LC-006

消費電力を考慮したウェイアロケーション型共有キャッシュ機構
A Power-Aware and Way-Allocatable Shared Cache Mechanism

小寺 功* 滝沢 寛之 * 小林 広明†
Isao Kotera Hiroyuki Takizawa Hiroaki Kobayashi

1 緒言

近年，マイクロプロセッサは性能向上が急速に進む一方で，高集積化による大量の電力消費や熱排出が問題となっている．特にテクノロジーの進歩に伴い，消費電力の中でも静的電力の占める割合が増加しており，2009 年には全消費電力の 80% が静的電力で占められると予想されている [1]．このため，電力消費の削減には動的電力の抑制以上に静的電力の抑制が求められている．

現在注目を浴びているマルチコアプロセッサ (Chip Multiprocessor, CMP) は，1 つのチップ上に複数のコアを集積し，スレッドレベル並列性を利用した処理の高速化に加え，電力あたりの処理能力を向上させることを目的として開発された．通常の CMP では，各コアに専用の 1 次キャッシュとコア間で共有する 2 次キャッシュを搭載する構成が一般的である．共有キャッシュを用いる利点として，各コア間で共有するデータへのアクセスをより高速にできること，1 つのコアが全てのキャッシュ空間にアクセスができるため，単一スレッド実行時には 1 つのコアで全容量を利用できることなどがある．しかしそのような共有キャッシュでは，コア間でデータを共有しない場合，キャッシュアクセスの競合が起こり，各コアで不均一に実行性能が低下するという欠点がある．これは各コアが実行するアプリケーションのデータ参照の振る舞いの違いに起因するものである．

CMP の共有キャッシュにおける不均一な性能低下の問題に対して，Kim らは各コアでの性能低下が均一になるように，セットアソシアティブキャッシュのウェイを動的に割り当てる手法を提案している [2]．その割り当て手法として，事前実行したプロファイルデータを用いて，共有した場合としていない場合のミス数の比率が各コアで一定になるようにウェイを割り当てている．また，Chandra らは各スレッドが均等に性能を出すために必要としているウェイ数を評価する詳細なモデルを提案している [3]．これは，キャッシュラインが参照される間隔を円列を用いて解析する手法である．しかし，いずれの提案も，現在重要な問題となっている消費電力削減の観点からの検討はなされていない．

プロセッサに集積される大規模オンチップキャッシュをすべてのソフトウェアが必要としているわけではないことに着目し、実行時にアプリケーションが必要としているキャッシュ容量を適切に予測して割り当てることができるウェイ適応性キャッシュが提案されている [4]．その結果，ウェイ適応性キャッシュは実行性能を低下させることなく，必要最小限の電力で稼動することが可能ある．これにより，無駄な電力消費を抑えることができ，プロセッサの省電力化を達成することができる．

本論文では，共有キャッシュにおけるコア間の公平性と消費電力の両方を考慮したキャッシュの制御機構を提案する．本機構では，シングルコアプロセッサの省電力化のために提案されたウェイ適応性の参照局所性の定量化手法を CMP の共有キャッシュに適用する．そして，共有キャッシュの容量を各コアに公平に割り当て，さらに不活性状態の領域をつくることで，均一な性能で省電力化を実現する．さらに，提案機構の性能をシミュレーションによって評価，検討する．

2 ウェイアロケーション型共有キャッシュ

2.1 概要

前節で述べたように，共有キャッシュメモリを使うことにより，参照の競合が CMP 全体の実行性能に影響を与える．また，アプリケーションが大容量のキャッシュメモリを必要としていない場合，無駄な電力を消費する原因になっている．そこで，実行性能の低下を回避しつつ，消費電力の最適化が実現可能なキャッシュ機構として，ウェイアロケーション型共有キャッシュ機構を提案する．まず，前提となる条件を述べる．

本論文で想定しているキャッシュメモリは，一般的なセットアソシアティブキャッシュであり，さらに各ウェイ単位で電力供給を停止，再開が可能なハードウェア機構を有している．電源供給を止めることによりリーク電流を削減でき，動的電力と静的電力をいずれも削減することができる．また，各ウェイを構成するサブアレイは，各コアによって排他的に参照可能である．

