ホストOS上のイメージファイル配置変更による仮想化環境におけるHadoopのI/O性能向上

中島健司[1] 藤島永太[1] 山口実靖[1]

概要： 近年，Hadoopのような大規模のデータ処理においてクラウドシステムを用いることが増えている．代表的なクラウドシステムの構築方法の1つに仮想計算機を用いる手法がある．単一物理計算機上に複数の仮想計算機を起動するようなクラウド環境においては，複数の仮想計算機が同時に単一のHDDにアクセスするような処理においてI/O処理がボトルネックとなってしまう問題がある．Hadoopのような大規模なデータを扱う処理では，ストレージにシーケンシャルにアクセスをすることが多く，Hadoopシステムの I/O性能を向上させる既存の研究としてファイルシステムにおけるファイル格納位置を制御し，シーケンシャルアクセス速度の高いHDDの外周部を積極的に活用する手法がある．しかし，この手法は大規模データ処理の代表的なシステムの一つである仮想化環境を用いたクラウドシステムを考慮していない．本稿では，この手法の仮想化環境への適用について考察する．具体的には，複数の仮想計算機を起動する仮想環境の課題を示し，その解決策としてVMイメージファイルを非連続的に配置する手法を示す．そして，複数のアプリケーションを用いて提案手法の有効性を示す．また，複数VM中の少数VMしか稼働していない状況における提案手法の性能を調査し，このような状況においても性能は同等か小さく劣化するにとどまり，本手法の負の影響が小さいことを示す．

Improving Hadoop I/O Performance by File Placing Control Virtualized Environment with Disk Image Files

KENJI NAKASHIMA[1] EITA FUJISHIMA[1] SANEYASU YAMAGUCHI[1]

1. はじめに

近年，世界中の情報量が爆発的に増加しており，その情報を収集・蓄積・分析して有効に活用することに注目が集まっている．膨大な情報を扱う方法としてHadoopがあり，Hadoopのような大規模のデータ処理においてクラウドシステムを用いることが増えている．代表的なクラウドシステムの構築方法の1つに仮想計算機を用いる手法がある．単一物理計算機上に複数の仮想計算機を起動するようなクラウド環境においては，複数の仮想計算機が同時に単一のHDDにアクセスするような処理においてI/O処理がボトルネックとなってしまう問題がある．Hadoopのような大規模なデータを扱う処理では，ストレージにシーケンシャルでアクセスをすることが多い．よってHadoopアプリケーションの性能を向上させるにはシーケンシャルアクセス性能を向上させることが重要となる．

HadoopシステムのI/O性能を向上させる既存の手法として，ファイルシステムにおけるファイル格納位置を制御しシーケンシャルアクセス速度の高いHDDの外周部を積極的に活用する手法[1]がある．しかし，この手法は大規模データ処理の代表的なシステムの一つである仮想計算機を用いたクラウドシステムを考慮していない．

本稿では，この手法の仮想化環境への適用について考察する．具体的には，複数の仮想計算機を起動する仮想化環境の課題を示し，その解決策としてVMイメージファイルを非連続的にストライプ状に配置する手法を提案する．そして，性能評価により提案手法の有効性を示す．具体的には，物理計算機の物理HDD内の全てのVMイメージファイルにアクセスが生じている状況(本手法が想定している状況)における性能を示し，提案手法によりアプリケーション性能が大幅に改善することを示す．さらに，物理HDD内の一部のVMイメージファイルにのみアクセスが生じている状況(本手法の想定と異なる状況)における性能を示し，このような状況であっても性能は同等か，小さく劣化するにとどまり，本手法の負の影響が小さいことを示す．

2. 既存手法

本章では，既存手法であるファイル格納位置の静的制御手法[1]について述べる．

定記録密度方式HDDのシーケンシャルI/O速度は内周側のゾーンより外周側のゾーンの方が高い．既存研究[1]ではこの特性を考慮し，ファイルを外周側のゾーンに優先的に格納し，シーケンシャルI/O性能を向上させている．

この手法の実装は，オープンソースファイルシステムであるext2/ext3を用いて行っている．これらのファイルシス

1 工学院大学大学院 工学研究科 電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School

テムは，ディスクは 4KB のブロックを単位に管理され，複数のブロックでブロックグループを構成している．そして，ブロックグループ毎にブロックビットマップ，inode ビットマップ，inode テーブルが用意されている．ブロックが使用中か未使用であるかはビットマップで管理されている．各ブロックグループのデータブロックビットマップのうち，ディスクの外周側に相当する低アドレス部以外のブロックのビットを使用中ビットへ変更し，内周側に相当する高アドレス部にデータが格納されることを回避している．

この手法は物理計算機環境においてその有効性が確認されているが，仮想化環境では物理 HDD と仮想 HDD のディスクアドレスが一致しないため適切な物理 HDD のアドレスにデータが格納されず，効果的に性能を向上できないと考えられる．

