様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成 ２９ 年　 ６ 月　 ４ 日現在

機関番号：１４３０１
研究種目：基盤研究(B)（一般）
研究期間：2014～2016
課題番号：２６２８００４４

研究課題名（和文）メニーコアプロセッサ向け高性能アプリケーション開発フレームワークの研究

研究課題名（英文）Research on the framework of high-performance application development targeting manycore processors

研究代表者
中島　浩（Nakashima, Hiroshi）

京都大学・学術情報メディアセンター・教授

研究者番号：１０２４３０５７

交付決定額（研究期間全体）：（直接経費）　　12,900,000円

研究成果の概要（和文）：メニーコアプロセッサを対象とした高性能アプリケーションの開発のための要素技術として、cache-awareなコード生成、SIMD化ループ変換、および計算・通信のパイプライン化を研究した。これらの要素技術と開発フレームワークを、超音波伝搬解析、PIC法プラズマシミュレーション、各種の線型方程式ソルバーなどの、実際的な高性能アプリケーションに適用した。その結果、たとえばPIC法プラズマシミュレーションにおいて、単一プロセッサレベルで従来実装の約8倍の性能向上を、また64ノードの大規模並列計算でマルチコアベースのシステムと比べて約10倍の性能向上を達成するなど、研究の有用性を示す重要な成果を得た。

研究成果の概要（英文）：We have pursued the research on three elemental technologies, namely cache-aware code generation, loop transformation for SIMD-vectorization and computation-communication pipelining, for high-performance application development targeting manycore processors. These technologies and our development framework were applied to practical high-performance applications such as supersonic propagation analysis, PIC plasma simulation, and various linear solvers. The importance and effectiveness of our research work were proved by the results of, for example, PIC plasma simulation whose single-processor performance improvement is 8-fold and 64-node performance is 10-times as high as that of a multi-core based system.

研究分野：計算機工学

キーワード：並列処理　メニーコアプロセッサ　プログラム変換

１．研究開始当初の背景

研究開始当初の代表的メニーコアプロセッサであった Xeon Phi KNC は、GPGPU などの演算加速機構とは異なり、従来型のプロセッサと同じ命令セットアーキテクチャを有することで、これまでに開発された高性能アプリケーションプログラムが（ほぼ）そのまま実行可能であるという、有用な特質を備えている。しかしこのプログラム可搬性は、従来型のプロセッサで得られていた実行効率（実効性能とピーク性能との比）が維持される性能可搬性を意味するものではなく、実際ほとんどの高性能アプリケーションでは従来型プロセッサの数分の１の実行効率しか得られず、結果的にピーク性能の向上が実効性能の向上に全く寄与しない状況となっていた。この低い実効効率の原因は、以下の３つの問題点にあった。

(1) 演算・メモリバンド幅のギャップ：Xeon Phi KNC のコアあたりメモリバンド幅は 6GB/s 弱、またキャッシュ容量は 512KB であり、従来型プロセッサ（たとえば Xeon E5-2600 系列）と比較すると 92%のバンド幅と 17%の容量となる。また命令実行機構が単純であるため、特にキャッシュ容量が小さいことを補うメモリアクセス遅延隠蔽が十分に機能せず、メモリバンド幅の不足が簡単に顕在化してしまう。

(2) SIMD 演算機構活用の困難さ：SIMD 演算機構は、少数のストリーム（１次元配列）を対象とした単純で依存関係がない演算ループには大きな効力を発揮するが、多数のストリームや配列要素の間接参照、部分的なループ運搬依存、ループボディ中の条件節など、多くの高性能アプリケーションのカーネルループに見られる比較的低レベルの複雑性に対して無力であり、アーキテクチャやコンパイラの適合性の限界が簡単に露呈してしまう。

(3) メニーコアノード間通信のバンド幅・遅延：研究開始時点でのメニーコアプロセッサは、ノード間通信を直接行うための機構を有しておらず、ホストプロセッサと呼ばれる従来型プロセッサを経由した通信とならざるを得ない。したがって演算性能に見合うノード間通信バンド幅を得ることは困難であり、また従来型の並列システムに比べて通信遅延が大きい。

