ページ共有を考慮したCPUキャッシュパーティショニング機構

滕　　昌民[†]　山田浩史[†]

近年，Amazon EC2 や Google Compute Engine をはじめとする Infrastructure as a Service (IaaS) 方式のクラウドサービスが普及している．IaaS ではハードウェア資源を仮想化し，VM としてユーザーに提供している．ユーザーは仮想ハードウェアのスペックを選べるようになっており，クラウド提供者は課金額に応じて資源を配分するこてができる．その際，同じ物理マシンで複数の VM が動作することもあるが，それらの VM は CPU の L3 キャッシュを共用するため，1 つの VM がキャッシュを多く占拠する処理を行うと，同じマシンで動作する他の VM の性能も低下してしまう．

このような状況を回避する手段として，Intel Cache Allocation Technology (Intel CAT) が挙げられる．Intel CAT は，CPU キャッシュのパーティショニングを支援するハードウェア機構である．CPU の内部にはいくつかの class of service (CLOS) が定義されており，各 CLOS に対して，cache allocation ができる L3 キャッシュの領域を表す capacity bitmask (CBM) を設定できる[3]．特定の CPU コア（Hyper-Threading 有効の場合は CPU スレッド）に対し CLOS を結びつけると，そのコアによる L3 キャッシュの利用は CBM の通りに制限される[6]．これを用いて，VM ごとに課金額などに応じて，適切なサイズの専用キャッシュを割り当てることで，他の VM との L3 キャッシュの競合が避けられる．

一方で，クラウド環境において，メモリを効率的に使用するために，ページ共有が行われることが多い．ページ共有とは，内容の同じ複数の物理ページを 1 つに集約することで，メモリ領域を節約する手法であり，仮想化環境においても，VMware ESX Server[5] や Satori[2] など，ページ共有の実装が多数存在する．

Intel CAT は cache allocation を制限するが，cache lookup に関しては制限しておらず，割り当てられていないキャッシュ領域であっても，キャッシュデータの読み取りは可能である[6]．この性質により，ページが共有されると，そのページに対応するキャッシュデータも共有されることになる．ページを共有したプロセスで，そのキャッシュデータを自分のキャッシュ領域に持たないものにとって，実質使用できるキャッシュ領域が増加したと考えることができる．ところが，既存のページ共有手法は Intel CAT を考慮しておらず，共有ページに対応するキャッシュデータが特定のキャッシュ領域に集中し，不公平性が生じてしまう．

本研究では，この不公平性を解決する手法を提案する．その第一歩としてまず，この不公平性の存在を実験を通して示す．実験マシンでは Xeon E5-1620 CPU が使われており，未改変の Linux 4.13.0 カーネルが動作している．特定の大きさの配列を確保しランダムにアクセスを行い，アクセス時間の平均を測定するテストプログラムが用意されており，ページ共有が適用できるよう，テストプログラムでは配列の半分は実験を通して，中身が変化しないようにしている．配列のサイズを拡大させながら，2 つのテストプログラムを交互に実行させ，Intel CAT を用いて，それぞれに重複のないよう 4MB の L3 キャッシュを割り当てる．ページ共有機構としては，Linux 2.6.32 以降のカーネルに標準実装されている KSM[1] を用いる．

実験結果を見ると，配列の大きさが L1 キャッシュのサイズである 32kB に収まるうちは，キャッシュを利用して高速なアクセスが可能だが，それを超えた瞬間に，L2 キャッシュを参照せざるを得なくなり，アクセス時間が一気に長くなる．同じ理由から，L2 サイズである 256kB でも，アクセス時間が急激に増長している．L3 キャッシュに関しては，ページを共有しない場合，双方のテストプログラムともに，配列サイズ 4MB を境に，L3 キャッシュに収まりきれなくなり，アクセス時間が急増していた．一方，ページ共有を用いて配列のうちの半分をシェアすると，シェア後に最初に実行したテストプログラムについて，L3 キャッシュからメモリに溢れるタイミングは同じく配列サイズが 4MB になる時であった．ところが，もう片方のテストプログラムは，アクセスできない領域にある共有ページのキャッシュデータを利用できるため，共有ページが 4MB，つまり全体で配列サイズが 8MB になるまで L3 キャッシュに全て収めることができ，ページ共有によるキャッシュ節約の恩恵を独占していた．これにより，配列サイズが 4MB と 8MB の間では，メモリアクセス時間で，最初に実行されるテストプログラムがもう片方の 4 倍にも上ることがあり，著しく公平性が損なわれていることがわかる．

† 東京農工大学
Tokyo University of Agriculture and Technology

現在，様々な手法を考案，実装し，このような不公平性を抑制できる方法を模索している．現段階で考えている案は，共有ページ専用のキャッシュ領域を Intel CAT で確保し，その領域にキャッシュデータが残り続けるよう，daemon を作成し，定期的に共有ページに対するアクセスを繰り返すというものである．今後はまず，前述のテストプログラムにおいて公平になるよう実装を行い，効果が確認されれば，仮想化環境において比較実験を行い，公平性が保てるよう実装を拡張する．その後，Dunn[4] で用いられたキャッシュ領域の自動振り分けを参考に，ページ共有を考慮した，Intel CAT による公平なキャッシュパーティショニング機構の実装を行う予定である．

参 考 文 献

1) H. D. Izik Eidus. How to use the Kernel Samepage Merging feature. `https://www.kernel.org/doc/Documentation/vm/ksm.txt`, 2009. [Online; accessed 28-November-2017].
2) G.Miłós, D.G. Murray, S.Hand, and M.A. Fetterman. Satori: Enlightened page sharing. In *Proceedings of the 2009 conference on USENIX Annual technical conference*, pages 1–1, 2009.
3) K. T. Nguyen. Introduction to Cache Allocation Technology in the Intel Xeon Processor E5 v4 Family. `https://software.intel.com/en-us/articles/introduction-to-cache-allocation-technology`, 2016. [Online; accessed 28-November-2017].
4) V. Selfa, J. Sahuquillo, L. Eeckhout, S. Petit, and M. E. Gómez. Application clustering policies to address system fairness with intel ' s cache allocation technology. In *Proceedings of the 26th International Conference on Parallel Architectures and Compilation Techniques (PACT'17)*, pages 194–205. IEEE, 2017.
5) C.A. Waldspurger. Memory resource management in vmware esx server. *ACM SIGOPS Operating Systems Review*, 36(SI):181–194, 2002.
6) M.Xu, L.Thi, X.Phan, H.-Y. Choi, and I.Lee. vcat: Dynamic cache management using cat virtualization. In *Proceedings of the 23rd IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'17)*, pages 211–222. IEEE, 2017.