3. 仮想化環境における静的制御手法

本章では，仮想化環境における既存のファイル格納位置制御手法の課題を述べる．

ファイルシステム上に生成・配置されたイメージファイルは HDD の連続境域に配置される．よって，最初に生成されたイメージファイルは HDD の外周側領域(すなわち高速領域)に生成されるが，2 つ目以降のイメージファイルは既にイメージファイルが配置された領域以外(すなわち低速領域)に生成される．このようにイメージファイルが生成・配置された状況では，仮想 HDD およびゲスト OS ファイルシステムにおける格納位置が物理 HDD における格納位置と一致しないことになる．そして，この環境下で静的ファイル格納位置制御手法を用いると，図 1 のように各仮想 HDD の低アドレス部に I/O が発生することになり，1 つ目の VM 以外は物理 HDD の低速領域を使うことになる．また，ファイル格納位置が離れることにより，シーク距離が長くなってしまう問題が生じる．

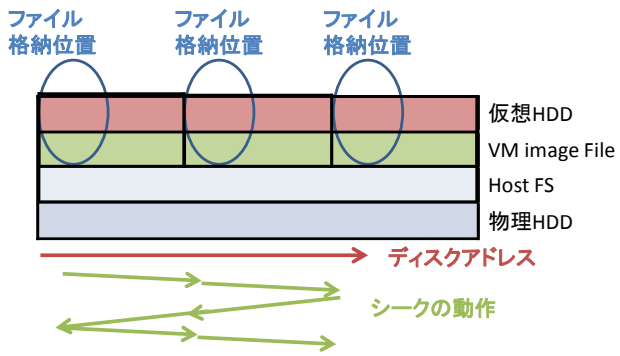

図 1 仮想化環境における静的ファイル格納位置制御手法

4. 提案手法

本章にて，既存手法を拡張し仮想化環境における Hadoop の I/O 性能の向上を実現する手法を提案する．本提案手法では，各 VM のイメージファイルを等間隔で分割し，図 2 のようにイメージファイルの断片を順番にストライプ状に配置する．そしてこの状態にてゲスト OS ファイルシステムに既存のファイル格納位置制御手法を適用する．各イメージファイルの断片をストライプ状に配置することにより，各 VM の仮想 HDD の低アドレス部が物理 HDD の低アドレス部に対応し，物理 HDD の高速な外周側を優先的に使用させることが可能となる．また，シーク距離も削減することができ，I/O 性能を向上させることができると考えられる．

提案手法の実装は，2 章で記述した既存手法と同様に ext3 ファイルシステムのデータブロックビットマップを書き換えることにより行うことが可能である．具体的には各々のイメージファイルを生成する領域以外はビットを使用中に書き換えた状態でイメージファイルの生成を行う．これにより，イメージファイルの断片を図 2 のように順番に配置させることが可能となる．

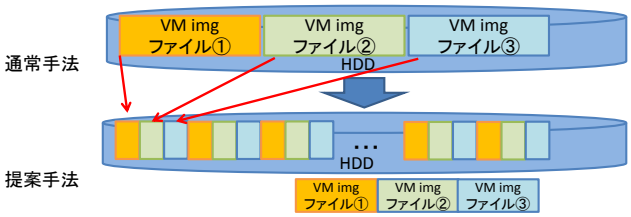

図 2 通常手法と提案手法

5. 性能評価

本章にて，提案手法の性能を評価する．

5.1 測定方法

Hadoop 環境を構築し，ベンチマークアプリケーションを実行し，既存手法および提案手法の性能評価を行った．

測定環境は 1 台の物理計算機で構成され，この物理計算機上に 3 台の仮想計算機を起動した．仮想計算機を SlaveNode，物理計算機を MasterNode として使用した．HDFS ブロックサイズは 64MB，ブロックの複製数は 3，各仮想 HDD のファイルシステムは ext3 とした．提案手法の実装方法は前章の通りであり，分割サイズは 1280MB とした．

測定に用いた物理計算機と仮想計算機の仕様は表 1 と表 2 の通りである．Hadoop で用いられる中間データなどのデータは ext3 ファイルシステムの HDD に格納される．この HDD の仕様の詳細は表 3 の通りである．

本環境にて通常手法および提案手法を用いて Hadoop 用ベンチマークアプリケーション TestDFSIO と TeraSort を実

行し, 両手法の性能を評価した. 稼働 SlaveNode 数は 1 台, 2 台，3 台と変化させて性能を評価した.

稼働 SlaveNode 数 3 は本手法が想定している状況であり, 物理 HDD 内の全ての VM イメージファイルアクセス要求が発生している状況である．稼働 VM 数が 1 や，2 の状況は，本手法が期待している状況とは異なり，本手法により性能が悪化することも予想される．例えば図 2 の VMimg ファイル①にのみアクセスが生じる場合は, VMimg ファイル①の断片の間に VMimg ファイル②と③の断片が配置されてしまっている状況(フラグメンテーション状態)であり, VMimg ファイル①が連続領域に配置されている状態よりも I/O 性能が劣化すると予想される.

