仮想計算機環境に適したI/Oスケジューラの研究

山口 実靖
工学院大学 工学部 情報通信工学科

〒 163-8677 東京都新宿区西新宿 1-24-2

I/O Scheduling in Virtualized Environment

Saneyasu Yamaguchi
Department of Information and Communications Engineering, Kogakuin University

1-24-2 Nishishinjuku, Shinjuku, Tokyo, Japan

1 はじめに

計算機が広く普及し，計算機の消費電力や設置スペースの増加が問題となっている [1]．その対策として，仮想計算機によるサーバ統合や，仮想計算機を用いたクラウドコンピューティングが注目されている．しかし，既存のOSにおいてはI/O最適化を行うI/Oスケジューラが仮想計算環境を考慮しておらず，仮想計算機環境におけるI/O性能は必ずしも高くないと考えられる．

そこで我々は，ベアメタル型仮想化システムXen [2] を用いて1台の物理計算機上に複数台のVMを稼働させI/Oスケジューラごとの I/O性能を評価し，その動作を解析した．解析の結果，複数の仮想計算機が稼働する環境ではストレージ内に巨大な仮想HDDイメージファイルが複数存在し，巨大イメージファイル間の長距離シークがI/O性能を大きく低下させることと，既存のI/Oスケジューラを用いると巨大イメージファイル間の長距離シークが頻繁に発生し，I/O性能が低くなっていることを確認した．この問題に対して我々は，I/Oスケジューラの I/O発行制限などを延長しI/O対象の仮想HDDイメージファイルの切り替え頻度を減少させ，I/O性能を向上させる手法を提案する．

本論文ではまず，ベアメタル型仮想化システムとホスト型仮想化システムを用いて，既存のI/Oスケジューラの動作解析結果を示し，その課題を示す．そしてI/O性能向上手法を示し，最後にLinuxに標準で搭載されている4つのI/Oスケジューラと提案手法で改善したI/Oスケジューラの仮想化環境におけるI/Oスループットと I/O応答性能を評価する．

本論文の構成は以下のとおりである．まず第2章でI/Oスケジューラの説明を行い，第3章で関連研究を紹介する．第4章において既存のI/Oスケジューラがベアメタル型仮想化システムや，ホスト型仮想化システムにおいてどの様な性能を示すか評価し，仮想化環境においてI/Oスケジューラが I/O性能に大きな影響を与えることを示す．第5章ではスケジューラによって発行されたI/Oの動作解析を行い，第6章では性能向上手法について述べ，I/Oスケジューラの改善によるI/Oスループットと I/O応答性能の評価と動作解析を行う．第7章において本論文のまとめを行う．

2 I/Oスケジューラ

I/Oスケジューラはプロセス群から発行されるI/O要求群を適切な順に並び替え，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にI/Oスケジューラが呼び出され，新しいI/O要求をキューのどの位置に追加するかを決定する．

LinuxにはNOOP，Deadline，AS[3][4]，CFQの4つのI/Oスケジューラが存在する．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を発行順に処理するスケジューラである．よって，スケジューリングによる負荷が小さく，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適している．Deadlineは現在処理しているI/O要求の近隣のI/O要求を優先して処理することでHDDヘッドの移動を削減するスケジューラである．近隣ではないI/O要求は保留されてしまうが，これには限界値が用意されており限界値より長く保留されたI/O要求は優先的に処理される．レイテンシの上限が

保障されているため，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されている．AS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I/O要求が発行されると予測された場合は，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それらを優先して処理する．ASはシーケンシャルアクセスの多い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り，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要求はプロセス単位で公平に処理される．CFQにもDeadlineやASと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．

仮想計算機では，アプリケーションから発行されたI/O要求群はゲストOS上のI/Oスケジューラによってスケジューリングされ，それらがVMMなどを通してホストOSに渡されホストOSのI/Oスケジューラによってスケジューリングされる．しかし，これらの既存のI/Oスケジューラは実計算機上に搭載されている物理ストレージデバイスへのスケジューリングを想定して実装されているため，仮想計算機環境では十分に高いI/O性能が得られないと考えられる．