なお研究開始から 3 年を経過した現在でも、たとえば KNC の後継である Xeon Phi KNL に関して、(1)の命令実行機構が out-of-order になったことと、(3)のホストプロセッサ経由の通信が自立型になった以外には、上記の状況は基本的に変わっていない。また後述するように、自立型になったことによる通信性能の改善は、演算性能の向上に見合うものとはなっておらず、(3)の問題は何ら解決していない。

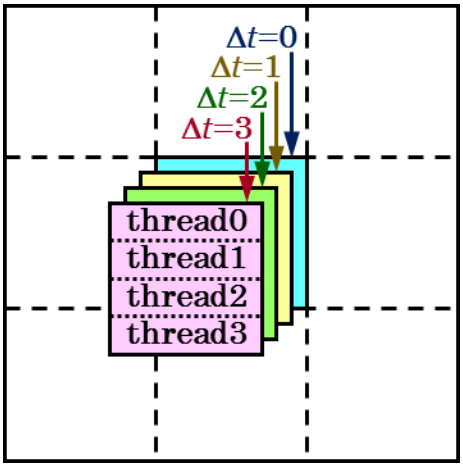

図 1: 時空間タイリング

２．研究の目的

本研究では、メニーコアプロセッサでの高性能計算における前述の問題点、すなわち①演算・メモリバンド幅のギャップ、②SIMD 演算機構活用の困難さ、③メニーコアノード間通信バンド幅・遅延を、アプリケーションコードのレベルで回避するために、以下の３つの高速化要素技法を開発することを目的として行った。

(1) 時間・空間の多重ループの実行順序を変更する時空間タイリングをベースとした cache-aware コード生成。

(2) キャッシュの存在を前提とした loop-splitting や strip-mining、およびループ運搬依存性を持つループをも対象とした、SIMD 化ループ変換。

(3) d 次元問題の d 次元分割において、メモリ空間を連続的に走査しつつ通信遅延を隠蔽し、かつ通信方向を $2\times d$ とする計算・通信のパイプライン化。

上記(1)～(3)の要素技法の適用条件やコード変換法は問題依存性が強いため、局所視点コーディングにより記述したプログラムから従来型のコードを自動生成するフレームワークとした。

３．研究の方法

(1) cache-aware コード生成

本項目のベースとなる時空間タイリングは、図 1 に示すように走査対象空間をタイルに分割し、時間発展計算に伴ってタイルを１格子点だけずらすことにより、各格子点に対する本来の計算順序を維持しつつ、時刻 t と t+1 の計算対象タイルの共通部分の格子点データをキャッシュ上で再利用することによって、参照の時間的局所性を大幅に改善する技法である。この技法の有効性は FDTD 法のみならず種々のアプリケーションにおいて確認しているが、本研究以前の我々の知見は主に従来型プロセッサでの評価に基づくものであり、メニーコアプロセッサでの有効性については未知の要素が少なくなかった。特にタイル計算のスレッド並列化を図に示すように単純な１次元分割で行っているが、たとえば Xeon

Phi の構成で最低限必要な 60 スレッド、あるいは演算遅延・キャッシュアクセス遅延の隠蔽のための Hyper-Threading を適用した場合の 120～240 スレッドの実行では、タイルの多次元分割やタイル単位のスレッド割当など、より高度な並列化が必要と予想されたため、タイルの形状や計算順序についてより高度な実装を検討した。

(2) SIMD 化ループ変換

たとえば 3 次元 FDTD 法の空間 3 重ループ；

```
for(t){
  for(z) for(y) for(x){
    BX[z][y][x]-=EY[z][y][x]+...;
    BY[z][y][x]-=...;
    BZ[z][y][x]-=...; }
  for(z) for(y) for(x){
    EX[z][y][x]+=...;
    EY[z][y][x]+=...;
    EZ[z]y][x]+=...; }
}
```

のボディには、合計 12 本のストリーム（電磁場ベクトル 3 次元配列の x 方向への連続参照）が存在するが、12 本というさほど大きくない数であってもレジスタの不足（特に配列要素アドレス計算のための汎用レジスタの不足）などにより、コンパイラによる SIMD 化が阻害されることが、予備的な調査によって明らかになっていた。一方 loop-splitting によって特定の磁場ベクトル成分のみを更新する 3 つのループに分割すれば、各ループのストリーム数が 5 本に減少して SIMD 化を実施できるが、ループ間で共通に参照される要素に関する時間的局所性が失われ、SIMD 化の効果が相殺されることも見出していた。そこで loop-splitting と strip-mining を組み合わせて

```
for(z) for(y){
  for(x) BX[z][y][x]+=...;
  for(x) BY[z][y][x]+=...;
  for(x) BZ[z][y][x]+=...;
}
```

