次世代省電力メモリを用いた並列プログラムの省電力化の評価

細萱祐人[†] 遠藤敏夫[†] 松岡聡[†,††]

近年大規模計算機の省電力化の要求の高まるにつれて, CPU だけでなくメモリの消費電力の削減が重要視されつつある. メインメモリとして使用される DRAM は揮発性メモリであるため, 消費電力, 特に待機電力が非常に大きい. しかし, スワップを避けるように設計される HPC では必要以上に DRAM を搭載しており, その結果多くの場合搭載された全メモリ使われてはいない. そこで, 我々は DRAM メモリの搭載容量を削減するためにメインメモリに MRAM と DRAM, スワップ領域に FLASH を配置した低消費電力システムを提案する. 本システムではページングによりメモリアクセスを高速な MRAM に集中させメモリアクセスの最適化を行っている. シミュレーションの結果, DRAM 搭載容量を削減することで実行時間の増加を 1.3 倍に抑え, 消費エネルギーを 1/3 に削減できることを示した.

Evaluation of Power Saving of Parallel Applications with Next Generation Low Power Memory

Yuto Hosogaya,[†] Toshio Endo [†] and Satoshi Matsuoka[†,††]

With the increasing demand for low power high performance computing, reducing power of not only CPUs but also memories is becoming important. In typical HPC environments large capacity of DRAM is installed to avoid memory swapping, although not all of the memory is used in many cases. Since DRAM is a volatile memory, such unused memory can waste a significant amount of power even in a standby state. We propose a next generation low power system that intends to reduce the DRAM capacity without causing application performance degradation. In this system, MRAM and DRAM is used as a main memory, while FLASH is used as a SWAP. Our profile-based paging algorithm optimizes memory accesses by avoiding I/O with slower memories and using faster memories as much as possible. Results from our simulation with parallel applications show that the power consumption can be reduced up to one third, with 30% performance loss for the applications.

1. はじめに

計算機の消費電力は高性能化に伴い非常に増加しており近年特に高い関心事となっている. 消費電力はすでにスパコンやクラスタの高性能化の足枷となっており, システムの省電力化への要求は非常に高い. 計算機において最も消費電力の大きい CPU の省電力化については, 動作周波数を動的に制御可能な Dynamic voltage scaling(DVS) を用いた研究などが広くなされている.[1)]

しかし, 高集積度化とそれに伴う廉価化が進んだメモリは計算機に大量に搭載されるようになり, その消費電力も無視できない程度になっている. たとえば FB-DIMM の消費電力はモジュールあたり 10W 程度となり, 複数枚搭載すると CPU の電力に匹敵しうる. また,HPC 用システムは非常に大量のメインメモリを搭載する傾向にある. これは様々なアプリケーションが実行されうる可能性がある中で, スワップの可能性を最小限にするためには, 充分な量のメインメモリが必要なためである. その結果, 実環境では搭載されたメモリの全容量を使っていない場合が多い[2)]. 一方, 現在メインメモリに広く使われている DRAM は, 一定周期でリフレッシュという操作を行いデータを保持している. そのため, 使用していないメモリモジュールも大きな待機電力を消費している. そこで搭載する DRAM 容量を削減することで HPC 用システムの省電力化が図れるのではないかと考えられる.

本研究では, DRAM 容量の削減を行っても性能低下を小さく抑えることを目的として, 以下のようなヘテロなメモリを持つシステムを提案する. まずスワップのコストを大きく削減するために, スワップ領域として HDD ではなく, アクセス遅延がはるかに高速な FLASH メモリを用いる. さらに, 次世代不揮発メモリ

† 東京工業大学
Tokyo Institute of Technology
†† 国立情報学研究所
National Institute of Informatics

として，DRAM よりも高速化が期待されている Magnetoresistive Random Access Memory (MRAM) を採用し，これをメインメモリ階層に DRAM と並列に配置する．この方式により，高速メモリをキャッシュとして (直列に配置して) 用いるよりも，DRAM 容量を効率的に削減できると期待される．そして並列配置したメモリのうち高速なものにアクセスを集中させるためのページングアルゴリズムを提案する．

