2013 年 4 月 30 日 情報システム基盤学合同輪講発表資料

Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems

著者： Min Kyu Jeong, Doe Hyun Yoon, Dam Sunwoo, Michael Sullivan, Ikhwan Lee, Mattan Erez
出典： *Proc. of the Int'l Sym. on High-Performance Computer Architecture, pp.53-64, 2012.*
発表者： 高性能コンピューティング学講座　本多・近藤研究室　1353014 佐々木沢

1　はじめに

DRAM からデータを読み出す際は、行バッファと呼ぶ領域に読み出しデータを一時的に保存してアクセスする。一度行バッファに読み出したデータは高速にアクセスできるため、空間的局所性が活用できるデータであれば、DRAM アクセスの遅延を消滅できる。しかし、チップあたりに搭載されているコア数の増加にともなって、複数のスレッドによる行バッファへの干渉が問題となっている。CPU の高速な演算能力を活用するためには、行バッファ切替の頻発によるアクセスレイテンシの増加を抑える必要がある。

本論文では、主記憶のアクセス性能の向上を目的とし、Bank-partitioning（BP）と Sub-ranking（SR）の二つの手法を組み合わせることを提案する。BP は、複数のスレッドによるデータ取得の干渉を防ぐため、物理アドレス空間で競合しない番地に割り当てる手法である。SR は、DRAM アクセスを細分化してバンクの数を増加させ、バンクレベルの並列性を高める手法である。

2　DRAM のアクセス効率

図 1 に DRAM の基本構成を示す。

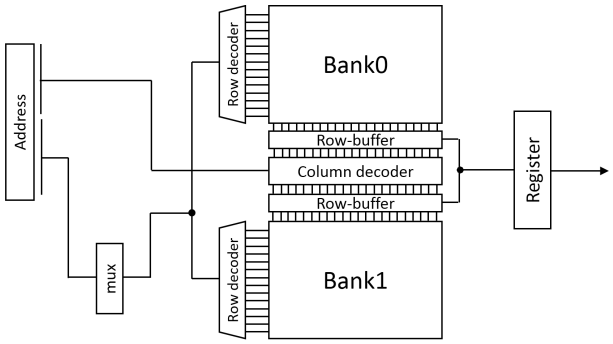

図 1: DRAM 基本構成

主記憶の内部は、バンクと呼ばれる複数のデータアレイに分割され、データは行と列に区切られたセルに格納される。CPU が要求するデータにアクセスするためには、行と列を指定して目的のセルを参照する必要がある。バンクは、それぞれに DRAM セルから読み出したデータを一時的に保持する行バッファを持つ。DRAM は値の保持をキャパシタで行っており、データを読み出した際に電荷が放出されてしまう。そこで、次の読出しのために行バッファをもう一度記録する必要がある。これをプリチャージと呼ぶ。DRAM へのアクセス時間において最も遅延が長いのは、行バッファ切替の際のプリチャージである。空間局所性の高いプログラムの場合、行バッファのデータに読み出したデータにプリチャージなしでアクセスできるために高速にアクセスすることが可能である。

3　CMP におけるバンク競合の発生

一度に DRAM のバンクへ並列にアクセスし、レイテンシを隠蔽をするために、データを複数のバンクに均等に割り当てることが一般的である。しかし、複数のスレッドから全バンクへアクセスされる可能性があることから、各バンクにデータを分けて配置した場合、バンク競合が発生する危険性が高い。従来は、行バッファの切替がなるべく起こらないように複数の DRAM アクセスコマンドの順番を入れ替える DRAM アクセススケジューリング手法が用いられてきた。しかし、アクセスを保持できるキュー内でしかスケジューリングは行えず、また、キューは有限であることから、スケジューリングには限界がある。さらに、根本的な原因であるスレッドの干渉を取り除いたとは言えない。

4　提案手法

4.1　Bank-partitioning

Bank-partitioning [1] は、それぞれのコアが実行するスレッドをある一部のバンクに配置し、スレッドの干渉が発生する可能性を排除する手法である。仮想・物理アドレス変換における物理フレームへのアドレス配置を改変し、バンクカラーリングによる割り当てを行う。バンクごとに異なるカラーを設け、スレッドごとに決まった色のバンクにのみアクセスを行う。

図 2 に BP を用いたデータをバンクに割り当てる様子を示す。BP を用いる前は、異なるスレッドの使用するデータがそれぞれ全てのバンクへ割り当てられ、行切替の競合が発生しやすい状況であった。BP によって、スレッド干渉が起きないように各スレッドのデータがバンクに割り当てられる。

BP を用いて遅延を削減する例を図 3 に示す。BP を用いる以前は、データ a2 が activate(行アクセス) され read された後に b2 への要求が発生した場合、行 0 から行 16 への切替が必要になるため、切替時間と precharge 時間に

