事前調査に基づくキャッシュヒット率推定による動的 VM メモリ割り当て

藤島永太[1]　　山口実靖[1]

概要：近年，サーバの消費電力増加や設置スペース肥大化が問題となっており，その解決策の一つとして仮想化技術を用いて複数の仮想マシンを一台の物理マシンに集約する手法がある．仮想化環境では，仮想マシンを停止させることなくメモリの割り当て量を変更することが可能である．一つの物理マシンにて複数の仮想マシンを稼働させ，それぞれの仮想マシンの負荷が均一でない場合，各仮想マシンに対して均等なメモリ割り当てを行うとメモリを効果的に活用できないことになる．不均一な負荷に対応するためには，各仮想マシンの負荷に応じてメモリ割り当て量を変更する必要がある．我々はこれまでに，読込みのみのアプリケーションのみが動作する状況にて適切にメモリ割り当てを行える手法や，特定の比率の読書混在のアプリケーションのみが動作する状況にて適切に割り当てを行える手法を提案し，有効性を確認してきた．しかし，両手法とも仮定した状況においてのみ適切に動作し，仮定を満たさない状況では効果的に機能しない．本稿ではアプリケーションに前提をもうけない手法として，ゲスト OS のアプリケーションのメモリ量とキャッシュヒット率の関係を調査し，キャッシュヒット率推定によりメモリ割り当て量の最適化を行う手法を提案する．そして，性能評価によりその有効性を示す．

Dynamic Memory Allocation based on Cache Hit Ratio Estimation

EITA FUJISHIMA[1]　SANEYASU YAMAGUCHI[1]

1. はじめに

データセンタ等におけるサーバの消費電力の増加や設置スペースの肥大化が問題となっている．この問題に対する解決策の一つとして，仮想化技術を用いて複数のサーバ OS を一台の物理マシンに集約する手法がある．

仮想化環境では，仮想マシン(VM)を停止させることなく割り当てメモリ量を変更することが可能である．一つの物理マシン上に複数の VM が稼働しており，それぞれの VM の負荷が均一でない場合，均等なメモリ割り当てを行うとメモリを効果的に活用できないことになる．不均一な負荷に対応するためには，仮想マシンの負荷に応じて VM メモリ割り当て量を変更する必要がある．

仮想化ソフトウェアの Xen には，VM の負荷に応じて VM メモリ割り当て量を変更させる機能として xenballoon がある．しかし，xenballoon がメモリ割り当てを行うために考慮する情報は，VM 内のプロセスが使用するメモリ量のみであり，ページキャッシュ(以下，単に“キャッシュ”と記す)として利用されるメモリの量を考慮していない．よって xenballoon では適切に I/O 性能向上を行えないと考えられる．

我々は，すべての VM にて読込みのみを行うアプリケーションが動作する環境においては，キャッシュヒット率が高い(100%に近い)VM に対してより多くのメモリを割り当てることが効果的であるとし，この方針により性能向上を行う手法を提案した[6]．また，すべての VM にて読込み書込み混在のアプリケーションのみアプリケーションが動作する環境では，100%でない特定の値(文献[2]の例にて 80%)を目標ヒット率とし，そのヒット率に近い VM により多くのメモリを割り当てることが効果的であるとし，この方針に従い VM メモリ割り当て量の調整を行う手法の提案を行った[7]．そして，読込みのみを行う FFSB や，読込み書込み混在の DBMS(TPC-C)などのアプリケーションを用いて性能評価を行い，キャッシュヒット率 100%や 80%を目標にする手法の有効性を確認した[1][2]．しかし，これらの手法は読込みのみのアプリケーションや読込み書込み混在のアプリケーションのみが全 VM で均一に動作している場合にのみ有効であり，各 VM にて特性(読込み書込み比率)の異なるアプリケーションが動作する状況に適応できる手法の考察が重要となっている．

本稿では，各 VM にて読込み書込み比率が異なるアプリケーションが並列に動作している状況を想定し，Xen を用いた仮想化環境における，メモリ量とキャッシュヒット率の関係性について調査を行う．その結果からメモリ量とヒット率の近似式を求め，近似式に基づく動的 VM メモリ割り当て量の最適化の手法の提案を行う．

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

2. xenballoon

xenballoon は，Xen が持つ機能の一つで，VM に割り当てるメモリ量の変更を動的に行う機能である．VM 上でデーモンとして動作し，ゲスト OS 内のプロセスの推定メモリ使用量(Commited_AS)を監視して，ホスト OS への要求メモリ量を調整する[3].