本論文では上記のようなシステムにおいて, 与えられた各メモリ容量に対してアプリケーション実行時間と消費電力量を見積もるモデルを提案する．そして提案システムにおいて，DRAM の搭載容量を削減しても，アプリケーション実行時間を 1.3 倍に抑え, 消費エネルギーを 1/3 に削減することをシミュレーションで示した．

2. 不揮発性メモリ

FLASH メモリは不揮発性メモリの中で最も利用が広まっている．数十 GB/s のアクセス速度を持つ USB メモリや Solid state disk(SSD) がすでに製品化され，HDD の代替または補助的に使われ始めている．Windows Vista の機能である Ready-Boost[3] は USB FLASH メモリを簡単にスワップ領域として拡張することが可能で, 高速かつ低消費電力化を実現している．また,HDD に FLASH メモリを付随させた HHDD(Hybrid Hard Disc Drive) も市販されている．今後更に, MLC(Multilevel Cell) や 3 次元構造の実現により集積は増し，速度向上すると期待される．本研究でも FLASH メモリをスワップ領域として用いることにより省電力化を図るが，主な対象は HPC アプリケーションであり，デスクトップアプリケーションとは特性が大きく異なる．

FLASH メモリは，速度や書き換え回数の制限のためにメインメモリとして用いるのは困難である．一方，メインメモリとして有望な不揮発性メモリとして，Magnetoresistive RAM (MRAM)[4], Phase-change RAM (PRAM), Ferroelectric RAM (FeRAM) などの研究が行われてきている．中でも MRAM は，DRAM よりも高速で，はるかに低消費電力が期待される点，小容量ではあるものの 2006 年に freescale 社[5] により製品が発売されている点から最も注目されている．MRAM は 2 枚の磁性体のスピンの向きの違いによる抵抗差によってデータを保存しているため, 電荷によってデータを保持している DRAM 等と違い不揮発性を実現している．しかし, 現時点で集積度の向上や書き込み電力の削減等に課題を抱えている．また, 大容量製品の量産化がなされるまではビット単価が DRAM よりも高価であることが予想されるため,HPC 用システムの全メインメモリを MRAM で構成するのは難しいと考えられる．そのため本論文では, メインメモリの一部を MRAM に置き換え，その容量と性能の関係について詳細に議論する．

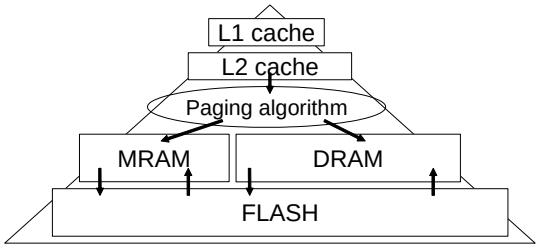

図 1 提案システムの概要

3. 提 案 手 法

本研究ではメインメモリに MRAM と DRAM, スワップ領域に FLASH を配置した次世代のメモリシステムを提案する (図 1)．キャッシュとして MRAM を使用しても DRAM の容量は削減することはできず, 本研究の目的にそぐわない．そのため,MRAM を DRAM と並列に配置して DRAM の容量を削減する．また,DRAM の容量を削減することに伴い発生するスワップのコストはスワップに HDD の代わりに FLASH を使用することで削減することができる．

3.1 ページング手法

メインメモリに DRAM と MRAM の 2 種類の RAM を配置したことによって，データをこの 2 種類の RAM に対してどのように配置するかは性能に非常に大きく影響を与え, これはソフトウェアで適切に処理する必要がある．MRAM は DRAM よりも高速かつ低消費電力なため, できるだけ多くのメモリアクセスを MRAM に集中させることがアプリケーションの性能向上につながり, それを実現するページング手法が必要になる．ページングに対する要件としては (1) アクセス数の多いページは MRAM 上に配置する, (2) アクセス数の多いページは LRU でスワップアウトの対象になってもメモリ上に残す. である．

