B-018

仮想化環境における下位キャッシュの局所性を高めるファイルシステム
A Caching Filesystem for Increasing Locality in the Second Cache in Virtualized Environment

吉田 光太朗† 山口 実靖†
Kotaro Yoshida Saneyasu Yamaguchi

1. はじめに

クラウド環境の普及に伴い，仮想化環境の重要性が高まっている．多くの実装で仮想化環境において物理 HDD へのアクセスは二重のページキャッシュを介して行われる．一つはゲスト OS ページキャッシュ，もう一つはホスト OS ページキャッシュである．前者は上位キャッシュであり，後者は下位キャッシュである．

これらのキャッシュにおいて，多くの場合キャッシュ置換アルゴリズムとして LRU が適用されている．この状況では，ホスト OS のページキャッシュへのアクセスは負の参照の時間的局所性[1]を持ち，ホスト OS のキャッシュヒット率はこの局所性により著しく低下する．

本稿では，ホスト OS キャッシュのヒット率の向上のための新しいファイルシステムを提案する．提案ファイルシステムはゲスト OS で動作し，ページキャッシュの変わりにメモリを管理する．本ファイルシステムでは，一部のデータへのアクセスを常にキャッシュヒットとし，他のデータへのアクセスは常にキャッシュミスとする．これにより，ホスト OS ページキャッシュへのアクセスの局所性が高まり，ホスト OS のキャッシュヒット率が向上することが期待できる．

2. 負の参照の時間的局所性

本章では，参照の局所性と負の参照の局所性について紹介する．

多くの場合，アプリケーションからのアクセスには参照の時間的局所性があり，最近参照されたデータは近い将来再び参照される可能性が高い．LRU などの多くのキャッシュ置換アルゴリズムは，この局所性に基づいている．これにより，キャッシュは低速ストレージのサイズよりはるかに小さいサイズでも効果的に機能することができる．

しかし，仮想化環境などの二重キャッシュ環境では，参照に逆向きの時間的局所性があり，一般的な置換アルゴリズムがうまく機能しないと予想される．ホスト OS ページキャッシュへのアクセスは，図 1 のようなゲスト OS キャッシュを介して実行される．最近参照されたデータへの要求は，ゲスト OS ページキャッシュで処理され，ホスト OS ページキャッシュには送信されない．したがって，ホスト OS ページキャッシュへのアクセスは，最近参照されたデータが近い将来再び参照されることがないという負の参照の時間的局所性を持つこととなる．この負の参照の局所性は，アプリケーションのデータサイズと上位キャッシュのサイズが増加するにつれて増加する[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

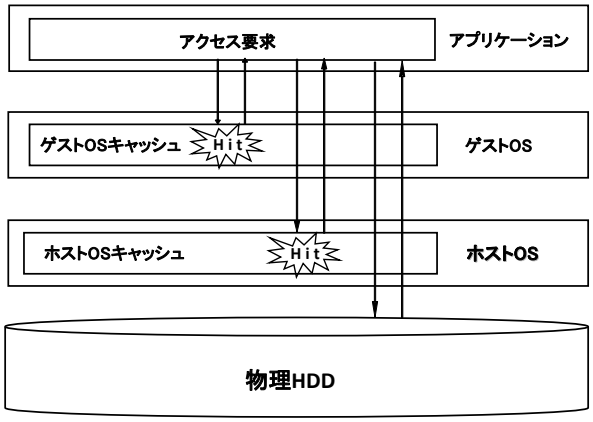

図 1 二重キャッシュ環境

3. 局所性を高めるファイルシステム

本章では，頻繁にアクセスされるデータが既知であると仮定して，下位キャッシュへの参照の局所性を向上させるファイルシステムを提案する．ゲスト OS に本ファイルシステムを適用することで，ホスト OS ページキャッシュのヒット率を向上させることができる．

提案ファイルシステムの動作を図 2 に示す．左が通常手法，右が提案手法を示している．通常の仮想化環境と同様に，これらのケースはゲスト OS とホスト OS のページキャッシュで構成されている．

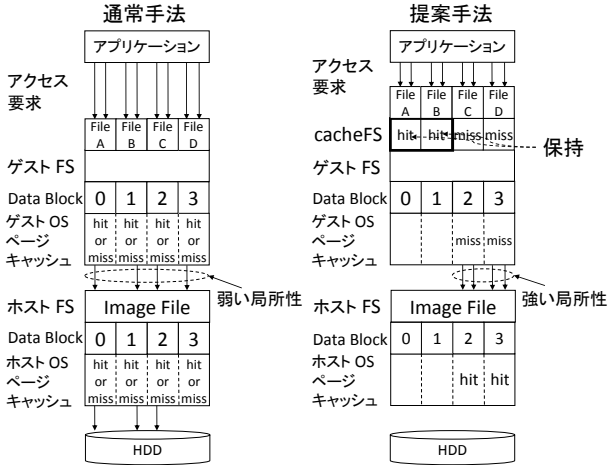

図 2 提案手法

始めに通常手法の動作を説明する．通常手法の場合，ゲスト OS ページキャッシュは LRU で管理され，キャッシュに格納されるデータは LRU に基づき動的に変化する．そのため，ゲスト OS のあるデータブロックへのアクセス要求はキャッシュヒットになる場合とキャッシュミスとなる場