しかし，異なるゲスト OS 間でメモリ量の調整を行う機能が無く，ホスト OS へ物理メモリ以上のメモリ量を要求してしまう事がある．この場合，メモリ量のオーバーコミットによるエラーなどは発生しないが，ゲスト OS へ割り当てたメモリを回収する機能がないため，より効率の良い状況へのメモリの再配分ができない．

3. 既存手法

本章では，既存手法である読込み専用アプリケーションを想定した手法[1]および読み書き混在アプリケーションを想定した手法[2]を紹介する．

3.1 読込み専用アプリケーションを想定した手法

すべての VM で読込みのみを行うアプリケーションが動作している環境において，ゲスト OS のページキャッシュヒット率を監視し，それを元に VM にメモリ割り当てを行う手法[1]がある．

この手法では，各 VM のキャッシュヒット率の値をホスト OS にて集計し，以下の手順により VM メモリ割り当て量を決定する．

i. 全 VM のメモリ割り当て量を 10%減少させ，これを再配分用メモリとする．
ii. 上記の手順 i にて得られた再配分用メモリを，キャッシュヒット率が 100%未満の VM へ各 VM のキャッシュヒット率に比例して配分する．

この手法は読込みのみのアプリケーションが動作する環境を考慮しており，読込み書込み混在のアプリケーションが同時に動作している環境では，適切な割り当てが行えないと考えられる．

3.2 読み書き混在アプリケーションを想定した手法

すべての VM で読込みと書込みの両方を行うアプリケーションが動作している状況を想定し，前節の手法[1]を調整した手法[2]がある．この手法では，目標キャッシュヒット率を 100%でない値に定め(文献[2]の状況にて 80%), 目標キャッシュヒット率未満の VM に各 VM のキャッシュヒット率に比例して配分する．

両手法[1][2]とも動作するアプリケーションに前提をおいており，任意のアプリケーションに適用可能な手法の検討が重要な課題となっている．

4. メモリ量とキャッシュヒット率の関係

本章では，メモリ量とキャッシュヒット率の関係の測定方法と測定結果について述べる．

4.1 キャッシュヒット率の測定方法

図 1 のようにホスト OS およびゲスト OS のカーネル内でキャッシュ前後の I/O 量を監視し，キャッシュヒット率を求めた．

キャッシュ前の I/O 量であるページキャッシュへのアクセス量を測定するためにカーネルの改変を行った．ゲスト OS では，カーネル構成ファイルの mm/filemap.c 内にある関数 find_get_page()の実行回数とブロックサイズの監視，ホスト OS では，drivers/xen/blkback/blkback.c 内にある関数 dispatch_rw_block_io()の実行回数とブロックサイズの監視をするための改変を行った．

キャッシュ後の I/O 量を測定するために VM の仮想ブロックデバイス(XVD)における I/O 量, 物理 HDD における I/O 量を測定する様にカーネルの改変を行った．ゲスト OS では，関数 do_blkif_request()の実行回数とブロックサイズの監視，ホスト OS では，drivers/scsi/sd.c 内にある関数 sd_prep_fn()の実行回数とブロックサイズの監視をするための改変を行った．

キャッシュ後の I/O 量がキャッシュミスした量となり，測定したキャッシュ前後の I/O 量の比よりキャッシュヒット率を求める．

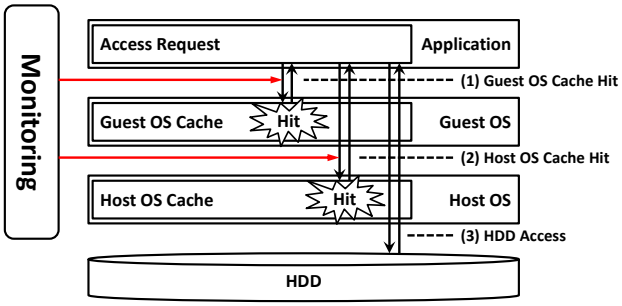


図 1. キャッシュヒット率の解析

4.2 メモリ量とキャッシュヒット率の関係の調査方法

本節にて，メモリ量とキャッシュヒット率の関係の測定方法について述べる．各環境にて，メモリ量とキャッシュヒット率の関係を 30 回測定し，そのうち上位 10%のヒット率および下位 10%のヒット率を計測値から除き，残りを測定結果とした．そして，取り除いた後の測定結果から最小二乗法により 3 次近似した曲線を求め，これを各環境におけるメモリ量とキャッシュヒット率の近似式とした．

