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

令和　５ 年　 ６ 月　 ２ 日現在

機関番号：１１３０１
研究種目：基盤研究(C)（一般）
研究期間：2018～2022
課題番号：１８Ｋ１１２３３

研究課題名（和文）超並列技術をML系高信頼言語SML#に統合した超並列関数型言語の実現と最適化

研究課題名（英文）Realizing massively parallel functional programming language by integrating parallel technologies into SML#, an ML-style functional language

研究代表者
大堀　淳（Ohori, Atsushi）

東北大学・電気通信研究所・名誉教授

研究者番号：６０２５２５３２

交付決定額（研究期間全体）：（直接経費）　　3,300,000円

研究成果の概要（和文）：超並列技術を統合したML系関数型言語を実現する上で最大の技術課題は並列・並行ゴミ集め（GC)方式と実装技術の確立であった。本研究では、新たな並列平行GC方式とマルチコア上で動作する実装技術を確立し、我々が開発を進めているSML#言語に実装し、１００万を超える軽量スレッドの並列実行が可能な関数型言語の実現に成功した。その性能は、ML系関数型言語はもとより、GC方式のメモリー管理を採用する種々の主要な言語と同等か上回ることが示されている。この成果に加え、本研究で達成した種々の先端機能を実現し拡張されたSML#言語コンパイラSML#4.00版をオープンソースソフトウエアとして提供した。

研究成果の学術的意義や社会的意義
本研究が実現した並列・並行ゴミ集め方式は、従来ML系関数型言語においては困難であった細粒度スレッドのマルチコア上での並行並列実行を可能にした。これら研究成果を取り入れて拡張されたSML#言語は、マルチコア上の１００万を超える軽量スレッドをサポートする世界的にも類を見ない関数型言語である。これら成果を実現したSML#言語コンパイラ4.00版はGitHubを通じてオープンソースソフトウェアとして提供され、世界の関数型言語コミュニティに貢献している。さらに我々は、本研究成果の社会への幅広い普及を目指しSML＃に関するWEBページや教科書の出版等を行い、ML系言語の幅広い普及に貢献している。

研究成果の概要（英文）：The major technical challenge in realizing a massively parallel functional language is to establish a parallel and concurrent garbage collection (GC) that supports a large number of light-weight system threads running on a multi-core CPU. Through this research, we have established novel parallel and concurrent garbage collection method and its implementation techniques, and have implemented the GC method in a full-scale ML-style functional language, SML#. The resulting SML# supports more than one million lightweight user threads running on multi-core CPU. The benchmark evaluation shows good scalability on a multi-core processor comparable to C, and it outperforms other GC-based compilers in most cases. In addition to this new GC method, we have extended SML# with various advanced features and have released the SML# compiler as open-source software through the GitHub repository.

研究分野：計算機科学

キーワード：関数型プログラミング言語　コンパイラ　SML#　並行並列GC　マルチコアCPU　軽量スレッド　超並列技術

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

（１）ML 系関数型言語は、その高水準な記述能力に加え、堅牢な静的型システムと型推論機構によって、ソフトウェアの高信頼化に貢献しうると認識されていたと考える。しかしこれらの優れた利点にも関わらず、ML 系関数型言語がソフトウェア生産に広く普及しているとは言い難い状況であった。ソフトウェア生産現場での ML 系言語の普及を阻害している要因には、実用システム構築上重要な C 言語で書かれたシステムライブラリやデータベースなどのとの連携の困難さ、さらに、近年コモディティハードウェアとして広く普及しているマルチコア CPU 上の並列並行スレッドのサポートの欠如などが含まれると考えられた。

（２）我々が２００３年から開発を続けている SML#言語は、当時すでに、C 言語やデータベースなどとのシームレスな連携を達成し、上述した実用ソフトウェア開発基盤としての ML 系関数型言語の弱点を克服しつつあった。しかしながら、当時の SML#言語のマルチコア CPU 上の並行スレッドサポートは限定的なものにとどまっていた。すでに OS が提供する POSIX スレッドの呼び出し機能を提供しており、OCaml 等の他の ML 系言語と比較すると、最先端の並列並行機能を提供していたとは言え、近年システムプログラミング分野で研究が進み普及しつつあった軽量スレッドのサポートは無く、C 言語等に比較すると、十分な並列並行機能を実現しているとは言えない状況であった。また、この並列並行機能のサポートの不十分さに加え、ML 系関数型言語には、広く普及している動的型付け言語に比較すると、種々の値の表示等の実用的な機能が不十分であるとの欠点もあった。

