

仮想計算機環境における VM イメージファイルの配置に関する一考察

山田将也[†] 山口実靖[†]

計算機システムの I/O 性能を向上させる手法としてストレージ内のデータの再配置がある。近年急速に普及が進んでいる仮想計算機環境では、ストレージ上に巨大な仮想 HDD イメージファイルが複数配置されることが多く、データの再配置による I/O 高速化が特に重要になると考えられる。しかしこれまでの再配置手法はデータが物理ストレージ上に直接配置されることを前提としており、これらの手法をファイルシステムなどが二重構造になる仮想計算機環境に直接適用するのは適切ではないと考えられる。

本稿では複数の VM イメージファイルが格納されているストレージ環境を想定し、各再配置手法の性能の評価と考察を行った。その結果、ホスト OS にて再配置を行う手法がゲスト OS にて再配置を行う手法よりも優れていることが確認された。また、処理負荷が少ないパーティション型 VM イメージファイル構築手法よりも、再配置自由度の高いイメージファイル型構成手法の方が再配置後の性能が高いことが確認された。

A Study on Layout Optimization of Virtual Machine Image Files

Masaya Yamada[†] Saneyasu Yamaguchi[†]

Storage re-organization is one of famous I/O performance improving methods. In these methods, the most frequently accessed data is placed in the middle of storage in order to decrease storage seek distance. These methods assume physical computer environment. Consequently, they cannot work effectively in virtual machine environments. In this paper, we explore I/O performance of several storage re-organization methods. Our evaluation demonstrated that re-organizing in the host OS layer is better than doing in the guest OS layer.

1. はじめに

計算機システムで扱うデータ量の増加に伴い、増加したデータを高速に処理するた

[†]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

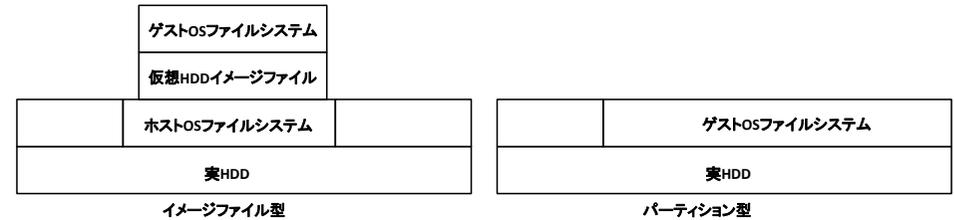


図1 二重ファイルシステム構造

めにストレージシステムには高い I/O 性能が要求されるようになってきている。しかし、主要なストレージデバイスであるハードディスクドライブ(HDD)はデータへアクセスする際、磁気ヘッドのシーク、磁気ディスクの回転といった動作を伴うためデータへのランダムアクセスの性能が高くない。そのため、ストレージシステムの I/O 性能の向上が計算機システムの重要な課題の一つとなっている。I/O 性能の向上を実現する手法の一つに、HDD のレイアウトを変更する手法がありこれまで研究されてきている。これらの手法では、高い頻度でアクセスされるデータをストレージの中央に配置して平均シーク距離を低減させたり、連続アクセスされるファイル群を近隣に配置することによりシーク距離を削減させる。

近年は仮想計算機(VM)環境が普及している。この環境では巨大な仮想 HDD イメージファイルがストレージ上に複数存在することが多く、その場合はストレージデバイスは巨大な仮想 HDD イメージファイルの間のシークを頻繁に行うことになり I/O 性能が低くなる。よって、VM 環境ではストレージ上のデータの再配置による高速化が特に重要になる。しかし、これまでに提案されてきた再配置手法は物理ストレージ上にデータ直接可能脳されていることを想定しており、図1のように二重にファイルシステムが構築される VM 環境を想定していない。よって、これらの既存の手法を VM 環境に直接適用するのは適切ではなく、VM 環境に適したディスク再配置に関する考察が重要であると考えられる。

2. 仮想計算機環境

CPU、メモリ、HDD などの計算機資源を仮想化し物理計算機上で動作する仮想的な計算機のことを仮想計算機(VM)という。実計算機上で動作する OS をホスト OS、VM 上で動作する OS をゲスト OS と呼ぶ。

VM における HDD の仮想化は、実計算機の HDD 内の指定の領域を VM に HDD として認識させることにより行う。VM に認識させた領域を仮想 HDD と呼び、仮想 HDD を作成するには 2 つの方法がある。1 つはホスト OS のファイルシステム上にイメー