調査に用いた物理マシンと仮想マシンの仕様は表 1 から表 4 の通りである．

表 1. 物理マシンの仕様

CPU	AMD Athlon 1640B 2.7 GHz
Memory	8 GB
HDD	1 TB

表 2. ホスト OS の仕様

OS	CentOS 6.3
Kernel	Linux-2.6.32.57
Memory	1,024 MB
仮想化システム	Xen 4.1.2

表 3. 仮想マシンの仕様

CPU	AMD Athlon 1640B 2.7 GHz
Memory	動的に変動
HDD	100 GB (Image file)

表 4. ゲスト OS の仕様

OS	CentOS 5.5
Kernel	Linux-2.6.18.8
Memory	1,000 MB ～ 7,000 MB

4.3 メモリ量とキャッシュヒット率の関係（ベンチマークデータサイズ均一，読込比率不均一）

VM のメモリ量とアプリケーションの読込み書込み比率とキャッシュヒット率の関係を調査するため，1 台の VM 上で FFSB を実行し，各 VM メモリ量におけるキャッシュヒット率を測定した．本実験では FFSB のベンチマークデータサイズは 2 [GB]とし，読込比率は 100，70，50 [%]とし，計測時間は 3,600 秒とした．

測定結果を図 2 に示す．図内の近似式は最小二乗法により求めた 3 次近似式である．図より，キャッシュヒット率が低いと近似曲線の傾きが大きいが，キャッシュヒット率が 100%に近づくと傾きが小さくなることが分かる．

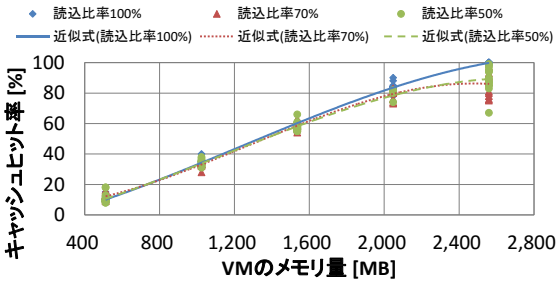


図 2. メモリ量とキャッシュヒット率の関係

4.4 メモリ量とキャッシュヒット率の関係（ベンチマークデータサイズ不均一，読込比率均一）

VM メモリ量とベンチマークデータサイズとキャッシュヒット率の関係を調査するため，3 台の VM 上で FFSB を実行し，キャッシュヒット率を測定した．本測定では FFSB のベンチマークデータサイズは VM1 を 1 [GB]，VM2 を 2 [GB]，VM3 を 3 [GB]とし，読込比率は 100，90，70 [%]から選択し，計測時間は 1,800 秒とした．

測定結果を図 3 から図 5 に示す．図より，キャッシュヒット率が低いと曲線の傾きが大きいが，キャッシュヒット率が 100%に近づくと傾きが小さくなることが分かる．また，ベンチマークデータサイズの小さい VM の方が，曲線の傾きが大きくなるという傾向を確認できる．

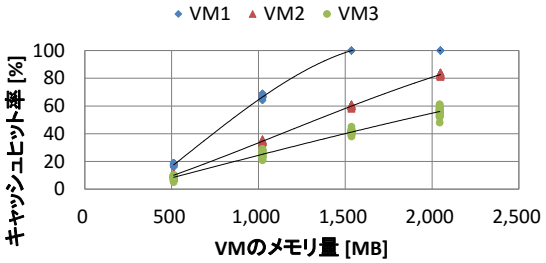


図 3. メモリ量とキャッシュヒット率と読込比率の関係 (読込比率 100%)

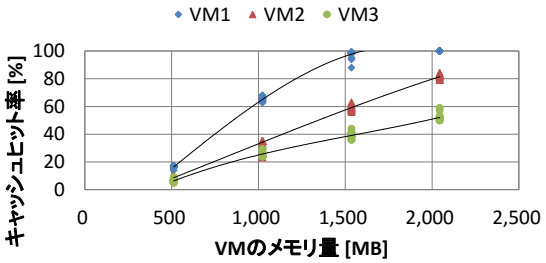


図 4. メモリ量とキャッシュヒット率と読込比率の関係 (読込比率 90%)

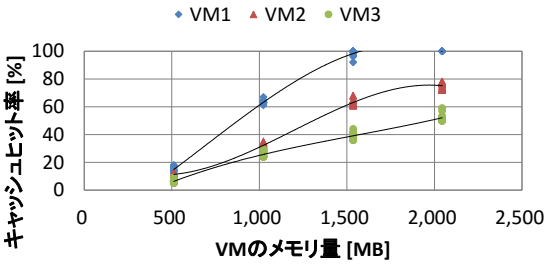


図 5. メモリ量とキャッシュヒット率と読込比率の関係 (読込比率 70%)