のように変換すると、SIMD 化と参照局所性の両立により高い性能が得られる。

上記のような変換は、幾何マルチグリッド法のガウスザイデルスムーザのループに含まれる逐次性の分離や、間接参照・条件節などの SIMD 化阻害要因の除去にも有効であり、種々のアプリケーションに対する共通技術として適用することを検討した。

(3) 通信遅延の削減・隠蔽

FDTD 法などの実空間上の物理場を扱う数値シミュレーションにおける領域分割プロセス並列化では、部分領域間で境界要素を交換する通信の遅延隠蔽を、図 2 左に示すように境界要素の計算①を先行して行い、その通信と並行して内部要素の計算②を行うことで実現する方法がしばしば用いられる。しかしこの方法は、境界要素の計算対象がメモリ上で連続しないため参照局所性が劣悪で SIMD 化

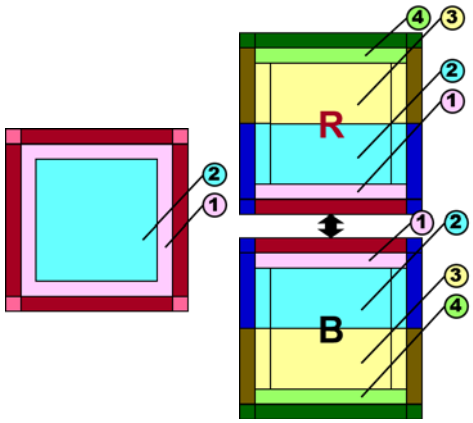

図 2: 境界要素交換通信の遅延隠蔽

も困難である。また部分領域の頂点や辺（3 次元の場合）の要素を交換するために、$3d-1$ 個（3 次元では 26 個）のプロセスとの通信が必要である。

そこで図 2 右に示すように、部分領域の計算を①～④の 4 フェーズ（d+2 フェーズ）に分割し、各々のフェーズが終了した時点で境界要素の通信を行うことで、計算対象をメモリ上で連続させるとともに、部分領域の $1/d$ 以上の計算と境界要素通信を並行させて遅延隠蔽を図る方法を考案している。この方法では、問題空間の特定の座標軸（図では y 軸）の方向で隣接するプロセスが red/black pair となり、この座標軸に関する計算順序をペア間で反転させることで、あるフェーズで計算される隣接プロセスが必要とする境界要素の通信時間を、他のフェーズの計算時間と重ね合わせることで、通信遅延が隠蔽される。またこの方法は、異なる座標軸方向の通信をシリアライズすることで（図の R プロセスでは、底辺、左右辺、上辺の順）、頂点（および 3 次元の場合は辺）の通信が不要となって通信相手のプロセスが $2d$ 個に限定されるメリットもある。

さらにメニーコアプロセッサでは、各コアの性能がマルチコアプロセッサに比べてかなり劣るため、通信回数自体の削減や、通信に関与するコアの数を適切に定める必要もあり、重要な研究テーマとして取り組んだ。

４．研究成果

3 節で示した各研究項目について、以下の成果が得られた。

(1) cache-aware コード生成

マルチスレッド化された 3 次元ステンシル計算の時空間タイリングを対象として、まず直方体形状のタイルを均等に分割して各スレッドに割り当てる方法と、タイル形状や分割方法の準最適解をパラメータチューニングにより見出す方法を開発した（学会発表②）。この研究では、チューニングの対象となるパラメータ空間が膨大であるだけでなく細かい起伏が極めて多数存在し、大局的な探索だけでは最適解が見出し難いことを明らかにし、局