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．

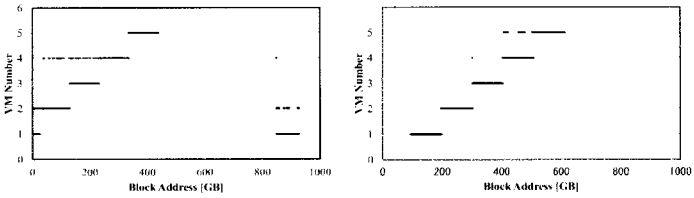
Boutcherらは文献[5]において，仮想化環境においてI/Oスケジューラが I/O性能にどのような影響を与えるか評価しており，仮想化環境において適切なI/Oスケジューラを選択することによりI/O性能が向上すること，ゲストOSのI/Oスケジューラとして優れているものはワークロードに依存すること，などを主張している．

Kesavan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，性能向上に関する推論を述べている．

Xuらは文献[7]において，プロセスのI/O要求の局所性を利用した新しいスケジューラを提案している．当該手法では，I/O要求を処理する際に次のI/O要求に強い局所性が見られればこれを処理せずに待機し，局所性の強いI/O要求群を連続して処理する．著者らは，I/O要求に強い局所性が見られた場合，それらのI/O要求は同一のプロセスあるいはその子プロセスから発行されたI/Oである可能性が高く，それらは待機し連続して処理を行った方が効率が良いと主張している．ディスクアレイ環境，Xenを用いた仮想化環境，PVFS[8]を用いた分散ファイルシステム環境において性能評価を行い，提案したスケジューラがそれらの環境において有効であることを示している．

表 1: 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	16KByte
1opあたりの読込み量	4KByte
1opあたりの書込み量	4KByte
スレッド数	16



（ベアメタル型仮想化システム）　（ホスト型仮想化システム）

図 1: 仮想イメージの配置

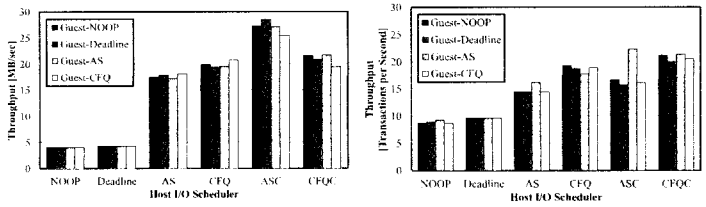
4 I/Oスケジューラの性能評価

4.1 測定環境

既存のI/Oスケジューラが仮想計算機の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．ベアメタル型仮想化システムのXenと，ホスト型仮想化システムを用い，OSとしてLinuxを用いた．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した．ベアメタル型仮想化システムを用いたときの各仮想ディスクイメージの物理ディスクにおける存在領域を図1左に，ホスト型仮想化システムを用いたときの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を図1右に示す．

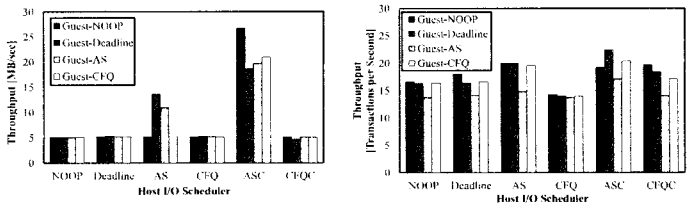
図1左よりVM1のイメージファイルは830[GB]付近から940[GB]付近と0[GB]付近から20[GB]に，VM2は20[GB]付近から100[GB]付近に，VM3は130[GB]付近から230[GB]付近に配置されている．これらより，VMのイメージファイルはほぼ連続的に配置されていることがわかる．同様にホスト型仮想化システムを用いた場合もVMのイメージがほぼ連続的に配置されていることがわかる．

