視野に入れた性能評価を行う．Gfarm は，メタデータサーバ，ファイルシステムサーバ，クライアントで構成されファイルシステムサーバは，Gfarm 仮想ファイルシステムによって束ねられている実際のローカルファイルシステムへのアクセスのために利用されるサーバである．また，ファイルシステムサーバは，各ファイルシステムを提供するノード上で実行される．このような Gfarm のアーキテクチャ上で，クラウドストレージの統合を検討した際に考えられる性能のボトルネックは，ファイルシステムサーバーとクラウドストレージ間のネットワークである．また，ファイルアクセスのアルゴリズムや，サービスとして隠蔽されている要素すなわち，サービスが提供するクラウドストレージと実体のディスク間の読み込み・書き込みシステムである．クラウドストレージのパフォーマンスは，ネットワークがボトルネックのためにシステムの設定やファイルアクセス方法などによる性能の振れ幅が大きいと考えられ，可能な限りの最適化が必要である．したがってネットワーク性能で飽和しない限り，ファイルアクセスのバンド幅を限界まであげるための要素を調査しながら性能評価を行う．

ディスク I/O パフォーマンスに影響を与える要因として，次が挙げられる．

- ブロックサイズ
- 多重アクセス
- Oracle Storage Cloud が提供する設定項目（3.2 節より）

以上を踏まえた上で，以下の方法で実験を行う．

(1) ブロックサイズの変更
ブロックサイズを変更しながら，大容量データの書き込み及び読み込み性能を測定する．

(2) クラウドストレージへの多重アクセス
多重アクセスには，2 通りのケースが有る．
- 1 つの同じ大容量ファイルに対して多重アクセスで性能を上げる方法
- 複数の大容量ファイルに対して多重アクセスで性能を上げる方法

また，多重アクセスとして，2 つの方法が考えられる
- 非同期 I/O を利用する方法
- 複数スレッド（プロセス）を用いる方法

したがって同期 I/O と非同期 I/O において，大容量データの書き込み及び読み込みの性能を調べる．また複数スレッドにおいて，複数のファイルデータの書き込み及び読み込みの性能を調べる．この時のブロックサイズは，（1）の結果を考慮した値を指定する．

(3) サービス側が提供しているセッティング
サービスが提供している設定項目を変えながら，性能を調べる．
- OSCSA インストールサーバのローカルストレージに保持されるキャッシュサイズを変化させ，性能を調べる．
- キャッシュミスが起こり得る状態で，プリフェッチ機能の設定を変化させたときの，読み込みの性能差を調べる．
- Sync Policy の POSIX Standard 設定と，Acynchronous 設定を切り替えて，書き込みの性能差を比較する．

4.4 性能評価の結果と考察

4.3 節に沿い，性能評価の結果と考察を行った．

4.4.1 ブロックサイズによる性能評価

ブロックサイズを 4 K, 8 K, 16 K, 32 K, 64 K, 128 K, 256 K, 512 K, 1 M, 2 M, ... , 1,024 MiB と変え，5 GiB のデータの読み込み・書き込み性能にどんな影響をおよぼすか調査する．ただし，バンド幅の平均が 25 MiB/s 未満となるものは，200 秒で実験を切り上げるものとする．Oracle Storage Cloud が提供する設定項目はデフォルトの設定とし，1 スレッドでの同期 I/O を利用する．

以上の要件での結果を示す．**図 1** は書き込みにおける，ブロックサイズによるバンド幅の性能である．ブロックサイズが十分小さいとき，性能は著しく悪い．また，ブロックサイズの増加による書き込み性能の増分は大きいものとなる．およそ 70 MiB/s ほどの書き込み性能で限界に達しており，32 MiB のブロックサイズから 1 GiB のブロックサイズまでは性能にあまり差が見られない．ブロックサイズの設定は，性能に大きく影響を及ぼす．

同様に**図 2** は読み込みにおける，ブロックサイズによるバンド幅の性能である．表 1 のネットワーク環境から，fio のシーケンシャル read を行った時は，ローカルのキャッシュ[*1]が効いていると思われる．ランダム read を行った時は，65,536 KiB のブロックサイズまではキャッシュミスが起こっていると思われ，その時のバンド幅は 8.8 KiB/s ～ 99.2 MiB/s であり，ブロックサイズによって書き込みと同じく増加が見られた．ローカルキャッシュを読んでいる場合も blocksize の増加に対して効果があると言える．

