様式 C-19

科学研究費補助金研究成果報告書

平成２３年　５月３１日現在

機関番号：１２６１２
研究種目：基盤研究（C）
研究期間：２００８年度～２０１０年度
課題番号：２０５００００２９
研究課題名（和文）多様な形態の密結合マルチコアアーキテクチャ向け並列プログラミングシステム
研究課題名（英文）A parallel programming system for various tightly coupled multi-core architectures
研究代表者
岩崎　英哉（IWASAKI HIDEYA）
電気通信大学・大学院情報理工学研究科・教授
研究者番号：９０２０３３７２

研究成果の概要（和文）：　本研究では，並列処理に慣れていない一般の利用者も，マルチコアからメニーコアというプロセッサ構成に向けての技術進歩の恩恵にさずかることを可能とする並列プログラミングシステムを構築し，以下の成果を得た。(1) 行列や要素数の変化するリストを提供する並列スケルトンライブラリを構築した。(2) GPGPU 向けに並列スケルトンに基づくフレームワークを開発した。(3) 従来の並列化コンパイラで十分に並列化できなかったリダクションを，半環上の行列乗算により定式化し自動並列化を達成した。(4) Java による高性能な Software Transactional Memory ライブラリおよび同期機構の静的選択システムを構築した。

研究成果の概要（英文）：In this research, we have developed parallel programming systems that enable inexperienced users of parallel programming to enjoy the benefits of today's multi-core and many-core architectures. The obtained results can be summarized as follows. (1) We have developed a parallel skeleton library that offers matrices and variable-length lists. (2) We have developed a skeletal parallel framework for GPGPU programming. (3) We have developed a general and powerful framework for automatic parallelization based on matrix multiplication over a semiring. (4) We have developed a system that enables the user to statically select a suitable synchronization mechanism from a lock or a software transactional memory.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
２００８年度	1,700,000	510,000	2,210,000
２００９年度	1,000,000	300,000	1,300,000
２０１０年度	800,000	240,000	1,040,000
年度			
年度			
総　計	3,500,000	1,050,000	4,550,000

研究分野：　計算機科学
科研費の分科・細目：　情報学・ソフトウェア
キーワード：　並列プログラミングライブラリ，並列スケルトン，マルチコア，自動並列化，同期機構

１．研究開始当初の背景

近年，ひとつの CPU の内部に複数の CPU コアを搭載させるマルチコア技術が一般的に利用可能となり，同一の CPU コアを搭載したホモジニアスマルチコアに加え，異なる CPU コアを実装したヘテロジニアスマルチコアも出現してきた．この流れは，ひとつの CPU に多数のコアを搭載するメニーコアへと向かっており，プロセッサ構成形態は多様化してきている．

このような背景のもと，ヘテロジニアスマルチコア，あるいはメニーコアという新しいアーキテクチャの能力を最大限に生かすような，並列処理ソフトウェア技術が求められている．

２．研究の目的

本研究は，並列処理プログラムに慣れていない一般の利用者も，上で述べたようなマルチコアあるいはメニーコアといった技術進歩の恩恵に浴することを可能とするような並列プログラミングシステムの構築を目的とする．

この目的を達成するために，代表者が行ってきた並列スケルトンライブラリ，言語処理系の実行時システムに関する研究と，分担者が行ってきたコンパイラ・インフラストラクチャにおける最適化の研究を有機的に結びつけ，コンパイラ内部における最適化，プログラム変換による最適化，有効な実行時システムの構築などを通して，目的を達成することを目指す．

３．研究の方法

本研究を進めるにあたって重要な設計上の選択は以下の２点である．

ひとつは，スケルトン並列プログラミングの考え方を基礎としてシステムを設計するという点である．スケルトン並列スケルトンでは，典型的な並列処理パターン，たとえば，すべてのデータに同一の関数を適用する map，データ間に二項演算子を挟んで計算する reduce，reduce の途中結果も残す scan のような「並列スケルトン」と呼ばれる基本関数を組み合わせてプログラムを作成する．各並列スケルトンは，目的とする並列構造を抽象化しているので，利用者はその内部の詳細を知る必要はない．その結果，並列スケルトンを用いれば，逐次プログラムを書く感覚で並列プログラムを開発することができる．この並列スケルトンの考え方は，本研究においてもきわめて有効であると考えられる．