まず, アプリケーションのメモリアクセスのプロファイルを取得し, アプリケーションの実行を通してページごとのアクセスが分かっていると仮定する．その上でメモリアクセスの多いページから搭載 MRAM 容量が許す限り MRAM 上に配置し, MRAM から溢れたページを DRAM 上に配置した．そして MRAM 上のデータはメモリ上に固定しスワップせず, DRAM 上のデータのみを LRU によってスワップを行った．これによりアクセスの多いデータを MRAM 上に配置することが出来, その結果高性能な MRAM へのアクセスを増やすことできる．また, アプリケーションの実行を通してアクセス数の多いページはメインメモリ上に残すことが可能となり, 非効率なスワップアウトを削減することができる．

しかし, この方法だと特に搭載 DRAM 容量が小さいとき, スワップアウトしうる対象のページが少なく効果的にスワップが行われないことがあった. また, メモリアクセスの局所性の低いアプリケーション, つまり MRAM 上に配置したページとその他のページへのアクセス数が同じようなアプリケーションでは, MRAM 上のデータをメモリ上に固定するアドバンテージが小さく効果的にスワップを減らすことはできない. 実際シミュレーションの結果, MRAM 上にデータを固定しない場合と比較して実行時間は 2 倍程になってしまった場合があった. そのため, 搭載メモリ容量やアプリケーション特性等に応じてページング手法は動的に振る舞いを変更する必要がある.

スワップアウトの振る舞いを決定する指標として, 次式で与えられる MRAM ヒット率を導入する.

$$MRAM\_HIT\_RATE = \frac{MRAM\_ACCESS}{TOTAL\_ACCESS}$$

式中の MRAM_ACCESS,TOTAL_ACCESS はそれぞれアプリケーション実行からその時点までの MRAM 上のデータへの総アクセス回数, MRAM と DRAM 両 RAM 上のデータへの総アクセス回数である. MRAM ヒット率が大きいことは, MRAM 上のデータへのアクセスが多いことを示しており, この時効果的にデータが MRAM 上に配置されている. MRAM ヒット率が小さい時は, プロファイルによって MRAM にアクセス数の多いページを MRAM 上に集めたにも関わらずアクセスがあまり集中していない. これはアプリケーションのメモリアクセスの局所性が小さい時に発生する. そのようなアプリケーションではデータを MRAM 上に固定するアドバンテージは小さいため MRAM 上のデータもスワップアウトすべきである.

そこで次式で定義される閾値と MRAM ヒット率を比較することで適切な振る舞いを決定する.

$$THRESHOLD = \alpha \times \frac{MRAM\_SIZE}{TOTAL\_SIZE}$$

上式中の MRAM_SIZE および,TOTAL_SIZE はそれぞれ搭載した MRAM の容量, MRAM と DRAM の総容量を示している. また,$\alpha(\approx 1)$ は定数である.

ページフォルトが発生したとき,MRAM ヒット率が閾値を上回れば, MRAM 上のデータはメモリ上に固定し DRAM 上のデータからスワップアウトするデータを選択する. 逆に, 閾値を下回れば MRAM,DRAM 両 RAM 上のページからスワップアウトする対象のデータを探す. この操作により適切なページングが可能になった.

尚,α は小さいほど MRAM 上のデータはメモリ上に固定されやすくなり, 大きいほどスワップしやすくなる. この値は予備実験から 0.9 が最もよい結果となったため以下, 本論文では 0.9 とする.

3.2 性能モデル

ここでは, 本提案システムの性能および消費エネルギーを見積もるためのモデルを提案する. まず, 以下のように定数を導入する.

T_{wj} : メモリ j の WRITE 遅延
T_{rj} : メモリ j の READ 遅延
E_{wj} : メモリ j の WRITE 時電力
E_{rj} : メモリ j の READ 時電力
N_{wj} : メモリ j への WRITE アクセス数
N_{rj} : メモリ j への READ アクセス数
N_{pm} : MRAM 上のデータのスワップ回数
N_{pd} : DRAM 上のデータのスワップ回数
S_j : メモリ j の搭載容量
E_j : 単位サイズ当りのメモリ j の待機電力
BS : ブロックサイズ
PS : ページサイズ
t_{calc} : メモリアクセスに非依存な計算時間

ここでメモリ j は MRAM,DRAM,FLASH を指すものとし, それぞれのメモリは頭文字をとって m,d,f と表すこととする. 例えば,T_{wm} は MRAM の WRITE 遅延を表し,E_{rd} は DRAM の READ 時電力を表す. また, 遅延と電力は MRAM についてはブロックサイズ, FLASH においてはページサイズ分のアクセスのものとする.