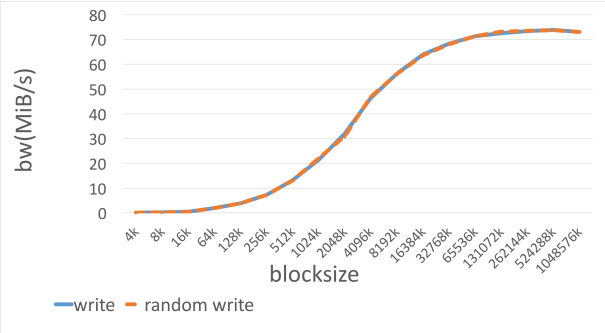

図 1 ブロックサイズによる書き込みバンド幅の性能

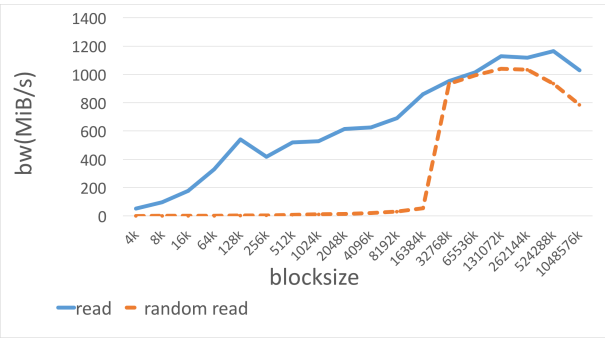

図 2 ブロックサイズによる読み込みバンド幅の性能

4.4.2 多重アクセスによる性能評価: 非同期 I/O

非同期 I/O を利用し，5 GiB のデータの読み込み・書き込み性能を調査する．非同期 I/O として，POSIX Asynchronous I/O（POSIX AIO）[11] と Linux Asynchronous I/O（libaio）[12] を利用し，それぞれの性能を調査する，同時に発行する I/O 数を 1 ～ 15 に変え，性能にどんな影響をおよぼすか調査する．Oracle Storage Cloud が提供する設定項目はデフォルトの設定とする．ブロックサイズは 4.4.1 節の結果を踏まえて，128 K, 512 K, 2,048 K, 8,192 K, 65,536 KiB とする．

まず POSIX AIO における性能調査の結果を示す．ブロックサイズが 128 KiB, 512 KiB, 2,048 KiB のときは，非同期 I/O（POSIX AIO）同時発行数（iodepth）による性能が横ばいだった．**図 3**，**図 4** は書き込みと読み込みにおける，非同期 I/O（POSIX AIO）同時発行数（iodepth）によるバンド幅の性能である．一例としてブロックサイズは 512 KiB のものを示す．ブロックサイズが 8,192 KiB のときと 65,536 KiB のときは，書き込み性能に変化が見られた．8,192 KiB のブロックサイズでの書き込み及び読み込

[*1] OSCSA のローカルキャッシュは任意のタイミングで削除ができない仕様のため

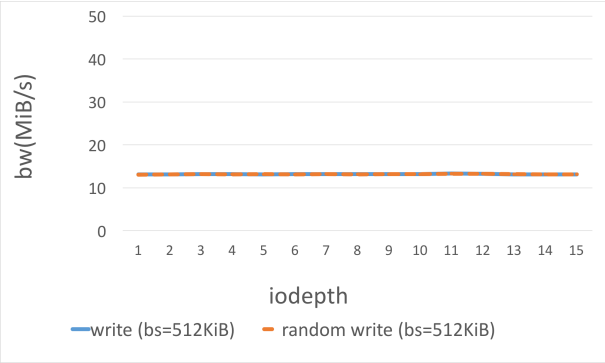

図 3 非同期 I/O(POSIX AIO) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 512 KiB)