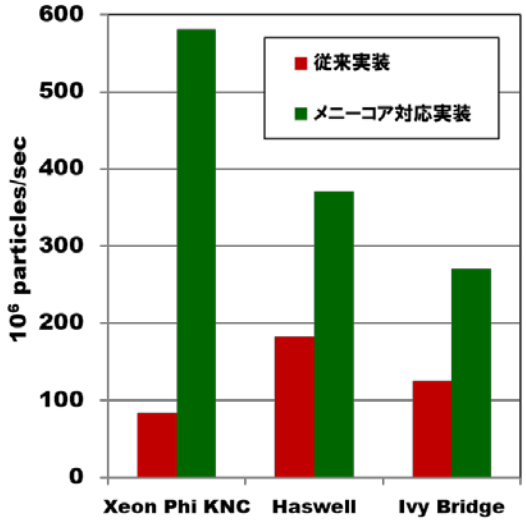

図 3: PIC 法の SIMD 化ループ性能

所的なモンテカルロ探索を組み合わせることの有効性を示した。

この初期の研究では、タイルを直方体形状に固定し、マルチスレッド並列化はタイル内部の計算に限定する方法を用いていた。しかしこの方法では、各スレッドが担当する計算領域（スレッドタイル）の形状に強い制約が生じることと、スレッド間の同期がタイムステップごとに必要であることの、二つの問題が明らかになった。前者については、スレッドタイルの最適形状を定めた上で、それを 3 次元的に組み合わせて直方体とは限らないタイルを構成する方法を考案し、その有効性を確かめた。また後者については、ダイヤモンドタイルと呼ばれるタイルサイズをタイムステップごとに伸縮させ、タイル間の並列性を利用可能とする方法を評価した（学会発表⑯、⑰）。

また計算対象空間を分割し、各部分領域に関する計算を時間方向に連続して行うという考え方は、マルチグリッド法における新たな乗法シュワルツ型スムーザの提案の基礎となっている（雑誌論文②、学会発表①）。この研究では、n 回の反復スムージングによる収束性改善が n 倍とはならないという問題を、部分領域に対する反復スムージングではキャッシュを活用することでコストが n 倍を大きく下回ることで解決し、4~5 回の反復が有効であることを示した。またより一般的な cache-aware なコードとして、後述する cache-aware な loop splitting の他、PIC 法で粒子集合の配列内シフト操作を、in-place かつマルチスレッド並列化と SIMD ベクトル化が可能にする技法を考案し、キャッシュの活用によりシフト操作の時間が約 2/3 に短縮できることを示した（学会発表⑩、⑬）。

(2) SIMD 化ループ変換

PIC 法の主要カーネルループに対して、ループ内で共通参照される配列要素を明示的にスカラー変数に置いて間接参照を回避する最適化と、それに伴うレジスタ割付の困難さの

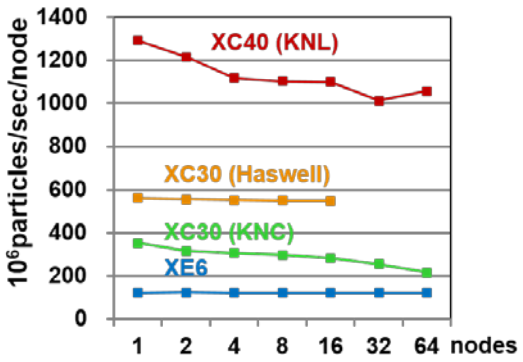

図 4: PIC 法のマルチプロセス性能

回避、および条件節の排除を目的として、粒子配列への参照が 1 次キャッシュに収まるようにした cache-aware な loop splitting（4 分割）を実施した。また分割されたループの一つでは、粒子の移動特性に着目した効率的かつ cache-aware な on-the-fly sorting を行い、カーネルループ全体の効率向上を図った。このループ変換の効果を、Xeon Phi 5120D (KNC)を用いて評価した結果、約 8 倍の性能向上が得られることを確認した（図 3, 雑誌論文①、学会発表③、④、⑤）。またこの最適化と、動的負荷分散を伴うマルチプロセス実装に必要な粒子集合管理ライブラリとを組み合わせて、最大 4096 コアの大規模並列シミュレーションを実施した結果、Xeon Phi 7250 (KNL)から構成されたスーパーコンピュータ Cray XC40 で 67.67×10^9 粒子／秒という極めて高い性能を達成するなど、ループ変換が実用的に優れた効果を発揮することを確認した（図 4、学会発表⑨、⑩、⑪、⑬、⑭、⑮）。

