C-010

上位キャッシュの破棄ブロック情報を用いた下位キャッシュ管理

Host OS Cache Managing based on Dropped Blocks in Guest OS Cache

杉本 洋輝†　　山口 実靖†
Hiriki Sugimoto　Saneyasu Yamaguchi

1. はじめに

近年，クラウドコンピューティングの普及に伴い仮想化環境の重要性が高まっている．仮想化環境下において，ホスト OS キャッシュ(下位キャッシュ)へのアクセスはゲスト OS キャッシュ(上位キャッシュ)を介して行われる．このような二重キャッシュ環境下において，上位キャッシュの置換アルゴリズムに LRU が使用されている場合，下位キャッシュにおいては一度アクセスされたデータが近い将来に再度アクセスされる可能性が低くなり，通常とは逆向きの負の参照の時間的局所性が存在することが実 OS で確認されている[1]．また，二重キャッシュ環境ではゲスト OS キャッシュとホスト OS キャッシュに同一のデータを重複して格納している可能性が高い．

ホスト OS キャッシュは負の参照の時間的局所性やゲスト OS キャッシュとのデータの重複により効果的に機能しない．そのため，負の参照の時間的局所性とデータの重複を考慮した置換手法が必要であると考えられる．

また，近年普及が進んでいるクラウドコンピューティング環境では，物理マシン上に複数の仮想マシンを稼働させて契約者に提供することがある．このような例では，資源提供者側が自由に仮想マシンの資源を増減させることができず，仮想マシンの数が少なく資源に余裕のある物理マシンではメモリ資源をホスト OS が管理した状態のまま性能向上を図ることが重要となる．

上位キャッシュが破棄したページは下位キャッシュと重複しないため，上位キャッシュが破棄したページを下位キャッシュが優先的に保持することで読み込み性能の向上が可能であると期待できる．本稿では仮想化環境のゲスト OS のページキャッシュが破棄したページをホスト OS キャッシュで優先的に保持する手法について考察する．

2. 負の参照の時間的局所性

アプリケーションが発行するデータアクセス要求には多くの場合偏りがあり，LRU などのキャッシュ管理技法の多くはこの参照の局所性の概念に基づいている．多くの場合，参照の時間的局所性を考慮したメモリ置換手法を用いることにより下層の低速記憶装置の記憶容量よりも小さいキャッシュでも十分な性能を発揮することができる．

しかし，仮想化環境のような二重キャッシュ環境では通常とは逆向きの負の参照の時間的局所性が存在する．ホスト OS キャッシュへのアクセスは，図 1 のようにゲスト OS キャッシュを介して行われる．アプリケーションから発行された要求がゲスト OS キャッシュ，ホスト OS キャッシュの両キャッシュでミスヒットした場合，

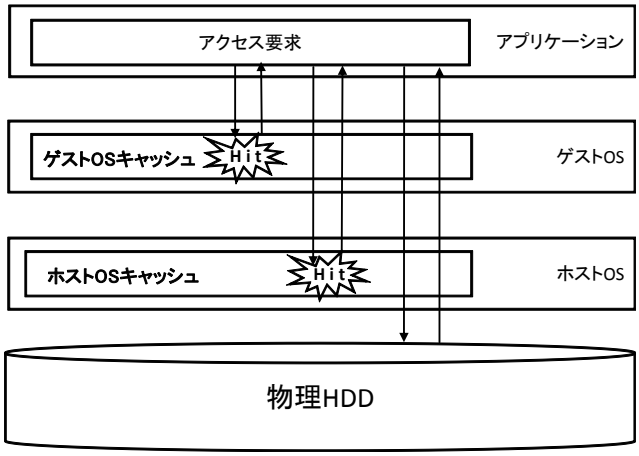

図 1　二重キャッシュ構造

両キャッシュには同一のデータが格納される．しかし，最近アクセスされたデータへのアクセス要求はゲスト OS キャッシュ上で処理され，ホスト OS キャッシュに届くことはない．従ってホスト OS キャッシュでは“最近アクセスされたデータは近い将来再度アクセスされる可能性が低い”という通常とは逆向きの負の参照の時間的局所性が存在する．

また，負の参照の時間的局所性は**ベンチマークのデータサイズが十分に大きい時**，上位キャッシュのサイズが大きいほど影響が大きくなることが確認されている[1].