ジファイルを作成して、それを仮想 HDD として認識させる方法である。この方法ではイメージファイル内にゲスト OS のファイルシステムを構築する。ゲスト OS 内でディスクアクセスが生じると、ホスト OS 上でイメージファイルへアクセスが発生する。もう 1 つは実計算機のパーティションを VM に割り当てる方法である。この方法ではストレージ上の空きパーティションに直接ゲスト OS のファイルシステムを構築する。ゲスト OS 内でディスクアクセスが生じると、割り当てたパーティションへのアクセスが発生する。

イメージファイル方式では、仮想 HDD のイメージファイルはホスト上では単一の巨大なファイルとして扱われるので、VM 環境を他の実計算機に移動することや、ディスクレイアウトの変更が容易である。パーティションを用いた場合、VM 環境の移動やディスクレイアウトの変更は困難である。よって再配置の柔軟性においてはイメージファイル方式が優れていると考えられる。逆に性能に関しては、処理が単純なパーティションが手法が優れていると考えられる。

イメージファイルを用いる場合、ゲスト OS 内でのアクセスがホスト OS 内ではイメージファイルへのアクセスとなるため、ゲスト OS、ホスト OS 両方のファイルシステムを経由することになる。それに対して、パーティションを用いる場合、ゲスト OS 内でのアクセスがホスト OS 内では実パーティションへのアクセスとなるので、イメージファイルを用いた場合に比べて必要な処理が少なくなる。

3. 関連研究

本章で、本研究と関連する研究を紹介する。

最初に ファイルシステムレベルの研究を紹介する。FFS[1]やその後続の研究[2,3]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。本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[4,5]。また文献[6]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[7 0 ,NILFS]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[8]。

Hierarchical File System(HFS)[9]の Hot File Clustering や Smart File System[10]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する。

FS2 (Free Space Filesystem)[11]では、ファイルの複製を用いて連続アクセスされるフ

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。

次に、動的レイアウト手法の紹介を行う。

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[12]が、シミュレーションによる研究として文献[13,14,5,15]がある。文献[12]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。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 がある[15]。本手法では organ pipe 手法により、シリンダーの順の並び替えを行う。また、並び替えの単位をシリンダーでなく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[5]がある。

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[4]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い。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[16,17,18,19]がある。

4. 再配置適用レイヤ

VM 環境ではホスト OS のファイルシステム上、ゲスト OS のファイルシステム上のそれぞれでホットスポットの再配置を行うことができる。

4.1 ゲスト OS におけるホットスポット再配置

複数の VM が稼働している環境で、各ゲスト OS 上で仮想 HDD へのアクセス要求をモニタリングし、特にアクセスの集中した領域(ホットスポット)をファイル単位で仮想 HDD 内の空き領域に再配置する。各 VM が他の VM の実 HDD 上の位置情報を持たない場合は、図 2 のように各 VM が独立してホットスポットを仮想 HDD の中央に再配置する。

各 VM が他の VM の位置情報を持ち VM 同士が連携してホットスポットの再配置を行う場合は、図 3 のようにホットスポットを他の VM に近づけて再配置する。この手法はイメージファイル型、パーティション型の両方に適用することができる。

4.2 ホスト OS におけるホットスポット再配置

複数の VM が稼働している環境で、ホスト OS 上で仮想 HDD へのアクセス要求をモニタリングし、図 4 のようにホットスポットをブロック単位で実 HDD 上の空き領域に再配置する。この手法はイメージファイル型にのみ適用することができる。

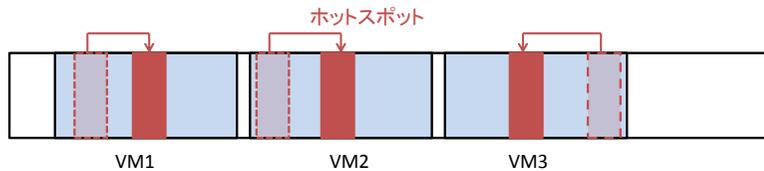


図 2 ゲスト OS 上でのホットスポット再配置(各 VM 独立)

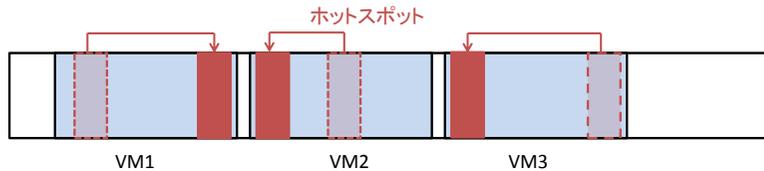


図 3 ゲスト OS 上でのホットスポット再配置(各 VM 連携)