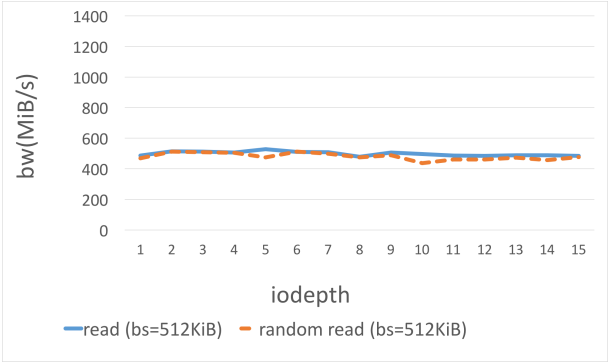

図 4 非同期 I/O(POSIX AIO) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 512 KiB)

み時のバンド幅の性能を**図 5**，**図 6** に示す．また，65,536 KiB のブロックサイズでの書き込み及び読み込み時のバンド幅の性能を**図 7**，**図 8** に示す．書き込み性能に関して，4.4.1 節の同期 I/O の性能評価や図 3 と比べ，バンド幅の最大値が 1.5 割程度上がっている．ブロックサイズが十分に大きい状態だと非同期 I/O（POSIX AIO）の同時発行は意味をなすといえる．また iodepth も 10 以上から性能が徐々に上がりはじめており，大きく取る必要があるようである．ただし，読み込みにおいてはあまり性能が上がらなかった．POSIX AIO は POSIX で定義されているインターフェースを利用して実装されたユーザレベルの非同期 I/O で，スレッドを複製することで多重アクセスを行っている．十分なバンド幅がありかつ，少ない多重度である場合，すぐに処理が終わってしまい非同期 I/O と同期 I/O の待ち時間に大差が無かったために，性能があがらなかったと考えられる．また，読み込みにおいてブロックサイズが 65,536 KiB の時は性能が右肩下がりとなった．これはブロックサイズと元々のバンド幅が大きいために性能が飽和してしまったものと考えられる．

次に libaio における性能調査の結果を示す．POSIX AIO との比較のため，ブロックサイズは 512 KiB, 8,192 KiB, 65,536 KiB のものをそれぞれ**図 9**，**図 10**，**図 11**，**図 12**，**図 13**，**図 14** に示す．読み込みでは，iodepth による大幅な性能の向上が見られた．ただし，性能からローカルキャッシュが効いているものと思われる．書き込みでは，iodepth による性能は横ばいであった．libaio は Linux システムコールによる非同期 I/O で NFS のサポートもあるが，あくまでも OSCSA は REST API でのアクセスを隠蔽しているだけのため，クラウドストレージにアクセスする過程で同期 I/O として処理されたと考えられる．書き込みは同時に非同期 I/O（libaio）を複数発行してもクラウド側（OSCSA）に受け入れられず，読み込みはローカルキャッシュにデータが存在していたことにより，ローカルキャッシュの読み込みに対して非同期 I/O が効き性能があがったと考えられる．

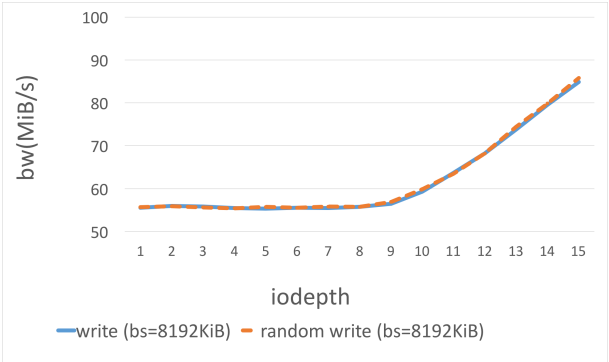

図 5 非同期 I/O(POSIX AIO) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 8,192 KiB)

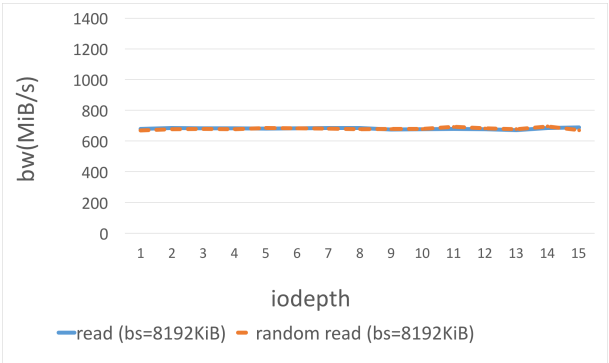

図 6 非同期 I/O(POSIX AIO) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 8,192 KiB)