次に，キャッシュを共有するスレッド間でいずれのアドレススペースも共有しないものとする．つまり，各コアで異なるプロセスを実行するものとする．同じデータを共有するような場合は比較的その参照の特徴が似ており，不均一な影響を与えることが少ないためである [3]．

また，L2 キャッシュは LRU 置換を用いる．これは実際のプロセッサにおいても，pseudo-LRU 置換のように LRU を近似的に実現する置換方法が用いられているからである．

以上の前提条件の下で，本論文ではウェイアロケーション型共有キャッシュ機構を提案する．本機構は，大容量の共有 2 次キャッシュを各コアに対して専用キャッシュ空間として動的に割り当てるものである．2 コア CMP のときの提案機構の制御フローを図 1 に示す．提案機構は，2 種類の制御機能を用いる．1 つ目は，各コアが実行性能を維持するために必要なキャッ

* 東北大学大学院情報科学研究科
† 東北大学情報シナジー機構

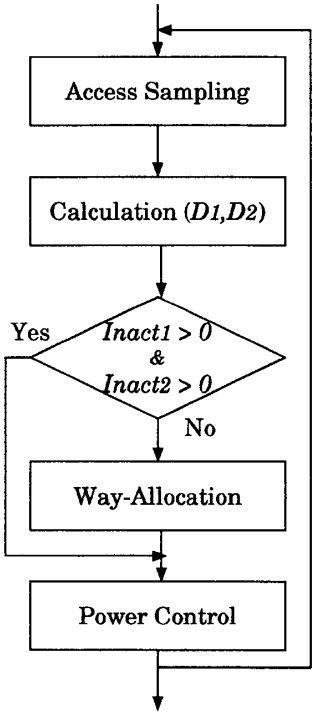

図 1　提案機構の制御フロー (2 コア CMP)

シュ容量を推定し，全コアが性能を発揮できるようにキャッシュ容量を公平に割り当てる機能である．2 つ目は，各コアで実行されているスレッドが割り当てられた全容量を必要としていない場合に，消費電力削減のために一部のウェイを不活性化する電力制御機能である．この 2 つの制御を一定期間毎に行う．通常の実行時はアクセスをサンプリングして統計情報を収集する．その情報を基に，局所性の程度を評価し，各ウェイに対して割り当てられるコアと電力供給の ON，OFF を決定する．ただし，ウェイアロケーション制御が各コアの性能に影響を及ぼすのは，割り当てられた全ウェイが活性状態の場合である．すなわち，コア i に割り当てられている不活性ウェイ数を $Inact_i$ とした場合，全てのコアが不活性ウェイを有するとき ($Inact_i > 0$)，ウェイアロケーションは行わない．

2.2　局所性評価量

まず，ウェイアロケーションと電力制御の双方で用いられる参照の時間的局所性の定量的評価について議論する．図 2 は同一ブロックへの参照の間隔をヒストグラム化したものである．横軸は参照の間隔を縦軸はアクセス数を示している．左図は時間的局所性が高い場合，右図は低い場合である．参照の時間的局所性から一般に MRU ブロックへのアクセス数が多く，LRU ブロックへのアクセス数は少ない．局所性が高い場合，LRU ブロックにヒットする確率は小さく，ウェイ数を減らしても性能の低下は小さいと推測できる．逆に分散していて局所性が小さい場合，LRU ブロックへのヒットする割合から考えて，キャッシュを広範囲に利用しているので，より多くのキャッシュを必要としていると推測され，ウェイ数を増加させることで性能向上が期待できる．

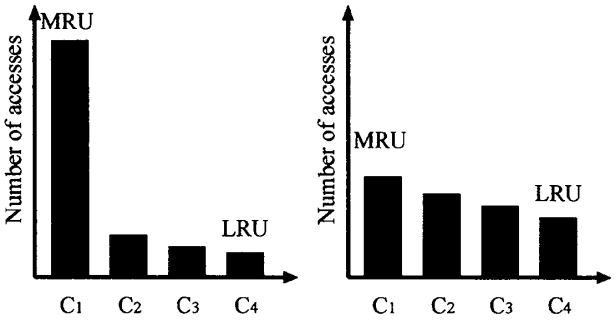

図 2　アクセスの局所性分析

ここで，局所性の程度は MRU ブロックと LRU ブロックへのアクセスの比で近似できることに着目し，アプリケーションのキャッシュ上での時間的局所性の振る舞いを定量化するために，局所性評価量