性能評価は，マイクロベンチマークとアプリケーションベンチマークにより行った．アプリケーションベンチマークではLinux



（シーケンシャルリード）　（FFSB）

図 2: 仮想化環境における I/O スケジューラの I/O 性能（ベアメタル型仮想化システム）



（シーケンシャルリード）　（FFSB）

図 3: 仮想化環境における I/O スケジューラの I/O 性能（ホスト型仮想化システム）

コマンドの dd を用いて 4[GB] のファイルに対するシーケンシャルリードを行い性能を評価し，アプリケーションベンチマークとしては FFSB（Flexible File System Benchmark）[?] を実行し性能評価を行った．FFSB ベンチマークの設定は表 1 のとおりであり，

4.2 I/O 性能

ベアメタル型仮想化システムを用いたときの I/O 性能の測定結果を図 2 に，ホスト型仮想化システムを用いたときの I/O 性能の測定結果を図 3 に示す．左図がシーケンシャルリード性能測定結果を表し，右図が FFSB を用いた測定結果を表す．縦軸が I/O 性能を表し，横軸がホスト OS の I/O スケジューラ，各線がゲスト OS の I/O スケジューラを表す．横軸のホスト OS の I/O スケジューラの CFQC，ASC は我々が改善したものであり，第 6 章にて後述する．

図 2 より，ベアメタル型仮想化システムを用いたときの I/O 性能は，シーケンシャルリードと FFSB の両測定においてホスト OS の I/O スケジューラを変更することにより大きく変化し，CFQ，AS，Deadline，NOOP の順に優れていることがわかる．これに対して，ゲスト OS の I/O スケジューラの変更による I/O 性能の変化は小さいことがわかる．

図 3 左より，ホスト型仮想化システムを用いたときのシーケンシャル性能は，ホスト OS の I/O スケジューラが AS の場合にのみ高く，それ以外 (NOOP,Deadline,CFQ) の場合はゲスト OS の I/O スケジューラによらず低いことがわかる．また，ホスト OS の I/O スケジューラが AS の場合はゲスト OS の I/O スケジューラを変更することにより性能が大きく変化し，Deadline と AS の性能は NOOP，CFQ の 2 倍以上になることがわかる．図 3 右より，FFSB を用いた測定では，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，AS, Deadline，NOOP，CFQ の順に優れていることがわかる．また，ゲスト OS の I/O スケジューラによって I/O 性能の差が生じていることも確認でき，NOOP，Deadline，CFQ，AS の順に優れていることがわかる．

以上をまとめると，ベアメタル型仮想化システム環境ではホスト OS の I/O スケジューラとしては AS, CFQ が適しており，ゲスト OS の I/O スケジューラは性能に大きな影響を与えないと言える．そして，ホスト型仮想化システム環境ではホスト OS の I/O スケジューラとしては AS が優れており，ゲストの I/O スケジューラとしては Deadline と NOOP が優れていると言える．

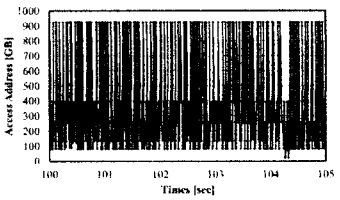
5 発行 I/O の解析

シーケンシャルリードおよび 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．発行 I/O 要求の解析について本章で示す．

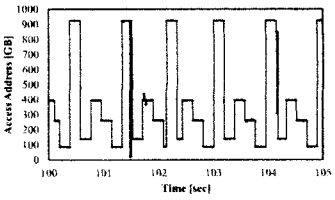
5.1 ベアメタル型仮想化システム

まず，ベアメタル型仮想化システムを用いたときの発行 I/O について考察する．シーケンシャルリードを実行した際に発行された I/O の解析結果を図 4 に示す．左図がホスト OS とゲスト OS の I/O スケジューラがそれぞれ NOOP，NOOP のときの発行 I/O を表し，右図が CFQ，NOOP における発行 I/O を表している．左図は，前章の測定で性能が低かった組み合わせであり，右図は，性能が高かった例である．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 の発行時刻を表している．同様に，FFSB 実行時に発行された I/O の解析結果を図 5 に示す．左図はホスト OS とゲスト OS の I/O スケジューラが NOOP，NOOP，右図は CFQ，NOOP における発行 I/O を表している．同様に前者が性能が低く，後者が性能が高い組み合わせである．