TestDFSIO は，稼働 SlaveNode 数が 1 台の場合は生成するファイル数を 1，データサイズを 24GB として行い，稼働 SlaveNode 数が 2 台の場合は生成するファイル数を 2，データサイズを 12GB として行い，稼働 SlaveNode 数が 3 台の場合は生成するファイル数を 3，データサイズを 8GB として行った．測定は read と write をそれぞれ 5 回ずつ行った. TeraSort は入力データサイズ 24GB にて 5 回行った. TeraSort における提案手法ではゲスト OS ファイルシステムにてファイル格納位置の静的制御手法を適用し，SlaveNode 数 1 台，2 台，3 台でそれぞれ仮想 HDD おける外周側 100GB，75GB，50GB 以外へのファイルの配置を禁止した.

表 1 物理計算機の仕様

OS	CentOS 6.5 x86_64 minimal
Kernel	Linux 2.6.32
CPU	AMD Turion Ⅱ Neo N54L Dual-core Processor 2.2GHz
メモリ	16 GB
ストレージ	500 GB × 3
Hadoop Version	2.0.0-cdh4.2.1
I/O スケジューラ	CFQ
仮想化システム	KVM

表 2 仮想計算機の仕様

OS	CentOS 6.5 x86_64 minimal
Kernel	Linux 2.6.32
CPU	AMD Turion Ⅱ Neo N54L Dual-core Processor 2.2GHz
メモリ	2 GB
ストレージ	64 GB (ext4) ， 130GB (ext3)
Hadoop Version	2.0.0-cdh4.2.1
I/O スケジューラ	CFQ

表 3 測定用 HDD の仕様

型番	DT01ACA050
インタフェース	SATA 3.0
インタフェーススピード	6.0 Gbps
容量	500 GB
バッファサイズ	32 MB
回転数	7,200 rpm
平均回転待ち時間	4.17 ms

5.2 測定結果

TestDFSIO により測った稼働 SlaveNode 数 1 台における read 処理，write 処理の I/O 速度および平均 I/O 速度を図 3 と図 4 に示す．同様に，稼働 SlaveNode 数 2 台における read 処理，write 処理の I/O 速度および平均 I/O 速度を図 5 と図 6 に，稼働 SlaveNode 数 3 台における read 処理，write 処理の I/O 速度および平均 I/O 速度を図 7 と図 8 に示す. 各グラフの横軸は測定回を表し，縦軸は各測定の I/O 速度[MB/sec]を示す．図 7 の稼働 SlaveNode 数 3 台の read 処理では通常手法に比べて提案手法は約 28.8%の I/O 速度の性能向上が確認でき, 本提案手法が単一物理 HDD 内の全 VM にアクセスが生じる状況にて有効であることが分かる．次に，一部の VM のみが稼働している状況における性能について考察する. 図 3 の稼働 SlaveNode 数 1 台の read 処理では，通常手法に比べて提案手法では約 14.6%の I/O 速度の性能低下が確認できる．そして，図 5 の稼働 SlaveNode 数 2 台の read 処理では通常手法に比べて提案手法は約 15.3%の I/O 速度の性能低下が確認できる．これらは，想定環境(図 7)における性能向上よりも小さく，想定と異なる状況(図 5 や図 6)が短い時間生じることがあっても提案手法は有効であると言える．一方で，図 4 と図 6 と図 8 の稼働 SlaveNode 数 1，2，3 台における write 処理では通常手法と提案手法で I/O 性能の速度差はあまりみられなかった．これは OS の I/O スケジューラが書き込み要求を遅延させ，待ち行列中に貯められた要求群を短時間で処理したため，ファイルの配置に依らずアクセスのランダム性の削減がなされたためであると考えられる.

次に, 稼働 SlaveNode 数 1 台, 2 台, 3 台における TeraSort の実行時間を図 9~図 11 に示す. 図 11 の稼働 SlaveNode 数 3 台における提案手法の平均実行時間は通常手法よりも約 23.5%短く, 提案手法が有効であることが確認できる. また, 図 9 の稼働 SlaveNode 数 1 台では通常手法と提案手法で平均実行時間がほぼ変わらないことが確認でき，図 10 の稼働 SlaveNode 数 2 台における提案手法の平均実行時間は通常手法よりも約 5.4%短いことを確認できる．以上により，アプリケーション TeraSort の例では，提案手法は想定環境や想定からややずれた環境において性能の向上を実現でき，想定から大きくずれた環境でも同等の性能を示せることが確認された.

