# 仮想化環境における I/O スケジューラの動作と性能に関する考察

新居 健一†　山口 実靖†
†工学院大学大学院 工学研究科電気・電子工学専攻

計算機が広く普及し，計算機の消費電力の増加や設置スペースの増大が問題となっている．その対策の一つとして仮想化技術があり，注目されている．しかし，既存の OS に備わっている I/O スケジューラは仮想化環境を考慮しておらず，仮想計算機における I/O 性能は十分に高いとはいえない．

そこで本稿では，代表的な仮想化システムである Xen[1]を用いて Linux に標準で搭載されている 4 つの I/O スケジューラが仮想化環境において I/O 性能に対してどのような影響を与えるか考察する．そしてその動作の解析を行い，解析結果をもとに仮想化環境に適した I/O スケジューラの改善手法を提案する．

# A Study of Performance and Behavior of I/O Schedulers in Virtualized Environment

Kenichi Nii†　Saneyasu Yamaguchi†
†Electrical Engineering and Electronics, Kogakuin University Graduate School

Virtualized environment can be considered as one of most important platform in current computer systems. However, existing I/O schedulers do not have consideration for virtual machines, so desirable performance cannot be obtained in virtualized environment. In this paper, we evaluate I/O performances of existing I/O schedulers of Linux OS in virtualized environment, and present detailed analyses of scheduled I/Os. We proposed a performance improving method based of these analyses. Our evaluation demonstrated the proposed method can improve I/O performance.

## 1. はじめに

計算機の増加に伴い，消費電力や設置スペースの増大が問題視され，仮想化によるサーバ統合などが重要視されている．しかし，既存の OS 上で I/O 最適化を行う I/O スケジューラは仮想化環境を考慮しておらず，仮想計算機の I/O 性能は十分に高いとはいえない．

仮想化環境における I/O スケジューラの性能評価に関する既存の研究としては Boutcher らの性能評価[2]がある．彼らは，ホスト OS の I/O スケジューラとしては単純に I/O 要求を到着順に処理する NOOP が優れているとしており，ゲスト OS の I/O スケジューラとして適するものはワークロードに依存するとしている．しかし，必ずしもこれが汎用的な結論であるとは言い切れず,さらに多くの評価，考察を行うことが好ましいと考えることができる．

そこで我々は，1 台の物理計算機上に 2 台の VM(仮想計算機)を稼働させ，I/O スケジューラごとの I/O 性能を評価した[3]．その結果，文献[2]の主張と異なる傾向が得られ，仮想化環境における I/O スケジューラの性能に関してさらなる考察を行う必要があることが明らかになった．

本稿では，代表的な仮想化システムである Xen を用いて，Linux 標準の 4 つの I/O スケジューラが仮想化環境において I/O 性能にどのような影響を与えるかについて考察する．そして，その動作解析と，仮想化環境に適した I/O スケジューラの改善手法の提案を行う．

## 2. I/O スケジューラ

I/O スケジューラは，アプリケーション群から発行された I/O 要求群を適切な順に並び変え，高性能や高公平性などを実現する OS 内のソフトウェアである．新しい I/O 要求が追加されたとき，I/O スケジューラが呼び出され，新しい

要素をキューのどの位置に追加するかを決定する．一例を挙げると，I/O スケジューラはキューをアクセスセクタ番号順に並び替えられた状態にする．これにより，I/O 要求を処理する際に HDD ヘッドがトラックからトラックへランダムに移動するのではなく，外側のトラックから内側のトラックへ一方向に移動するため，ディスクのシーク量を減少させることができる．

Linux には，NOOP，Deadline，AS，CFQ の４つの I/O スケジューラが存在する[4].

