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　２ 年　 ２ 月 １７ 日現在

機関番号：１１３０１
研究種目：若手研究(B)
研究期間：2015～2018
課題番号：１５Ｋ１５９６４

研究課題名（和文）実用プログラミング言語のための系統的言語開発基盤の実現

研究課題名（英文）A systematic approach for developing practical programming languages

研究代表者
上野　雄大（Ueno, Katsuhiro）

東北大学・電気通信研究所・准教授

研究者番号：６０５５１５５４

交付決定額（研究期間全体）：（直接経費）　　3,000,000円

研究成果の概要（和文）：実用プログラミング言語の系統的な設計と実装のための枠組みを研究し，以下の成果を得た．（１）レコード多相性の考え方を応用してRuby言語を独立性の高い計算系の集合に分割し，それらを系統的に合成することで言語全体の操作的意味論を構築した．その結果からインタプリタを導出できることを確認した．（２）Ruby開発チームとの共同研究として，Rubyのための抽象解釈に基づく型プロファイラおよび漸進的型付けの考え方を取り入れた型検査器を開発した．（３）関数型言語コンパイラ構築の要素技術として，静的型付け言語から外部JSONデータを型付きで操作する体系，およびミューテータを止めない並行ごみ集め方式を開発した．

研究成果の学術的意義や社会的意義
あらゆるサービスがソフトウェアによって運用される情報化社会が信頼性と安全性を保ちつつ発展するには，サービスの複雑さや質を落とすことなく信頼性の高いソフトウェアを構築する技術の確立が望まれる．プログラミング言語の研究開発はその中核をなす重要な課題である．本研究成果は，広く産業利用され社会的に重要な地位をすでに築いている実用プログラミング言語を高信頼化するための技術開発，および世界的に研究が進められている高信頼プログラミング言語を実用化するための基礎研究の両面から高信頼ソフトウェア開発基盤の実現に貢献したことに，学術的および社会的な意義がある．

研究成果の概要（英文）：We conducted research on systematic design and implementation of a practical programming language. Major achievements includes the following: (1) By applying the idea of record polymorphism, we decomposed the Ruby language into a set of simple calculi and composed them systematically into the operational semantics of the whole langauge. In addition, we built an interpreter from the semantics. (2) In cooperation with the Ruby development team, we developed a type profiler and type checker for the Ruby language. A part of this development is based on the decompositional understanding of Ruby's operational semantics. (3) For functional programming language development, we developed a technique for typed JSON manipulation in a statically-typed language and concurrent garbage collector that does not stop the world.

研究分野：プログラミング言語

キーワード：プログラミング言語　スクリプト言語　関数型言語　操作的意味論　型検査　静的解析

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

今日、産業界で広く活用されているプログラミング言語のうち、業務上の要求に対する定型的な処理を短い記述で実現するための機能を重点的に提供する、スクリプト言語と呼ばれる言語が注目を集めている。Ruby、Perl、Python、PHP、JavaScript などの言語はその典型的な例である。これらの言語の多くは動的な型付けやリフレクション機能を持ち、豊富なライブラリを利用したプログラムをコンパイルエラーに悩まされること無くすぐに実行できる、手軽な開発環境をプログラマに提供する。その手軽さに由来する高いソフトウェア開発効率のため、スクリプト言語は Web アプリケーションを中心にソフトウェア産業の至る所で利用されている。

スクリプト言語は、言語実装者が諸機能をひとつの処理系実装に実現することを主軸として経験的に実現されているものが多い。この言語設計手法は、言語設計者が想定する使用法の範囲では、方法上、実用性の高い言語を迅速に作ることができる方法である。しかし、その言語構成方法は ad-hoc であり、各言語機能が理論的な整合性を持つ形で言語に組み入れられているかは、十分に検討されていない。この脆弱さは、その言語が広く普及するにつれ、また言語の規格が策定され規格適合性が議論されるにつれて露呈する。近年、実際に、スクリプト言語は言語設計者が意図しない巨大な開発プロジェクトに利用されるようになり、その成功とは裏腹に、そのいくつかの欠点が明らかになってきた。スクリプト言語の産業上の重要性を考えるならば、スクリプト言語を系統的に理解し構築する理論基盤の整備は喫緊の課題である。