4.4.3 多重アクセスによる性能評価: 複数スレッド

スレッド数に応じて分割した計 5 GiB のデータに対する読み込み・書き込み性能を調査する．同期 I/O を利用する．スレッド数を 1 〜15 に変え，計 5 GiB のデータに対して読み込みと書き込みを行い，性能にどんな影響をおよぼすか調査する．Oracle Storage Cloud が提供する設定項目はデフォルトの設定とする．ブロックサイズは 4.4.1 節の結果を踏まえて，128 K, 512 K, 2,048 K, 8,192 K, 65,536 KiB とする．ブロックサイズが 128 KiB, 512 KiB, 2,048 KiB のときはスレッド数が増えるに従って性能の向上が見られた．一例としてブロックサイズが 512 KiB のときの

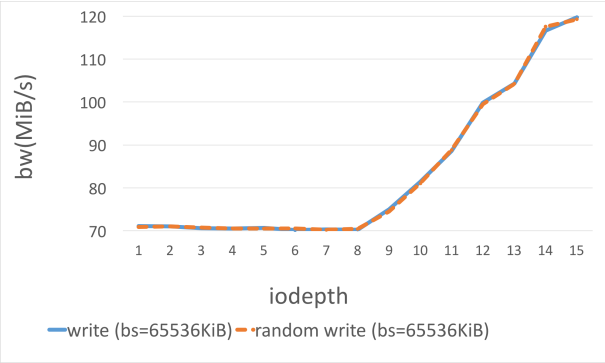

図 7 非同期 I/O(POSIX AIO) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 65,536 KiB)

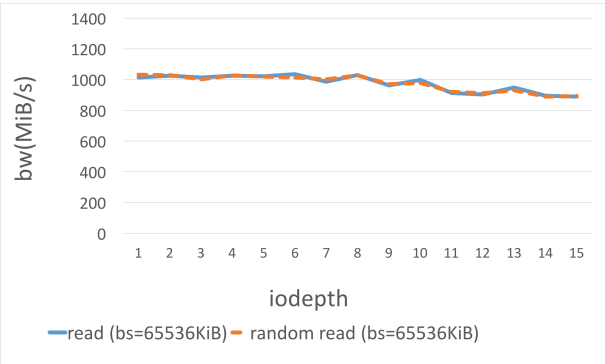

図 8 非同期 I/O(POSIX AIO) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 65,536 KiB)

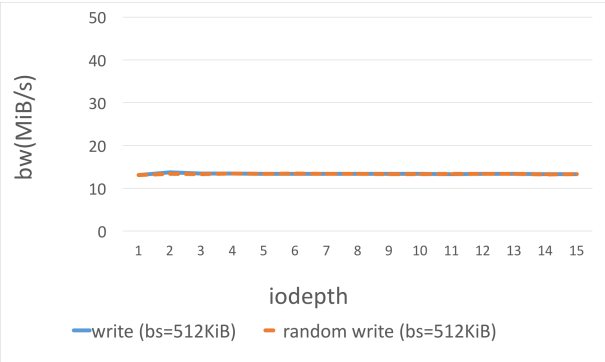

図 9 非同期 I/O(libaio) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 512 KiB)

図 10 非同期 I/O(libaio) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 512 KiB)

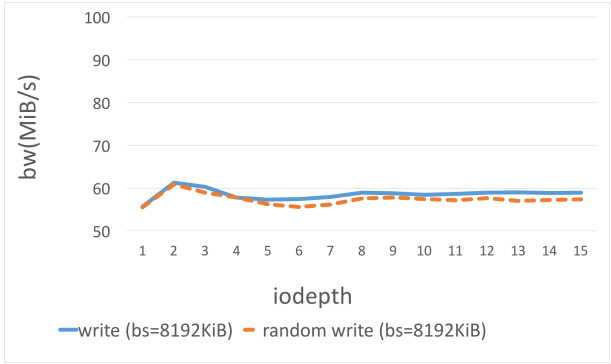

図 11 非同期 I/O(libaio) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 8,192 KiB)