２．研究の目的

（１）本研究の目的は、マルチコアプロセッサにスケールする超並列技術を、研究代表者等によって開発された ML 系多相型システムを備えた関数型言語 SML#に統合し、大規模分散計算や超並列計算などを含む高度な機能を必要とする高信頼ソフトウェア開発の基盤となりうる ML 系関数型言語を実現することである。この実現のため、SML#が持つ OS ライブラリとの直接連携機能を駆使して、ML 系言語での超並列機能の実現方式と最適化等に関する研究を行い、その成果を統合した SML#言語の実現方式を確立する。

（２）SML#への超並列技術の統合に加え、ソフトウェア開発の基盤として使用しうる ML 系高信頼プログラミング言語の完成を目指し、SML#のさらなる改良のための理論と方式・実装技術の開発を行い、それら成果を統合した SML#コンパイラを完成させ、オープンソースソフトウェアとして広く公開する。これらの活動を通じて、関数型プログラミングの研究コミュニティへの貢献、さらに、高機能な ML 系言語である SML#の普及を通じて、高信頼ソフトウェアの生産から高信頼言語によるプログラミングの学習等の幅広い分野に貢献することを目的とする。

３．研究の方法

（１）SML#ですでに達成している C 言語との直接連携機能を基礎に、システムのスレッドライブラリとの効率的な直接連携を実現し、C を代表とする手続き型プログラミング言語で実現されている超並列機能と比肩する性能を、ML 系関数型言語で実現するためのメモリー管理方式とその最適化手法を確立する。この実現のため基本的なアイデアは以下の通りである。宣言的で高水準な関数型言語のプログラムの実行は，CPU タイムスライスに加えて，静的に用意できない無制限のメモリー資源が外部から供給される必要がある。そこで、OS が行っている物理的なコア数で制約される CPU 資源を静的に見積もれない多量の論理スレッドに最適に分配するスレッドスケジュラーと同等の機能を、言語処理系が管理するヒープメモリー資源にも導入し、論理スレッドにヒープメモリー資源を動的かつ最適に分配するメモリーアロケータの概念を実現する。

（２）上述のメモリーアロケータの概念を実現する上での最大の技術的な課題は、マルチコア CPU 上で動作している多量の軽量スレッドと共存し、使用済みメモリーを低オーバーヘッドで回収可能な、新たな並列並行ゴミ集め（GC）方式とその実装技術の確立である。マルチコア CPU 上で動作しているスレッドの下で動作する GC は、各スレッドが使用中のメモリー資源を正確に把握するために、概念的にはすべてのスレッドと同期する必要がある。この同期を多量の軽量スレッドに対して実際にシグナル等を用いて実行するのは非現実的である。そこで本研究では、軽量スレッドの CPU 資源の分配方式をヒントに、GC に対して、user-worker の２段階のコンテクストを導入し、各 worker 毎に動作する仮想的なメモリーアロケータを、このコンテクストで動くコード集合として構築し、この仮想的なアロケーターを、すでに我々が開発しているスレッドを停止しない GC 方式とを統合した新たな GC 方式を開発し、この困難な技術課題の解決を目指す。

（３）以上の新たな GC 方式の開発以外に、実用性の高い超並列関数型言語の実現には、数多くの課題がある。本研究開始時点で検討した課題は、超並列軽量スレッドを使いこなすための高水準な宣言的な言語機能の実現、SML#が達成済みの先端機能であるデータベースとの統合の並列

拡張、などである。これらの課題についても、超並列軽量スレッドの実現方式を基礎に、SML#の機能を活かして実現可能性を検討する。

（４）マルチコア上の超並列処理が可能なフルスケールの関数型言語 SML#の実現と普及にむけ、本研究成果を SML#コンパイラに統合し、新たな SML#を実装し種々のサポートライブラリとともに、オープンソースソフトウェアとしてリリースする。さらに、実用ソフトウェア開発基盤としての SML#言語の普及を通じた幅広い社会的な貢献を目指し、本研究によって実現される超並列機能を含む SML#コンパイラに関して、その広報のためのホームページの開設やプログラミング手法を解説する書籍等の出版を行う。

４．研究成果

（１）最大の技術的な課題であったマルチコア上の軽量スレッドをサポートする並行並列ゴミ集め（GC）方式とその実装方式を確立した。その基本戦略の概要は以下の通りである。

