科学研究費助成事業　　研究成果報告書

平成 ２６ 年　 ６ 月　 ６ 日現在

機関番号：１７１０４
研究種目：基盤研究(B)
研究期間：2009～2013
課題番号：２１３００００８

研究課題名（和文）安全な計算状態操作機構の実用化

研究課題名（英文）Enhancing and Exploiting a Language Mechanism for Legitimate Manipulation of Execution States

研究代表者
八杉　昌宏（YASUGI, Masahiro）

九州工業大学・大学院情報工学研究院・教授

研究者番号：３０２７３７５９

交付決定額（研究期間全体）：（直接経費）　　13,800,000円、（間接経費）　　4,140,000円

研究成果の概要（和文）：安全な計算状態操作機構L-closureにより、平常時の高い実行性能の元で、計算中のソフトウェアの動的再構成・保全の機能を持つ言語処理系が実装可能となる。本研究では、真の末尾再帰の新しい高効率実装において提案機構を応用した。また、提案機構を応用した負荷分散により並列システムにおける資源を有効活用する手法の提案と評価を行った。実装面からは、L-closureの償却呼出しコストを低減した。コンパイラ実装においてはx86-64をサポートし、短時間しか実行されない関数におけるレジスタ割り当ての効果を評価した。

研究成果の概要（英文）：We can implement language systems with the functionality of dynamic maintenance of software execution by using a language mechanism (called L-closures) which enables legitimate manipulation of execution states. In this study, we applied the proposed mechanism to the efficient implementation of proper tail recursion. We also proposed and evaluated various schemes for exploiting available resources in parallel systems by applying the proposed mechanism to dynamic load balancing. Using a new implementation of the proposed mechanism, we reduced amortized invocation costs of L-closures. We also supported x86-64 in the compiler implementation, and evaluated the effect of register allocation for short functions.

研究分野：総合領域

科研費の分科・細目：情報学・ソフトウエア

キーワード：プログラミング言語処理系　プログラミング言語　計算機システム　並列処理　ディペンダブル・コンピューティング　ハイパフォーマンス・コンピューティング　性能評価　負荷分散

１．研究開始当初の背景

計算システムは十分な信頼性・性能・容量が求められるのに加えて、最近は消費電力量の削減も求められる。並列化率向上によりクロック周波数向上と同じ性能が得られれば消費電力量の大幅な削減が可能となるため、マルチコア、計算機クラスタ、グリッド計算など、さまざまなレベルで並列システムの提供が拡大している。そのような多様化・複雑化する計算システムを、そのままの形で効率よく簡単・安全に利用することは困難であり、適切に抽象化されたプログラミングインタフェースが求められている。そのようなプログラミングインタフェースを提供する高信頼・高性能プログラミング言語の処理系を実装するには、しばしば、「ごみ集め」など計算中のソフトウェアの動的再構成・保全の機能が必要となる。

我々は、これまでの安全な計算状態操作機構の研究で、拡張 C 言語の仕様として入れ子関数（クロージャ）を利用して呼出し元に眠る変数の値への安全で正式なアクセスを可能とし、負荷分散、ごみ集め、マイグレーション、チェックポインティングなど、計算中のソフトウェアの動的再構成や保全を記述可能とした。

実装技術としては、初期化や保存を呼出しまで遅延させることで、クロージャ生成コスト削減や、変数へのレジスタ割当てを可能とし、提案機構の追加を意識させない高い実行性能を得た。その研究成果は、メジャーな国際会議にて発表したアセンブリ言語レベルの技術を用いた実装と、標準 C 言語への翻訳方式での実装として結実している。

２．研究の目的

(1) 本研究では、これまでの研究成果を発展させ、計算状態操作機構の実用化を広く進めることを目的とする。具体的には、計算状態操作機構の応用技術に磨きをかけるとともに、設計・実装面から同機構を確立・改良する。

(2) より具体的には、ソフトウェアの動的再構成・保全の面からは、ごみ集めや一級継続や真の末尾再帰がサポートされた高水準言語の実装、負荷分散やマイグレーションにより並列システムにおける資源を有効活用する手法の開発、などがあり、設計・実装面からは、計算状態操作機構を持つ拡張 C 言語の言語仕様や実装・性能モデルの改良、言語仕様の普及活動、他言語での計算状態操作機構の設計、アセンブリ言語レベルのコンパイラ実装においては急速に普及している x86-64 (AMD64/Intel64)命令セットアーキテクチャのサポート、などが挙げられる。