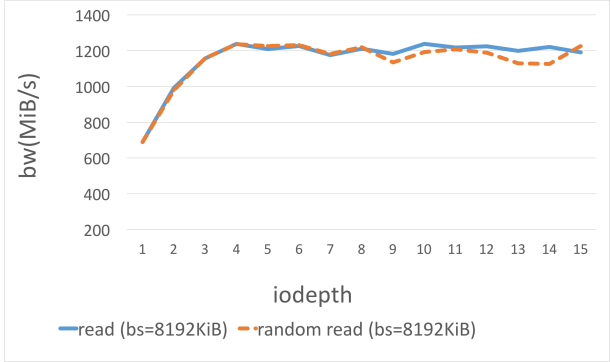

図 12 非同期 I/O(libaio) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 8,192 KiB)

スレッド数における書き込み性能を**図 15** に示す．複数スレッドを用い並列に書き込みを行うことで，全体としてのバンド幅を大きくすることができることがわかった．8,192 KiB, 65,536 KiB の時は，ブロックサイズが大きいせいか，スレッド数による性能が安定しなかった．一例としてブロックサイズが 8,192 KiB のときのスレッド数における書き込み性能を**図 16** に示す．

読み込み性能に置いては，ローカルキャッシュによって適当な評価が出来なかった．

4.4.4 Oracle Storage Cloud の設定による性能評価

Oracle Storage Cloud が提供する設定項目による性能評価を行う．

まず，OSCSA インストールサーバのローカルストレージに保持されるキャッシュサイズを変更して性能を調べる．影響するのは読み込み性能のため，読み込み性能について調査する．ローカルのキャッシュサイズが十分に確保できないケースを想定して，テストを行った．指定できる範囲での最低の設定である，1 GiB をセットし 4.4.1 節と同じ条件で 5 GiB のデータの読み込みを行った．OSCSA の仕様でキャッシュを明示的にフラッシュすることが出来ないので，明確には言えないが，読み込み容量よりも小さい容量設定をしているためキャッシュミスが起こると考え

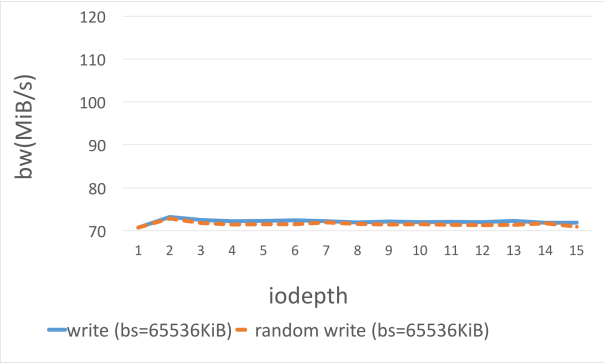

図 13 非同期 I/O(libaio) 同時発行数による書き込みバンド幅の性能 (ブロックサイズは 65,536 KiB)

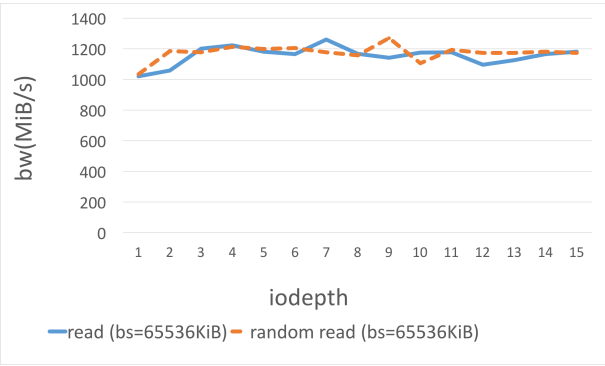

図 14 非同期 I/O(libaio) 同時発行数による読み込みバンド幅の性能 (ブロックサイズは 65,536 KiB)