- 軽量スレッドの標準的な実現モデルである user-worker モデルにおいて、各 user スレッドが worker として動作するためのコンテクストを管理し、GC 機能全体を worker コンテクストで動作するコードの集合として構成する。
- 各 user スレッドは、十分に短いタイミングで GC 状態をチェックし、それぞれの状態に応じて GC の処理の一部を実行する。
- GC 状態には GC の実行フェーズが含まれ、worker コンテクストで実行される各 GC コードは、GC の実行フェーズに応じて、平行かつ分散してヒープメモリーのトレースと回収を実行する。

この方式は、従来の関数型言語の GC においてマルチコア上の軽量スレッドにスケールことを阻害する原因となっていたスレッドローカルなヒープや同期を取るための stop-the-word 処理などを取り除き、スレッドと GC の待ち合わせ等を一切必要とせずにメモリー負荷に応じて GC 処理に必要な CPU 資源を自動的に確保するなどの優れた特性をもつ、新規性の高いものである。

本研究では、この方式に基づき GC アルゴリズムを開発し、その正しさを示し、その実現のための詳細な実装技術を開発し、それらを SML#に実装することに成功した。実装された SML#コンパイラのベンチマークテストの結果から、本研究成果である SML＃は、C と比肩するスケーラビリティを示し、並列並行実行が可能な種々の主要な言語とほぼ同等かよりよい性能を示していることが確認できた。特に、既存の ML 系関数型言語、例えば SML#とほぼ同等の言語である OCaml と比較すると、格段に優れた性能を達成している。

（２）以上の主要な目標であった超並列実行をサポートする GC 方式の開発と実装に加え、SML#言語をより実用的な言語として完成するための種々の機能や最適化方式の開発と実装を行った。その主なものを以下報告する。

① 動的型付け機構の実現

ML 系関数型言語は静的に型付けられた言語であり、種々のデータレイアウトはコンパイラによって静的に決定される。このため、プログラムによる実行時の動的な構造の分析や操作は困難であり、そのため、型に応じた値の表示や型依存の動作のするプログラムを定義できない。この弱点を克服するために、静的言語に動的型付け機構を導入する研究を行い、代表者がかつて開発した多相レコードのコンパイル方式を応用し、柔軟な動的型付け機構を SML#に導入することに成功した。この機構により、ほぼ任意の型の値をプリントする関数

```
# Dynamic.pp;
val it = fn : ['a#reify. 'a -> unit]
```

などを実現することができ、SML#の機能向上に繋がった。

② 有限多相性の理論に基づくコンパイラ最適化方式

SML#の多相関数は、研究代表者の開発した多相型プログラムの型主導コンパイルを通じて実現している。このコンパイル方式は、従来困難であったレコード多相性などを含む種々の多相性を系統的に実現しうる優れた方式であるが、ほぼ単相的に使用される多相関数に対しては、無視できない実行時オーバーヘッドを伴うという問題があった。本研究では、有限多相性に関する新たな型理論を構築し、それを基礎に従来の型主導コンパイル方式を洗練し、不要な型適用をコンパイル時に除去することによって、型主導コンパイルに伴う実行時オーバーヘッドを大幅に軽減することに成功した。

③ 主要な型判定を持つ ML の型推論理論

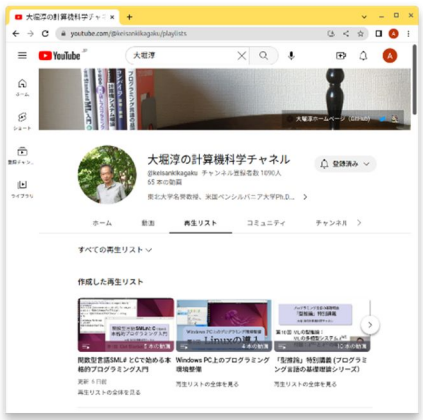
本研究全般が基礎とする理論的な枠組みは、ＭＬ系多相型言語の型推論理論である。ML 系言語の型推論理論は、Damas と Milner による１９８２年の論文以来、よく理解されているとみなされているが、その理論の詳細は複雑であり、理解されている内容は不正確な要素を含む。実際、我が国で出版されている教科書などにも「式のすべての型を代表する主要型」などの直観的な説明がなされるが、これらの説明は技術的に正確なものとは言い難い。この問題の本質は、従来の理論が基礎とする型システムは、式の主要な型判定を持たないことに起因する。本研究では、この問題に取り組み、ML の型推論の本質を理解する枠組みを再構築し、式の主要な型判定が存在