３．研究の方法

本研究は、安全な計算状態操作機構を基軸にさまざまな実用化を進めるため、ある程度独立に進められる個別研究課題の集合となっていた。このため、適宜、順序を入れ替えたり、新たな個別研究課題を追加したりすることで、常に有益な研究を続けるよう対応できた。

高水準言語の実行ステップ（セマンティクス）に関する一貫性・頑健性・時間効率・空間効率を冗長性・後戻り・再実行・予測近似といった手段により実現したいという点が本研究の本質的な動機であった。その際に、「隠された計算状態」が存在すると抜本的再構成・保全が不可能となるが、そのような「隠された計算状態」の存在を防ぐために計算状態操作機構を利用するというのが本研究に共通する手法である。

(1) 提案機構の応用

提案機構の応用では、既存の言語処理系に対して提案機構を用いた新しい実装手法を考案して適用することや新規に言語処理系を開発することにより研究を進めた。また、実アプリケーションの開発などを行い、これらの言語処理系ならびにその改善を評価した。

(2) 提案機構の改良

提案機構には、コンパイラ実装と翻訳系による実装があった。コンパイラ実装では対応する命令セットアーキテクチャの拡大、元にする GCC のバージョン更新への対応、適正な性能評価システムの作成などにより研究を進めた。また、翻訳系による実装は、GCC のバージョン更新の制約を受けないため、計算状態操作機構 L-closure の複数回の償却呼出しコストを低減させる方式の開発は翻訳系による実装を改良する形で進めた。

４．研究成果

(1) 真の末尾再帰の新しい高効率実装における提案機構の応用

ごみ集めや真の末尾再帰がサポートされた高水準言語の実装において、有限のスタックを用いて正規化を遅延させるために提案機構を応用した。

Scheme 処理系は真に末尾再帰的であることが要求されており、アクティブな末尾呼び出しの数に制限がない場合もサポートしなくてはならない。Clinger は真の末尾再帰の形式的定義の１つを空間効率の点から与えており、その定義に従えば、末尾呼び出しの最適化（末尾呼び出しをトランポリンなどによりジャンプに置き換えて実装する方法）だけでなく、Baker の CPS（継続渡しスタイル）変換を用いた C 言語における Scheme の実装手法も、真に末尾再帰的と分類できる。Baker の実装手法は、CPS 変換された末尾呼び出しにおいて新たな継続を生成せず、C 言語の実行スタックに対してもごみ集めを行うため、空間効率が良い。本研究では拡張 C 言語による真に末尾再帰的な Scheme インタプリタの実装手法を提案した。本手法は CPS 変換を用いず、C の実行スタックがあふれそうになれば、残りの計算に必要な“ Frame ”オブジェ

クトのみを含むリストとして表現された空間効率の良い一級継続を生成し、すぐさまその継続を呼び出すというアイディアに基づく。ごみ集めや継続のキャプチャにおいては、実行スタックに合法的にアクセスできる、つまりデータ構造や変数の値としてアクセスできる L-closure という言語機構を用いている。ベースとなる Scheme インタプリタは、Java アプリケーション組み込み用 Lisp ドライバである JAKLD をもとに C 言語で再実装されたものとした。

また、真の末尾再帰の理論的定義から、CPS 変換は必ずクロージャ変換を伴うべきと推奨するとともに、プログラム中の変数への値の代入に基づく環境を用いないインタプリタ実装でも有効な真の末尾再帰の新しい理論的定義を提案した。