$$D = \frac{LRUcount}{MRUcount} \tag{1}$$

を用いる [4]．$LRUcount$ は LRU ブロックへのアクセス数，$MRUcount$ は MRU ブロックへのアクセス数である．

2.3　ウェイアロケーション機能

次に，ウェイアロケーション型共有キャッシュ機構の 1 段階目であるウェイアロケーション機能について述べる．一般的なキャッシュメモリはウェイと呼ばれる単位に分割されている．ウェイアロケーションはこのウェイ単位で各コアにキャッシュを割り当てる．

この制御手法として前節で述べた局所性評価量 D を用いて，キャッシュ参照の局所性を考慮に入れたアロケーションを提案する．ここでは，適切なウェイ割り当て数とは，局所性の高さに比例するという仮定に基づいている．まず，コア $i(i = 0, 1)$ からのアクセスについて D_i を求める．D_i について，

$$\text{if } D_k < D_l \quad \begin{cases} Alloc_k & +=1 \\ Alloc_l & -=1 \end{cases} \tag{2}$$

を行う．ここで，$Alloc_i$ は，コア i にアロケーションされたウェイ数であり，全ウェイ数 $Alloc_{all}$ をとしたとき

$$Alloc_{all} = \sum Alloc_i \tag{3}$$

を満たす．また，$D_0 = D_1$ の場合，再アロケーションは行わない．

この手法により，参照局所性の程度に応じたウェイアロケーションを実現することができる．この割り当てられた容量について，次節で述べる電力制御を適用することで必要に応じて不活性 (電力供給停止) 状態のウェイを作り全体の消費電力の削減を図る．

2.4　電力制御機能

ウェイアロケーション機能においては，D_i の相対的な比較によって適切なウェイアロケーションを行った．本節の電力制御機能では，ウェイ適応性キャッシュの制御法 [4] を利用し，

D_i を絶対値との比較によって，D_i に応じたウェイ数を活性化し，性能低下を許容範囲内に留める．

効果的で適切なウェイ数の制御を行うためには，アプリケーションが今必要としているキャッシュ容量を利用状況から予測する局所的判断と，過渡的な要求の変化に過剰に反応しないように時系列から全体的な傾向を把握する大域的判断の二つが必要である．提案機構の制御には，局所的判断のために局所的評価量と，大域的判断のために N ビットステートマシンをそれぞれ用いる．

2.4.1 局所的判断

局所的なキャッシュの必要量を求めるために，局所性評価量 D を用いる．閾値 (t_1, t_2) を導入し，D がある閾値 t_1 以下の場合は，MRU ブロック近辺に著しくアクセスが集中していると判断し，1 ウェイを無効化するべきと判断する．また，D が別の閾値 t_2 を越えている場合は，参照局所性を示すヒストグラムの分布が比較的広く分散していて局所性が低いといえる．この場合，ウェイ数を増加させることで大きな性能向上が得られると考えられることから，不活性化されているウェイを 1 つ活性化する．

2.4.2 大域的判断

2.4.1 節で，D を閾値と比較することで，増加，減少，維持の 3 つの判定を行うことを示した．このような判定方法は，常に安定したアクセス分布を示すアプリケーションに対しては有効に機能する．しかし，アクセスの分布が急激に変化するアプリケーションの場合，2.4.1 節で示した局所的評価だけで制御すると活性化・不活性化短期間に繰り返すことになる．しかも，これは本来の目的とするタイミングとは異なるので，ミス率の増加やキャッシュの増減制御のオーバヘッドの原因となる．このような場合，より長い期間で安定した状況を見つけ出し，キャッシュを安定して制御することでウェイ数の増減によって引き起こされるオーバヘッドを抑える必要がある．