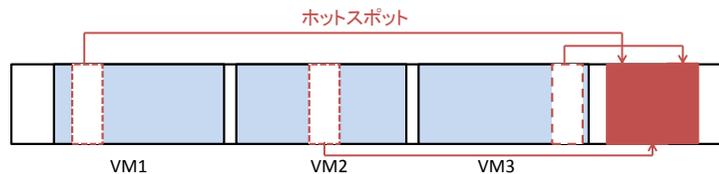


図 4 ホスト OS 上でのホットスポット再配置

5. 性能評価

VM 環境にデータ再配置手法を適用することの有効性を確認するために、性能評価実験を行った。

5.1 再配置適用レイヤの比較

1 台の実計算機上にイメージファイルサイズが 100[GB]の VM を 3 台稼働させ全 VM で同時にベンチマークソフト FFSB を実行した。その際、ゲスト OS 上で仮想 HDD へのアクセス要求を、ホスト OS 上で実 HDD へのアクセス要求をモニタリングした。モニタリングの結果から得られたホットスポットをゲスト OS、ホスト OS それぞれで再配置し性能の比較を行った。また、100[GB]のパーティションを使用する VM を 3 台稼働させ、ゲスト OS 上での再配置を行い、イメージファイル型とパーティション型の性能を比較した。この時の測定環境を表 1、表 2、表 3 に、測定結果を図 5 に示す。

表 1 測定環境 1 (実計算機)

OS	CentOS 5.5 x86_64
CPU	Phenom II X4 955
Memory	256 [MB]
仮想HDD	100 [GB]
kernel	2.6.18.8

表 2 測定環境 2 (仮想計算機)

OS	CentOS 5.5 x86_64
CPU	Phenom II X4 955
Memory	4 [GB]
HDD	1 [TB]
	7200 [rpm]
kernel	2.6.18.8
Xen	3.4.3

表 3 FFSB 設定

実行時間 [sec]	500
スレッド数	1
ファイル数	200
ファイルサイズ [Byte]	10485760
読み込みサイズ [Byte]	16384
読み込みブロックサイズ [Byte]	4096
書き込みサイズ [Byte]	16384
書き込みブロックサイズ [Byte]	4096

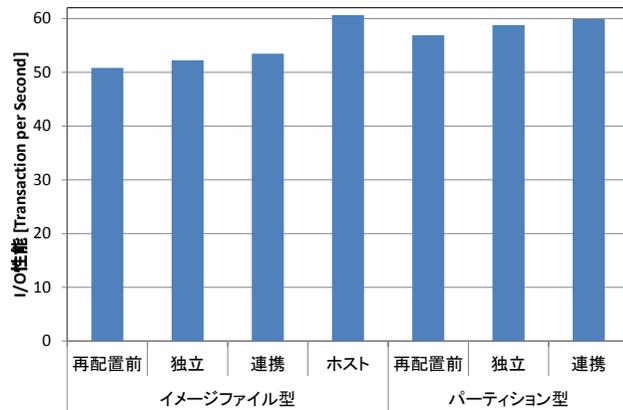


図 5 性能比較結果

図 5 から、イメージファイル型、パーティション型共に、各 VM が連携し再配置を行う場合の性能が最も高く、各 VM が独立して再配置を行う場合が 2 番目、再配置を行わない場合が最も性能が低くなっていることが確認できる。イメージファイル型で再配置前の実 HDD へのアクセス要求を図 6 に、各 VM が独立して再配置を行った後の実 HDD へのアクセス要求を図 7 に、各 VM が連携して再配置を行った後の実 HDD へのアクセス要求を図 8 に、ホスト上で再配置を行った後の実 HDD へのアクセス要求を図 9 に示す。図 6 から、実 HDD 上の 60[GB]付近、120[GB]付近、300[GB]付近にホットスポットができていのがわかる。各 VM で独立して再配置を行い、図 7 のように 50[GB]、150[GB]、250[GB]付近にホットスポットを再配置した。

次に各 VM が連携して再配置を行った結果、図 8 のようにホットスポットが 100[GB] 付近、105[GB] 付近、210[GB] 付近に再配置された。ホスト上でホットスポットの再配置を行うことにより、300[GB] 付近にアクセス要求が集まった。図 6, 7, 8, 9 から再配置を行うことにより、VM 間でホットスポットの距離が小さくなっている。このときの実 HDD で生じているシークの距離と時間を図 10, 図 11, 図 12, 図 13 に示す。図 10 から、再配置を行わないと最大で 250[GB] の巨大なシークが発生することがわかる。各 VM が独立してホットスポットの再配置を行うことにより、発生する最大のシークが 200[GB] に減少していることが図 11 から確認できる。各 VM が連携してホットスポットを再配置することにより、図 12 の通り発生するシークが最大 110[GB] になった。