4.5 メモリ量とキャッシュヒット率の関係（ベンチマークデータサイズ不均一，読込比率不均一）

様々なベンチマークデータサイズと読込み書込み比率のアプリケーションが混在する環境における VM メモリ量とキャッシュヒット率の関係を調査するため，3 台の VM 上で FFSB を実行し，キャッシュヒット率を測定した．本測定では測定毎に各 VM のベンチマークデータサイズと読込比率を変更し，ベンチマークデータサイズは 1, 2, 3 [GB]から選択し，読込比率は 100，90，70，50 [%]から選択し，計測時間は 1,800 秒とした．

測定結果を図 6 から図 9 に示す. 図名の括弧の中は各 VM のベンチマークデータサイズと読込比率を示している．図より, キャッシュヒット率が低いと曲線の傾きが大きいが,

キャッシュヒット率が 100%に近づくと傾きが小さくなることが分かる．また，読込比率の高い VM の方が曲線の傾きが大きくなりやすいという傾向を確認できる．そして，ベンチマークデータサイズの小さい VM の方が曲線の傾きが大きくなりやすいという傾向を確認できる．

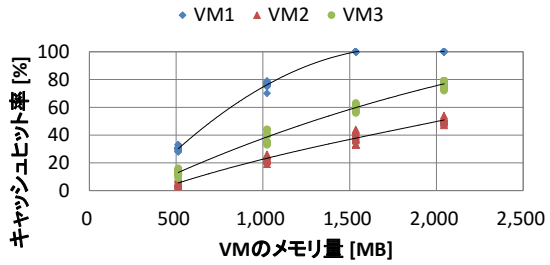


図 6. メモリ量とキャッシュヒット率と読込比率の関係 (VM1: 1GB 100%, VM2: 3GB 50%, VM3: 2GB 50%)

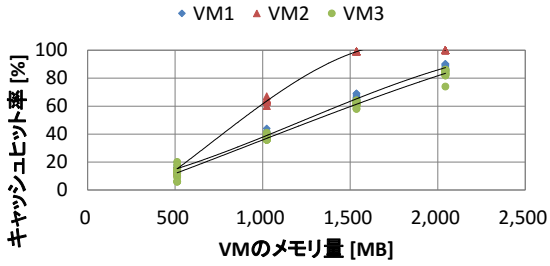


図 7. メモリ量とキャッシュヒット率と読込比率の関係 (VM1: 2GB 90%, VM2: 1GB 90%, VM3: 2GB 70%)

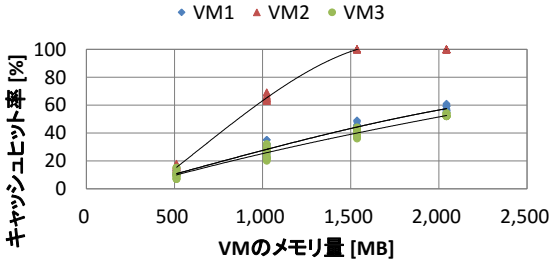


図 8. メモリ量とキャッシュヒット率と読込比率の関係 (VM1: 3GB 50%, VM2: 1GB 100%, VM3: 3GB 50%)

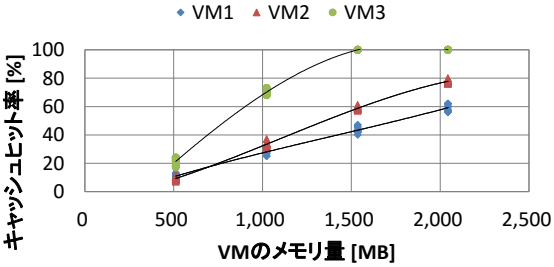


図 9. メモリ量とキャッシュヒット率と読込比率の関係 (VM1: 3GB 70%, VM2: 2GB 100%, VM3: 1GB 100%)

5. 提案手法

本章では，4 章で調査した近似曲線を元に VM にメモリ割り当てを行う手法を提案する．本手法では，以下の手順により VM メモリ割当量を決定する．

i. 全 VM のメモリ割り当て量を 10%減少させ, これを再配分用メモリとする．

ii. 上記の手順 i にて得られた再配分用メモリを，メモリ 1 Byte 投入あたりの$(\text{キャッシュヒット率})^n$の向上が最も大きい VM に与える．キャッシュヒット率は事前調査により得た近似曲線より推定する．