２．研究の目的

本研究では、この課題に対し、関数型言語の設計アプローチをスクリプト言語に対しても応用することで、スクリプト言語を従来より系統的に定義し、さらにその定義から言語処理系を機械的に導出することを可能とする枠組みを与えることを目的とする。計算機科学におけるプログラミング言語理論研究の分野では、ある特定の機能をモデル化することに絞った小さな計算系を定義し、その性質を示すことで、プログラミング言語の諸機能を形式的に議論してきた。そのひとつの集成が、Standard ML に代表される ML 系関数型言語である。Standard ML は高階関数、多相型型システム、モジュールシステムなどの高度な言語機能を持つ。これらの機能はそれぞれ独立に、ラムダ計算を基礎とする独自の計算系によって数学的および論理学的にモデル化され、その基礎理論と実装技術が研究されてきたものである。フルセットの言語定義はそれらモデルの合成として与えられる。これが、Standard ML コンパイラおよび Standard ML プログラムの正しさを裏付ける理論基盤となる。さらに、Standard ML のビッグステップ操作的意味論は、その構成上、インタプリタの実装とほぼ対応しており、言語のリファレンス実装は言語定義からほぼ自動的に生成できる。これら関数型言語の設計上の特徴をスクリプト言語にも応用することで、従来のスクリプト言語構成方法を洗練し、スクリプト言語の系統的論理的な理解を与え、スクリプト言語の更なる開発の促進および高信頼化に貢献することを目指す。

３．研究の方法

本研究では、関数型言語の設計手法を基礎として、ひとつのスクリプト言語を複数の独立した計算モデルに分解し再構成することで、その言語の系統的な開発を可能とする理論的枠組みと基盤技術を構築する。その基本的な方法は以下の通りである。

- スクリプト言語が持つ特徴的な制御構造を、生成的な例外など、よく研究されている制御構造を応用することで、高水準かつ一般的な計算モデルとして定義する。
- 相互依存し一見不可分な言語機能集合を、相互依存するわずかな箇所を操作的意味論上で特定し抽象化することで、直交する計算モデルに分割する。
- 言語全体の意味論を、相互依存の抽象を互いの計算モデルで補い合うことで系統的に導出する。

この方法に従い、実用スクリプト言語に対し、その言語を構成する各言語機能が独立した計算系として抽出できることを示し、さらにそれら計算系の集合からスクリプト言語全体の操作的意味論を系統的に導出できることを示す。さらに、そのようにして導出された操作的意味論から、その言語のインタプリタを導出し、規格適合性を評価する。

以上の方法を実施するため、本研究では、具体的な対象言語として Ruby を選択する。Ruby は処理系独立な言語仕様が存在し（JIS X 3017:2011 および ISO/IEC 30170:2012）処理系の実装を読み解かなくてもその仕様を読み解く準備が整っており、本研究の対象として妥当である。

一方、関数型言語的アプローチによる仕様からのコンパイラの導出には、関数型言語コンパイラ開発のための要素技術を洗練することが必要不可欠である。本研究では、研究代表者らがこれまでに取り組んで来た関数型言語 SML#の開発と密接に連携しながら、関数型言語コンパイラの要素技術の開発を行う。