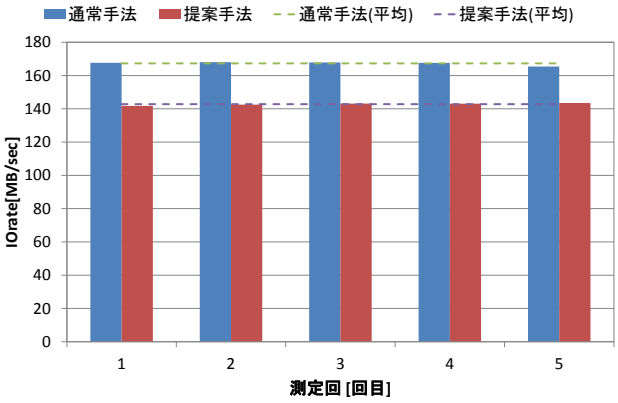

図 3 TestDFSIO(read, SlaveNode 数 1)

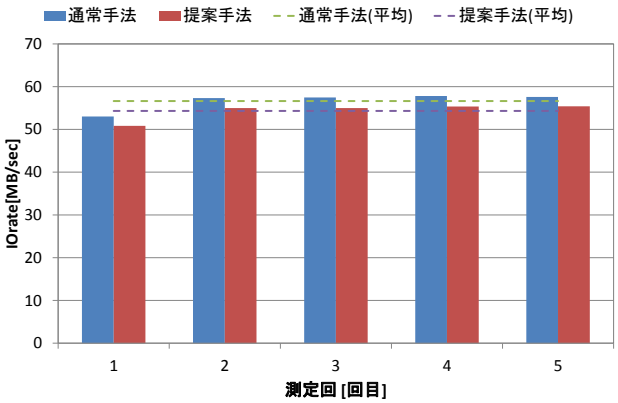

図 4 TestDFSIO(write, SlaveNode 数 1)

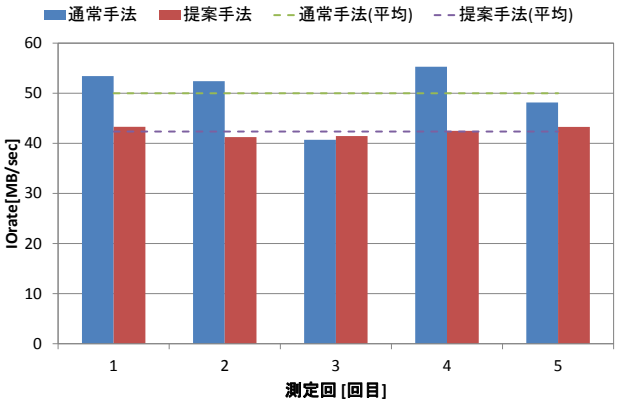

図 5 TestDFSIO(read, SlaveNode 数 2)

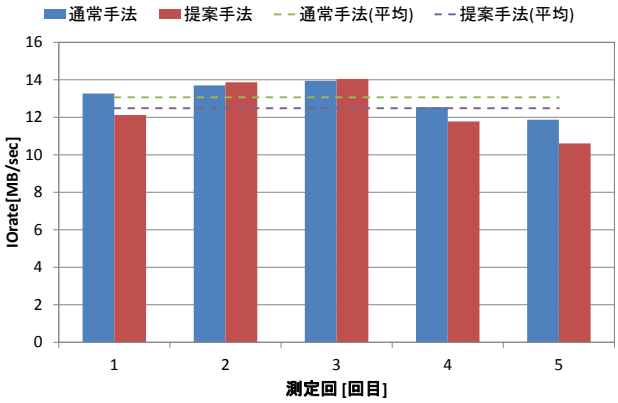

図 6 TestDFSIO(write, SlaveNode 数 2)

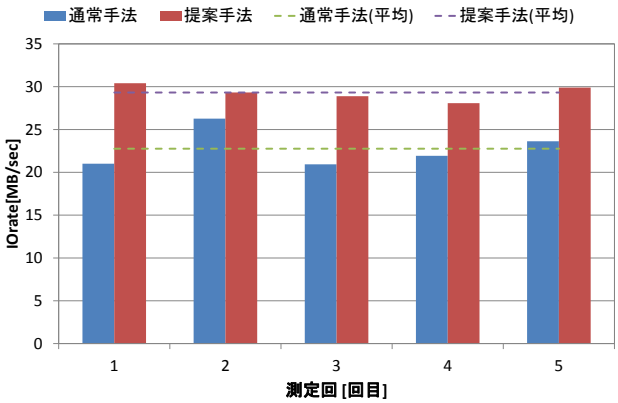

図 7 TestDFSIO(read, SlaveNode 数 3)

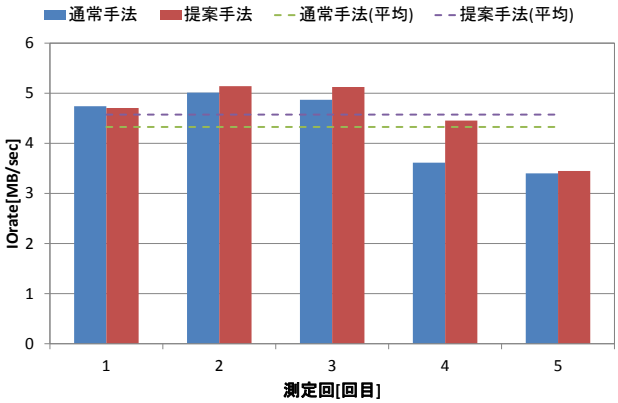

図 8 TestDFSIO(write, SlaveNode 数 3)