事前調査により得た近似曲線は，本稿の例では図 2 から図 9 が該当する．メモリ 1 Byte 投入あたりの$(\text{キャッシュヒット率})^n$の向上とは，近似曲線を$y = f(x)$としたときのdy^n/dxであり，最も性能向上が期待できる VM に資源を与えることを趣旨としている．多くの場合，性能はヒット率の 1 乗以上の速度で向上するため，nは 1 以上とすることが適切であると考えられる．

6. 性能評価

提案手法の有効性を検証するために性能評価実験を行った．本章にて，性能評価結果について述べる．

6.1 評価方法

実験を行うため，3 台の VM 上で FFSB を実行した．ベンチマークデータサイズは 1，2，3 [GB]から選択し，読込比率は 100，90，70，50 [%]から選択し，測定時間は 1,200 秒とした．実験に用いた物理マシンと仮想マシンの仕様は 4 章と同様である．提案手法におけるnは 2 とした．

6.2 I/O 性能の評価

図 10 から図 12 に，提案手法，メモリを均等に配分した手法(1,685 [MB])，xenballoon 手法，既存手法[1]の I/O 性能を示す．

図から，10 件中 9 件において提案手法が最も高い性能を示していることが分かる．図 10 における読込比率 90:90:70 の測定結果においてのみ，提案手法が最も高い性能となっていないが，最高の性能を示した既存手法の 93%の性能となっており，最高性能とほぼ同等の性能となっている．また 10 件中 3 件において, 性能が 2 番目に高い既存手法と比べて大幅な性能向上を実現していることが分かる(図 10 の読込比率 100:70:50 の測定結果にて 3.88 倍，同図の読込比率 70:70:100 の測定結果にて 8.39 倍，図 11 の読込比率 100:100:100 の測定結果にて 1.54 倍).

また，提案手法は各 VM 上で動作するアプリケーションの読込み書込み比率が不均一である環境を想定しているが，図 11 に着目すると提案手法は同一の読込み書込み比率のアプリケーションが動作している環境においても良い性能を出せることが確認できる．

測定中のメモリ量の推移は付録の図 13 から図 22 に示す．それらの図より，データサイズが小さく読込比率が高く，

メモリ増加によりキャッシュヒット率向上の効果が出やすいVMに対して積極的なメモリ配分を行っていることが確認できる．

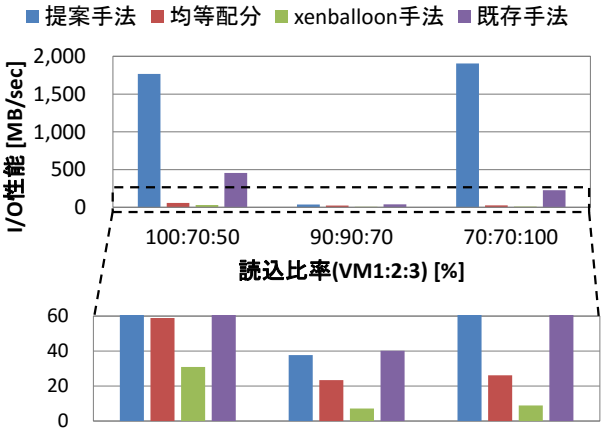


図 10. 性能評価
(ベンチマークデータサイズ均一，読込比率不均一)

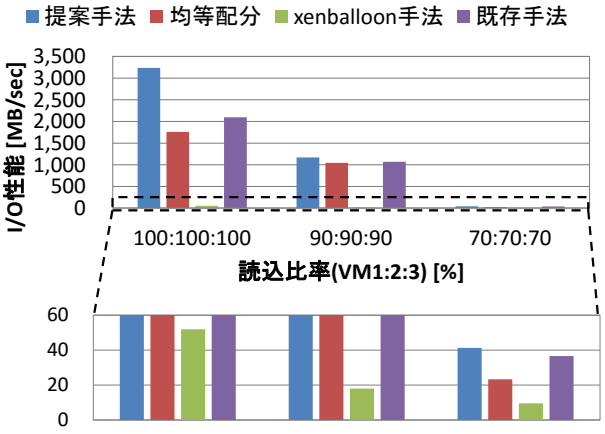


図 11. 性能評価
(ベンチマークデータサイズ不均一，読込比率均一)

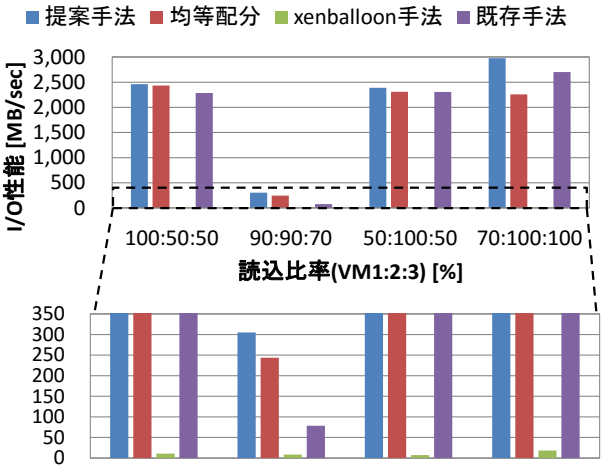


図 12. 性能評価
(ベンチマークデータサイズ不均一，読込比率不均一)