(2) 並列システムにおける負荷分散への提案機構の応用

① 並列システムの高並列環境における評価

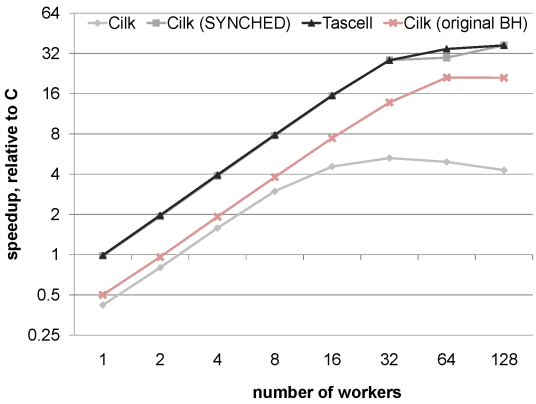
我々は不規則な問題、環境において粒度が大きくバランスのとれた負荷分散を可能にする並列プログラミング／実行フレームワーク Tascell を提案している。Tascell ワーカがタスクを要求した時に一時的にバックトラックを行うことで、並列化のコストを本質的に極小化した遅延分割型負荷分散を実現しており、例えば論理スレッドを利用する Cilk と比較しても優れた性能を示す。Tascell は複数の拠点に設置されたクラスタを WAN で接続した広域分散環境にも適用可能である。本研究では、高並列環境における Tascell の有効性を検証するために、Niagara 2 プロセッサ 2 台を備えたサーバ 128 ハードウェアスレッド環境、および InTrigger のうち最大 5 クラスタ 288 コアを用いた環境における評価を行った。その結果、メモリバンド幅や通信遅延、ワークスティーリングの偏り等の影響によるスケーラビリティの制約は観測されたものの、本フレームワークが広域分散環境を含む高並列環境においてもおおむね有効にはたらくことが確認できた。

② 並列システムの実アプリケーションへの適用：並列グラフ走査

ワークスティールに基づくメニーコア環境向け並列プログラミング／実行フレームワークは、不規則な並列アプリケーションにおいても効率よい負荷分散を可能とする。しかし、グラフに基づくデータ構造を辿るような並列プログラムはスタックオーバフローや極端な負荷の偏りを生じさせやすい。本研究では、確率的にバランスのとれた分割統治的グラフ走査ができる並列プログラムを開発した。オーバーフローする呼出しを、次の並列ステージ用に保存しておくプログラミング技法を用いた。Tascell と呼ぶワークスティールフレームワークにおいて、各ワーカが作業空間を長期間独占利用できるようなプログラミング技法を提案し、論理スレッドを利用する Cilk における同様の技法も導いた。

③ 並列システムの実アプリケーションへの適用：多体シミュレーション

我々は、多体シミュレーションに用いられる高速版 Barnes-Hut アルゴリズムを、近年提案している動的負荷分散に基づく並列プログラミング言語 Tascell を用いて並列化した。一般に動的負荷分散は不規則なアプリケーションの並列化に対して有効であり、Cilk や X10 などのマルチスレッド言語ではワークスティーリングに基づいて効率的な動的負荷分散を提供する。Barnes-Hut アルゴリズムは多体シミュレーションに広く使用されている。Barnes-Hut アルゴリズムには、単一の作業空間を保持し、それを逐次的に更新する特徴を持った、高速なアルゴリズムが存在する。同時実行可能な各論理スレッドの作業空間を注意深く管理しなければならないマルチスレッド言語とは異なり、論理スレッドフリーなワークスティーリングフレームワークである Tascell では、高速版のアルゴリズムを容易に並列化することができた。実際に 128 ハードウェアスレッド環境上で 100 万の質点に及ぼされる力を計算した結果、我々の Tascell プログラムは通常の手法で並列化された Cilk プログラムと比較して 6.95 倍の高速化を達成することができた。



④ 並列システムの実アプリケーションへの適用：グラフマイニング

我々は、各頂点がアイテムの集合を持つようなグラフから閾値以上の数の共通アイテム集合を持つ連結部分グラフを全抽出する並列アルゴリズム、および並列言語 Tascell による実装を提案した。このようなグラフマイニングはソーシャルネットワークや生体ネットワークの分析などに応用できる。効率良い逐次探索アルゴリズムとして、我々は既に COPINE を提案している。COPINE は探索空間を制限するために、同一の部分グラフを重複列挙しないようにする枝刈り、共通アイテム数による枝刈り、アイテム集合の包含関係を利用する枝刈りを導入している。このうち包含関係による枝刈りのために枝刈りテーブルを用いるが、並列探索において過剰な枝刈りを回避するためには、他ワーカが登録したテーブルエントリの参照に対してある種の制約を加えなければならない。この制約に注意して COPINE を拡張することにより並列