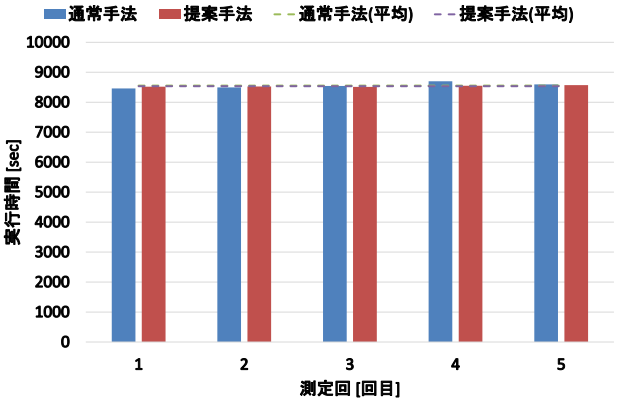

図 9 TeraSort(SlaveNode 数 1)

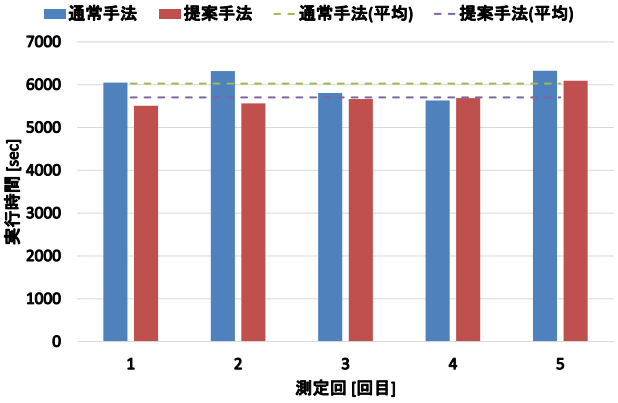

図 10 TeraSort(SlaveNode 数 2)

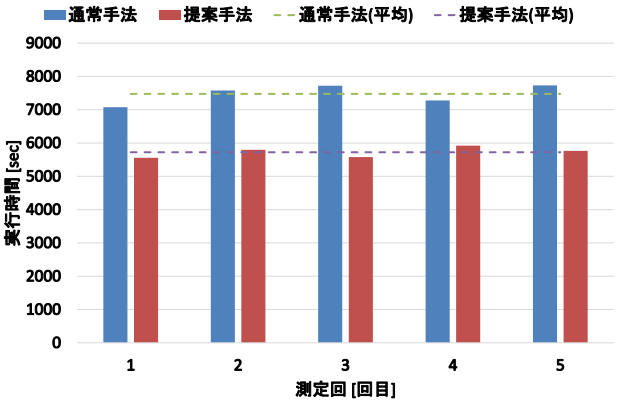

図 11 TeraSort(SlaveNode 数 3)

5.3 ブロックデバイスシーケンシャルアクセス性能

本節にて，提案手法適用時と非適用時における基本 I/O 性能評価として，ゲスト OS 上における仮想ストレージデバイスのスペシャルファイル(/dev/sdb，/dev/sdc) へのシーケンシャルリード/ライト速度の評価を行う．

ゲスト OS 上における仮想ストレージデバイスのスペシャルファイルに対して 64MB の読込/書込を 2046 回行うプログラムを実行し，仮想 HDD のゾーン毎のシーケンシャルリード/ライト速度を測定した．測定に用いた物理計算機と仮想計算機の仕様，HDD の仕様の詳細は 5.1 節と同様である．

測定結果を図 12~図 17 に示す．各グラフの横軸は仮想 HDD 内のディスクアドレス[GB]，縦軸は 64MB の読込/書込一回にかかった時間[sec]を示している．

図 12~図 14 の通常手法では仮想 HDD により低アドレス部の I/O 速度に大きな差が生じていることが分かる．これは vm01 のイメージファイルは物理 HDD の外周部(高速部)に配置されているが，他の VM のイメージファイルは物理 HDD の非外周部(非高速部)に配置されてしまっているためである．よって，このような状態で静的ファイル格納位置制御手法を用いても一部の VM 以外は物理 HDD の外周部を用いることができず，性能向上につながりにくいことが分かる．一方図 15~図 17 の提案手法では，全ての VM イメージファイルの低アドレス部が物理 HDD の外周部に配置されているため，全ての VM の低アドレス部にて高い性能が得られている．よって，この状況でファイル格納位置制御手法を用いると I/O 性能の向上を実現できる．