合の両方がありえる．キャッシュミスしたアクセス要求はホスト OS 内のデータブロックのアクセス要求となるため，全てのブロックにアクセス要求が生じる可能性がある．すなわち，ホスト OS 内のブロックへの参照の局所性は弱く，アクセスは多様性が高くなる．

一方，提案手法は以下のように動作する．提案されたファイルシステムはメモリを保持し，ページキャッシュの代わりにデータキャッシュ機能を提供する．本ファイルシステムのキャッシュは常に一定のブロックを保持し，格納されたブロックは動的に置き換えられない．したがって，これらのブロックへのアクセスは，常にキャッシュヒットとなる．同様に，これら以外のブロックへのアクセスは常にキャッシュミスとなる．これによりホスト OS のデータブロックへの参照の局所性は強く，多様性は低くなる．以上のことから，ホスト OS のキャッシュヒット率の向上が期待できる．

提案手法は，頻繁に参照されるデータが既知であり，OS のブロック層のページキャッシュの代わりにメインメモリを管理しキャッシュ機能を提供することを前提としている．本稿では，提案ファイルシステムは FUSE を用いて実装した．

4. 評価実験

本章にて，提案手法の性能を評価する．本章では，ゲスト OS のページキャッシュとホスト OS のページキャッシュをそれぞれゲストキャッシュおよびホストキャッシュと呼ぶ．

評価は，ファイルシステムのベンチマークである Flexible File System Benchmark(FFSB)[2]を使用して行う．FFSB の設定は表 1 の通りである．アクセスされるデータのサイズは 4GB～15GB と 1GB 刻みで変更した．ベンチマークでは，ランダムに選択されたファイルのランダムに選択された場所への 1MB の読み込み要求の発行が繰り返される．ランダム選択の分布は一様分布である．

実験環境の仕様は表 2 および表 3 の通りである．提案ファイルシステムには 2.7GB のメモリを割り当てた．

表 1 FFSB の設定

Data Size	4GB to 15GB
File Size	1GB
Num. File	4 to 15
OP Size	1MB
read/write weight	read 100 %, write 0 %

表 2 実計算機の仕様

OS	CentOS 6.5
HDD	1TB
CPU	Intel Core i3
Kernel	Linux 2.6.32.27
Memory	6GB
仮想化システム	KVM
使用ファイルシステム	ext2

表 3 仮想計算機の仕様

OS	CentOS 6.7
HDD	50GB
CPU	QEMU Virtual CPU
Kernel	Linux 2.6.32.27
Memory	3GB
使用ファイルシステム	ext2

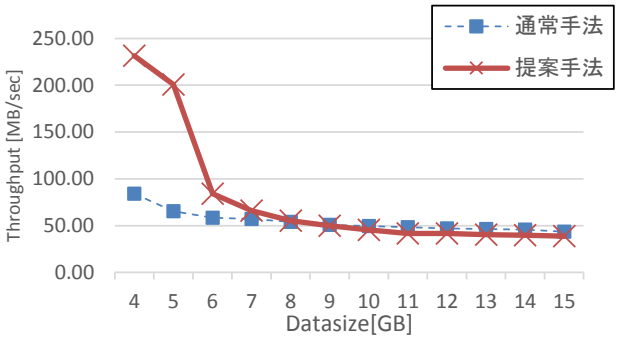

図 3　スループット

実験結果を図 3 に示す．図 3 の縦軸は，FFSB のスループットである．図 3 より，提案手法の性能は通常手法の性能よりも著しく高いか同等であることが分かる．データサイズが 4GB の場合，提案手法により性能が大幅に改善されている．一方，データサイズ 8GB の場合，性能は改善されてはいない．これは，データサイズがキャッシュサイズよりも大幅に大きいため，キャッシュが性能に大きな影響を与えないためである．

5. おわりに

本稿では，負の参照の時間的局所性を紹介した．そして，ホスト OS ページキャッシュのヒット率を向上させる新しいファイルシステムを提案した．ファイルシステムは，特定のデータを保持し，ホスト OS ページキャッシュへの参照の局所性を高める．ベンチマークにより提案ファイルシステムの I/O 評価を行い，提案ファイルシステムは仮想化環境において I/O 性能を向上させることを確認した．

謝辞

本研究は，JST，CREST JPMJCR1503 の支援を受けたものである．

本研究は，JSPS 科研，26730040，15H02696，17K00109 の助成を受けたものである．

参考文献

[1] Yusuke, N. and Saneyasu, Y. 2011. A Server Cache Size Aware Cache Replacement Algorithm for Block Level Network Storage. In Proceedings of the 10th International Symposium on Autonomous Decentralized Systems (ISADS 2011) (Jun. 2011), 573-576

[2] Flexible File System Benchmark https://sourceforge. net/projects/ffsb/