アルゴリズムを設計した。実装においては、他ワーカが登録した枝刈り情報をできるだけ多く利用できるように、効率良くワーカ間でテーブル情報を共有することが求められる。本研究では共有手法として、ワークスティール時に victim ワーカのテーブルを複製する手法、1 つのテーブルをロックにより共有する手法を実装した。また、タンパク質ネットワークの実データを用いて性能評価を行い、その結果、ロックによる共有手法において 16 ワーカで 3.29 倍の性能向上が得られた。

(3) 計算状態操作機構の実装モデルの改良

設計・実装面からは、実装・性能モデルの改良として、計算状態操作機構 L-closure の複数回の償却呼出しコストを低減させる方式を開発した。

本研究では L-closure という言語機構に関して 2 つの実装を提案した。1 つは複数回同じ L-closure が呼び出された場合などの呼び出しコストの削減、もう 1 つは GNU C コンパイラ（GCC）4 系列への元々呼び出しコストの低い closure の実装である。L-closure は入れ子関数定義を評価すると生成される軽量レキシカルクロージャで、これを持つ拡張 C 言語を、高水準言語コンパイラの中間言語として採用することで、ごみ集めなどの高水準サービスを効率良く実装できる。現在 L-closure の実装として、GCC 3 系列の拡張によるコンパイラ実装と標準 C 言語への翻訳による実装があり、これらは L-closure の初期化処理を遅延したり、低い維持コスト（アクセスされる変数のレジスタ割当て）を実現したりしているが、呼び出しコストは高い。翻訳による実装では、C のスタックとは別のスタック（明示的スタック）を用意し、L-closure 呼び出し時に C のスタックの内容を明示的スタックへ一時的に移すことで、入れ子関数を持つ関数の局所変数へのアクセスを実現していた。本研究では、L-closure からのリターンの後、C スタック全体を再構築するのではなく、フレームごとに再構築することで、再度 L-closure が呼び出されたときのスタック間の値の移動を減らし、呼び出しコストを削減する。L-closure の研究の一環として、生成/維持コストはかかるものの呼び出しコストが低い closure も提案してきた。本研究ではまた、高度な最適化のために内部構造が刷新された GCC 4 系列において、これを実装した。

(4) コンパイラ実装

アセンブリ言語レベルのコンパイラ実装においては x86-64 命令セットアーキテクチャをサポートし、その際、短時間しか実行されない関数におけるレジスタ割り当ての効果の評価のため、新しい性能評価方式も開発した。

メモリ階層や投機実行の仕組みなどが複雑になり、正確な性能評価が困難になっている。特に機能の追加によるオーバヘッドや、プログラムの小規模な変更による、微小な性能の変化を知りたい場合に問題となる。問題は、ほぼ同じ機械語命令列で構成されたプログラムであっても、命令が配置されるアドレス(コード配置) が異なると、しばしば性能が大きく変化することにある。この変化は、コード配置が分岐予測、命令キャッシュなどに影響を与えるために起こっていると考えられる。本研究ではコード配置効果を考慮し適正に性能評価を行うために、評価対象プログラムとはほぼコード配置のみが異なる複数のプログラムの性能測定結果を統計処理することで、「コード配置の偶発性」を打ち消す手法を提案した。また、提案手法による性能評価を支援するシステムとして開発した「コードシェーカ」の構造、機能、有効性について議論した。

５．主な発表論文等

〔雑誌論文〕（計 9 件）

① Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi and Jun Sese, Parallelization of Extracting Connected Subgraphs with Common Itemsets, IPSJ Transactions on Programming, 査読有, 2014 (To appear)

② 田附 正充, 八杉 昌宏, 平石 拓, 馬谷 誠二, L-Closure の呼び出しコストの削減, 情報処理学会論文誌 プログラミング, Vol. 6, No. 2, pp.13-32, 査読有, 2013
http://ci.nii.ac.jp/naid/110009602856

③ Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani, Taiichi Yuasa, Parallel Graph Traversals using Work-Stealing Frameworks for Many-core Platforms, Journal of Information Processing, Vol. 20, No. 1, pp.128-139, 査読有, 2012
DOI: 10.2197/ipsjjip.20.128