まず, スワップの発生しないアクセス全体のアクセス遅延 (T_{main}), 消費エネルギー (E_{main}) は以下のようになる.

$$T_{main} = \sum_{j=m,d} N_{wj}T_{wj} + N_{rj}T_{rj}$$
$$E_{main} = \sum_{j=m,d} N_{wj}T_{wj}E_{wj} + N_{rj}T_{rj}E_{rj}$$

次に,page fault が発生してスワップが起こったアクセス全体のアクセス遅延 (T_{SWAP}), 消費エネルギー (E_{SWAP}) は以下のようになる.

$$T_{SWAP} = (N_{pm} + N_{pd})(T_{wf} + T_{rf})$$
$$E_{SWAP} = (N_{pm} + N_{pd})(T_{wf}E_{wf} + T_{rf}E_{rf}) + \sum_{j=m.d} (N_{pj}T_{wj}E_{wj} + N_{pj}T_{rj}E_{rj})\frac{PS}{BS}$$

page fault 発生時にはスワップインとスワップアウトの両方の処理が起こっているため, FLASH には読み込みと書き込みの両方の処理が起こっている. また, スワップが発生するとメインメモリ階層には, スワップアウトするページの読み込みと, スワップインするページの書き込み処理が起こっていることを考慮する. しかし, メインメモリにスワップアウトする読み込み処理, スワップインする書き込み処理の遅延は, FLASH にスワップアウトと並列に行われると考えられるので遅延に含めていない.

よってアプリケーションの実行時間 (T) と消費エネルギー (E) は以下のようになる.

L2 cache サイズ	1MB
ブロックサイズ	64B
連想度	1
ページサイズ	4KB

表 1 シミュレート環境

	MRAM	DRAM	FLASH
読み込み遅延 (ns)	15	22.5	40000
書き込み遅延 (ns)	15	22.5	80000
読み込みエネルギー (nJ)	6.24	1.23	760000
書き込みエネルギー (nJ)	6.24	2.36	760000
待機電力 (μW/MB)	867	173	0
アクセス単位 (Byte)	64	64	4096

表 2 設定したパラメータ

$$T = T_{main} + T_{SWAP} + T_{clac}$$
$$E = E_{main} + E_{SWAP} + T \sum_{j=m,d,f} E_j S_j$$

消費エネルギーの第 3 項はシステムの待機電力による消費エネルギーの合計である.

4. 評　　価

本研究は次世代のメモリ環境を想定しており, 現在では MRAM は入手困難なため, シミュレーションによって評価を行った.

本シミュレーションは実アプリケーションのメモリアクセスのトレースしたログを使用した. メモリアクセストレースは Valgrind[6] に L2 cache までをシミュレートするパッチ実装して行った. ログファイルには時系列に従って L2 cache miss が発生したアクセスのアドレスと実行される処理 (READ または WRITE) が出力される.

シミュレートする環境を表 1 に, モデル式に当てはめる各メモリのパラメータは表 2 に示す. パラメータの値は数年後に各メモリに期待される数値を考えている. また,FLASH の容量は無限にあるものとしたため, 待機電力は 0 とした. 評価に用いたベンチマークは Nas Parallel Benchmark3.2 の CG(class B),MG(class A),SP(class B) と HPL で, それぞれ 1 ノードで実行した. ログサイズとシミュレーション時間の関係上, NPB の各アプリケーションは Iteration2 回で行い,HPL は Matrix size を 6720, Block size を 224 に設定し行った. また,CG,MG,SP,HPL の総メモリ使用量は 412MB,448MB,344MB,368MB であった.