する新な型理論とそれに基づく健全で完全な型推論アルゴリズムが存在することを示した。この理論は、正確かつ直観的に理解可能であり、SML#、OCaml、Haskell などを含む ML 系言語の理解や実装に新たな知見を与えるものである。

（3）本研究成果によって実現可能になった新たな SML# 言語コンパイラ SML#4.00 をオープンソースソフトウェアとしてリリースした。このリリースを機に、これまでに SML#の開発に賛同いただいたプログラムコミュニティの協力の下、SML#の GitHub リポジトリを開設し、さらに SML#ホームページ（smlsharp.github.io/ja）を開設した。右の図は、SML#ホームページのスクリーンショットである。これらのインタネットリソースによって、ソースコードを含む SML#の研究開発資源をだれでも容易に利用できるようになり、また、SML#やそれに関連する新たな研究開発への参加や貢献なども容易になり、SML#の研究開発コミュニティの形成に寄与すると期待される。さらに、学術研究から生まれた我が国発のプログラミング言語のオープンソースとして公開は、国内外のプログラミングコミュニティ発展に資すると期待される。

（4）本研究の最終的な目的は、超並列機能で拡張された新しい ML 系関数型言語 SML#の実現を通じて、国内外の高信頼なソフトウェア開発に貢献することである。この目的達成のために、以上報告した学術的な成果に加え、研究成果である SML#の普及を目的とした書籍の出版や公開ビデオ講義などを通じた活動を行った。これらは本科学研究費の直接経費による研究成果ではないが、科学研究費の大きな目標である学術研究による社会貢献の考えに沿った貢献であるため、本報告に含める。

執筆した教科書は、ML プログラミングの方法論を解説した「SML#で始める実践 ML プログラミング」および ML 系言語のコンパイラの理論と実装を学ぶための「コンパイラ原理と構造」である。前者は、本研究の成果である軽量スレッドライブラリを駆使した並列プログラミング技法の解説も含んだ ML プログラミングの解説である。後者は、本研究で実現した動的型付け機構などを利用し SML#言語を用いてコンパイラを学習する内容である。



また、初心者がプログラミングや計算機科学学ぶための種々のビデオ教材を配信する YouTube チャネル「大堀淳の計算機科学チャネル」（youtube.com/@keisankikagaku）を開設し、そこで SML#の導入と SML#プログラミング学習のためのビデオ教材や、本研究の成果である主要な型判定を持つ ML の型推論理論をその背景から解説する講義などを含む複数のビデオ教材を配信済みである。

これらの広報活動は、本研究成果の成果である SML#の普及、さらにはそれらを通じたプログラミングリテラシの向上などに幅広く貢献するものと信じる。

５．主な発表論文等

〔雑誌論文〕　計3件（うち査読付論文　3件／うち国際共著　0件／うちオープンアクセス　3件）

１．著者名 逢坂美冬，上野雄大，大堀淳	４．巻 vol. 35, no. 3
２．論文標題 部分動的レコードを活用した型付きテンプレートエンジンの実現	５．発行年 2018年
３．雑誌名 コンピュータソフトウェア	６．最初と最後の頁 3_79 - 3_95
掲載論文のDOI（デジタルオブジェクト識別子） 10.11309/jssst.35.3_79	査読の有無 有
オープンアクセス オープンアクセスとしている（また、その予定である）	国際共著 －

１．著者名 Atsushi Ohori, Katsuhiro Ueno, Hisayuki Mima	４．巻 Volume 2 Issue ICFP
２．論文標題 Finitary polymorphism for optimizing type-directed compilation	５．発行年 2018年
３．雑誌名 Journal Proceedings of the ACM on Programming Languages (PACMPL)	６．最初と最後の頁 Article No. 81
掲載論文のDOI（デジタルオブジェクト識別子） 10.1145/3236776	査読の有無 有
オープンアクセス オープンアクセスとしている（また、その予定である）	国際共著 －

１．著者名 大野一樹，上野雄大，大堀淳	４．巻 vol. 11, no. 3
２．論文標題 関数型言語SML#のためのコードレベルデバッグ環境の実現方式	５．発行年 2018年
３．雑誌名 情報処理学会論文誌プログラミング（PRO）	６．最初と最後の頁 1-13
掲載論文のDOI（デジタルオブジェクト識別子） なし	査読の有無 有
オープンアクセス オープンアクセスとしている（また、その予定である）	国際共著 －