また線型方程式ソルバーのメニーコア対応実装方式について、マルチグリッド法のガウスザイデルスムーザ、ICCG 法などクリロフ部分空間法の IC 分解前処理、および一般的に用いられる疎行列ベクトル積を対象として、効率的な SIMD ベクトル化ループの生成法を検討した。マルチグリッド法については、ガウスザイデルスムーザに本質的に内在する逐次性を持つ近似解更新操作を、他の近似解更新操作と分離する cache-aware な loop splitting を行うことで、効率的な SIMD ベクトル化が実施できることを示した（雑誌論文②、学会発表①）。IC 分解前処理については、分解処理により得られる前処理行列の非ゼロ要素が、係数行列の非ゼロ要素に対応する要素だけでなく、非ゼロ要素を含む一定サイズのブロック中にも出現することを許容して収束性を向上しつつ、非ゼロ要素の増加に伴う演算オーバヘッドを SIMD 機構の活用で抑制する方法を提案し、KNC を用いた評価によりその有効性を実証した（学会発表⑥、⑦）。疎行列ベクトル積については、種々の行列表現形式に対する SIMD ベクトル化ループの性能を評価し、diagonal 形式を対象に cache-aware な tiling を施したものが、一般的な CRS 形式を対象としたものに比べて 3~4 倍高速であることを、KNL を用いた評価によっ

て示した（学会発表⑱）。

(3) 通信遅延の削減・隠蔽

3 節(3)で示した境界要素交換通信の遅延隠蔽法が、時空間タイリングを施したステンシル計算や、後述する粒子移送通信回数の削減を行った PIC 法シミュレーションのように、通信対象の境界要素の幅が大きい場合に有効であることを確認した。またこの評価の過程で、メニーコアプロセッサの 1 コアの性能が、ノード間通信のバンド幅を埋めるには不十分であり、たとえば KNL ではプロセッサあたり 8 プロセス以上の構成として、多数のコアを MPI 通信に関与させることが必要であることを明らかにした（学会発表⑩、⑬）。

このようにプロセッサあたりのプロセス数を大きくすることは、大規模並列計算では一般に得策ではなく、特に PIC 法シミュレーションのように種々の通信を行うアプリケーションでは並列性能に対する影響が大きい。そこで従来のマルチコア実装でも用いている、粒子移送について領域境界を拡大する手法を、メニーコア対応の粒子管理と整合させるためのアルゴリズム考案し、粒子管理ライブラリへの組み込みを実施した。またこれらの通信遅延削減・隠蔽を実施した結果、従来のマルチコア実装では小さかった大域的縮約通信のオーバヘッドが相対的に大きくなり、このコスト削減や遅延隠蔽が今後の重要な課題であることを示した（学会発表⑩、⑬）。

５．主な発表論文等

〔雑誌論文〕（計 2 件）

① Hiroshi Nakashima. Manycore Challenge in Particle-in-Cell Simulation: How to Exploit 1 TFlops Peak Performance for Simulation Codes with Irregular Computation, Computers and Electrical Engineering, 44, 81-94, 2015. 10.1016/j.compeleceng.2015.03.010

② Masatoshi Kawai, Takeshi Iwashita, and Hiroshi Nakashima. SIMD Implementation of a Multiplicative Schwarz Smoother for a Multigrid Poisson Solver on an Intel Xeon Phi Coprocessor, LNCS 8969, 57-65, 2015.

〔学会発表〕（計 17 件）

① Masatoshi Kawai, Takeshi Iwashita, Hiroshi Nakashima. Implementation of a Multiplicative Schwarz Smoother for a Multigrid Poisson Solver on an Intel Xeon Phi Coprocessor, Intl. Mtng. High-Performance Computing for Computational Science (VECPAR), 2014.

② Takeshi Minami, Motoharu Hibino, Tasuku Hiraishi, Takeshi Iwashita, Hiroshi Nakashima. Automatic Parameter Tuning of Three-Dimensional Tiled FDTD Kernel, Intl. WS. Automatic Performance Tuning (IWAPT), 2014.

③ Hiroshi Nakashima. PIC Simulation in Many-Core Era. Forum on Advanced Scientific Computing, 2014.