スケルトン並列プログラムは「既製品」である並列スケルトンを組み合わせてプログラムを記述する関係上，そのまま実行したのでは必ずしも実行効率が良いとは限らないという弱点がある．これを克服するためには，並列スケルトンの連続した呼び出しを一つにまとめる最適化（「融合変換」という）が極めて重要である．本研究では，融合変換の必要性を念頭において並列スケルトンの考えを取り入れる．

設計上の選択の２点目は，研究分担者が開発に深くかかわったコンパイラ・インフラストラクチャ COINS を積極的に活用するという点である．COINS は，その内部中間表現などの内部構造が公開されており，コンパイラに手を加えるような作業を比較的容易に行うことができるという利点を備えている．この利点を最大限に生かし，システムの実装の手間を軽減させることとした．

対象とするアーキテクチャは，一般的な PC で用いられているホモジニアスマルチコア，ヘテロジニアスマルチコアとして GPU および Cell Broadband Engine とする。

４．研究成果

本研究で得られた成果を，以下にまとめる．

(1) 研究代表者が開発中の並列スケルトンライブラリ SkeTo (Skeletons in Tokyo) の機能を，マルチコア環境における利用を見据えて以下のように拡張し効果を確認した．

① 行列用スケルトン

行列用(二次元配列)スケルトンに関して，マルチコアアーキテクチャの特性を活かすように再設計を行い，実装を行った．

具体的には，内部的にはタスク並列的な実装に基づき，全体の計算をタスクに分割し，ノードにタスクを分散させる．タスクを受け取ったノードは，さらにこれを細かく分割（コアタスクと呼ぶ）し，これをコアに分散させる．この仕組みを動的に行うことにより，ノード間の負荷分散およびコア間の負荷分散が適切に行われるようにする．

さらに，与えられたスケルトン計算を即座には行わず，値が必要となった時点で計算をまとめて実行することにより，融合変換を実現している．以上の仕組みは，C++言語のテンプレート機能によるプログラミング技法を駆使して実装を行った．いくつかの実験により，以上のような仕組みには，一定の効果があることを確認した．

この成果は，並列処理に関する国際会議 ICPP 2009 において論文が採択され発表を行った．

② 可変長リスト用スケルトン

SkeTo が扱うことのできる新しいデータ型として可変長リストを実現し，可変長リスト用スケルトンを実装した．

可変長リストとは，計算中に大きさが伸縮するリストを指す．このような可変長リストに関しては，既存の固定長リストのスケルトンをもとにして，可変長に対応させるようなデータ構造の拡充を行った．さらに可変長リストを必要とするような問題をいくつかのタイプに分類し，それぞれの問題を扱うプログラムを記述するためには，ライブラリがどのようなスケルトンを用意すべきかを検討した．

これらの検討の結果，新たに，concatmap スケルトン（リストの各要素へ関数を適用し，その結果生じた複数のリストを平らにしてつなげる），filter スケルトン（リストから

ある条件を満足する要素だけを残す), リストの先頭・末尾位置における要素の追加・削除を行う操作, リストを連結する操作を API として実現した. その結果, 固定長リストだけでは記述することが難しかった問題, たとえば双子素数問題, 凸包問題, マンデルブロー集合を求める問題などを解くプログラムを, 効率的に記述し実行できるようになった.

この成果は, 国際会議 Euro-Par 2009 において論文が採択され発表を行った.

(2) 並列スケルトンの考え方を, メニーコアの一種である GPU (Graphics Processing Unit) を汎用計算に利用する GPGPU (General Purpose GPU) 向けのプログラミングに応用し, GPGPU プログラミングの難しさを隠蔽するようなシステムを設計し実装した.

このシステムにおいて, 利用者は並列性を意識せずに, 本システムが提供するスケルトンを用いて C 言語プログラムを記述する. 本システムは, 利用者が記述したこのプログラムから, GPU 向けに最適化された CUDA コードを出力する. ここで CUDA とは, GPU 上で作動するプログラムのために標準的に用いられている言語である. 利用者は出力された CUDA コードを CUDA のコンパイラでコンパイルすることにより, GPU 上で動く並列プログラムを得ることができる.

このシステムを利用することにより, 利用者は, GPU というハードウェアを十分に考慮した高い並列性を持つプログラムを直接記述しなくてもすむようになる.

本システムの大きな特長の一つは, 提供する並列スケルトンに対する融合変換の機能を備えていることである. 融合変換は, 本システムによって生成されたプログラムの実行効率を大きく改善することに役立っている.