3. 上位キャッシュ破棄ブロック情報を用いた下位キャッシュページの優先保持

本章で，上位キャッシュ破棄ブロック情報を用いた下位キャッシュページの優先保持手法を提案する．

仮想化環境のような二重キャッシュ環境では 2 章で説明した負の参照の時間的局所性と両キャッシュのデータの重複によりホスト OS キャッシュは効果的に機能しない．

そこで，上位キャッシュの破棄ブロックを監視し，ホスト OS において当該ブロックを優先的に保持する手法を提案する．

提案手法を評価するために，Linux と KVM を用いてユーザ空間で上記処理を行う試作システムを実装した．試作実装ではアクセス対象ファイルが既知であると仮定し，ゲスト OS 内にて `mincore` 関数を用いてページキャッシュに格納されているブロックを常時監視する．ゲスト OS キャッシュでページが破棄されると破棄ページの情報はホスト OS に送信される．ゲスト OS キャッシュで破棄されたページを通達されたホスト OS は，当該ページがホスト OS ページキャッシュに格納されている場合は当該ページに対して読み込みを行う．これによりホスト OS キャッシュの LRU にて最も破棄されづらいページとなる．

†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

4. 評価実験

本章において，提案手法の評価実験を行う．ゲスト OS 上に 20GB のファイルを作成し当該ファイルの先頭から 16GB 目までに対してランダムリードを行うベンチマークプログラムを実行し，提案手法を適用した場合と提案手法を適用しない場合でホスト OS ページキャッシュのヒット率を比較した．ランダムアクセスは一様分布乱数と指数分布乱数で行った．表 1, 2, 3 に実験環境を，図 2, 3 に実験結果を示す．

表 1 実計算機の仕様

OS	CentOS 6.3 (64bit)
Kernel	Linux 2.6.32.57
CPU	Intel Celeron(R) CPU G530
HDD	250GB
Memory	16GB
仮想化システム	KVM
ファイルシステム	ext2

表 2 仮想計算機の仕様

OS	CentOS 6.3 (64bit)
Kernel	Linux 2.6.32.57
CPU	Intel Celeron(R) CPU G530
HDD	50GB
Memory	8GB,10GB
使用ファイルシステム	ext2

表 3 ベンチマークプログラムの仕様

データサイズ	16GB
総読み込み量	64GB
Read size	16MB
アクセスアドレスの偏り	一様分布，指数分布

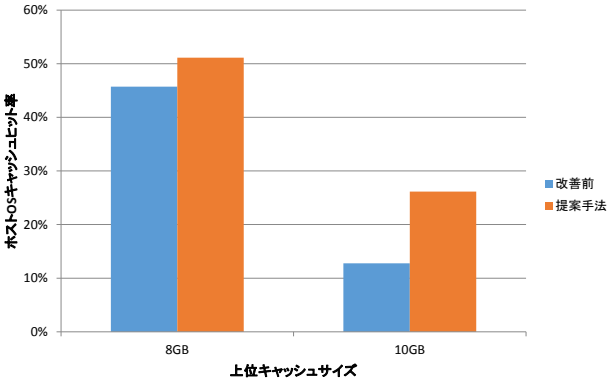

図 2 指数分布時のホスト OS キャッシュヒット率

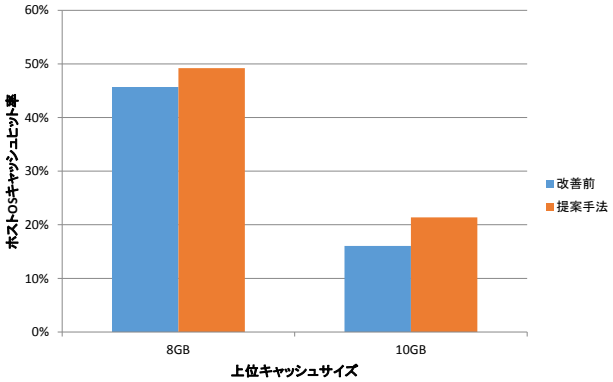

図 3 一様分布時のホスト OS キャッシュヒット率

図 2 の読み込みの偏りが指数分布の場合を見ると，上位キャッシュに 8GB,10GB 割り当てた時いずれも提案手法のホスト OS キャッシュヒット率が改善前に比べて向上していることが確認できる．

図 3 の読み込みの偏りが一様分布の場合も同様に上位キャッシュに 8GB,10GB 割り当てた時いずれも提案手法のホスト OS キャッシュヒット率が改善前に比べて向上していることが確認できる．

提案手法を適用した結果ホスト OS キャッシュヒット率が向上した理由としては，上位キャッシュで破棄されたデータをホスト OS キャッシュにおいて優先的に保持することにより，両キャッシュのデータの重複を削減することができたためと考えられる．

5. まとめ

本稿では，上位キャッシュの破棄ブロック情報を用いた下位キャッシュ管理手法を提案し，ユーザ空間でキャッシュの監視と制御を行う試作実装を作成しキャッシュヒット率の評価を行った．評価の結果，提案手法はホスト OS ページキャッシュのヒット率を向上できることが確認された．

今後は，提案手法をホスト OS とゲスト OS のカーネル内に実装し I/O 処理速度の評価を行っていく予定である．

謝辞

本研究は JSPS 科研費 24300034, 25280022, 26730040 の助成を受けたものである。

参考文献

[1] 竹内洸祐，山口実靖，“複数サーバ接続ネットワークストレージ環境での参照の局所性の解析“第 24 回 コンピュータシステム・シンポジウム (ComSys 2012)