④ Hiroshi Nakashima. Manycore Challenge in Kyoto: What We Learned from HPC Programming with KNC, Intl. WS. Enhancing Parallel Scientific Applications with Accelerated HPC (ESAA), 2014.

⑤ Hiroshi Nakashima. Manycore Challenge in Kyoto: What We Learned from HPC Programming with Intel Xeon Phi, Intel HPC Developer Conf., 2014.

⑥ Takeshi Iwashita, Naokazu Takemura, Hiroshi Nakashima. A Fill-In Strategy for Fast ICCG Solver with SIMD Vectorization, Annual Meeting on Advanced Computing System and Infrastructure (ACSI), 2015.

⑦ Takeshi Iwashita, Naokazu Takemura, Akihiro Ida, Hiroshi Nakashima. A New Fill-In Strategy for IC Factorization Preconditioning Considering SIMD Instructions, Proc. Intl. Symp. Parallel and Distributed Processing with Applications (ISPA), 37-44, 2015.

⑧ Yuto Kato, Yoshiharu Omura, Yohei Miyake, Hideyuki Usui, Hiroshi Nakashima. Dependencies of the Generation Process of Whistler-Mode Emissions on Temperature Anisotropy of Energetic Electrons in the Earth's Inner Magnetosphere. URSI-Japan Radio Science Meeting, 2015.

⑨ 木倉佳祐, 三宅洋平, 臼井英之, 中島浩. プラズマ粒子シミュレーションのメニーコアプロセッサ向け最適化手法の探求, ハイパフォーマンスコンピューティングと計算科学シンポジウム, 2015.

⑩ Hiroshi Nakashima, Keisuke Kikura, Yohei Miyake. Prototype Implementation and Its Fundamental Performance Evaluation of a Manycore-Aware Oh-Help'ed PIC Simulation Code, IPSJ SIG Notes, 2016-HPC-153-15, 1--10, 2016.

⑪ Hiroshi Nakashima. Regularity: A New Important Player in the Game of High-Performance Simulations in Manycore Era, Intl. Conf. Simulation Technology (JSST), 2016.

⑫ Yuto Kato, Yoshiharu Omura, Yohei Miyake, Hiroshi Nakashima, Hideyuki Usui, Keiichiro Fukazawa. Electron Hybrid Code Simulations with OhHelp Load Balancer for the Study of Relativistic Electron Acceleration in Planetary Magnetospheres, Intl. Conf. Simulation Technology (JSST), 2016.

⑬ Hiroshi Nakashima, Yoshiki Summura,

Keisuke Kikura, Yohei Miyake. Large Scale Manycore-Aware PIC Simulation with Efficient Particle Binning, Intl. Parallel and Distributed Processing Symp. (IPDPS), 202-212, 2017.
⑭ 三宅洋平, 木倉佳祐, 寸村良樹, 中島浩. メニーコア型スーパーコンピュータ向けプラズマ粒子シミュレーション高効率実装の検討, 第 21 回計算工学講演会, 2016.
⑮ 三宅洋平, 寸村良樹, 木倉佳祐, 中島浩. メニーコア型スーパーコンピュータ向けプラズマ粒子計算手法の研究, ハイパフォーマンスコンピューティングと計算科学シンポジウム, 2016.
⑯ 深谷猛, 岩下武史. マルチコア・メニーコア環境における反復型ステンシル計算と時空間タイリング, 日本応用数理学会 2016 年度年会, 2016.
⑰ 深谷猛, 岩下武史. 反復型ステンシル計算のマルチコア・メニーコア向け実装に関する考察, 日本応用数理学会「行列・固有値問題の解法とその応用」研究部会, 2016.
⑱ 中島浩. メニーコア時代のプログラミング, PC クラスタワークショップ, 2017.

〔その他〕
ホームページ等
① OhHelp ライブラリ公開ページ, http://www.para.media.kyoto-u.ac.jp/ohhelp/

６．研究組織
(1) 研究代表者
中島　浩（NAKASHIMA HIROSHI）
京都大学・学術情報メディアセンター・教授
研究者番号：10243057

(2) 研究分担者
岩下　武史（IWASHITA TAKESHI）
北海道大学・情報基盤センター・教授
研究者番号：30324685

(3) 連携研究者
平石　拓（HIRAISHI TASUKU）
京都大学・学術情報メディアセンター・助教
研究者番号：60528222