NOOP はスケジューリングにおける並び替えは行わずに単純に I/O を到着順に処理する．NOOP はスケジューリング負荷が小さく，フラッシュメモリなどのランダムアクセスが高速なハードウェアには適していると考えられる．

Deadline は近隣の I/O 要求を優先して処理ことで HDD ヘッドの移動を削減する．近隣の I/O 要求を優先して処理するため，近隣でない I/O 要求は後回しにされるが，後回しにされる時間には限界値(deadline)が用意されており，この値より長く待たされている I/O 要求が発生したら，その I/O 要求を優先して処理する．レイテンシの上限が保証されているため，リアルタイムアプリケーションやデータベース管理システムなどに適していると言われている．

AS(Anticipatory)[5]は I/O 要求を処理する際に，近隣への I/O 要求がすぐ後に発行されるかを予測する．近隣への I/O 要求が発行されると予測されたときは I/O 要求の処理を遅延させ，近隣の I/O 要求が発行されるのを待ってから I/O を処理する．ただし Deadline と同様に，待ち時間の長い I/O 要求が発生した場合には近隣 I/O 待ちを中断しそれらを優先して処理する．AS はデータにシーケンシャルにアクセスする Web サーバなどに有効とされている．

CFQ はプロセスから発行される I/O 要求をプロセス毎のキューに割り振っていくスケジューラである．処理対象のキューを切り替えながら I/O 要求を処理していく．処理対象キューを一定時間で変更するため，I/O 要求は公平に処理される．CFQ にも Deadline や AS 同様に待ち時間の長い I/O 要求を優先的に処理する機能が備わっている．

## 3. 関連研究

仮想化環境における I/O スケジューラに関する研究としては，以下のものがある．

Boutcher らは文献[2]において，仮想化環境において I/O スケジューラが I/O 性能にどのような影響をあたえるか評価しており，仮想化環境において適切な I/O スケジューラを選択することにより I/O 性能が向上すること，ホスト OS の I/O スケジューラとしては処理が最も単純な NOOP が優れていること，ゲスト OS の I/O スケジューラとして優れているものはワークロードに依存すること，などを主張している．

Mukil らは文献[6]において，仮想化環境における仮想ディスクの振る舞いと，通常の物理ディスクの振る舞いの違いを調査し，両者には大きな違いがあることを示している．具体的には I/O 要求に対するレイテンシの大きさを調査し，仮想ディスクのレイテンシの大きさは同一物理計算機内に集約された別の VM の振る舞いに大きく依存することを示している．また，性能向上を実現するための手法についての考察を述べている．

文献[2]は，さまざまな環境における I/O 性能評価結果を示すとともにその性能評価結果に対する考察を行っているが，スケジューリング結果の調査や動作解析などは行っていない．文献[6]も，具体的な性能向上手法や，その実装を用いた性能評価などは行っていない．よって，スケジューリング結果に対する詳細な解析を行い，性能改善手法を提案し，性能向上を実現している本研究とは貢献の内容が異なっている．

I/O スケジューラに関する既存の研究としては，Sitaram らによる Anticipatory スケジューラの提案[5]があるが，これは仮想化環境を考慮したものではない． I/O スケジューラの性能に関する研究としては，Steven らによる性能評価[7]などがあるが，これらも仮想化環境を考慮したものではない．仮想化環境を想定したものではないが，仮想化環境への応用が期待できる研究としては，2 階層の I/O スケジューリングを考慮した Cello[8]や，Zhang らによる自己学習型(self-learning) ディスクスケジューラの研究[9]がある．

# 4. I/O スケジューラの性能評価

既存の I/O スケジューラが仮想化環境の I/O 性能にどのような影響を与えるか調べるために，性能評価実験を行った．実験に用いた物理計算機の仕様は表 1 の通りであり，仮想計算機の仕様は表 2 の通りである．

仮想計算機の仮想ディスクは，ホスト OS のファイルシステム上にファイルとして作成されている．各仮想ディスクイメージファイルの物理ディスクにおける存在領域は図 1 の通りである．横軸がイメージファイルの存在アドレス，縦軸がＶM番号である．

図より VM1 のイメージファイルは 550GB 付近から 640GB 付近に配置されていることがわかる．同様に VM2 は 600GB 付近と 640GB 付近から 740GB 付近に，VM3 のイメージファイルは 900GB 付近から 1000GB 付近に配置されていることが分かる．このことにより，VM のイメージはほぼ連続的に配置されていることがわかる．

性能評価は，ファイルシステムベンチマークソフト FFSB(Flexible File System Benchmark)[10] を用いて行った．ベンチマーク設定は表 3 の通りである．測定では，1 台の物理計算機上に 6 台の VM を起動し，各 VM 上で FFSB を実行し I/O 性能を測定した．その際ホスト OS と，ゲスト OS の I/O スケジューラは変更して，それぞれの性能の比較を行った．測定結果を図 2 に示す．