キャッシュアクセスの大域的な振る舞いを評価し，制御を安定させるために N ビットステートマシンを用いる．例として図 3 に非対称型の 2 ビットステートマシンの状態遷移図を示す．まず，2 ビットカウンタで表すことのできる 4 個の状態を考える．00 を増加状態，11 を減少状態とし，それ以外の場合を維持状態とする．入力は局所的判断で得られた結果であり，入力に応じた遷移後の状態に対応する処理（増加，減少，維持）を出力をとする．局所的判断によりウェイ数を増加するべきと判定された場合，状態 00 に遷移し，逆に減少するべきと判定した場合 +1 した状態に遷移する．このように活性化・不活性化の持続性を評価することで，高い周期で増加・減少を繰り返すことを防ぐことができ，安定した制御を行うことができる．また，増加判定に傾いた非対称型にすることで，ウェイを減らすような制御は慎重に，ウェイの増加要求に対しては急速に対応することが可能であり，不用意な性能低下を防ぐことができる．

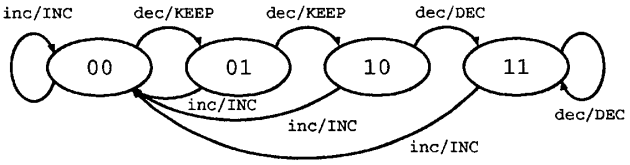

図 3　非対称 2 ビットステートマシン

表 1　主なシミュレーションパラメータ

Parameters	Value
fetch width	8 instrs
decode width	8 instrs
issue width	8 instrs
commit width	8 instrs
Inst. queue	64 instrs
LSQ size	32 entries
L1 Icache	32kB, 2-way, 32B-line
	1 cycle latency
L1 Dcache	32kB, 2-way, 32B-line
	1 cycle latency
L2 cache	1024kB, 32-way, 64B-line
	14 cycle latency
main memory	100 cycle latency

3　評価実験と考察

3.1　実験環境

提案手法の評価には CMP のシミュレーションツールである M5[5] を用いる．評価するプロセッサモデルは L2 キャッシュを共有する 2 コアの CMP であり，Alpha 命令セットアーキテクチャを用いている．ベンチマークアプリケーションとしては SPEC2006 から gcc, mcf, sjeng, dealII の 4 個のアプリケーションを用いて，各コアでそれぞれ 1 個づつ実行させる．実行命令数は 5 億命令とする．提案キャッシュ機構の制御間隔は 100,000 アクセス毎とする．主なシミュレーションパラメータを表 1 に示す．

3.2　ウェイアロケーション機能の評価

全てのベンチマークの組合せについての結果を図 4 に示す．図 4 の縦軸の Normalized IPC は，従来の共有キャッシュを用いて実行した場合の IPC で正規化したコア 0 の IPC を示している．横軸は，ベンチマークの組み合わせを示しており，例えば，gcc-mcf はコア 0 で gcc を実行し，コア 1 では mcf を実行した結果である．いくつかのベンチマークの組み合わせにおいて Normalize IPC が 1 を越え，平均値も 1% の IPC 向上を示している．従って，提案機構は従来の共有キャッシュにくらべ，競合が少なく，適切なウェイアロケーションができたといえる．また，各コアへウェイを割り当てる際の基準として参照の局所性を用いることの妥当性も明らかになった．

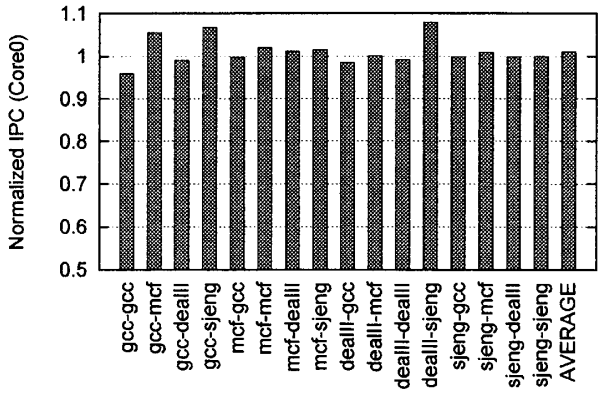

図 4　ウェイアロケーション評価 (IPC)

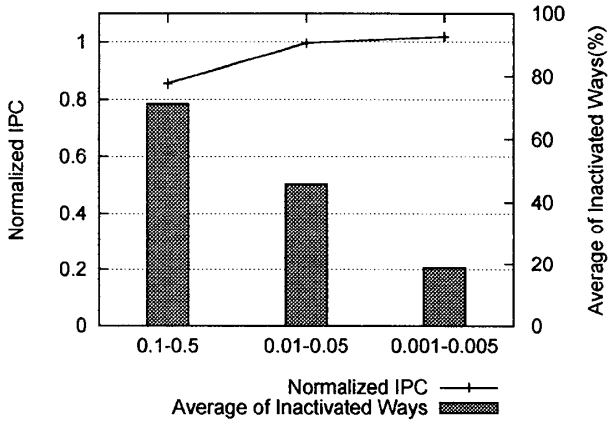

図 5　電力制御機能評価 (gcc-mcf)