両図より，I/O 性能の低い図 4 左や，図 5 左（ともにホスト OS の I/O スケジューラが NOOP）では，アクセス対象の VM イメージの変更が頻繁に発生し，ストレージは長距離のシークを頻繁に行っていることがわかる．これに対して I/O 性能の高い図 4 右や図 5 右（ともにホスト OS の I/O スケジューラは CFQ）では，アクセス対象 VM の変更が少なく，I/O 性能が高くなっていることが確認できる．

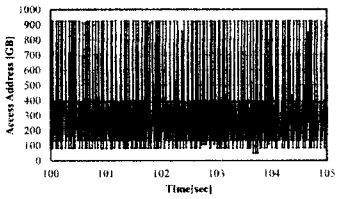


(HostOS–NOOP, GuestOS–NOOP)

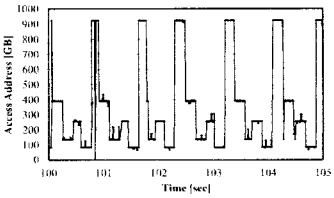


(HostOS–CFQ, GuestOS–CFQ)

図 4: 発行 I/O 要求（ベアメタル型仮想化システム，シーケンシャルリード）



(HostOS–NOOP, GuestOS–NOOP)



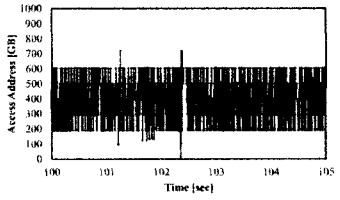
(HostOS–CFQ, GuestOS–NOOP)

図 5: 発行 I/O 要求（ベアメタル型仮想化システム，FFSB）

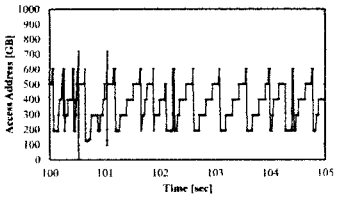
5.2 ホスト型仮想化システム

次に，ホスト型仮想化システムを用いたときの発行 I/O の解析結果を示す．シーケンシャルリードを実行した際に発行される I/O の解析結果を図 6 に示す．左図がホスト OS とゲスト OS の I/O スケジューラが NOOP，NOOP のときの発行 I/O を表し，右図が AS，Deadline のときの発行 I/O を表している．同様に，FFSB 実行時の発行 I/O の解析結果を図 7 に示す．左図がホスト OS とゲスト OS の I/O スケジューラが NOOP，NOOP，右図が AS，NOOP ときの発行 I/O を表している．図 6，7 ともに，左図が性能が低い例であり，右図が性能で優れている例である．

両図より，前節の仮想化システムにベアメタル型仮想化システムを用いた場合と同様に，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 イメージの変更が頻繁に発生しており，ヘッドの移動距離が長い I/O が多数発生していることがわかる．そして両図の右図より，ホスト OS にて AS を用いた時はアクセス対象 VM の切り替え回数が少なく，I/O 性能が高くなっていることがわかる．

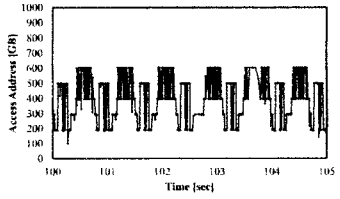


(HostOS–NOOP, GuestOS–NOOP)

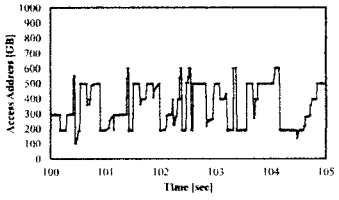


(HostOS–AS，GuestOS–DL)