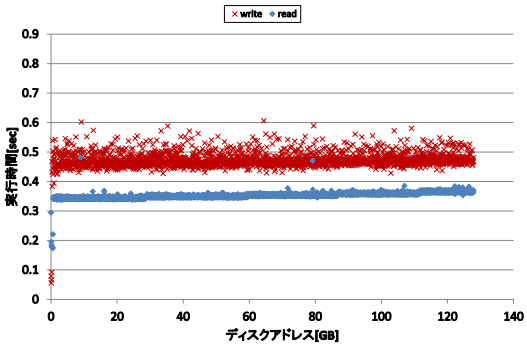

図 12 仮想 HDD のシーケンシャル I/O 速度 (通常手法 vm01)

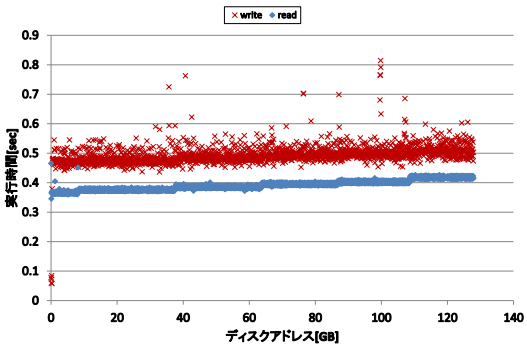

図 13 仮想 HDD のシーケンシャル I/O 速度 (通常手法 vm02)

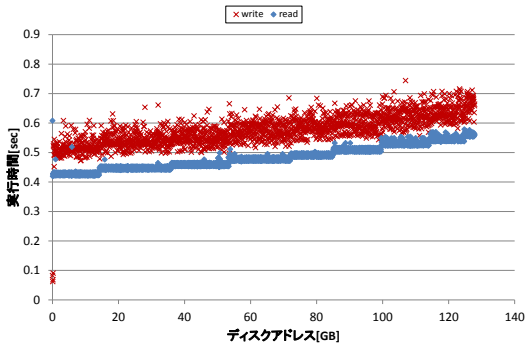

図 14 仮想 HDD のシーケンシャル I/O 速度
(通常手法 vm03)

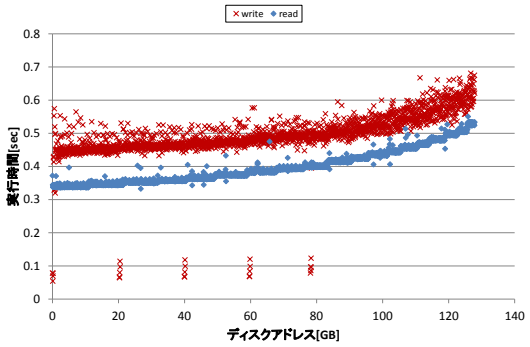

図 15 仮想 HDD のシーケンシャル I/O 速度
(提案手法 vm01)

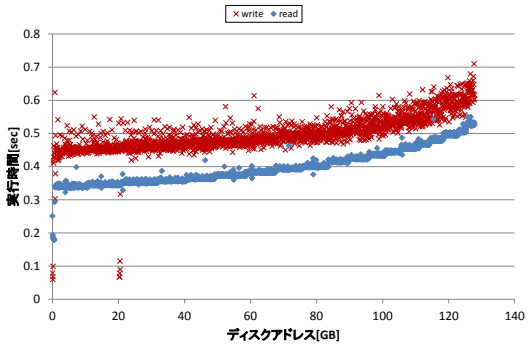

図 16 仮想 HDD のシーケンシャル I/O 速度
(提案手法 vm02)

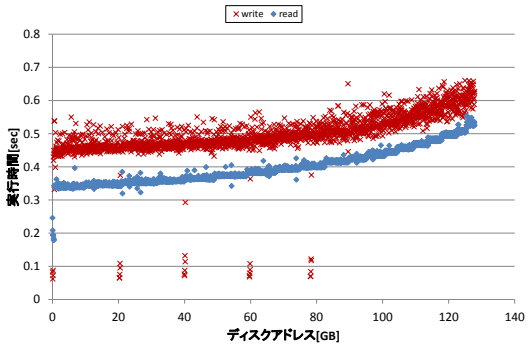

図 17 仮想 HDD のシーケンシャル I/O 速度
(提案手法 vm03)

6. 考察

6.1 VM イメージファイルの分割サイズ

5 章における性能測定では，VM イメージファイルの分割サイズを 1280MB として性能評価を行っている．分割サイズの大きさの性能への影響について考察する．

Hadoop アプリケーションは多くの場合はストレージ装置に対して高いシーケンシャル性でアクセスする．よって，分割サイズは小さすぎず，物理 HDD へのアクセスが高いシーケンシャル性で行われるようにすることが好ましいと考えられる．しかし，分割サイズが過度に大きいと分割片内における内部フラグメンテーションが生じ，結果的に不使用の領域により HDD の外周部が占有されてしまう問題や，ファイル配置間に不使用領域が生じシーク距離が増大してしまう問題につながり，好ましくないと予想される．また，ext2/3/4 においてはブロックグループサイズやエクステントサイズが 128MB となっており，これと同一とする設定もファイルシステムの処理負荷の軽減につなげることができると期待できる．