7. 考察

7.1 データサイズや読込み書込み比率が変化するアプリケーション

本提案手法は，事前にメモリ量とキャッシュヒット率の関係を調査し，その調査結果がその後に有効であり続けることを前提にしている．しかし，実際のサービスでは運用に伴いデータサイズが変化し，読込み書込み比率が変化していくことが予想される．この問題は，サービス運用中にも常にメモリ量とキャッシュヒット率の関係を測定し続け，「メモリ量とキャッシュヒット率の関係」の情報を更新し続けていくことにより解決または緩和できると考えられる．

7.2 さらなる性能向上について

キャッシュヒット率が 100%の VM からメモリを奪いキャッシュヒット率が 100%をわずかに下回る状態(99%など)にすると，その VM のアプリケーション性能は著しく低下する．よって，これを回避する工夫をすることによりさらなる性能向上が実現できると期待される．たとえば，キャッシュヒット率 100%の VM からメモリを削減することに対しては消極的にし，n 回連続で削減するべきであるとの判定が出された後に削減するなどが考えらえる．

8. 関連研究

VM のメモリ割り当て量の最適化の研究として，以下のものがある．

Zhao らは，LRU ヒストグラムを取得してキャッシュヒット率を予測し ballooning を行う手法を提案している[4]．しかし，仮想記憶や PTE に注目した手法であり，本稿の様な I/O 性能に関して寄与のある手法とはなっていない．

仮想化環境の I/O に着目し，VM のメモリ割り当てについて考察した研究として，Jones らの提案[5]がある．当該手法ではバッファキャッシュへの追加と削除から必要メモリ量の推定を行っている．しかし，キャッシュヒット率についてのみ考察されており，最終的な性能については議論されていない．また，キャッシュヒット率の推定は必ずしも正確ではなく，実際のヒット率を求めて使用している本稿の提案手法と比較し，正確さにおいて劣っている部分がある．

9. おわりに

本稿では読込み書込み比率が様々なアプリケーションを同時に実行した際のキャッシュヒット率推定による VM メモリ割り当て手法を提案し，評価を行った．評価の結果，提案手法の有効性が確認できた．

今後は，メモリ量とキャッシュヒット率の関係の事前調査をせずに，アプリケーション実行中にメモリ量とキャッシュヒット率の関係を測定しながら，適切なキャッシュヒット率の推定が行える手法を提案し，評価を行っていく予定である．

謝辞 本研究は JSPS 科研費 25280022, 26730040, 15H02696 の助成を受けたものである．

本研究は，JST，CREST の支援を受けたものである．

参考文献

[1] 坂本雅哉, 山口実靖, "複数アプリケーション動作環境におけるキャッシュヒット率を考慮した動的 VM メモリ割り当て", 情報処理学会第 77 回全国大会 (2015).

[2] 坂本雅哉, 山口実靖, "キャッシュヒット率を考慮した動的 VM メモリ割り当てによる DBMS 性能の向上", 電子情報通信学会データ工学研究会第一種研究会「データ工学と食メディア」(2015).

[3] Stephen Spector, “Memory Overcommit”, August 27, 2008, http://blog.xen.org/index.php/2008/08/27/xen-33-feature-memory-overcommit/

[4] Weiming Zhao, Zhenlin Wang, “Dynamic memory balancing for virtual machines,” Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, pp, 21-30. (2009).

[5] Stephen T, Jones Andrea C, Arpaci-Dusseau Remzi H, Arpaci-Dusseau, “Geiger: monitoring the buffer cache in a virtual machine environment,” Proceedings of the 12th international conference on Architectural support for programming languages and operating systems, pp, 14-24. (2006).

[6] 坂本雅哉, 山口実靖, "仮想化環境におけるキャッシュヒット率を考慮した VM メモリ割当", 第 12 回情報科学技術フォーラム FIT2013 RC-009, (2013).

[7] 坂本雅哉, 山口実靖, "ワーキングセットとキャッシュヒット率を考慮した動的 VM メモリ割り当て", 第 14 回情報科学技術フォーラム FIT2015 M-014, (2015).