図 6: 発行 I/O 要求（ホスト型仮想化システム，シーケンシャルリード）



(HostOS–NOOP, GuestOS–NOOP)



(HostOS–AS, GuestOS–NOOP)

図 7: 発行 I/O 要求（ホスト型仮想化システム，FFSB）

6 VM 間移動の削減による I/O スケジューラの改善

6.1 I/O スケジューラの変更

本章にて，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 に対して変更を行い，I/O 性能を向上させる手法を提案する．. 以下に，その詳細を記す．

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 資源の飢餓状態であると判定する閾値（default_read_expire：読込み期限切れ時間，default_write_expire：書込み期限切れ時間）」と「I/O 要求をどれだけ連続して処理するかを定める許容時間（default_read_expire：連続読込み時間，default_write_expire：連続書込み処理期限時間）を増加させる．飢餓状態とは，I/O 要求が長時間処理されずいる状態のことである．

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．読込み期限切れ時間と書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．連続読込み処理時間と連続書込み処理時間の変更により，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「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 資源の飢餓状態であると判定する閾値（cfq_fifo_expire[0]：書込み期限切れ時間，cfq_fifo_expire[1]：読込み期限切れ時間）」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（書込み/読込み期限切れ時間）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 3 倍に，読込み期限切れ時間，書込み期限切れ時間，連続読込み処理期限時間，連続書込み期限時間を 64 倍にしたものを ASC と呼ぶ．ASC の I/O 性能を図 2 と図 3 の”ASC”に示す．CFQ の前節の 4 値を 8 倍にしたものを CFQC と呼ぶ．CFQC の I/O 性能を図 2 と図 3 の”CFQC”に示す．

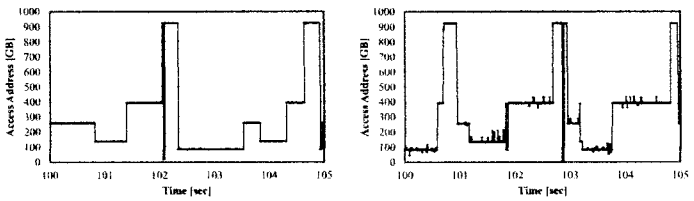
図 2，3 より，ホスト型仮想化システムにてシーケンシャルリードを行った場合の CFQC を除き，提案手法の適用により I/O 性能が大きく向上することが確認された．ホスト型仮想化システムにてシーケンシャルリードを行ったときの CFQC のみ，性能向上が見られなかったが，性能の低下はなく，提案手法はホスト型仮想化システムの I/O 性能向上に関しても有効であるといえる．

6.3 動作解析

ベアメタル型仮想化システムを用いた時の提案手法の動作を図 8 に，ホスト型仮想化システムを用いた時の動作を図 9 に示す．これらを，図 4 から図 7 と比較することと，提案手法の適用により I/O 対象 VM の切り替えに伴う長距離シークがさらに削減されていることがわかる．

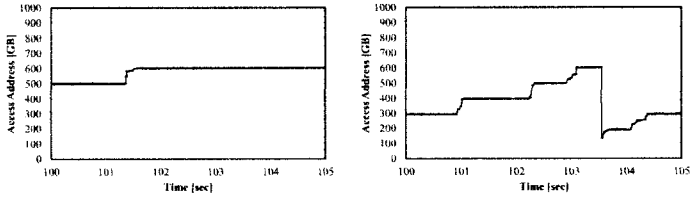
6.4 応答性能

提案手法は処理対象 VM の切り替えの頻度を低下させるため，各 I/O 要求の応答性能が低下する可能性が考えられる．本節において，提案手法の適用による応答性能の変化について調査する．提案手法の応答性能を評価するために，各 VM 上で FFSB を実行し，各読込み要求の応答性能の測定をした．応答時間とは，読込み要求の発行から読込み処理終了までに要した時間のことである．