本システムは COINS を用いて実装した. COINS は, 利用者が記述したプログラムを読み込み, 内部中間表現(HIR)をメモリ中に作成する. 融合変換は HIR から HIR への変換系として実装し, 最終的には HIR から CUDA ソースコードへの出力系により CUDA ソースを出力する.

この成果は, プログラミング言語とシステムに関するアジアシンポジウム APLAS 2009 において論文が採択され発表を行った.

(3) 従来の並列化コンパイラにおける doall 並列化では十分に扱うことのできなかったリダクション (総和計算の一般化で並列スケルトンにおいては reduce, scan に相当する) を, 半環上の行列乗算を用いて定式化した。

この定式化に従うと, 並列化の対象であるループ本体は, 半環を構成するふたつの二項演算子 (＋と×, 最大値をとる Max 演算子と＋など) を適切に定めると, その半環の上の行列乗算の形で表現できる場合がある。行列乗算は結合的なので, 行列乗算の形を導出することができさえすれば, ループ全体は木構造への分割統治の形で並列に効率よく計算することができる。

ところがこれを実際に実システムとしてコンパイラに実現しようとすると, いくつかの困難な点に遭遇する。最大の問題点は, コンパイラが対象とする (たとえばC言語の) プログラムは, 必ずしもきれいな形で記述されているとは限らず, 場合によっては入れ子になった if 文による複雑な分岐などがあり, 上の定式化された形に持ち込むことを困難としている点である。特に, Max は言語組み込みの演算子として用意されていることはほとんどないため, 最大値をとる計算は必然的に if 文を伴う。このため, Max と＋による半環上の行列乗算形を得るためには, if 文の入れ子から Max 演算子を抽出することが必要になる。

本研究では, Max 演算子抽出の問題を充足可能性判定問題に変換し, 既存の SMT ソルバを利用して Max 演算子を抽出する手法を提案した。

本論文での定式化は, COINS を利用してCコンパイラに実装した。SMT ソルバとしては Yices を利用した。このコンパイラは OpenMP を利用する並列Cプログラムを出力する。その結果, 従来はできなかった並列化を達成することができ, 実際に並列効果も確認した。

本研究は, 数学的な定式化と, 並列化コンパイラの実装との間にあるギャップを埋めることに成功したという点で注目に値すると考えている。

本研究の成果は, プログラミング言語の設計と実装に関する国際会議 PLDI 2011 において論文が採択された。会議の開催は 2011 年 6 月である。

(4) 並行プログラミングにおける排他制御機構のロックと Software Transactional Memory (STM) を, 同一の Java ソースプログラムから適切に選択し使い分けることを可能とするシステムを, アスペクト指向プログラミングを利用して実現した。

排他制御機構としては, 一般的にはロックによる同期が広く用いられている。しかし, 同期の記述がアプリケーションロジック中に混在するという記述形式の問題点, および, 高いスループットを得るためにはクリティカルセクションを細かく指定する必要があるという性能の問題点がある。STM は主に後者の問題点の解決法として提案されているが, 競合が頻繁に起こる場合の性能低下が大きいため, ロックと STM を適切に使い分ける

のが好ましい。本研究では，アプリケーションごとに適切な同期機構を静的に選択するシステム SAW を提案した。

提案した機構では，上の二つの問題点を次のように解決する。記述形式の問題点に関しては，利用者は二つのアノテーションを Java のソースプログラム中に指定する。ひとつは排他的に実行する部分（クリティカルセクション），もうひとつは複数のスレッドにより共有されるオブジェクトである。これらにより直感的で簡潔な記述を可能としている。性能の問題点に関しては，STM を導入し，ロックと STM からより適している同期機構を利用者が選べるようにすることで解決する。

SAW は，利用者による同期の記述に基づき，利用者が指定した同期機構用のコードをアスペクトとして自動生成する。生成されたコードは標準的な AspectJ コンパイラによりロジックの中に埋め込まれる。

SAW が有効であるためには，性能の高い STM ライブラリを利用することが必須である。そうでないと常にロックを選択する方が性能が良いということになってしまう。そこで本研究では，SAW からの利用に適した STM ライブラリである AL-STM（Aspect-led STM）を実装した。AL-STM は利用者がその API を直接利用するのではなく，自動生成するアスペクトでだけ利用することを前提としている。そのため，低レベルで無駄のない API だけから構成できる。

SAW の評価は記述コストと性能の両面から行った。その結果，十分簡潔な記述が可能であることが確認できた。また適切な同期機構はアプリケーションの特徴やスレッド数によって決まることを確認し，ソースコードを変更せずに容易に同期機構を選択できる SAW の有効性も示した。