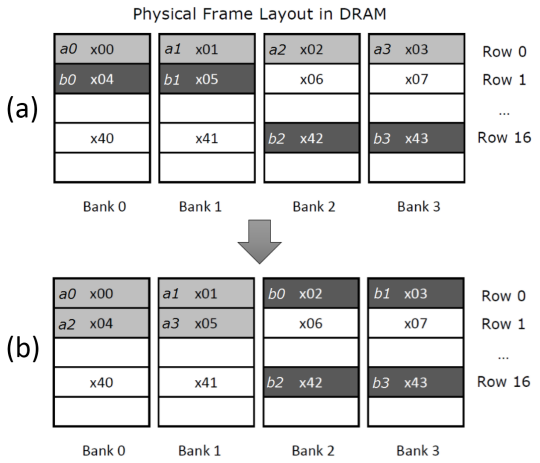

図 2: Bank-partitioning を用いたバンク割り当て

よる遅延が発生する。BP を用いることで、競合するアクセスが排除され、a2 と b2 は連続的なアクセスが可能になっていることが示されている。行切替の必要もないため prechage は実行されず、続けて a3 の activate に移行する。

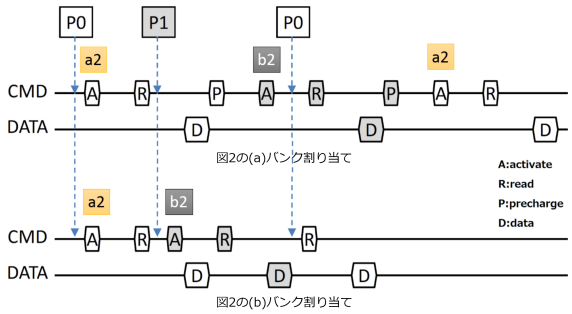

図 3: BP による遅延削減の様子

4.2 Sub-ranking

行バッファ上での局所性を活用できないプログラムは、DRAM アクセスが効率が悪くなる。そこで、レイテンシを削減するために、バンクレベルの並列性を用いることは重要である。しかし、BP はスレッドごとにバンクを割り当てるため、並列に使用することのできるバンク数が減少する。

不足したバンクレベルの並列性を補うためにここで Sub-ranking (SR)[2] を用いる。ｓｓ SR は、DRAM へのアクセス単位であるランクを分割し、並列に動作するバンクの数を増やす。今回は、全体で 64 ビット幅のランクを 32 ビットずつに分割された 2 つのランクとして扱う。

5 性能評価

Zesto シミュレータ上で、SPEC-CPU2006 を用いて性能評価を行った。コア数は 8 コアを仮定し、1 コアで 1 スレッドを動作させる。使用したベンチマークは、SPEC-CPU2006 の 15 個のプログラムから 8 個を組み合わせるが、ここで、IPC と行バッファのヒット率が異なるように 20 通り作成する。IPC と行バッファへのヒット率の高さから HIGH、MIX、LOW に区分される。図 4 は、スループットの評価結果である。各値は、提案手法を用いていない場合の値に正規化している。局所性の高い HIGH に分類されるプログラムのスループットは、BP を用いることで 10 から 15 %の向上が見られる。MIX および LOW の局所性の低いプログラムが混在したベンチマークにおいて、BP と SR の組み合わせが高いスループットを示している。さらに、図 5 の電力効率を考慮すると本提案が有用であることがわかる。一般的に、メモリあたりのバンク数を増加させると、高いコストがかかる。それに対し、本提案手法は、将来的にコア数がさらに増加した場合に、バンク数が少ない環境でも高いアクセス効率を示すことが可能であることがわかった。

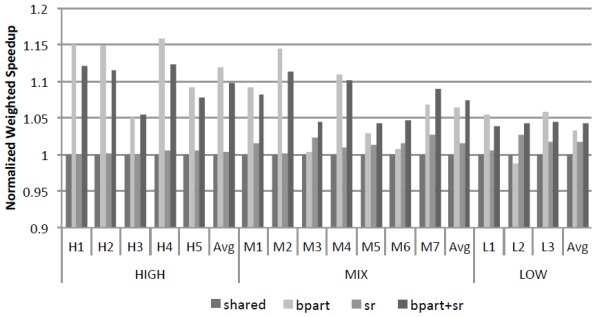

図 4: スループットの評価結果

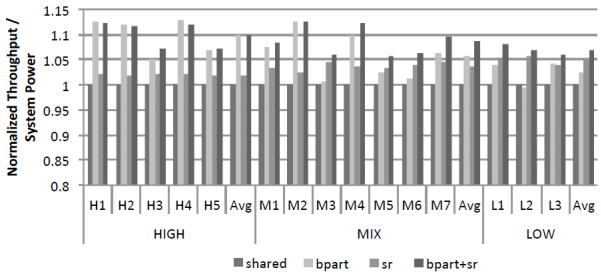

図 5: 電力効率の評価結果

6 おわりに

本論文では、BP と SR を組み合わせることで局所性と並列性のバランスをとる手法について提案した。コア数に対するバンク比が低い環境において著しいアクセス効率の向上があることがわかった。

参考文献

[1] JWei Mi et al. Software-Hardware Cooperative DRAM Bank Partitioning for Chip Multiprocessors, NPC'10 Proceedings of the 2010 IFIP international conference on Network and parallel computing, pp 329-343

[2] Doe Hyun Yoon et al. granularity memory systems: a tradeoff between storage efficiency and throughput, ISCA '11 Proceedings of the 38th annual international symposium on Computer architecture, pp295-306