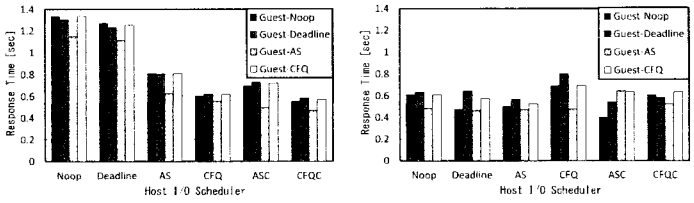
（シーケンシャルリード）（HostOS–CFQC, GuestOS–CFQ）　（FFSB）（HostOS–CFQC, GuestOS–NOOP）

図 8: 発行 I/O 要求（ベアメタル型仮想化システム）



（シーケンシャルリード）（HostOS–ASC, GuestOS–Deadline）　（FFSB）（HostOS–ASC, GuestOS–NOOP）

図 9: 発行 I/O 要求（ホスト型仮想化システム）



（ベアメタル型仮想化システム）　（ホスト型仮想化システム）

図 10: 仮想化環境における平均応答性能

ベアメタル型仮想化システムを用いたときの応答時間を図 10 左に，ホスト型仮想化システムを用いたものを図 10 右に示す．縦軸が発行された読込み要求の応答時間（ターンアラウンドタイム）の平均を表している．

図 10 左より，ベアメタル型仮想化システムを用いた場合，ホスト OS の I/O スケジューラを変更することにより応答時間が大きく変化し，CFQC，ASC，CFQ，AS，Deadline，NOOP の順に優れていることがわかる．また，ゲスト OS の I/O スケジューラを変更することでも応答時間が変化し，AS，NOOP，Deadline，CFQ の順に優れていることがわかる．

図 10 右より，ホスト型仮想化システムを用いた場合，ホスト OS の I/O スケジューラの変更による応答時間の変化はほとんどなく，ゲスト OS の I/O スケジューラを変更することで応答時間が小さく変化し，AS，NOOP，CFQ，Deadline の順に優れていることがわかる．

既存のスケジューラである AS，CFQ と，改善手法である ASC，CFQC を比較すると，多くの例において応答時間に小さな改善がみられ，提案手法は平均応答時間に関しても多くの場合は有効であると考えられる．一部の例において平均応答時間の小さな増加がみられるが，程度は限定的である．

7 おわりに

本論文では，仮想化環境における I/O スケジューラの改善手法を示した．そして，ベアメタル型仮想化システムとホスト型仮想化システムを用いた場合における改善手法の性能を評価し，両仮想化システムにおいて I/O 性能が向上することを確認した．

謝辞

本研究の遂行にあたり，公益財団法人 高柳記念財団の助成を受けました．ここに記し，感謝の意を表します．

参考文献

[1] J. G. Koomey, "Estimating total power consumption by servers in the U.S. and the world," [Online]. http://enterprise.amd.com/Downloads/svrpwrusecompletefinal.pdf

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. and A. Warfield, "Xen and the Art of Virtualization," SOSP'03, pp.164-177, New York, USA, 2003

[3] S. Iyer and P. Druschel, "Anticipatory scheduling: A disk scheduling framework to overcome deceptive idleness in synchronous I/O," SOSP'01:Proceedings of the eighteenth ACM symposium on operationg systems principles. ACM, 2001.

[4] Andrew Morton, "Linux: Anticipatory I/O Scheduler," http://kerneltrap.org/node/567

[5] D. Boutcher and A. Chandra, "Does Virtualization Make Disk Scheduling Passe?," SOSP Workshop on Hot Topics in Storage and File System(Host Storage '09)

[6] M. Kesavan, A. Gavilovska and K. Schwa, "On Disk I/O Scheduling in Virtual Machines," WIOV'10, May 2010

[7] Y. Xu and S. Jiang, "A Scheduling Framework that Makes any Disk Schedulers Non-work-conserving solely based on Request Characteristics," FAST2011

[8] P. Carns, W. Ligon, R. Ross, and R. Thakur, "PVFS: A Parallel File System For Linux Cluster," Proc. of IEEE International Conference on Cluster Computing, 2008