付録

I/O 性能測定中の VM のメモリ量の推移を図 13 から図 22 に示す．これらの図は，図 13 から図 15 がベンチマークデータサイズ均一かつ読込比率不均一，図 16 から図 18 がベンチマークデータサイズ不均一かつ読込比率均一，図 19 から図 22 がベンチマークデータサイズ不均一かつ読込比率不均一の測定結果を示している．以上の図と事前調査で得られた図 2 から図 9 より，キャッシュヒット率が最も良くなる VM に対して再割当て用メモリを与えていることが確認できる．

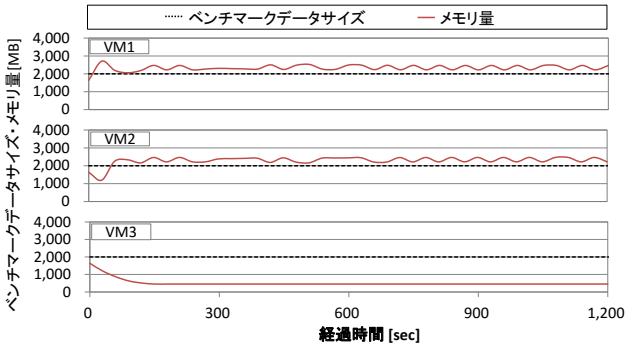


図 13. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 2GB 90%, VM2: 2GB 90%, VM3: 2GB 70%)

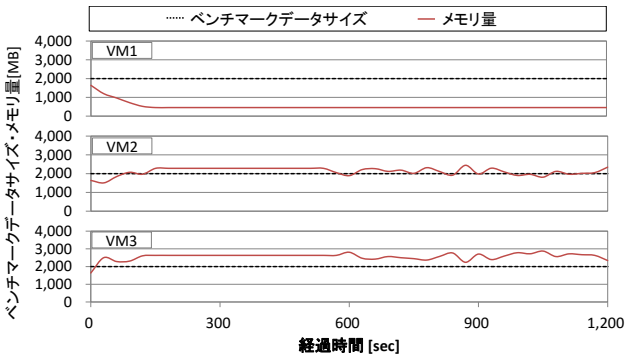


図 14. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 2GB 70%, VM2: 2GB 70%, VM3: 2GB 100%)

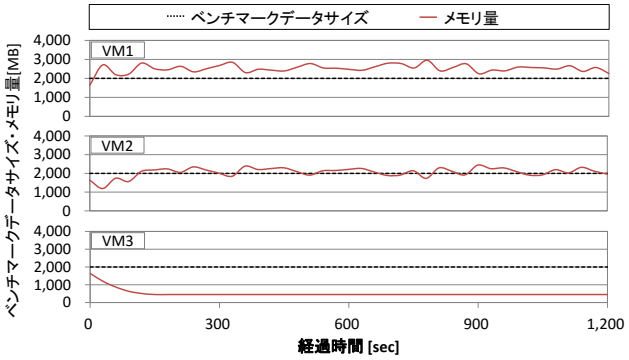


図 15. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 2GB 100%, VM2: 2GB 70%, VM3: 2GB 50%)

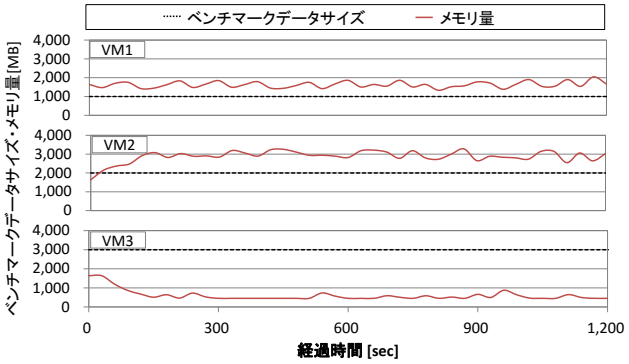


図 16. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 1GB 100%, VM2: 2GB 100%, VM3: 3GB 100%)

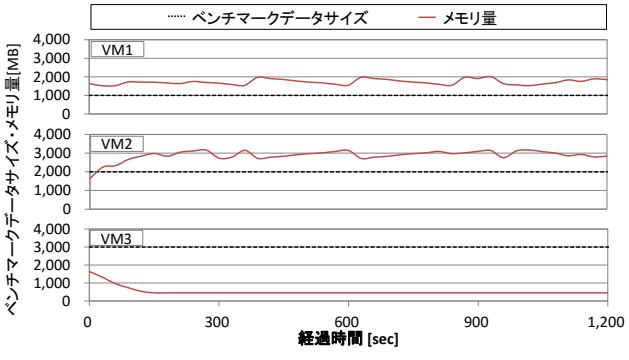


図 17. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 1GB 90%, VM2: 2GB 90%, VM3: 3GB 90%)