4.1 ページング手法の評価

まず, 提案したページング手法について評価する. 図 2 は 128MB の MRAM を搭載したシステムで, 搭載 DRAM 容量を変更しながらそれぞれ CG,MG をシミュレートした結果である. 各系列はページング手法の違いを示しており, profile を行わない手法,profile をして MRAM 上のデータをメモリロックする手法, 提案手法を比較した. CG のグラフから分かるように profile を行った結果, アクセス数の多いページを MRAM 上に集めることが出来, 実行時間が早くなることが分かる. また,MG のグラフから特に DRAM 容量が少ないとき閾値を導入することで, DRAM と MRAM からスワップアウトするページを適切に決定し効果的なスワップを実現出来ており, 実行時間の低下を抑制できていることがわかる.

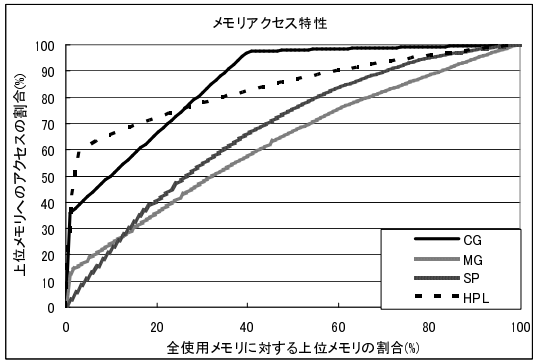

図 4 メモリアクセス局所性の違い

4.2 実行時間の変化

次に, 提案手法において搭載する DRAM 容量を変化させたとき, 実行時間と消費エネルギーがどのように変化するかを見ていく. 図 3 は搭載するメモリ容量を変化させた時の各アプリケーションの実行時間の変化である. グラフは可視性を高めるため, 適宜拡大している. 横軸は搭載する DRAM の容量, 縦軸は実行時間, 各系列は搭載する MRAM 容量を示している.

グラフを見てわかるとおりスワップが発生すると顕著に性能が低下するアプリケーションと, 性能がある程度維持できるアプリケーションに分かれることがわかる. 例えば,CG,HPL では DRAM 容量を削減してもある程度は性能が維持できているのに対して, MG,SP ではスワップの使用で性能が顕著に低下している.

このスワップ使用での性能低下の違いはメモリ特性, 特にアクセスの局所性に起因している. 図 4 は各アプリケーションのメモリアクセス局所性を示している. 横軸はメモリアクセスが多いページから順番にどれくらいとるかを示しており, 縦軸はその上位ページにあるアクセスが全メモリアクセスに対してどれくらいあるかを示している. このグラフを見てわかるように CG,HPL はメモリアクセスの局所性があることがわかり, 特に CG は約 40%のページに 95%以上のメモリアクセスが集中していることがわかる. 一方,MG,SP はグラフが線形に近い形であることから, 全使用メモリにほぼ同様なアクセスがあることがわかる. MG や SP のようなメモリアクセスの局所性が低いアプリケーションでは, メインメモリが不足するとスワップ回数が急激に増加してしまい, 顕著な性能低下を招いてしまう.