表 1. 物理計算機の仕様

| | |
|---|---|
| CPU | AMD Athlon Dual Core Processor 5050e (2.6[GHz]) |
| Memory | 2 [GB] |
| HDD | ST31500341AS （7200 [rpm]） |
| Xen | 3.4.2 |
| Kernel | Linux-2.6.18.8 |
| OS | CentOS 5.3 |

表 2. 仮想計算機の仕様

| | |
|---|---|
| Memory | 256 [MB] |
| HDD | 100 [GB] |
| Kernel | Linux-2.6.18.8 |
| OS | CentOS 5.3 |

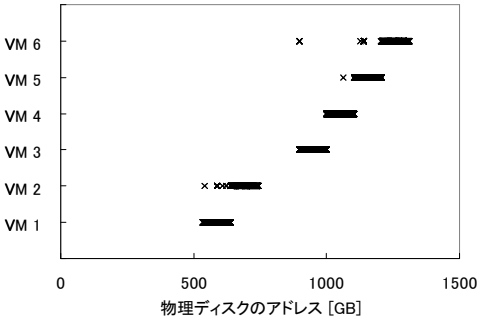


図1　VM イメージの配置

表 3. ベンチマークソフト(FFSB)の設定

| | |
|---|---|
| ファイル数 | 65536 |
| ファイルサイズ | 16 [KB] |
| 1op あたりの読み込み量 | 4 [KB] |
| 1op あたりの書き込み量 | 4 [KB] |
| スレッド数 | 128 |

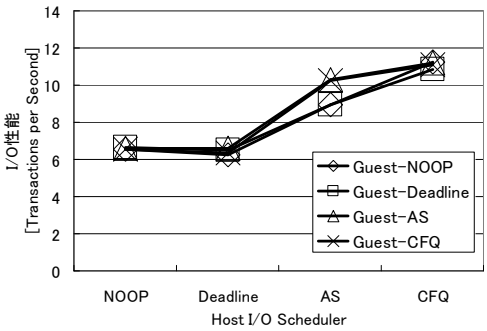


図 2　仮想化環境における I/O スケジューラの性能

縦軸が I/O 性能を表し，単位は Transactions per Second である．横軸がホスト OS の I/O スケジューラを表し，グラフの各線がゲスト OS の I/O スケジューラに対応する．

図より，ホスト OS の I/O スケジューラを変更することにより I/O 性能は大きく変化し，CFQ，AS，Deadline，NOOP の順に優れていることが分かる．

また，ホスト OS の I/O スケジューラが AS であるときのみ，ゲスト OS の I/O スケジューラが I/O 性能に影響を与え，AS，CFQ，Deadline，NOOP の順に優れていることが分かる．

また，文献[2]において，Boutcher らは，処理内容によらずホスト OS の I/O スケジューラとしては NOOP が優れると主張しているが，必ずしも一般的に成り立つことではないことがわかる．

## 5. 発行 I/O の解析

FFSB をゲスト OS 上で実行した際に発行された I/O 要求をホスト OS 上でモニタリングした．ホスト OS の I/O スケジューラが NOOP で，ゲスト OS の I/O スケジューラが CFQ のときの発行 I/O を図 3 に示す．縦軸が発行された I/O の物理ディスクにおけるアクセスアドレスを，横軸が I/O の処理時刻を表している．同様に，ホスト OS の I/O スケジューラが AS，ゲスト OS の I/O スケジューラが AS のときの発行 I/O を図 4 に，ホスト OS の I/O スケジューラが CFQ，ゲスト OS の I/O スケジューラが AS のときの発行 I/O を図 5 に示す．図 3(ホスト OS-NOOP，ゲスト OS-CFQ)は，文献[2]において最も高い I/O 性能が得られるとされている組み合わせである．図 4，図 5 は前章の測定において高い I/O 性能を示した 2 つの組み合わせである．