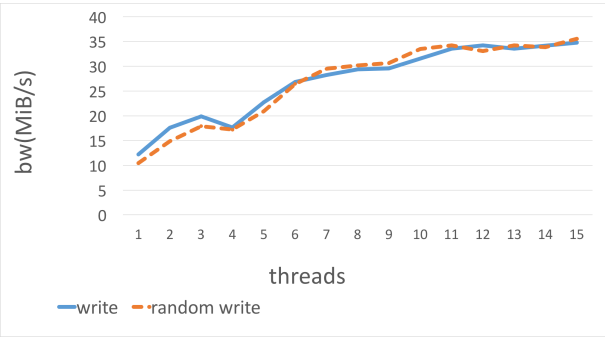

図 15 スレッド数による書き込みバンド幅の性能 (ブロックサイズは 512 KiB)

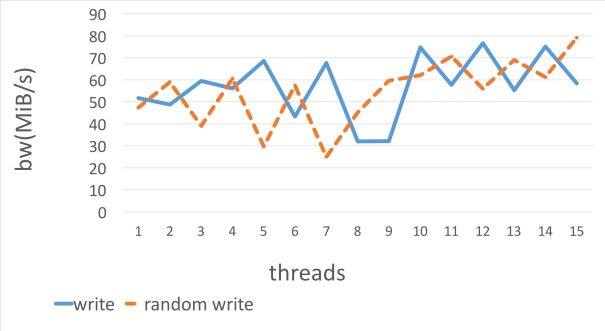

図 16 スレッド数による書き込みバンド幅の性能 (ブロックサイズは 8,192 KiB)

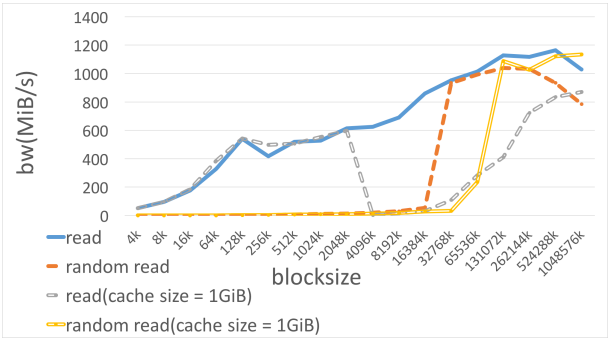

図 17 ローカルキャッシュサイズの設定による読み込み性能の違い

られる．結果を**図 17** に示す．また，4.4.1 節との読み込みの結果とも比較する．4.4.1 節でのキャッシュサイズはデフォルトの設定であり 300 GiB である．シーケンシャル read を実行した時，高いブロックサイズ（4,096 KiB ～）で性能の低下が発生した．ローカルキャッシュが読取りのバッファとしても機能しているため，圧迫されるとキャッシュの削除を行う必要があり，その削除のタイミングで性能が悪化したとみられる．性能を維持するためにはローカルキャッシュの容量を十分に確保することが重要となる．

次に，プリフェッチ機能の設定を 1 MiB, 2 MiB, 4MiB, 8 MiB, 16 MiB, 32 MiB と変化させたときの性能差であるが，OSCSA の仕様でキャッシュを明示的にフラッシュすることが出来ないため純粋な性能として正当なサンプルが取れなかった．

最後に，Sync Policy の POSIX Standard 設定と，Acynchronous 設定を切り替えて，書き込みの性能差を比較する．4.4.1 節と同じ条件でブロックサイズを変えながら 2 つの設定の比較をした．結果を**図 18** に示す．Acynchronous 設定では，ローカルキャッシュからクラウドへは非同期で書き込み処理を行うため，ネットワークを通した書き込みの完了を同期的に待つ必要がないために，性能が格段に上がっていると言える．ただし I/O 発行元への通知の時点では書き込みが完了したのはローカルキャッシュであり，実際のクラウドにデータが書き込まれたかどうか，ユーザが知ることが出来ない．ディスク上の一貫性に影響を与える可能性が有るので，同期トランザクションに依存するファイルシステム操作には適さない．また，内部システムのためかブロックサイズが 2,048 KiB から待ちが発生し同期書き込み設定と性能がほぼ変わらなくなり，また 256 MiB から性能が少し上がっている．

