B-010

仮想化環境下におけるホスト OS キャッシュの参照の時間的局所性の解析

An Analysis of Locality of Reference in Virtualized Environment

竹内 洸祐† 山口 実靖†
Kosuke Takeuchi Saneyasu Yamaguchi

1. はじめに

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．このような多段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と予想される．これに関連する既存の研究として，長廻らのシミュレーションによる解析[1]と Zhou らによる DBMS や DISK I/O を用いた解析[2]がある．しかし，実システムの動作を十分に考慮しているとは言えず，また，仮想化環境下における調査はまだ無く，実 OS 上で実アプリケーションを動作させての解析が必要であると考えられる．

本研究では，ゲスト OS 上でアプリケーションを動作させ，ホスト OS キャッシュへのアクセスパターンの解析を行い，負の参照の時間的局所性の存在の調査を行う．また，ホスト OS キャッシュにおける各キャッシュ置換アルゴリズムのヒット率の調査を行う．

2．負の参照の時間的局所性

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．ゲスト OS キャッシュ置換手法には多くの場合 LRU が用いられており，この場合最近参照されたデータはゲスト OS キャッシュに格納される．このため,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し,従来の参照の時間的局所性を期待している LRU キャッシュ置換手法は効果的に機能しない．

図 2 はブロックレベルアクセスのシミュレーションにより得た上位キャッシュと下位キャッシュにおける再アクセス時間と再アクセス確率の関係である[1]．再アクセス時間が 0[アクセス]～約 1000[アクセス]のときは下位キャッシュにアクセス要求が来ず,下位キャッシュにおける再アクセス確率が 0[%]になっている．これより，当該シミュレーション環境においては負の参照の局所性が存在することがわかる．しかし，本シミュレーションは実環境で用いられるファイルシステムなどの振る舞いが考慮されておらず，実際の OS を実計算機上で稼働させ，その上で実アプリケーションを動作させた実環境における考察が必要であると考えられる．

†工学院大学大学院電気・電子工学専攻

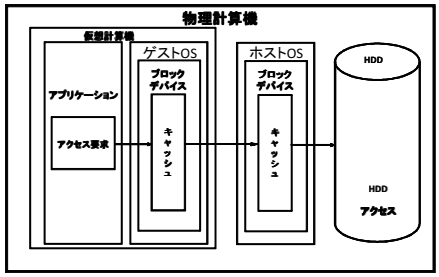

図 1 .二重キャッシュ構造

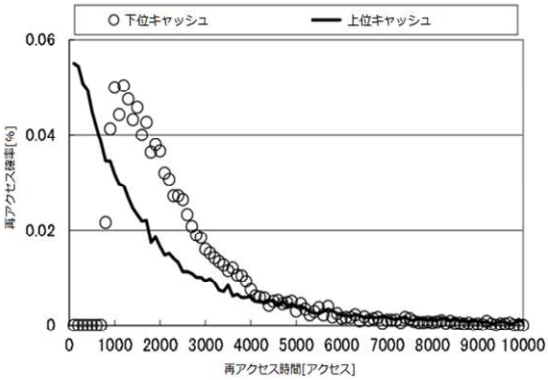

図 2 .負の参照の時間的局所性

3．ホスト OS キャッシュにおける参照の時間的局所性の解析方法

ゲスト OS 上でベンチマークソフトである FFSB (Flexible File System Benchmark)を実行し，ホスト OS キャッシュへのアクセス要求をモニタリングすることにより，負の時間的局所性の調査を行った．ホスト OS，ゲスト OS 共に OS は Linux2.6.18.8(CentOS5.5) を使用した．物理メモリは 4.0GB 搭載されており，ホスト OS に 2.0[GB]，ゲスト OS に 2.0[GB]割り当てた．また，両 OS ともに 2.0[GB]のうち約 1.8 [GB]がキャッシュに使用された．モニタリングはゲスト OS のブロックデバイス層にて行い，ファイルシステムは ext3 を用いた． なお，仮想化システムは Xen 3.4.3 を使用した．

4．解析結果

各ブロックにおける再アクセス間隔とアクセス発生確率(確率密度関数)の関係を図 3 に示す．図 3 は FFSB を用いた測定結果であり，総データサイズ 16[GB]，オペレーションサイズ 16[KB]，1MB[MB]の測定結果である．図 3 より再アクセス間隔 1.8[GB]以下の再アクセス確率が極めて低く負の参照の時間的局所性が存在することが確認された．

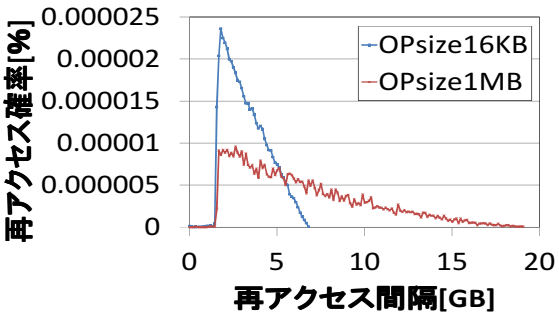

図 3. FFSB（OPsize16KB, 1MB） 測定結果

5．キャッシュ置換アルゴリズムのヒット率評価

本章にて，ホスト OS キャッシュにおける既存アルゴリの実験で得られた実アクセスログ(FFSB, DataSize=16GB, OPsize=16KB, 1MB)を用い，キャッシュヒット率の計算はシミュレーションにより行った．置換手法としては LRU, RANDOM，FIX，MQ を用いた．FIX はキャッシュに格納されたデータを置換せず保持し続ける手法である．RANDOM は，キャッシュから無作為に選択したデータを置換対象とする置換手法である．MQ は多段キャッシュ環境に適したアルゴリズムとされ，長時間データを保持することにより，キャッシュヒット率を高めることを目的としたアルゴリズムである[2].