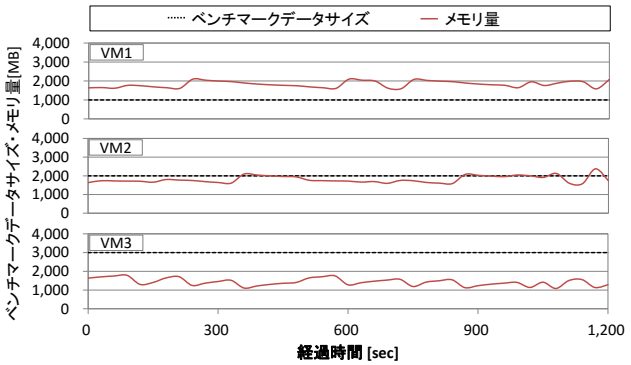


図 18. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 1GB 70%, VM2: 2GB 70%, VM3: 3GB 70%)

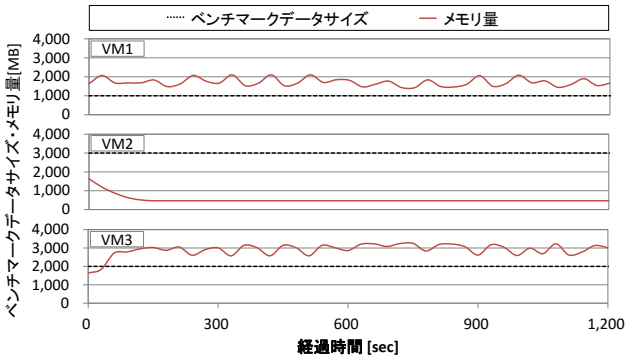


図 19. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 1GB 100%, VM2: 3GB 50%, VM3: 2GB 50%)

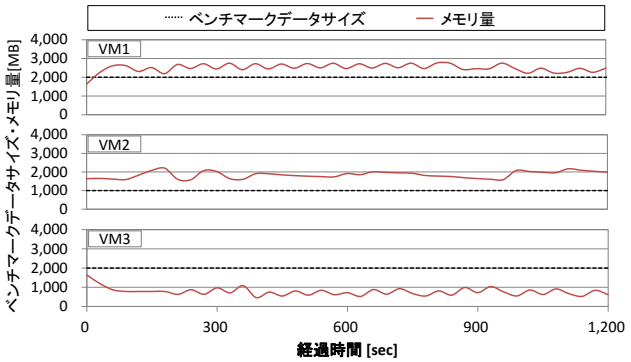


図 20. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 2GB 90%, VM2: 1GB 90%, VM3: 2GB 70%)

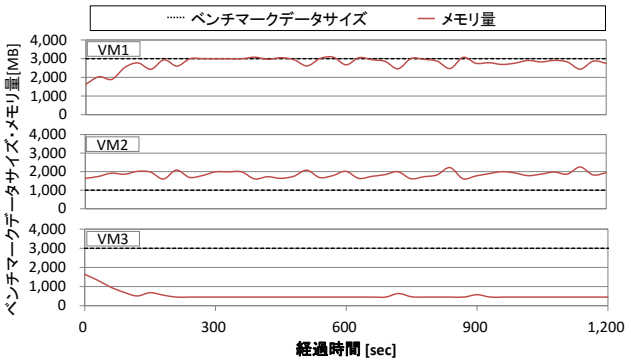


図 21. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 3GB 50%, VM2: 1GB 100%, VM3: 3GB 50%)

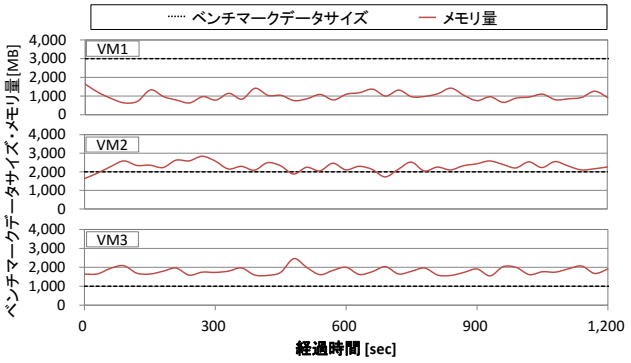


図 22. ベンチマークデータサイズと VM のメモリ量の推移
(VM1: 3GB 70%, VM2: 2GB 100%, VM3: 1GB 100%)