5. クラウドストレージにおける Gfarm の組み込みへの検討

まず，本論文を通しての実験結果を以下にまとめる．

- ブロックサイズによって，read, write の性能を純粋に上げることが出来る．ブロックサイズが十分大きい時，性能が飽和して緩やかに増加するため，コストと

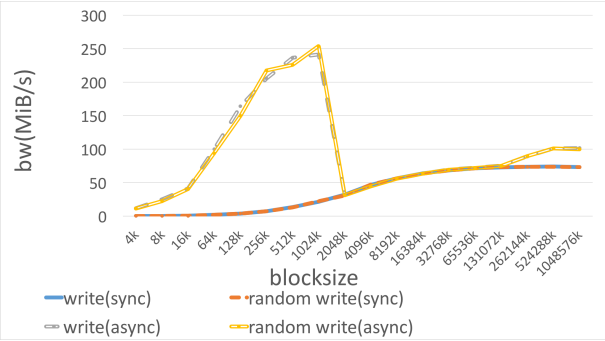

図 18 OSCSA の POSIX Standard 設定と，Asynchronous 設定の切り替えによる書き込みバンド幅の性能

のバランスを考えると 8 MiB 〜 64 MiB 程度が好ましい．

- write はクラウドの状況にもよるが 60 〜 70 MiB/s ほどの性能を持つ．
- write はスレッドによる I/O の多重発行が有効であり，同一データもしくは多数データにおいて並列に処理を行うことで性能を上げることが出来る．
- write は libaio による I/O の多重発行に対して無効である．
- write は OSCSA の Sync policy を Asynchronous 設定にすることによって，ローカルキャッシュへの書き込みの時間だけで完了通知が返ってくるので，結果として時間あたりの I/O の発行数を増やすことが出来る．ただし，クラウドへのデータ移送は完了していない可能性があることに気をつける．ユーザ側でクラウドへのデータ移送が完了していることを知りたければ，スレッドによる多重アクセスを利用するのが好ましい．ただし，スレッドの数にも限界があるため，併用することで性能が上がる可能性がある．
- read はローカルキャッシュが効くため，性能がローカルキャッシュのディスク性能に依存する．
- read のローカルキャッシュが効かないときの性能はクラウドの状況にもよるが，write と同程度の性能と推定される．
- read は libaio による I/O の多重発行に対してローカルキャッシュに対してのみ有効である．
- read はスレッドによる I/O の多重発行が有効か否かは，キャッシュによって正当な評価ができなかった．
- read はキャッシュによって読み込み性能が大きく改善するため，キャッシュサイズを多くとるのが好ましい，ただしローカルストレージの容量を圧迫するため適当に定める．
- read はプリフェッチ機能によって性能を上げることが出来るとされるが，キャッシュによって正当な評価ができなかった．

以上の性能評価の結果をうけて，Gfarm で組み込むことを検討する．OSCSA を利用する場合は NFS を利用するため，Gfarm でそのまま利用が可能である．組み込みの際は，上の性能評価の結果を元に最適な設定の上で運用を行う．クラウドストレージへの読み込みと書き込みは，ローカルストレージとくらべて数十倍の性能差があるため，ローカルストレージとともに利用するのであればデータの参照頻度によって階層的に配置することが 1 つの適切な利用として考えられる．そのような利用を考えた時に階層的に配置を行いながら統合するシステムの追加設計が必要になる．

また Oracle Storage Cloud だけでなく，一般のクラウドストレージについても言及する．オブジェクトストレージのアクセスに用いる，REST API を仮想ファイルシステムに組み込むことを検討する．OSCSA は REST API を隠蔽しているため，性能評価の結果は REST API を組み込む際の良いサンプルの 1 つとなる．REST API を利用する場合もスレッドや非同期通信による多重アクセスで性能をあげられると考えられる．そのため Gfarm 仮想ファイルシステムにおける，接続インターフェースの隠蔽の設計の際には，多重アクセスを検討する．また，ローカルキャッシュが有用であるため，キャッシュを置きデータを管理するシステムの実装も検討の必要があるといえる．