ホスト OS キャッシュのヒット率を図 4，5 に示す．横軸は，ゲスト OS キャッシュサイズとホスト OS キャッシュの比を表しており，“2.0 倍”は，ホスト OS キャッシュのサイズがゲスト OS キャッシュの 2.0 倍であることを示している．図 4，5 の全ての場合において LRU および MQ のヒット率が極めて低く，多段キャッシュ環境においてこれらの手法が効果的に機能しないことが分かる．RANDOM は LRU や MQ よりは性能が高いが FIX よりも低く，これも効果的な手法とはなっていない．

ゲスト OS キャッシュのサイズと，ホスト OS キャッシュのサイズの比率について注目すると，ホスト OS キャッシュのサイズが小さいほど LRU と MQ の性能が悪くなっていることがわかる．これは，ゲスト OS キャッシュのサイズがホスト OS キャッシュのサイズと比較して相対的に大きくなるほど，ホスト OS キャッシュのデータがゲスト OS キャッシュに包含される確率が増え，負の参照の時間的局所性が強まっているからだと言える．

MQ は，多段キャッシュ環境用に提案されたキャッシュ置換アルゴリズムであるが，今回の評価では LRU とほぼ同等の性能となり，非常に低い性能となった．MQ は参照回数によって，データをどの LRU キューに保存するか決定する手法であり，標準的決定方法として log2(参照回数)によりキューを選択する方法が示されており，標準的なキュー数として 8 本が示されている[2]．本評価もこの標準に従って行ったが，この場合は参照回数が 128 回以上になると，全てのデータが最上位の LRU キューに格納されてしまい，実質的に MQ の動作が LRU の動作とほぼ同一になってしまうことになる．これが MQ の性能が低くなった原因と考えられる．このことから，MQ を効果的に使用するにはパラメータを環境毎に適切に変更する必要があり，最適化を行えないと性能が極めて悪くなってしまうことがわかる．

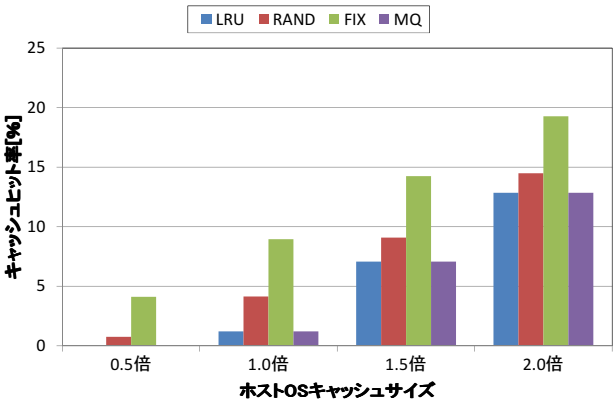

図 4. シミュレーション結果(OPsize16KB)

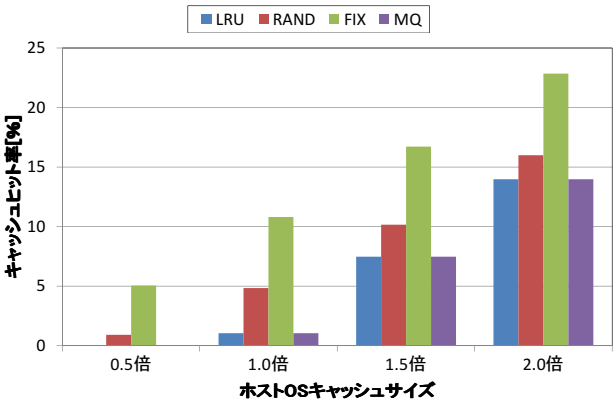

図 5. シミュレーション結果(OPsize1MB)

6．おわりに

本稿では，仮想化環境下における負の参照の時間的局所性の調査を行った．その結果，負の参照の時間的局所性が存在することを確認した．また，既存アルゴリズムの有効性を確認するために，当該調査で得られたアクセスログを使用してキャッシュヒット率の調査を行った．その結果，LRU と MQ の性能は低く，仮想化環境において，LRU や MQ が効果的に機能しないことが分かった．

今後は負の参照の時間的局所性を考慮したキャッシュ置換アルゴリズムの提案を行っていく予定である．

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] Yusuke Nagasako and Saneyasu Yamaguch, “A Server Cache Size Aware Cache Replacement Algorithm for Block Level Network Storage,” In Proceedings of The 4th Ad Hoc, Sensor and P2P Networks Workshop (AHSP) 2011.

[2] Yuanyuan Zhou and James F. Philbin, Kai Li, “The Multi-Queue Replacement Algorithm for Second Level Buffer Caches,” In Procedings of the 2001 USENIX Annual Technical Conference, 2001.