また，各組み合わせにおける発行 I/O のシーク距離とシーク時間の関係を図 6 から図 8 に示す．

図 3，図 6 より，ホスト OS の I/O スケジューラに NOOP を用いた場合は，アクセス対象の VM イメージの変更が多く発生していることが分かる．よって，HDD ヘッドの移動距離が長く多くの時間を要する I/O が多数発生していることになる．これに対して，図 4，図 7（ホスト OS-AS，ゲスト OS-AS）の例では．アクセス対象 VM の変更が図 3，図 6 よりも少なく，I/O 性能も高くなっている．図 5，図 8（ホスト OS-CFQ，ゲスト OS-AS）の例では，アクセス対象 VM の変更がさらに少なく，I/O 性能もさらに高くなっている．

VM イメージファイルが 100GB であるため，図 6〜8 においてシーク距離が 100GB 程度より大きいものは VM の切り替えを伴うものであるが，その割合は図 6〜8 の順に 90%，14%，6%となっている．

以上より，アクセス対象 VM イメージの変更の回数が I/O 性能に大きな影響を与えていると考えられる．

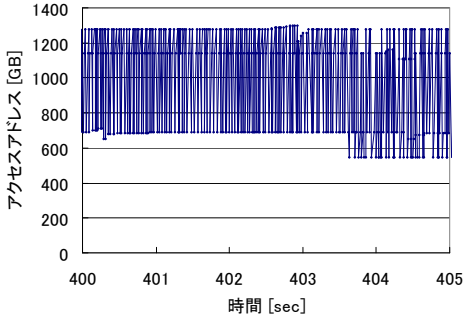


図 3　発行 I/O
(ホスト OS-NOOP，ゲスト OS-CFQ)

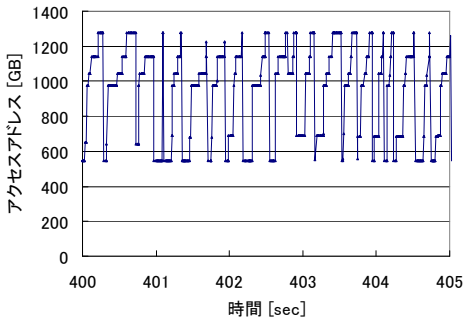


図 4　発行 I/O
(ホスト OS-AS，ゲスト OS-AS )

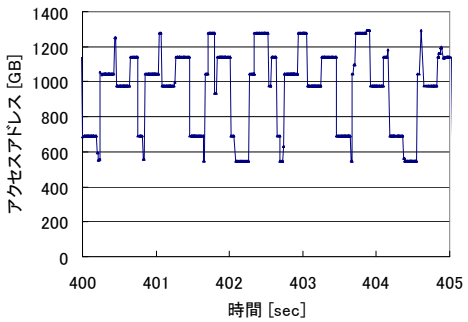


図 5　発行 I/O
(ホスト OS-CFQ，ゲスト OS-AS)

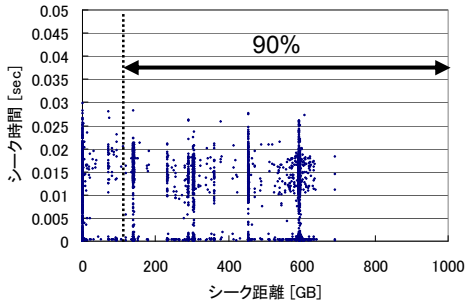


図 6　発行 I/O のシーク距離と時間
(ホスト OS-NOOP，ゲスト OS-CFQ)

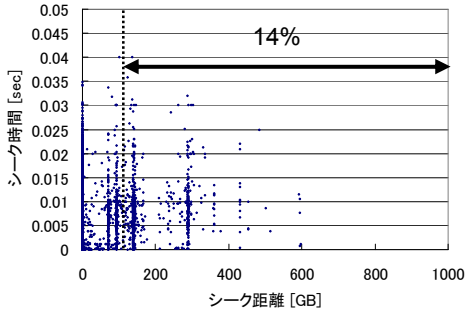


図 7　発行 I/O のシーク距離と時間
(ホスト OS-AS，ゲスト OS-AS)