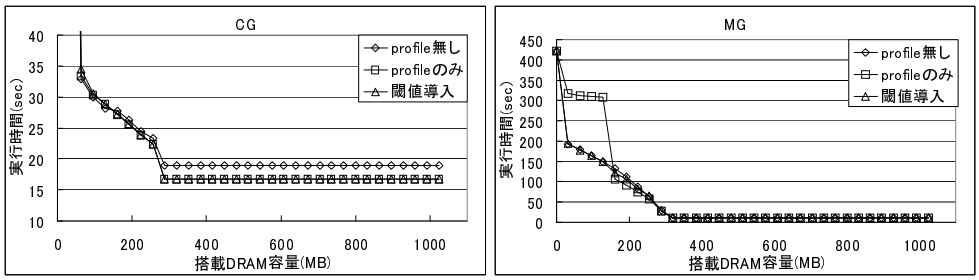

図 2 ページング手法の評価 - MRAM 128MB

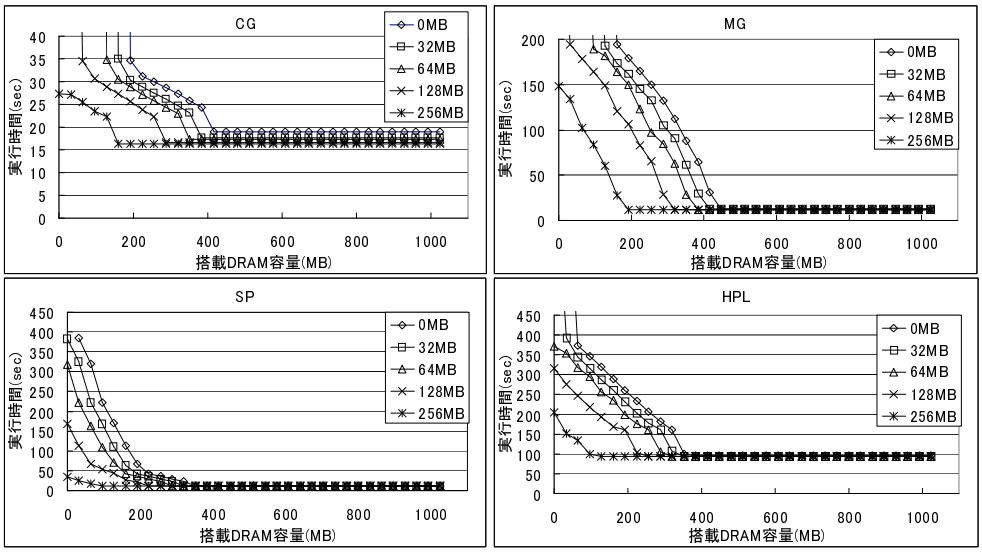

図 3 実行時間の推移

4.3 消費電力の変化

図 5 は各アプリケーション実行時にメモリチップで消費する消費エネルギーを示している．搭載した DRAM 容量に比例してシステムの待機電力による消費エネルギーが増加するため, DRAM の搭載容量を増加させると消費エネルギーは増加していく．一方 DRAM 容量を削減させ,SWAP が発生すると実行時間が増加するため, 長い時間システムを稼動することになり, 結果として待機電力による消費エネルギーが増加する.

グラフから分かるように, メモリアクセスの局所性を持つアプリケーションでは, 搭載する DRAM 容量を削減することは消費エネルギーを削減することに非常に有効であることがわかる．これは DRAM のみ搭載しているシステムでも言えることで, 例えば, DRAM を 1GB から 192MB まで削減して CG を実行すると実行時間は約 1.8 倍となるが, 消費エネルギーは 1/2 まで削減される．また,MRAM を導入し, 提案手法を用いると, 128MB の MRAM を搭載し,DRAM 容量を 192MB まで削減すると, DRAM1GB のシステムと比べ, 実行時間は約 1.3 倍に抑えることができ, 消費エネルギーは 1/3 まで削減が可能になることが分かる.

一方, メモリアクセスの局所性の低いアプリケーション, 特に MG では SWAP が発生しない程度の DRAM 容量の削減は消費エネルギーの削減につながる．しかし,SWAP の発生と同時に急激な実行時間の増加に伴い消費エネルギーも大きく増加してしまう．そのため, このようなアプリケーションに対しては現状のまま SWAP を伴う程の DRAM の削減は難しく, 更なる工夫が必要であり, これは今後の課題としている.

5. 関 連 研 究

組み込みシステムではメモリは一般的に階層構造を成していなく, 並列に配置されている．これはキャッシュを用いないことで, 電力やスペースのロスやキャッシュの書き換えなどの無駄な処理を避けるためである．そのため, データをどのメモリに配置するかはソフトウェアに依存し, これは性能に対して大きな影響を与える．論文[7] ではコンパイル時にプロファイルを元にアプリケーションの実行を通して最もメモリアクセス