〔学会発表〕　計13件（うち招待講演　0件／うち国際学会　2件）

１．発表者名 大堀 淳
２．発表標題 主要な型判定をもつMLの型理論と型推論アルゴリズム
３．学会等名 日本ソフトウェア科学会 第25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 PPL 2023
４．発表年 2023年

１．発表者名
Katsuhiro Ueno, Atsushi Ohori

２．発表標題
Concurrent and parallel garbage collection for lightweight threads on multicore processors

３．学会等名
Proceedings of the 2022 ACM SIGPLAN International Symposium on Memory Management (ISMM 2022), pp 29-42, DOI：https://doi.org/10.1145/3520263.3534652（国際学会）

４．発表年
2022年

１．発表者名
Atsushi Ohori, Katsuhiro Ueno

２．発表標題
A Compilation Method for Dynamic Typing in ML

３．学会等名
19th Asian Symposium on Programming Languages and Systems, APLAS 2021 - Chicago, 2021/10/17-2021/10/18（国際学会）

４．発表年
2021年

１．発表者名
上野 雄大, 大堀 淳

２．発表標題
SML#の並列処理機能とその性能

３．学会等名
日本ソフトウェア科学会 第37回大会講演論文集

４．発表年
2020年

１．発表者名
大堀 淳, 上野 雄大

２．発表標題
関係代数を基礎とするプログラムに現れる名前解析システム

３．学会等名
日本ソフトウェア科学会第37回大会講演論文集

４．発表年
2020年

１．発表者名
大堀 淳，上野 雄大，高城 光平

２．発表標題
外部データの解釈を文脈ごとに与える動的型付け機構

３．学会等名
日本ソフトウェア科学会第37回大会講演論文集

４．発表年
2020年

１．発表者名
上野 雄大

２．発表標題
自然なデータ表現を持つ多相型言語のLLVM IRへのコンパイル方式

３．学会等名
日本ソフトウェア科学会第37回大会講演論文集

４．発表年
2020年

１．発表者名
上野 雄大、大堀 淳、田浦 健次朗

２．発表標題
SML#とMassiveThreadsの統合による超並列言語の実現

３．学会等名
日本ソフトウェア科学会第36回大会

４．発表年
2019年

１．発表者名
大堀 淳、上野 雄大

２．発表標題
SML#の動的型付け機構

３．学会等名
日本ソフトウェア科学会第36回大会

４．発表年
2019年

１．発表者名
大堀 淳、上野 雄大、大塚 祐貴、高城 光平

２．発表標題
SML#の永続性拡張に向けて

３．学会等名
日本ソフトウェア科学会第36回大

４．発表年
2019年

１．発表者名
魚谷孝太，上野雄大，大堀淳

２．発表標題
Java PathFinderによるMLプログラムの捕捉されない例外の検証

３．学会等名
The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018)

４．発表年
2018年

１．発表者名
高城光平，上野雄大，大堀淳

２．発表標題
ML系言語とストリーミングデータベースの統合

３．学会等名
The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018)

４．発表年
2018年

１．発表者名
大塚祐貴，Karim HAMDI，上野雄大，大堀淳

２．発表標題
高水準IoTプログラミング環境の実現に向けて

３．学会等名
The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018) （ポスター発表）

４．発表年
2018年

〔図書〕　計2件

1．著者名 大堀 淳	4．発行年 2021年
2．出版社 共立出版	5．総ページ数 196
3．書名 コンパイラ―原理と構造―	

1．著者名 大堀 淳、上野 雄大	4．発行年 2021年
2．出版社 共立出版	5．総ページ数 242
3．書名 SML#で始める実践MLプログラミング	

〔産業財産権〕

〔その他〕

SML#プロジェクトホームページ
https://smlsharp.github.io/ja/
本研究成果の成果であるを取り入れて開発されたSML#言語の紹介の他、SML#コンパイラをオープンソースとして配布しているGitHubレポジトリへのリンクなどがリンクされている。

大堀淳の計算機科学チャネル
https://www.youtube.com/@keisankikagaku
SML#の導入からSML#プログラミング、さらに、本研究成果である主要な型判定を持つMLの型推論などに関するオンラインビデオ講義を配信している。

6．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考
研究分担者	上野　雄大 (Ueno Katsuhiro) (60551554)	新潟大学・自然科学系・准教授 (13101)	

7．科研費を使用して開催した国際研究集会

〔国際研究集会〕　計0件

８．本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関