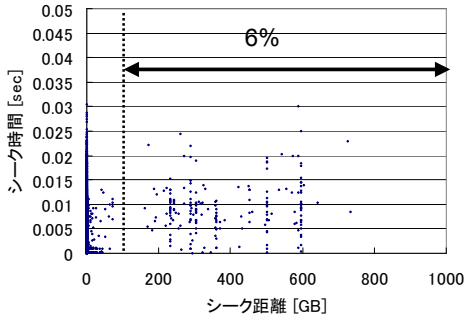


図 8　発行 I/O のシーク距離と時間
(ホスト OS-CFQ，ゲスト OS-AS)

# 6. VM 間移動を削減するスケジューリング手法

## 6.1 I/O スケジューラの変更

アクセス対象 VM の変更頻度を低減させるためにホスト OS の I/O スケジューラの AS と CFQ に対して以下の変更を行い，I/O 性能を向上させる手法を提案する．

まず，AS の「近隣の I/O が発行されると予測し I/O の処理を遅延する時間の限界値（default_antic_expire：**予測期限切れ時間**）」を増加させる．そして，「I/O 資源の餓餓状態であると判定する閾値（default_read_expire：**読み込み期限切れ時間**，default_write_expire：**書き込み期限切れ時間**）」と「I/O 要求を一度にどれだけ連続して処理するかを定める許容時間（default_read_batch_expire：**連続読み込み処理期限時間**，default_write_batch_expire：**連続書き込み処理期限時間**）」を増加させる．

**予測期限切れ時間**の変更により，同一 VM から発行された I/O 要求をより長い時間集め，VM 切り替え回数が減少すると期待される．**読み込み期限切れ時間**と**書き込み期限切れ時間**の変更により，飢餓発生による近隣 I/O 待ちの中断およびそれによる VM の切り替えが抑制されると期待される．**連続読み込み処理期限時間**と**連続書き込み処理期限時間**の変更により，連続 I/O 処理の中断とそれによる VM の切り替えが抑制されると期待される．

次に CFQ に対しては，「スケジューリングスライス時間（cfq_slice_sync：**同期 I/O 実行中プロセスのスライス時間**，cfq_slice_async：**非同期 I/O 実行中プロセスのスライス時間**）」を増加させ，「I/O 資源の餓餓状態であると判定する閾値（cfq_fifo_expire[0] ：**書き込み期限切れ時間**，cfq_fifo_expire[1]：**読み込み期限切れ時間**）」を増加させる．前者(スライス時間)の変更により，スライス時間の使い切りによる VM の切り替えを削減できると期待される．後者(書き込み/読み込み期限切れ時間)の変更によ

り，飢餓発生による同一 VM のキュー処理の中断およびそれによる VM の切り替えを抑制できると期待される．

## 6.2 性能評価

AS の**予測期限切れ時間**を 1 倍から 6 倍に変更させた．これらを ASae1～ASae6 と呼ぶ．そして，**読み込み期限切れ時間**，**書き込み期限切れ時間**と**連続読み込み処理期限時間**，**連続書き込み処理期限時間**を 1 倍から 64 倍に変更させた．これらを ASrw1～ASrw64 と呼ぶ．

変更後の性能を図 10 に示す．図より，上記の値を増加させることにより I/O 性能が向上すること，**予測期限切れ時間**は 3 倍（ASae3）が優れていること(ただし一部おいて 4 倍が優れる例もある)，**読み込み/書き込み期限切れ時間**と**連続読み込み/書き込み処理期限時間**を増加させることによりほぼ単調に増加することが分かった．

次に，CFQ の前節の 4 値を 1 倍から 8 倍に変更したときの性能を図 11 に示す．これらを CFQ1 から CFQ8 と呼ぶ．図より，CFQ のスライス時間および書き込み/読み込み期限切れ時間を増やすことにより，I/O 性能が向上することが分かる．VM 間移動を削減した I/O スケジューラ(ASrw64ae3，CFQ8)と，既存の I/O スケジューラの性能比較を図 12 に示す．図より，CFQ8，ASrw64，CFQ，AS，Deadline，NOOP の順に高い I/O 性能が得られていることが分かり，提案した改良を施した I/O スケジューラの性能は，既存のもので最も高い性能が得られた例と比較しても，約 10%の性能向上が得られていることが分かる．

