2016 年度 パッヘ研究奨励金 I-A-2（特定研究助成･一般）申請書（簡易版）

2016 年 4 月 15 日

氏 名:横山 哲郎	所 属: 理工学部ソフトウェア工学科 研究科 専攻

（ ） 本学に赴任後 3 年以内の教員で、初申請の方は、左の括弧内に○をつけてください。また、その場合、「研究費」欄は 40 万円を上限として記入してください。

次のいずれかに該当される方は、下記の該当する括弧内に○をつけてください。
研究代表者（ ）研究分担者（ ）２０１６年度科学研究費補助金申請者（採択）
研究代表者（ ）研究分担者（ ）２０１６年度科学研究費補助金申請者（不採択）
研究代表者（ ○ ）研究分担者（ ）２０１６年度科学研究費補助金継続採択
（ ）2015 年 4 月 20 日から 2016 年 4 月 20 日までの間に科学研究費補助金以外の公募制研究助成金（機関公募に限る）の採否の結果を受けた者
助成金名：
採否：

研究課題 研究の内容を具体的に表すよう 40 字以内で簡潔に記入してください。

可逆変換を用いた高水準プログラミング言語のプログラム変換

研究費（合計金額は 30 万円（赴任後 3 年以内の初申請は 40 万円）を上限としてください。）

費目、内訳、金額（千円単位）を記入

研究経費の妥当性・必要性
「研究計画・方法」を踏まえ、研究費経費の妥当性・必要性・積算根拠について記入してください。

参考となる書籍・規格・資料を収集することで先行研究や標準的な技術に関する調査を進める。また関連資料の収集には ACM のデジタルライブラリも用いる。5 年以上前に購入したパソコンにおいて実験を行う。このために、パソコンのメモリやストレージに交換して計算機能力を補う。研究成果を発表するために旅費もしくは学術雑誌掲載料を使う。可読性の評価は人間が行う必要がある。対象コードの扱いも含めて研究の妥当性を示すために研究遂行者がこれにあたるのではなく協力者に評価をしてもらう。

＜研究費＞

書籍代 25 千円
複写費 1 千円
国内旅費(日本ソフトウェア科学会の学会、3 日間) 30 千円
印刷製本費・学術雑誌掲載料 91 千円
消耗品費 98 千円
謝礼費(可読性の評価・対象ソースコードの収集、5 名) 20 千円
諸会費 35 千円

合 計	300,000 円

※大学が指定する「研究倫理教育」を受講していない方、および過去にパッヘ研究奨励金Ⅰ-A の助成を受けて、申請時点で研究成果が公刊されていない方は申請の資格がありませんのでご留意ください。

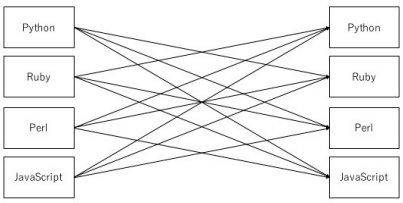
＜研究目的（概要）＞

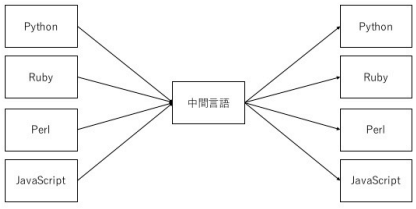
すでに良くテストされ稼働実績があるアプリケーションの算法・解法・論理を再開発することは困難でリスクを伴うためにコストがかかる。したがって、類似する新規のアプリケーションの開発において既存のアプリケーションの算法・解法・論理の一部分を統合することでコスト削減をすることが妥当であることは少なからずある。しかし、正しさが厳密に保証された高水準言語間のプログラム変換を行うことはコードの可読性を損ない変換後のコードの可読性や再利用性を下げる問題がある。われわれは変換前後の基本データ型および単純な派生型（クラス・モジュール等の型に準ずるものを含む）の間に自然な対応関係をつけ、振る舞いの保証は限られた範囲でのみすることにする。すなわち、プログラム変換の全域性を犠牲にする代わりに可読性と再読性を上げるアプローチを取る。われわれは、命令型やオブジェクト指向型の性質を備えたいわゆる軽量プログラミング言語を用いて開発される数十〜数百行からなるコードレッドのみを対象とする。

言語 A で開発されたコードレット a を、言語 B のコードレット b に自動変換できるようにすることの利点は主に 3 つある。第一に、a の正しさや算法の効率の良さを仮定すれば、b のそれらも保証される。このことにより、b を人手で開発するよりもコーディング、テスト・証明のコストが削減されることが期待される。第二に、開発者が言語 A に未習熟であっても、b を読むことで a の算法・論理を高い確度で推測することを可能にする。ソースコードリポジトリやウェブ上に蓄えられた巨大なコードレットの集積から、参考になる算法・論理を用いて作成されたコードレットを読む際にも言語間の構文のささいな違いに拘泥してしまう問題を避ける助けとなる。第三に、言語 A で開発された基本コードレット群を他の開発者に複数の言語で配布することができるようにする。

＜研究計画・方法＞

対象プログラミング言語は Python、Ruby、Perl、PHP、JavaScript といった代表的な軽量プログラミング言語とする。高水準言語間のプログラム変換器は既存研究があり、左図の矢印に対応する変換器のいくつかは実用的な水準である。しかし、このアプローチでは m 個の言語を対象としたとき m^2 に比例する量、すなわち $O(m^2)$ のプログラム変換器を準備する必要が出てきてしまう。翻訳機の開発において標準的に使われている中間言語を導入することで右図のような変換が可能になる。すなわち、原始言語から中間言語にプログラム変換して、さらに中間言語から対象言語に変換するのである。このことによって必要なプログラム変換器の数は m に比例した数（2×m）になる。中間言語は構造図で管理するような情報を適切に保持するように設計する。





プログラミング言語は版が上がると構文や意味論が変更される。このためわれわれが開発するプログラム変換器もそれに伴って更新が必要である。原始言語から中間言語のプログラム変換および中間言語から対象言語への変換を独立して開発する場合はこれらの対応が場当たり的かつ非統一的になる。われわれはプログラム変換を可逆的に行うことでこの問題を解決する。具体的には、字句解析・構文解析・プリティプリントには各言語に公式にサポートされているかすでに実績があるものを用いて、その結果得られる抽象構文木をさらに、BNFC(The BNF Converter)によって字句解析・構文解析とプリティプリントを双方向にする。