6. まとめと今後の課題

本論文では，Gfarm 上での運用を目標とした，Oracle Storage Cloud の性能評価について着目した．クラウドストレージはネットワークが性能のボトルネックのために，設定やアクセス方法などによって大きく性能が変わることを確認し，最適化が必要であることがわかった．アプリケーションの上でクラウドストレージを運用する前に把握しておくべき性能，仕様を調査し，明らかにすることが出来た．また，性能評価の結果から Gfarm への統合に向けたシステムの検討が出来た．今後は，他のクラウドストレージについても，実験を行ってみて最適な方法を調査する予定である．またクラウドストレージ毎に比較し，アプリケーションへの適性や，適切なユースケースを見る．そしてその性能調査の結果をベースとし，Gfarm への組み込みを行う．組み込みにおいては，組み込み後の性能や分散透明性のバランスを考えながら，統合を図っていく．ストレージを統合する際には大規模データに対する，ローカルストレージとクラウドストレージの適切な立ち位置を考える．加えて，クラウドストレージだけでなく Ceph/RADOS[13] などの分散ストレージに対しても組み込みを検討したい．また，統合のために REST API などのインターフェースの違いを隠蔽し，仮想ファイルシステムとして提供する仕組みを研究開発する．

謝辞 本研究の一部は JST-CREST JPMJCR1303「EBD：次世代の年ヨッタバイト処理に向けたエクストリー

ムビッグデータの基盤技術」，JST-CREST JPMJCR1413「広域撮像探査観測のビッグデータ分析による統計計算宇宙物理学」，JSPS 科研費 JP17H01748 による．

参考文献

[1] Osamu Tatebe, Kohei Hiraga, Noriyuki Soda: *Gfarm Grid File System*, New Generation Computing, Volume 28, Issue 3, pp 257-275 (2010).
[2] Fielding R.T.: *Architectual Styles and the Design of Network-based Software Architectures*, PhD Thesis (2000).
[3] Oracle: Oracle Storage Cloud (オンライン), 入手先 ⟨https://docs.oracle.com/en/cloud/iaas/storage-cloud/index.html⟩ (参照 2017-10-24).
[4] Xiaonan Zhao, Zhanhuai Li, Leijie Zeng: *A Hierarchical Storage Strategy Based on Block-Level Data Valuation*, Fourth International Conference on Networked Computing and Advanced Information Management (2008).
[5] Hazen Damian, Hick Jason: *GPFS HPSS Integration: Implementation Experience* (2008).
[6] D. Reiner, G. Press, M. Lenaghan: *Information lifecycle management: the EMC perspective*, 20th International Conference on Data Engineering (2004).
[7] Amazon: Amazon S3 (オンライン), 入手先 ⟨https://aws.amazon.com/jp/s3/details/⟩ (参照 2017-11-23).
[8] Google: Google Cloud Storage (オンライン), 入手先 ⟨https://cloud.google.com/storage/⟩ (参照 2017-10-24).
[9] Oracle: Oracle Cloud Infrastructure Storage Software Appliance (オンライン), 入手先 ⟨https://docs.oracle.com/en/cloud/iaas/storage-cloud/cssgu/oracle-cloud-infrastructure-storage-software-appliance.html⟩ (参照 2017-10-24).
[10] fio: fio (オンライン), 入手先 ⟨https://github.com/axboe/fio⟩ (参照 2017-10-24).
[11] The IEEE and The Open Group: The Open Group Base Specifications Issue 7 IEEE Std 1003.1, -2008, 2016 Edition (オンライン) 入手先 ⟨http://pubs.opengroup.org/onlinepubs/9699919799/⟩ (参照 2017-11-23).
[12] Suparna Bhattacharya, Steven Pratt, Badari Pulavarty, and Janet Morgan: *Asynchronous I/O Support in Linux 2.5*, Proceedings of the Linux Symposium, pp 35-44 (2003).
[13] Sage A.Weil, Andrew W.Leung, Scott A.Brandt, and Carlos Maltzahn: *RADOS: A Scalable, Reliable Storage Service for Petabyte-Scale Storage Clusters*, Proceedings of the 2nd International Workshop on Petascale Data Storage, PDSW '07, held in conjunction with Supercomputing '07, pp 35-44 (ACM, 2007).