6J-02

キャッシュヒット率向上のための仮想計算機への動的メモリ割り当て手法

青木 雄佐 † 芝 公仁 †
† 龍谷大学

1 はじめに

近年，仮想化技術が広く普及し，様々な分野で仮想計算機が利用されるようになってきている．仮想計算機ごとにメモリ管理が行われるため，限られたメモリを複数の仮想計算機で共有する場合には，仮想計算機へのメモリの配分を適切に行う必要がある．しかし，システムが効率的に動作するのに必要なメモリ量をシステム構築時に正確に予測することは困難であるため，それぞれの仮想計算機に対して適切な量のメモリを割り当てることは難しい．

本稿では，ディスクキャッシュが有効に機能するよう仮想計算機へのメモリ割り当て量を設定する手法と，本手法に基づきメモリ割り当てを行うシステムについて述べる．提案手法では，仮想計算機の動作を監視し，メモリ割り当て量とキャッシュヒット数を用いてキャッシュが有効に機能するメモリ割り当て量を算出し，各計算機へのメモリ割り当てを行う．本手法により，仮想計算機のキャッシュが有効に機能するようにメモリを配分することが可能になる．そのため，メモリのオーバーコミットの状態で仮想計算機を起動し，メモリ負荷が高くなった仮想計算機へとメモリを割り当てて処理時間を短縮させることができる..

2 システム構成

仮想計算機へのメモリ割り当てに関しては，事前に必要なメモリ量を予測することや動的な負荷の変化への対応が難しい．これに対応するため，仮想計算機上で動作しているシステムに対して，ワーキングセットサイズの推測を行う手法が提案されている [1, 2, 3]．これらは，仮想計算機上で動作しているシステムに対して，メモリ割り当て料を変更する，キャッシュ領域を監視するなどを行いワーキングセットサイズを推測する．提案システムでは同様に，仮想計算機のメモリ割り当て料の変更と動作の監視を行い，仮想計算機へ割り当てる適切な量のメモリを推測する．

提案システムは，単一のホスト計算機上で複数のゲスト計算機が動作する環境において考える．ホスト計算機で動作する Linux カーネルは物理メモリの管理を行い，ゲスト計算機にメモリを分配する．このような環境においては，ゲスト計算機間で物理的な計算機資源が共有されているため，限られた資源を有効に機能するよう分配することが重要である．

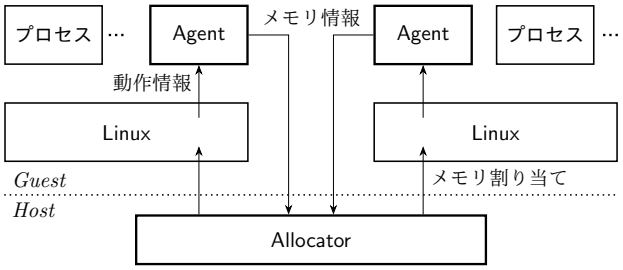

図 1 ゲスト計算機へのメモリ割り当て

各ゲスト計算機ではそれぞれ固有のプロセスが動作し，それらにより各ゲスト計算機で行われるべき処理が実現される．このとき，プロセスは自身が必要とする量のメモリを使用して動作し，通常，ゲスト計算機が持つメモリ量に応じて動作を変えることはない．カーネルやプロセスが必要とする以上のメモリが使用できる場合，当該メモリはカーネルがディスクキャッシュとして使用する．すなわち，ゲスト計算機に新たにメモリが割り当てられたとき，そのメモリはディスクキャッシュとして使用される．逆に，メモリを減らす場合には，ディスクキャッシュを破棄し，そのメモリを解放する．

通常，キャッシュ量が多いほどキャッシュヒット率が高くなるため，多くのメモリをディスクキャッシュに使用できると I/O 性能が向上する．提案システムは，ゲスト計算機のキャッシュの動作をもとに，ゲスト計算機へのメモリ割り当て量の設定を行い，システム全体でのメモリ利用の効率化を図る．

図 1 に示すように，提案システムでは，次の 2 つのコンポーネントが協調して動作し，ゲスト計算機へのメモリ割り当てを行う．Allocator は，ホスト計算機で動作するプロセスである．提案システムの中核として動作し，ゲスト計算機の動作状況をもとに割り当てるべきメモリ量を計算し，ゲスト計算機へのメモリ配分を行う．Agent は，各ゲスト計算機で 1 個ずつ動作するプロセスである．ゲスト計算機で動作する Linux カーネルの動作を監視し，メモリ使用状況を Allocator に通知する．

Dynamic memory allocation for virtual machines to improve cache hit rate
†Yusuke Aoki, †Masahito Shiba
†Ryukoku University