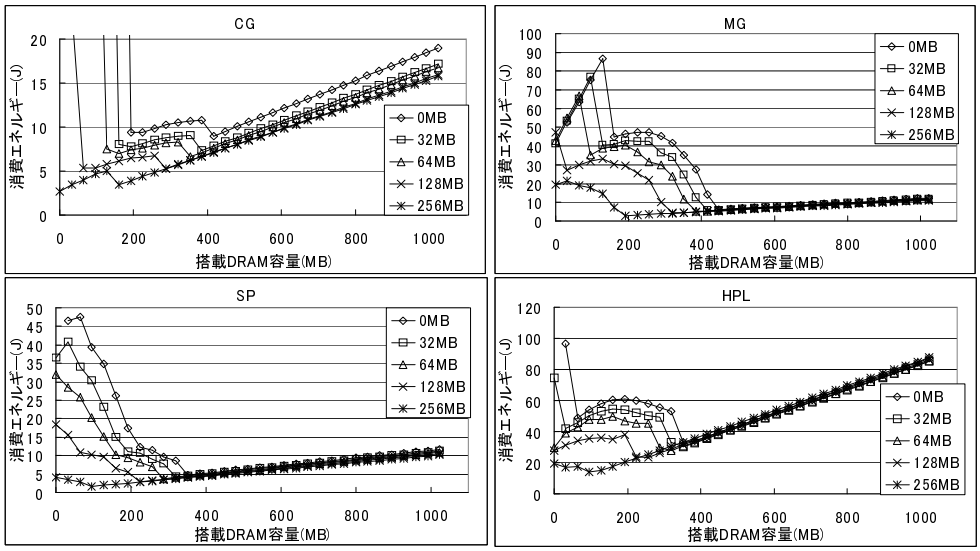

図 5 消費エネルギーの推移

遅延が小さい配置を 0/1 整数計画問題を解くことによって探し出している．しかし,HPC ではコンパイル時にアプリケーションが動作する環境が予測不可能であることや, 他のアプリケーションと物理メモリを共有する可能性も十分考えられるのでデータ配置は動的に行われなくてはならない．

スワップの使用を考慮した研究としてはリモートスワップの研究があげられる．論文[8] では, 10GbEthernet 上に RDMA を用いたリモートスワップを実装している．スワップアウトの回数を減少させるために, 書き込みを頻繁に行うデータはメモリ上に固定し, またデータはファイルマップとして持つ．そして, 変更していないデータはスワップアウトせずにそのままメモリ上から削除するようにすることで, スワップアウトの処理を削減することを実現している．本システムでもスワップアウトを削減する同様な手法は非常に有効であると考えられ, 今後の課題にあげられる．

6. おわりに

6.1 まとめ

本研究では, ヘテロなメモリを持つ省電力システムを提案し, その性能と消費電力をシミュレーションによって評価を行った．評価の結果, 特にメモリアクセスの局所性を持ったアプリケーションでは DRAM の搭載容量を削減し, 消費電力を大きく削減できることを示した．

6.2 今後の課題

FLASH メモリの特性上 FLASH には書き込み回数の制限や, READ はある程度高速で,WRITE の遅延は非常に大きいという性質を持っている．しかし, 本システムは READ,WRITE の違いは考慮に入れておらず, 今後データへのアクセスの特性を考慮したページングアルゴリズムが必要だと考えている．

参考文献

1) Chun Liu, et al. Exploiting Barriers to Optimize Power Consumption of CMPs. In *IEEE International Parallel Distributed Processing Symposium 2005 (IPDPS 2005)*, pp. 5a– 5a, March 2005.
2) Ganglia:: TGC –TSUBAME Grid Cluster Grid Report. http://ganglia.cc.titech.ac.jp/.
3) Microsoft Corporation. *Windows PC Accelerators*, November 2006.
4) Saied Tehrani, et al. Magnetoresistive Random Access Memory Using Magnetic Tunnel Junctions. In *PROCEEDINGS OF THE IEEE vol91 No.5*, pp. 703–714, May 2003.
5) Web-site:Freescale. http://www.freescale.com/.
6) Web-site:Valgrind. http://valgrind.org/.
7) Oren Avissar, et al. Heterogeneous Memory Management for Embedded System. In *ACM 2nd international Conference on CASES 2001*, 2001.
8) 後藤正徳ほか. 10Gb Ethernet 上の RDMA を用いた遠隔スワップメモリの実装. 電子情報通信学会 (CPSY2006), 2006.