6.2 格納位置制御の負荷

本手法では，VM イメージファイルをストライプ状に配置してから実行する必要がある．この格納位置制御に要する負荷について考察する．

通常の VM イメージファイルの作成時間と，ストライプ状配置での作成時間を比較すると，ほぼ同等となる．通常の手法では，ストレージに対してシーケンシャルライトを行い巨大なイメージファイルを 1 つ作成する．n 個の VM イメージファイルで構成されるストライプ状配置の場合は，1 つ目のイメージファイルの分割サイズ分のシーケンシャルライトを行い，続いて 2 つ目のファイルのシーケンシャルライトを行い，3 つ目のファイルのシーケンシャルライトを行い，以下同様に n 個目のファイルまでのシーケンシャルライトを行う．続いて，1 つ目のイメージファイルの 2 つ目の分割サイズ分のシーケンシャルライトを行い，2 つ目のファイルのシーケンシャルライトを行っていく．よって，ストレージデバイスにかかる負荷は通常の手法によるイメージファイルの作成とほぼ同等である．唯一の差違は，提案手法におけるファイルシステムのブロックビットマップの変更処理の負荷のみであり，これは上記の巨大なイメージファイルの作成処理の負荷と比較して非常に小さい．

6.3 コピーオンライトとの比較

仮想計算機用の VM イメージファイルの作成方法の 1 つに CoW(Copy on Write)手法がある．この手法を用いると，ゲスト OS 内で使用されている領域用の分のみイメージファイルが作成される．よって，巨大な VM イメージファイル内に大きな未使用領域が存在し，これを超えてシークするために長いシーク時間を要することはなくなる．しかし，CoW 手法を用いてインクリメンタルにファイルの拡張を

行うと，ファイルが細かく分割されてしまい，ゲスト OS 内における仮想ストレージへのシーケンシャルアクセス性能が低下し，結果としてシーケンシャルアクセスを多用する Hadoop アプリケーションの性能が低下してしまうと予想される．

7. 関連研究

本章にて，本研究と関連する既存の研究を紹介する．初期のディスクレイアウトの理論に関する研究として文献[2]があり，シミュレーションによる研究として文献[3]~[6]がある．文献[2]では，最高頻度アクセスデータをストレージの中央に配置する organ pipe 手法がランダムアクセスに適していることが示されている．また，organ pipe 手法を現実のワークロードに適用した研究として cylinder shuffling[6] がある．文献[6] ではシリンダー単位で並び替えを行うが，並び替えの単位をブロックとすることによりさらなる高速化を実現した研究として文献[5] がある．また，実システムにおける block shuffling について最初に述べた研究として文献[7] がある．

また，これら再配置の実現には HDD の幾何学的構造が既知である必要があることが多く，物理ディスクから幾何学的構造を調査する研究として文献[8]~[11]がある．

次にファイルシステムレベルの研究を紹介する．FFS[12]やその後続の研究[13][14]にて，関連するデータブロックと inode をディスク上の近隣に配置することにより I/O 速度を向上させる方法が提案されている．これらの手法は動的なアクセスパターンを考慮しないため，性能向上の程度が制限されてしまうことが指摘されている[5][7]．また文献[15]にて，参照の局所性ではなく，微小ファイルの距離を近づけることに着目して性能を上げる方法が提案されている．

ログ構造化ファイルシステム[16]では，大幅な書き込みの性能の向上が実現されている．また，アクセス頻度の高いファイルをディスクの外周に配置することによりログ構造化ファイルシステムをさらに高速化する研究がなされている[17]．

HFS[18]の Hot File Clustering や Smart File System[19]では，ファイルシステムが実行時アクセスパターンを観察し，高頻度アクセスデータを予約領域に移動を行っている．FS2[22]では，ファイルの複製を用いて連続アクセスされるファイル，あるいはその複製を近隣に配置することによりさらなる高速化を実現している．文献[20]の手法では，高頻度データを近隣に再配置しシーク距離の削減を実現している．

Hadoop などのデータ処理基盤に着目した研究としては，2 章で紹介した文献[1]や，それを発展させた文献[23]などがある．これらの手法は，仮想化環境を想定しておらず，クラウド環境などにそのまま適用することはできない．

文献[25]では仮想化環境を考慮した手法が提案されているが，稼働 VM 数が少ない状況における評価や，VM 内における仮想 HDD シーケンシャルアクセス性能に関する考察が行われていない．

8. おわりに

本稿では，ベアメタル(非仮想化)環境において有効性が確認されている静的ファイル格納位置制御手法を仮想化環境において適用する手法についての考察を行い，VM イメージファイルの非連続配置手法を提案し，性能評価による有効性の検証を行った．具体的には，既存手法である静的ファイル格納位置の制御を紹介し，仮想化環境における本既存手法の課題を示し，VM イメージファイルの非連続配置手法を提案した．そして性能評価を行い，想定している環境である全ての VM イメージファイルにアクセスが生じる環境にて 大きな性能向上が達成され，想定と異なる一部の VM イメージファイルにのみアクセスが生じる環境においても性能は向上されるか小さな低下が生じるのみであることを確認し，提案手法の有効性を示した．また，VM 上における仮想 HDD シーケンシャルアクセス性能を示し，提案手法により Hadoop アプリケーション性能が向上する理由を示した．