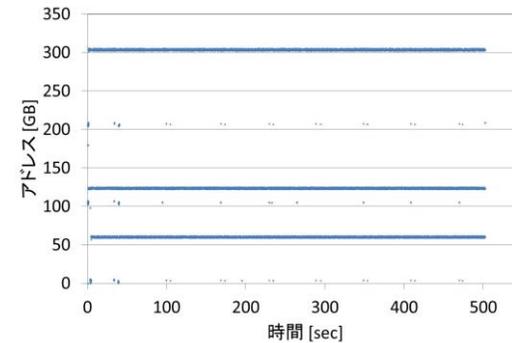


図 6 実 HDD へのアクセス要求分布 (再配置前)

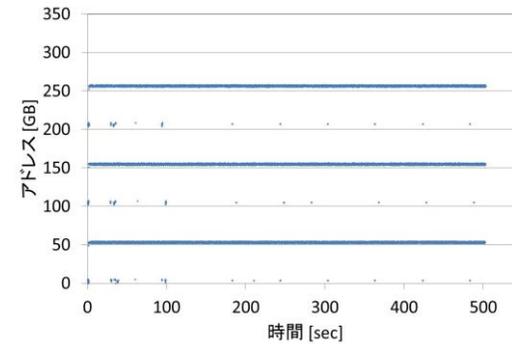


図 7 実 HDD へのアクセス要求分布 (各 VM 独立・再配置後)

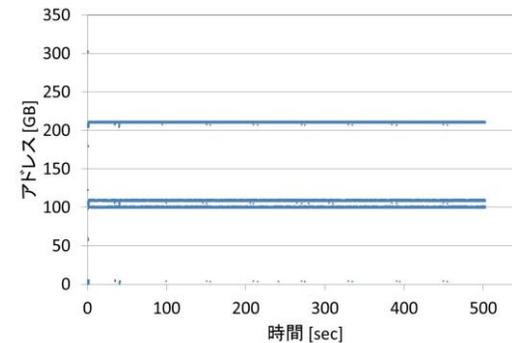


図 8 実 HDD へのアクセス要求分布 (各 VM 連携・再配置後)

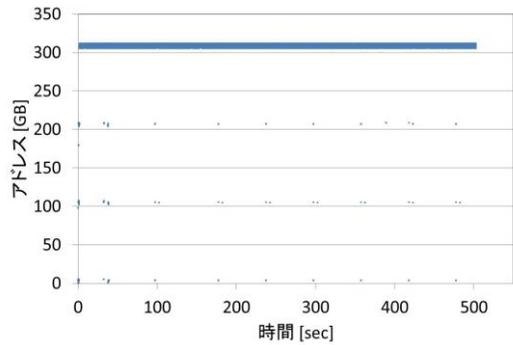


図 9 実 HDD へのアクセス要求分布 (ホスト OS 上・再配置後)

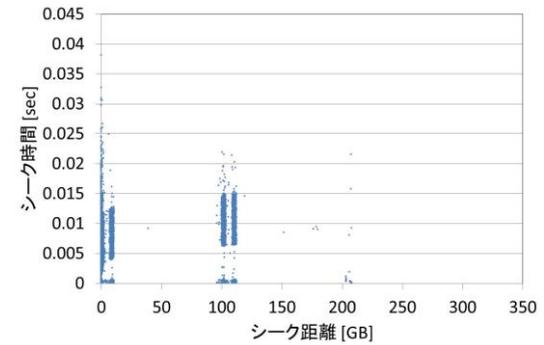


図 12 ホスト OS におけるシーク距離と時間 (各 VM 連携・再配置後)

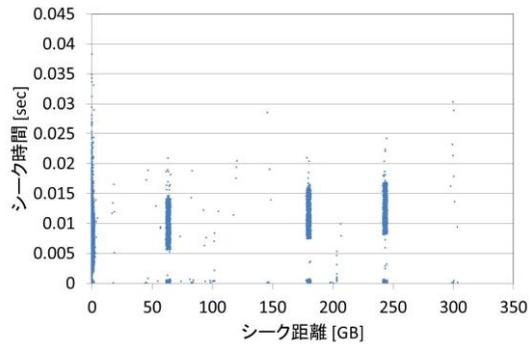


図 10 ホスト OS におけるシーク距離と時間 (再配置前)