本研究の成果は，国際会議への投稿の準備中である。

(5) その他の成果としては，以下のものが挙げられる。

① COINS の高水準中間表現を利用したベクトル化の試験実装を行い，ヘテロジニアスマルチコアである Cell Broadband Engine 向けの，イントリンシックルーチンを含む C 言語ソースを出力できるようにした。さらに，特別なデータ構造を有する配列の処理の高速化のための，処理前と処理後のデータ変換や処理コードの自動変換について検討した。

② 共通鍵暗号方式の一つである AES (Advanced Encryption Standard) 暗号の処理を題材として，その高速化と省電力化の両立を目指す SIMD (Single Instruction Multiple Data-Stream) 命令セットを提案した。さらにシミュレーションによりその効果を確認した。本研究の成果は，現在論文誌に投稿中である。

③ ②の命令セット向けのコンパイラ最適化を検討し，実装を開始した。そのための技術的課題は，②の命令セットが対象とするデータ構造やオペレーションに合致するプログラムコードを，最適化の対象とするプログラムから発見し，適切なプログラム変換を施すことであり，そのために①で得た技法を応用する予定である。

④ ヘテロジニアスマルチコアは，複数のコアが均一でないため，プログラミングのコストが増大している。この研究では対象を Cell Broadband Engine における DMA 転送処理に焦点を絞り，プログラマによる DMA 転送の記述の負担を軽減するようなライブラリを作成した。このライブラリでは，DMA 転送処理に関する記述を隠蔽し，さらに DMA ダブルバッファリングにより転送処理を自動的に効率化する。

５．主な発表論文等

〔雑誌論文〕（計６件）

① Sato, S., Iwasaki, H.: Automatic Parallelization via Matrix Multiplication, Proc. 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2011), 2011, 採録済, 査読有

② 山田佑二, 鵜川始陽, 岩崎英哉: Java における適切な同期機構の選択システム, 第 13 回プログラミングおよびプログラミング言語ワークショップ論文集 (PPL 2011), pp.1-15, 2011, 査読有

③ Sato, S., Iwasaki, H.: A Skeletal Parallel Framework with Fusion Optimizer for GPGPU Programming, Proc. 7th Asian Symposium on Programming Languages and Systems (APLAS 2009), Lecture Notes in Computer Science 5904, Springer-Verlag, pp.79-94, 2009, 査読有

④ Karasawa, Y., Iwasaki, H.: A Parallel Skeleton Library for Multi-core Clusters, Proc. 38th International Conference on Parallel Processing (ICPP 2009), pp.84-91, 2009, 査読有

⑤ Tanno, H., Iwasaki, H.: Parallel Skeletons for Variable-length Lists in SkeTo Skeleton Library, Proc. 15th International Euro-Par Conference (Euro-Par 2009), Lecture Notes in Computer Science 5704, Springer-Verlag, pp.666-677, 2009, 査読有

⑥ 岩崎英哉, 胡振江: 並列計算パターン（スケルトン）による並列プログラミング, 情報処理, Vol.49, No.12, pp.1385-1394, 2008, 査読有

〔学会発表〕（計３件）

① Sato, S., Iwasaki, H.：Parallelization via Matrix Multiplication, 8th Asian Symposium on Programming Languages and Systems (APLAS 2009), poster session, 30 November 2010, Shanghai

② 安仁屋宗石，黒田和宏，鈴木貢：AES 暗号を高速化する SIMD 拡張命令セット，第12回IEEE広島支部学生シンポジウム，2010年11月6，7日，島根大学

③ 山田佑二，鵜川始陽，岩崎英哉：アスペクト指向に基づく適切な同期機構の選択システム，2010年並列/分散/協調処理に関する『金沢』サマー・ワークショップ，2010年８月５日，金沢

④ Sato, S., Iwasaki, H.: A Skeletal Framework with Fusion Optimizer for Rapid GPGPU Application Development，日本ソフトウェア科学会第２６回大会講演論文集，2009年９月17日，島根大学

〔その他〕
SkeTo ホームページ
http://www.ipl.t.u-tokyo.ac.jp/sketo

６．研究組織
(1)研究代表者
岩崎　英哉（IWASAKI HIDEYA）
電気通信大学・大学院情報理工学研究科
・教授
研究者番号：90203372

(2)研究分担者
鈴木　貢（SUZUKI MITSUGU）
島根大学・総合理工学部・准教授
研究者番号：50272753