４．研究成果

以上の研究を実施し、以下の研究成果を得た。

- Ruby 言語全体を、コアとなる計算系と、そのコアを拡張するサブ計算系の集合にそれぞれ分割した。これらの計算系の評価規則に現れる環境をレコード多相的に与えることで、先行研究よりも機能間の独立性が高く形式的合成が可能な操作的意味論を構築した。さらに、モジュラーな意味論を実現するための基盤技術として自然結合を用いることの着想を得た。これまでの本研究における意味論の構築にはオラクル関係と言語拡張を機能ごとに使い分けていたが、自然結合に基づく方法ならば、これらを統合したより系統的な意味論の構築が可能になると期待できる。
- 前述の操作的意味論から Ruby のインタプリタを手作業で導出し、操作的意味論からインタプリタをおおよそ機械的に作り出せることを確認した。このインタプリタは、SML#で記述され、前述した形式的合成後の計算系に対する評価器と、Ruby の全構文をその計算系に翻訳するアルゴリズムからなる。このインタプリタの Ruby への適合性を検査するため、このインタプリタで Ruby 言語付属のテストケースを実行した結果、Ruby 2.0.0p648 に付属の 3966 件のテストケースのうち 623 件のテストに成功した。
- Ruby 開発チームとの産学連携共同プロジェクトを実施し、Ruby の信頼性を高めるための研究を行い、Ruby プログラムの静的解析方式を開発し、2 つのツールの試作を支援した。ひとつは抽象解釈に基づく型プロファイラ、もうひとつは漸進的型付けの考え方を取り入れた型検査器である。どちらの静的解析方式も、その開発には、Ruby の意味論に根ざした緻密な分析が必要であった。この分析の一部は本研究による実用言語の系統的形式的理解に基づいて行われた。この開発は本研究の具体的応用のひとつであり、この応用を通じて、本研究が提案する方式の適用可能性および有用性の一部は示されたと言える。本共同研究の成果の一部は、Ruby の次期バージョンである Ruby 3 に取り込まれることが検討されている。
- 関数型言語コンパイラ構築の要素技術について以下の成果を得た。(1) 静的型付け言語から外部 JSON データを型付きで操作する理論を構築し、その理論に基づく機能を SML#コンパイラに実装した。(2) ミューテータを止めない並行ごみ集め方式を開発し、高い並列性能を得た。(3) 高度な言語機能を実現する基盤となる型主導コンパイル方式のためのコード最適化の理論を開発し、その実装技術を確立した。この研究により、SML#コンパイラが挿入する属性抽象の約 8 割を削減し、 約 1 割の高速化を達成した。(4) 関数型言語と SQL の統合方式を見直し、多くの SQL 構文のサポートを可能とした。これらの成果は、SML#コンパイラに統合され、オープンソースソフトウェアとして公開されている。

５．主な発表論文等

〔雑誌論文〕（計　４件）

(1) 大野一樹，上野雄大，大堀淳: 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，情報処理学会論文誌プログラミング（PRO），Vol. 11, No. 3, pp. 1-13, 2018, 査読あり．
https://ipsj.ixsq.nii.ac.jp/ej/index.php?active_action=repository_view_main_item_detail&page_id=13&block_id=8&item_id=191425&item_no=1

(2) 逢坂美冬，上野雄大，大堀淳: 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，コンピュータソフトウェア，vol. 35, no, 3, pp. 3_79-3_95, 2018, 査読あり．
https://doi.org/10.11309/jssst.35.3_79

(3) Katsuhiro Ueno, Atsushi Ohori: A foreign language interface from ML to shell, New Generation Computing, Vol. 34, No. 3, pp. 239--256, 2016, 査読あり．
https://doi.org/10.1007/s00354-016-0303-1

(4) Katsuhiro Ueno, Atsushi Ohori: A Type Safe Access to Key-value Stores from Functional Languages, Journal of Information Processing, Vol. 24(2016), No. 1, pp.141-151, 2016, 査読あり．
https://doi.org/10.2197/ipsjjip.24.141

〔学会発表〕（計１９件）

(1) 遠藤侑介，松本宗太郎，上野雄大，住井英二郎，松本行弘: Progress report: Ruby 3 における静的型解析の実現に向けて，第 21 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（PPL2019），2019.

(2) Atsushi Ohori, Katsuhiro Ueno, Hisayuki Mima: Finitary Polymorphism for Optimizing Type-Directed Compilation, In Proceedings of the 23rd ACM SIGPLAN International

Conference on Functional Programming (ICFP'18), 2018.
https://doi.org/10.1145/3236776

(3) 大塚祐貴, Karim HAMDI, 上野雄大, 大堀淳: 高水準 IoT プログラミング環境の実現に向けて, The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018), 2018.