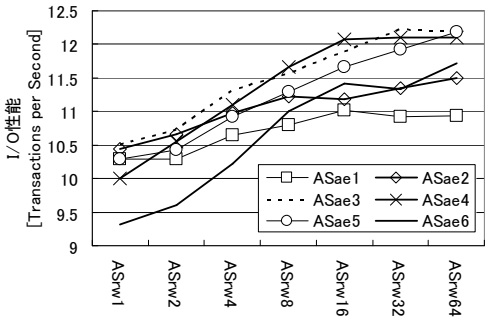


**図 10**　VM 間移動を削減した AS の性能

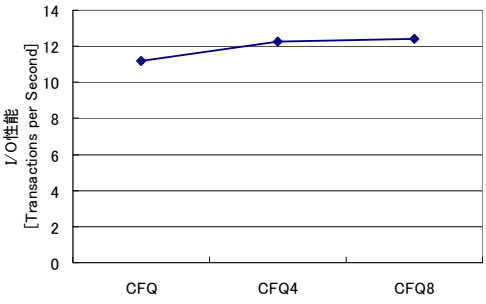


**図 11**　VM 間移動を削減した CFQ の性能

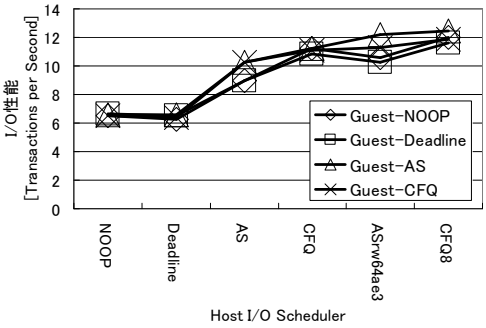


**図 12**　仮想化環境における I/O スケジューラの性能

## 6.3 動作解析

ホスト OS の I/O スケジューラを ASrw64ae3，ゲスト OS の I/O スケジューラを AS としたときに発行された I/O をホスト OS 上でモニタリングした結果を図 13 に示す．同様に，ホスト OS の I/O スケジューラを CFQ8，ゲスト OS の I/O スケジューラを AS としたときの発行 I/O を図 14 に示す．発行 I/O のシーク距離とシーク時間の関係を図 15，図 16 に示す．

図 13～図 16 より，ホスト OS の I/O スケジューラとして ASrw64ae3 や CFQ8 を用いることで既存の CFQ などより VM 間移動が削減されることが確認された．

次に，VM 間移動の頻度を図 17 に示す．横軸がホスト OS とゲスト OS の I/O スケジューラの組を表し，1 つ目がゲスト OS の I/O スケジューラ，2 つ目がホスト OS の I/O スケジューラを示している．例えば CFQ/NOOP はゲスト CFQ，ホスト NOOP を表している．縦軸は単位時間あたりの処理対象 VM の切り替え発

生回数を示している．図より既存の I/O スケジューラでは CFQ を用いた場合が最も切り替わりが少なく，VM 間移動を削減したスケジューリング手法である CFQ8 や ASrw64ae3 はこれよりさらに VM 間移動が削減していることが確認できる．

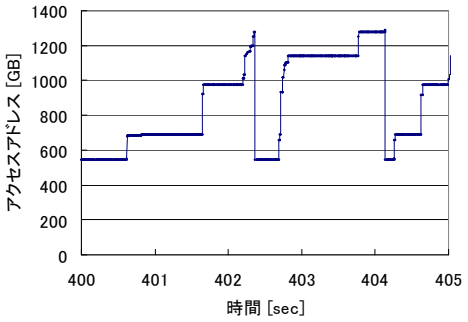


図 13　発行 I/O
(ホスト OS-ASrw64ae3，ゲスト OS-AS)

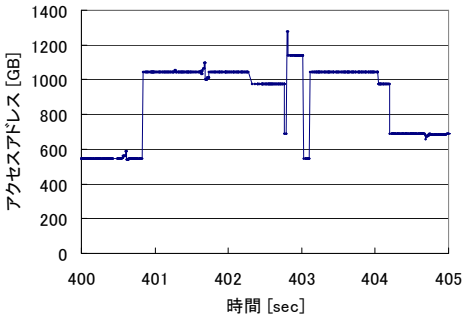


図 14　発行 I/O
(ホスト OS-CFQ8 ゲスト OSAS)