一方で gcc-gcc では約 5% の性能低下を示したが，これは gcc は参照の局所性の低いアプリケーションであり，ウェイアロケーションすることが逆に性能低下につながっていると考えられる．このようなアプリケーションの組み合わせにおいては，ウェイアロケーション機能を無効にし，従来の共有キャッシュとして動作させる必要がある．

3.3　電力制御を加えたウェイアロケーションの評価

次に電力制御機能を加えた提案キャッシュ機構の性能を評価する．評価指標としては提案キャッシュ機構の活性化率と CMP 全体の IPC を測定する．電力制御機能の閾値は $(t_1, t_2) = (0.1, 0.5), (0.01, 0.05), (0.001, 0.005)$，ステートマシンは 3-bit の非対称型を用いる．

代表例として．図 5 にベンチマークが gcc-mcf の組み合わせの時の結果を示す．図 5 は，従来の共有キャッシュを用いた場合の IPC に対する Normalized IPC と平均の不活性率を示している．閾値が小さい値 $(0.001, 0.005)$ の場合，平均活性化率は約 80% であり，全てのキャッシュが活性化しているわけではないにもかかわらず，2 つのコア全体で 2% の IPC 向上が得られた．また，閾値が大きい値 $(0.1, 0.5)$ をとる場合，IPC が約 30% 低下したが，約 72% のキャッシュが不活性化しており，消費電力の大きな削減が期待できる．これらのことから，閾値が小さい場合は性能指向であり，大きい場合は省電力指向であるといえる．したがって，閾値 (t_1, t_2) はその値を操作することで，性能と消費電力の間でトレードオフをとる指標として使うことができる．

4　結言

本論文では CMP の共有キャッシュメモリの電力消費を低減し，最適なキャッシュサイズを提供できるウェイアロケーション型共有キャッシュ機構を提案した．参照の局所性を考慮したウェイアロケーションを導入することにより，従来の共有キャッシュよりも CMP 全体の IPC を向上させることが可能であることを示した．また，電力制御機能を導入し，閾値を性能指向に設定することにより共有キャッシュを用いる場合よりも約 2% の IPC 向上と約 20% の活性化率の削減を達成した．また，省電力指向にすることにより約 15% の IPC 低下で約 70% の活性化率の削減を達成した．

今後の課題としては，ハードウェア構成に関して詳細な検討を加え，具体的に電力の削減量をシミュレーションにより定量的に明らかにしていく．また，参照の局所性の面から省電力化が可能と考えられるキャッシュ以外のハードウェア資源に関しても，同手法の適用を検討する予定である．

謝辞

本研究の一部は、文部科学省科学研究補助金基盤研究 (B)（課題番号 18300011）による。

参考文献

[1] International technology roadmap for semiconductors. http://public.itrs.net.

[2] Seongbeom Kim, Dhruba Chandra, and Yan Solihin. Fair cache sharing and partitioning in a chip multiprocessor architecture. In *the Proceedings of the International Conference on Parallel Architectures and Compilation Techniques*, 2004.

[3] Dhruba Chandra, Fei Guo, Seongbeom Kim, and Yan Solihin. Predicting inter-thread cache contention on a chip multi-processor architecture. In *the Proceedings of the 11th Int'l Symposium on High-Performance Computer Architecture*, 2005.

[4] Hiroaki Kobayashi, Isao Kotera, and Hiroyuki Takizawa. Locality analysis to control dynamically way-adaptable caches. *SIGARCH Comput. Archit. News*, Vol. 33, No. 3, pp. 25–32, 2005.

[5] Nathan L. Binkert, Ronald G. Dreslinski, Lisa R. Hsu, Kevin T. Lim, Ali G. Saidi, and Steven K. Reinhardt. The m5 simulator: Modeling networked systems. *IEEE Micro*, Vol. 26, No. 4, pp. 52–60, 2006.