(4) 魚谷孝太, 上野雄大, 大堀淳: Java PathFinder による ML プログラムの捕捉されない例外の検証, The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018), 2018.

(5) 高城光平, 上野雄大, 大堀淳: ML 系言語とストリーミングデータベースの統合, The 2nd. cross-disciplinary Workshop on Computing Systems, Infrastructures, and Programming (xSIG 2018), 2018.

(6) 美馬久行, 上野雄大, 大堀淳: 多相関数を含むプログラムの抽象解釈を用いた最適化, 日本ソフトウェア科学会第 34 回大会, 2017.

(7) 大野一樹, 上野雄大, 大堀淳: SML#のためのコードレベルデバッグ環境の構築に向けて, 日本ソフトウェア科学会第 34 回大会, 2017.

(8) Atsushi Ohori, Kenjiro Taura, Katsuhiro Ueno: Making SML# a general-purpose high-performance language, ACM SIGPLAN ML Family Workshop, 2017.

(9) Tomohiro Sasaki, Katsuhiro Ueno, Atsushi Ohori: SML# with Natural Join, ACM SIGPLAN Workshop on ML, 2016.

(10) Katsuhiro Ueno, Atsushi Ohori: A Fully Concurrent Garbage Collector for Functional Programs on Multicore Processors, in Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming (ICFP 2016), pp. 421--433, 2016.
https://doi.org/10.1145/2951913.2951944

(11) 佐藤友昭, 上野雄大, 大堀淳: SML#のよりシームレスな外部関数インターフェースの実現に向けて, 日本ソフトウェア科学会第 33 回大会, 2016.

(12) 佐々木智啓, 上野雄大, 大堀淳: 自然結合制約を含む型推論アルゴリズムの実装方式, 日本ソフトウェア科学会第 33 回大会, 2016.

(13) 美馬久行, 上野雄大, 大堀淳: SML#による Vertex-centric プログラミングに向けて, 日本ソフトウェア科学会第 33 回大会, 2016.

(14) 逢坂美冬, 上野雄大, 大堀淳: 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試作, 日本ソフトウェア科学会第 33 回大会, 2016.

(15) 徳永航平, 上野雄大, 大堀淳: OS を関数型言語のみで開発するための検討と試作, 日本ソフトウェア科学会第 33 回大会, 2016.

(16) Atsushi Ohori, Katsuhiro Ueno, Tomohiro Sasaki, Daisuke Kikuchi: A Calculus with Partially Dynamic Records for Typeful Manipulation of JSON Objects, in Proceedings of the 30th European Conference on Object-Oriented Programming (ECOOP 2016), pp. 18:1--18:25, 2016.
http://dx.doi.org/10.4230/LIPIcs.ECOOP.2016.18

(17) 新田祐児, 上野雄大, 大堀淳: 関数型と組型がネストした型を持つ変数を含んだ式の自動生成手法, 情報処理学会プログラミング研究会 第 108 回プログラミング研究発表会, 2016.

(18) 田畑憲太, 上野雄大, 大堀淳: コンパイラ実装言語で中間表現データ構造を記述するための言語機構, 情報処理学会プログラミング研究会第 108 回プログラミング研究発表会, 2016.

(19) 逢坂美冬, 菊地大介, 上野雄大, 大堀淳, 佐々木加奈子: 関数型言語による高水準な Web アプリケーション開発環境, 情報処理学会プログラミング研究会第 104 回プログラミング研究発表会, 2015.

〔図書〕（計　0件）

〔産業財産権〕
○出願状況（計　0件）

○取得状況（計　0件）

〔その他〕
ホームページ等
SML# Project
https://www.pllab.riec.tohoku.ac.jp/smlsharp/

６．研究組織

(1)研究分担者
なし

(2)研究協力者
なし