今後は，VM イメージファイルの分割サイズを大きさの影響，アクセス頻度を考慮した格納位置制御の適用について考察していく予定である．

謝辞

本研究は JSPS 科研費 26730040, 15H02696 の助成を受けたものである．

本研究は、JST、CREST の支援を受けたものである．

参考文献

[1] Eita Fujishima, Saneyasu Yamaguchi, “I/O Improving on Reduce Phase of Hadoop,” International Symposium on Computing and Networking (CANDAR’15), (2015).

[2] C.K.Wong, ”Algorithmic Studies in Mass Storage Systems,” Computer Sciences Press, 1983.

[3] Robert English and Stephnov Alexander, ”Loge: A Self-Organizing Disk Controller,” Proceedings of the Winter 1992 USENIX Conference, 1992.

[4] David Musser, ”Block Shuffling in Loge,” HP Technical Report CSP-91-18, 1991.

[5] C.Ruemmler and J. Wilkes, ”Disk Shuffling,” HP Technical Report, HPL-CSP-91- 30, 1991.

[6] P.Vongsathorn and S. D. Carson, ”A System for Adaptive Disk Rearrangement,” Software Practice Experience, 20(3): 225-242, 1990.

[7] Sedat Akyurek and Kenneth Salem, ”Adaptive Block Rearrangement, Computer Systems,” 13(2): 89.121, 1995.

[8] M.Aboutabl and A.Agrawala and J.Decotignie, ”Temporally Determinate Disk Access: An Experimental Approach,” Technical Report, CS-TR-3752, 1997.

[9] Zoran Dimitrijevic et al., ”Diskbench: User-level Disk Feature Extraction Tool,”Technical Report, UCSB, 2004.
[10] J.Schindler and G.Ganger, ”Automated Disk Drive Characterization,” Technical Report CMU-CS-99-176, 1999.
[11] N.Talagala and R.Arpaci-Dusseau and D.Patterson, ”Microbenchmark-based Extraction of Local and Global Disk Characteristics,” Technical Report CSD-99-1063,1999
[12] M.K.McKusick et al., ”A Fast File System for UNIX,” ACM Transactions on Computing Systems (TOCS), 2(3), 1984.
[13] R.Card and T.Ts ’o and S.Tweedle, ”Design and Implementation of the Second Extended Filesystem,” First Dutch International Symposium on Linux, 1994.
[14] Stephen Tweedie, ”Journaling the Linux ext2fs Filesystem,” LinuxExpo, 1998.
[15] Greg Ganger and Frans Kaashoek, ”Embedded Inodes and Explicit Groups: Exploiting Disk Bandwidth for Small Files,” USENIX Annual Technical Conference, 1-17. 1997.
[16] M.Rosenblum and J. Ousterhout, ”The Design and Implementation of a LogStructured File System,” ACM Transactions on Computer Systems, 26-52, 1992.
[17] J.Wang and Y.Hu, ”PROFS.Performance-Oriented Data Reorganization for Logstructured File System on Multi-Zone Disks,” The 9th International Symposium on Modeling, Analysis and Simulation on Computer and Telecommunication Systems,285.293,
[18] HFS Plus Volume Format, http://developer.apple.com/technotes/tn/tn1150.html
[19] C.Staelin and H.Garcia-Molina, ”Smart Filesystems,” USENIX Winter, 45-52,1991. 2001.
[20] 山田将也，山口実靖，”仮想計算機環境における二重ファイルシステム構造を考慮した仮想 HDD イメージファイルの再配置”， WebDB Forum 2011，2011
[21] Masaya Yamada, Saneyasu Yamaguchi, "Filesystem Layout Reorganization in Virtualized Environment", The 9th IEEE International Conference on Autonomic and Trusted Computing (IEEE ATC 2012), ATC4-2
[22] Hai Huang, Wanda Hung, Kang G. Shin ”FS2: Dynamic Data Replication in Free Disk Space for Improving Disk Performance and Energy Consumption,” SOSP’05, pp.263-276, October. 2005.
[23] Eita Fujishima, Saneyasu Yamaguchi, "Dynamic File Placing Control for Improving the I/O Performance in the Reduce Phase of Hadoop", the Tenth International Conference on Ubiquitous Information Management and Communication (IMCOM2016), 8-2, 2016
[24] 藤島永太・山口実靖, "ファイル格納位置制御による Hadoop MapReduce ジョブの性能の向上", FIT2015 第 14 回情報科学技術フォーラム, RC-003,2015
[25] 中島健司，藤島永太，山口実靖，“ホスト OS ファイルシステムにおける VM イメージファイルの非連続配置による仮想化環境における Hadoop I/O 性能の向上”， SIGOS 第 137 回，2016