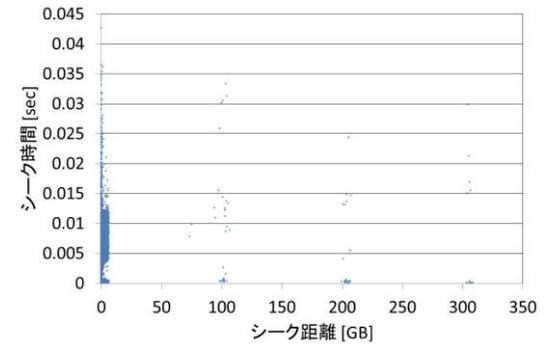


図 13 ホスト OS におけるシーク距離と時間 (ホスト OS 上・再配置後)

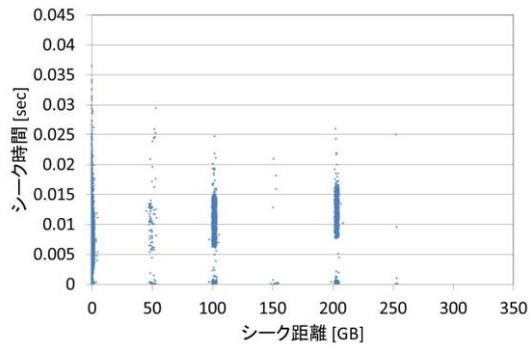


図 11 ホスト OS におけるシーク距離と時間 (各 VM 独立・再配置後)

ホスト上で再配置を行うことで、これまでに生じていた巨大なシークの削減ができた。ゲスト OS 上での再配置は VM に割り当てられている仮想 HDD 領域を超えたデータの移動が行えない。それに対してホスト OS 上での再配置は、仮想 HDD 領域に捕らわれないため、限りなくホットスポットの距離を削減することができる。

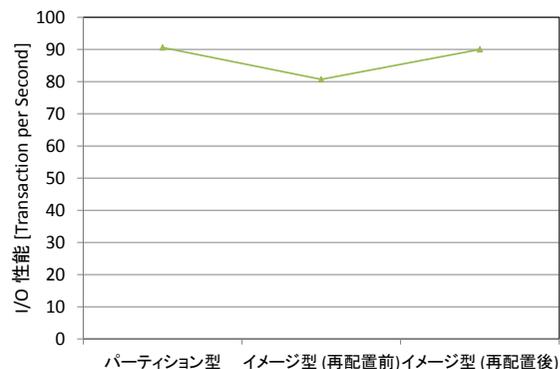


図 14 稼働 VM 数 2 台時のパーティション型とイメージファイル型の比較

5.2 イメージファイル型とパーティション型の比較

1 台の実計算機上に 100[GB]のパーティションを使用する VM を 2 台稼働させた場合の性能と、イメージファイルサイズが 100[GB]の VM2 台をホスト上で再配置を行った場合の性能の比較を行った。その結果を図 14 に示す。再配置を行うことによってイメージファイル型の性能がパーティション型と同等の性能になることを確認した。また、図 5 から稼働 VM 数が 3 台の場合、再配置を行ったイメージファイル型の性能が、再配置を行ったパーティション型の性能を上回った。このことから、稼働 VM 数が増加すると、ホスト上で再配置を行った際の性能向上率が高くなると考えられる。

5.3 VM 数と性能向上率

VM 数と性能向上率の関係についての確認実験を行った。

1 台の実計算機上で、VM 数が 2 台～4 台の場合で、ホスト上におけるホットスポット再配置を実行し、その前後で性能の比較を行った。この実験で使用した環境を表 4、表 5 に示す。測定の結果を図 15 に示す。図 15 から、VM 数が増加することにより、再配置を行った際の性能向上率が大きくなっていくことが確認できた。これは、VM 数が増加するとイメージファイル数が増え巨大なシーク多数発生し、それを削減したためである、

表 4 測定環境 3 (実計算機)

OS	CentOS 5.5 x86_64
CPU	AMD Athlon(tm) Processor 1640B
Memory	4 [GB]
HDD	1 [TB]
	7200 [rpm]
kernel	2.6.18.8
Xen	3.4.3

表 5 測定環境 4 (実計算機)