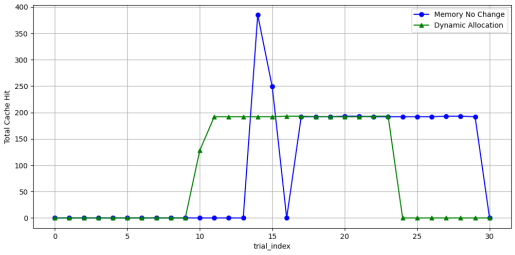

図 2　各試行でのキャッシュヒット数

Allocator は，式 1 を用いて，各ゲスト計算機へ割り当てるべき最適なメモリ量の探索を行う．e が評価値，h がキャッシュヒット数，m がキャッシュミス数，memsize がメモリサイズである．評価値の差が 500 未満の場合，評価値の高い仮想計算機のメモリを減少させ，評価値の低い仮想計算機のメモリを増加させる．増減させる値は 100 MB である．提案システムは，以上の処理を繰り返し，メモリ割り当て量を最適する．

$$e = \frac{h - m}{memsize} \tag{1}$$

各ゲスト計算機上で動作する Agent は，割り当てられたメモリでゲスト計算機がどのように動作するかを観測する．具体的には，eBPF を使用して Linux カーネルの動作を調べ，キャッシュヒット数とキャッシュミス数を取得する．また，これらを Allocator に通知する．

3　評価

提案システムと仮想計算機を立ち上げたときにメモリを割り当てた場合を比較する実験を二つ行った．

本実験では，ファイル読み出し処理を行う負荷プロセスが動作するゲスト計算機 G1 と G2 の 2 台を用意した．負荷プロセスは，G1 では 2GB のファイルに，G2 では 4GB のファイルにランダムアクセスを行う．このとき，乱数は正規表現であり，偏りのあるアクセスが行われる．Allocator は，6GB のメモリをこの 2 台のゲスト計算機に分配する．ただし，各ゲスト計算機には少なくとも 1 GB のメモリが割り当てられる．すなわち，各ゲスト計算機のメモリ量は，1 GB から 5 GB までの間で変化することになる．最初に割り当てて変更しない場合は，G1 に 3 GB，G2 に 3 GB のメモリを割り当てる．(3G3G) グラフの横軸は試行回数であり，縦軸は各仮想計算機の合計キャッシュヒット数である．丸点が本システムを用いて割り当てたとき，三角点が一度割り当てて変更しなかったときである．試行回数を重ねるごとに，提案システムの合計キャッシュヒット数はメモリ量を変更しなかった場合のキャッシュヒット数よりも多くなった．

負荷プロセスが読み出すファイルのサイズを 50 回読み込むと変更するようにして実験した．G1 は 2 GB から 4 GB，G2 は 4 GB から 2 GB に変更するようにした．各試行でのキャッシュヒット数を図 2 に示す．試行回数を重ねるごとに，提案システムの合計キャッシュヒット数はメモリ量を変更しなかった場合のキャッシュヒット数よりも多くなった．G1 は，提案システムを用いた場合と一度メモリを割り当てて変更しない場合両方が 2 GB のファイルが試行回数 11 のときに，4 GB のファイルが試行回数 16 のときに読み終わった．G2 は，提案システムを用いた場合は 2 GB のファイルを試行回数 9 に，3 GB のファイルを試行回数 24 に読み終えた．G2 はそれぞれ試行回数が 11 と 30 であった．

G1 が 2 GB のファイルを読み終わったあと，G2 が 4 GB のファイルを読み終わるまではキャッシュヒット数が提案システムを使っていない場合のほうが多かったが，それ以外は提案システムを使った場合のほうがキャッシュヒット数が多く，実行結果は提案システムを使った場合のほうが速く読み終わった．

提案システムを用いたときは，提案システムを用いてない場合と比べて試行回数 6 回分速く終わった．また，試行回数 24 までに提案システムを用いた時 19 回上回った．

4　おわりに

本稿では，ディスクキャッシュが有効に機能するよう，仮想計算機へのメモリ割り当て量を設定する手法について述べた．本手法では，ゲスト計算機の動作を監視し，メモリ割り当て量とキャッシュヒット数を用いてキャッシュが有効に機能するメモリ割り当て量を算出し，各ゲスト計算機にそのメモリ割り当て量を適用する．本手法によって，各ゲスト計算機のキャッシュが有効に機能するようなメモリの割り当てが実現される．これにより，限られた物理メモリ資源を効率的に活用し，システム全体の性能を向上させることが可能になる．

参考文献

[1] Yamaguchi, S. and Fujishima, E.: Optimized VM memory allocation based on monitored cache hit ratio, Association for Computing Machinery (2016).

[2] Jones, S. T., Arpaci-Dusseau, A. C. and Arpaci-Dusseau, R. H.: Geiger: Monitoring the Buffer Cache in a Virtual Machine Environment, *ASPLOS XII: Proceedings of the 12th international conference on Architectural support for programming languages and operating system*, pp. 14–24 (2006).

[3] Harby, A. A., Fahmy, S. F. and Amin, A. F.: More Accurate Estimation of Working Set Size in Virtual Machines, *IEEE Access*, Vol. 7, pp. 94039–94047 (2019).