④ 平石 拓, 八杉 昌宏, 湯淺 太一, SC 言語処理系における変形規則の再利用機構, コンピュータソフトウェア, Vol. 28, No. 1, pp.258-271, 査読有, 2011
https://www.jstage.jst.go.jp/browse/jssst/28/1/_contents/-char/ja/

⑤ 八杉 昌宏, 小島 啓史, 小宮 常康, 平石 拓, 馬谷 誠二, 湯淺 太一, L-Closure を用いた真に末尾再帰的な Scheme インタプリタ, 情報処理学会論文誌 プログラミング, Vol.3, No.5, pp.1-17, 査読有, 2010
http://ci.nii.ac.jp/naid/110007970943

〔学会発表〕（計 47 件）

① Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi

and Jun Sese, Parallelized Mining of Subgraphs Sharing Common Items using Task-Parallel Language Tascell, HPC in Asia Poster (in conjunction with ISC'14), 2014年6月26日,(発表確定)

② Masahiro Yasugi, Yuki Matsuda, and Tomoharu Ugawa, A Proper Performance Evaluation System That Summarizes Code Placement Effects, the 11th ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering (PASTE '13), 2013年6月20日,米国ワシントン州シアトル市 Red Lion 5th Avenue

③ 松井 健, 平石 拓, 八杉 昌宏, 馬谷 誠二, 高速版 Barnes-Hut 多体シミュレーションの並列実装, 先進的計算基盤システムシンポジウム (SACSIS2012), 2012年5月18日, 神戸国際会議場

④ 八杉 昌宏, 並列／高信頼プログラミング言語と実装技術, 東京大学コンピュータ科学専攻講演会 (招待講演), 2011年7月21日, 東京大学

⑤ 平石 拓, 八杉 昌宏, 馬谷 誠二, 動的負荷分散フレームワークTascellの広域分散およびメニーコア環境における評価, 先端的計算基盤システムシンポジウム SACSIS 2011, 2011年5月25日, 東京都秋葉原コンベンションホール

⑥ Masahiro Yasugi, Tasuku Hiraishi, Seiji Umatani, Taiichi Yuasa, Dynamic Graph Traversals for Concurrent Rewriting using Work-Stealing Frameworks for Multicore Platforms, 16th International Conference on Parallel and Distributed Systems (ICPADS 2010), 2010年12月10日, 中国上海市 衡山賓館

⑦ Masahiro Yasugi, Tsuneyasu Komiya, Tasuku Hiraishi, Seiji Umatani, Managing Continuations for Proper Tail Recursion, 2010 International Lisp Conference (ILC 2010), 2010年10月21日, 米国ネバダ州リノ市 John Ascuaga's Nugget

⑧ Tasuku Hiraishi, Masahiro Yasugi, Takuya Kouno, Seiji Umatani, Taiichi Yuasa, Tascell: a Backtracking-based Load Balancing Framework, 24th International Conference on Supercomputing (ICS10), poster presentation, 2010年6月2日, 茨城県つくば市 エポカルつくば

⑨ 八杉 昌宏, 高水準プログラミングによる細部の自由と計算状態操作機構, 自動チューニング技術の現状と応用に関するシンポジウム 招待講演, 2009年10月22日, 東京大学

〔図書〕(計 1 件)

① Masahiro Yasugi, Springer, On Efficient Load Balancing for Irregular Applications. In Concurrent Objects and Beyond (Festschrift for Prof. Yonezawa), 2014 (To appear)

〔その他〕

- ホームページ
  http://super.para.media.kyoto-u.ac.jp/tascell/
- アウトリーチ活動
  自動チューニング研究会 (http://atrg.jp/ja/) が主催する「自動チューニング技術の現状と応用に関するシンポジウム」において講演し、研究成果を広く発信した (2010年度から2013年度まで毎年度1回)。

６．研究組織

(1)研究代表者
八杉 昌宏 (YASUGI, Masahiro);
九州工業大学・大学院情報工学研究院・教授
研究者番号：30273759

(2)研究分担者
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

(3)連携研究者
小宮 常康 (KOMIYA, Tsuneyasu)
電気通信大学・大学院情報システム学研究科・准教授
研究者番号：80283638