OS	CentOS 5.5 x86_64
CPU	AMD Athlon(tm) Processor 1640B
Memory	256 [MB]
仮想HDD	100 [GB]
kernel	2.6.18.8

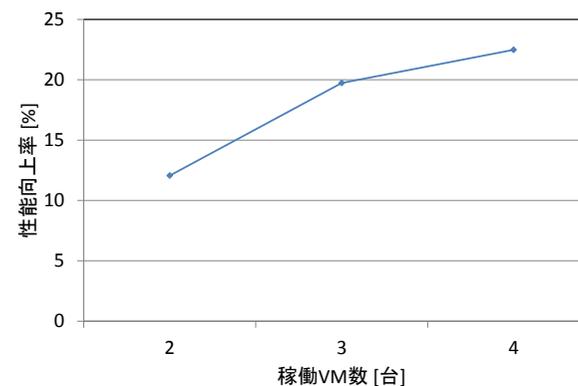


図 15 稼働 VM 数と性能向上率の関係

6. おわりに

本稿では VM 環境におけるデータ再配置手法の性能評価と考察を行った。ゲスト OS 上でホットスポットを再配置した場合に比べ、ホスト OS 上でホットスポットを再配置した場合の方が性能が高くなることを確認した。また、再配置を行うことによりイメージファイル型の性能がパーティション型の性能より高くなることを確認した。今後は動的な再配置について考察する予定である。

謝辞

本稿は科研費（22700039）の助成を受けたものである。

参考文献

- [1] M. K. McKusick et al., A Fast File System for UNIX, ACM Transactions on Computing Systems (TOCS), 2(3), 1984.
- [2] R. Card and T. Ts'o and S. Tweedle, Design and Implementation of the Second Extended Filesystem, First Dutch International Symposium on Linux, 1994.
- [3] P. Barford and M. Crovella, Generating Representative Web Workloads for Network and Server Performance Evaluation, Proceedings of the ACM Measurement and Modeling of Computer Systems, 151.160, 1998.
- [4] Sedat Akyurek and Kenneth Salem, Adaptive Block Rearrangement, Computer Systems, 13(2): 89.121, 1995.
- [5] C. Ruemmler and J. Wilkes, Disk Shuffling, HP Technical Report, HPL-CSP-91-30, 1991.
- [6] Greg Ganger and Frans Kaashoek, Embedded Inodes and Explicit Groups: Exploiting Disk Bandwidth for Small Files, USENIX Annual Technical Conference, 1-17. 1997.
- [7] M. Rosenblum and J. Ousterhout, The Design and Implementation of a Log-Structured File System, ACM Transactions on Computer Systems, 26-52, 1992.
- [8] J. Wang and Y. Hu, PROFS . Performance-Oriented Data Reorganization for Log-structured File System on Multi-Zone Disks, The 9th International Symposium on Modeling, Analysis and Simulation on Computer and Telecommunication Systems, 285.293, 2001.
- [9] HFS Plus Volume Format, <http://developer.apple.com/technotes/tn/tn1150.html>.
- [10] C. Staelin and H. Garcia-Molina, Smart Filesystems, USENIX Winter, 45.52, 1991.
- [11] Hai Huang, Wanda Hung, Kang G. Shin "FS2: Dynamic Data Replication in Free Disk Space for Improving Disk Performance and Energy Consumption", SOSPO'05, pp.263-276, October. 2005.
- [12] C. K. Wong, Algorithmic Studies in Mass Storage Systems Computer Sciences Press, 1983.
- [13] Robert English and Stephnov Alexander, Loge: A Self-Organizing Disk Controller, Proceedings of the Winter 1992 USENIX Conference, 1992.
- [14] David Musser, Block Shuffling in Loge, HP Technical Report CSP-91-18, 1991.
- [15] P. Vongsathorn and S. D. Carson, A System for Adaptive Disk Rearrangement, Software Practice Experience, 20(3): 225?242, 1990.
- [16] M. Aboutabl and A. Agrawala and J. Decotignie, Temporally Determinate Disk Access: An Experimental Approach, Technical Report, CS-TR-3752, 1997.
- [17] Zoran Dimitrijevic et al., Diskbench: User-level Disk Feature Extraction Tool, Technical Report, UCSB, 2004.
- [18] J. Schindler and G. Ganger, Automated Disk Drive Characterization, Technical Report CMU-CS-99-176, 1999.
- [19] N. Talagala and R. Arpaci-Dusseau and D. Patterson, Microbenchmark-based Extraction of Local and Global Disk Characteristics, Technical Report CSD-99-1063, 1999.