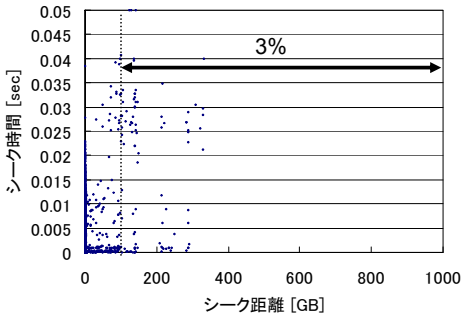


図 15　発行 I/O のシーク距離と時間
(ホスト OS-AS rw64ae3，ゲスト OS-AS)

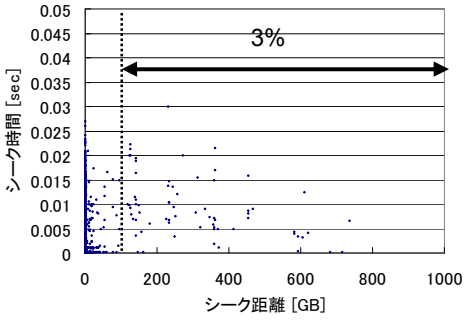


図 16　発行 I/O のシーク距離と時間
(ホスト OS-CFQ8，ゲスト OS-AS)

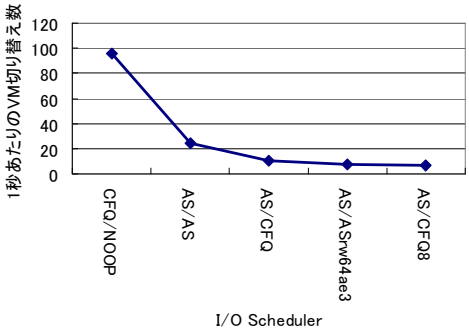


図 17　VM 切り替えの発生頻度

## 7. まとめ

本稿では，仮想化環境における I/O スケジューラの性能評価と，スケジューリング結果の解析を行い，性能低下の大きな原因として処理対象 VM の切り替えがあることを示した．そして，VM 切り替え回数を削減する手法を提案し，その性能を評価した．性能評価の結果，提案手法により I/O 性能が約 10%向上することが確認された．

今後は，ゲスト OS の I/O スケジューラの動作解析と性能向上に関する考察，公平性に関する考察を行う予定である．

## 謝辞

本研究は科研費(22700039)の助成を受けたものである．

# 参考文献

[1] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harries, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A., “Xen and the art of virtualization,” SOSP ’03: Proceedings of the nineteenth ACM symposium on Operating systems principles, New York, NY, USA, ACM Press, 2003, pp. 164-177.

[2] David Boutcher and Abhishek Chandra, “Does Virtualization Make Disk Scheduling Passé? ,” SOSP Workshop on Hot Topics in Storage and File System (Hot Storage ’09).

[3] 新居 健一, 山口 実靖, “仮想化環境における I/O スケジューラと I/O 性能の関係に関する一考察”, 第 72 回情報処理学会 全国大会 (4L-6)

[4] Jens Axboe, “Linux Block IO – present and future”, In Proceedings of the Ottawa Linux Symposium, pages 51-61. Ottawa Linux Symposium, July 2004

[5] Sitaram Iyer, Peter Druschel, “Anticipatory scheduling: A disk scheduling framework to overcome deceptive idleness in synchronous I/O”, In SOSP ’01: Proceedings of the eighteenth ACM symposium on Operating systems principles. ACM, 2001.

[6] Mukil Kesavan, Ada Gavrilovska, Karsten Schwan, “On Disk I/O Scheduling in Virtual Machines,” In WIOV ’10, May 2010.

[7] Steven L. Pratt, Dominique A. Heger “Workload Dependent Performance Evaluation of the Linux 2.6 I/O Schedulers,”, In Proceedings of the Linux Symposium, 2004

[8] Prashant Shenoy and Harrick M. Vin. “Cello: A disk scheduling framework for next generation operating systems,” In Proceedings of ACM SIGMETRICS Conference, pp. 44-55, 1997.

[9] Yu Zhang, Bharat Bhargava “Self-learning disk scheduling,” IEEE Trans. on Knowl. and Data Eng.,21(1), pp. 50-65, 2009.

[10] FFSB http://sourceforge.net/projects/ffsb/