A-031

コンパイラ構築の証明論的枠組み
A Proof Theoretical Framework for Compiler Construction

大堀 淳*

概 要

本発表では，コンパイラを系統的に構築することを可能にする証明論的な枠組みを提案する．本枠組みでは，ソース言語，ターゲットの機械語言語，さらにコンパイル段階に現れる中間言語は，すべて，論理学の証明システムとして表現され，コンパイルの各段階は，それら証明システム間の証明変換として表現される．さらに，それら証明システム間の証明変換は，証明システムのカット除関係を保存することを示すことができる．この表明論的枠組みは構成的であり，証明システム間の変換が可能であると言う性質の証明から，対応するコンパイル段階を実現するアルゴリズムが抽出できる．このアルゴリズムは，その構成方法から，型と操作的意味を保存することが帰結する．

1　正しいコンパイラ構築の試み

今日の計算機システムにおいては，すべてのプログラムは，最終的には，機械語に翻訳され，フォン・ノイマン型逐次機械によって実行される．この事実は，すべてのプログラムの正しさは，プログラムを機械語コードに翻訳するコンパイラの正しさに依存していることを意味している．インタープリタを含む高水準のプログラミングシステムも，そのシステム全体を考えれば，そのシステムの操作的意味を実現しているプログラムが高水準記述に対して実行する機械語のコード列に翻訳され実行されていると見なせるため，その正しさは，操作的意味を実現しているプログラムをコンパイルしたコンパイラに依存している．

このような重要な位置づけから，正しいコンパイラ構築の試みは，計算機科学の発展とともに探求されてきた最重要な課題の一つである．コンパイラ研究の長い歴史の中で，型の解析や意味解析，コード生成，レジスタ割付け等の数々の研究がなされ，より正しいコンパイラの構築技術が蓄積されている，しかし残念ながら，正しさが保証されたコンパイラが構築できる技術が確立されたと言える状況にはいたっていない．本発表では，正しさが保証されたコンパイラ構築の新たな枠組みを提案する．

コンパイラもプログラムの一つであり，正しさが保証されたコンパイルの構築は，正しさが保証されたプログラム構築の一種である．この洞察から，コンパイラの正しさの研究には，従来，コンパイラを分析対象のプログラムと捕らえ，その対象の性質を記述するメタ論理を構築し，プログラムとしてのコンパイラの正しさを証明する手法が試みられている．しかしながら，そのようなプログラム検証のアプローチは，２０万行を越え１５以上ものコンパイル段階を含む[1]ような現実の最先端のコンパイラに適用できる手法とはなっていないのが現状である．

本発表で提案する手法は，コンパイラが言語の翻訳システムであることの洞察を基礎とし，コンパイラを，あらかじめ確立された正しい翻訳の方式に従って構成する，というものである．構成に関するミスを取り除く形式的なチェックは必要であるが，この方式で構築されたアルゴリズムは，その構成上正しいと言える．このアプローチは，もし成功するならば，例えばグラフ彩色などのアルゴリズムそのものをメタレベルで検証する試みにおける検証の複雑さの問題を一挙に解決する可能性を持つ．さらにこのアプローチは，正しさの証明に関しても，無条件に信頼すべきコンポーネントが極めて少なく，de Millo[3]等の指摘する複雑な定理自動証明システムに関する問題点も少ないため,「より信頼性の高い正しさ」と言える．

なお，本アプローチのより技術的な内容に関しては，日本ソフトウエア科学会の大会にて発表予定である．

2　コンパイラ構築の証明論的枠組み

本提案が基礎とするのは，直感主義的論理学の証明論とプログラミング言語との Curry-Howard 同型関係 [1, 2] の考え方である．この考え方に基づき，自然演繹システムと型付きラムダ計算との同型関係が確立し，それに基

*東北大学 電気通信研究所. RIEC, Tohoku University

[1]現在我々が開発中の次世代 ML 言語，ＳＭＬ＃コンパイラ（http://www.pllab.riec.tohoku.ac.jp/smlsharp/）の例

づき，証明論的な結果や考え方を基礎としたプログラミング言語の基礎付けの研究が多数なされている．本提案では，この考え方をコンパイルの全過程に適用することによって，正しいコンパイルありゴリズムを系統的に導出する枠組みを構築する．その基本となる洞察は以下の通りである．

- 機械語を含め，コンパイルの各段階に現れる全ての言語は，有限の規則によって形式的に定義された計算系であり，それは，ソース言語のモデルであるラムダ計算と等価である．従って，それら各々に対応する直感主義的な証明システムが存在するはずである．
- コンパイルは，一つのソース言語を別のターゲット言語に翻訳する処理の繰り返しである．ソース言語とターゲット言語が証明システムに対応するのであれば，この翻訳は，同一の論理学を記述する等価な二つの証明システム間の証明変換であるはずである．

この洞察から，コンパイルの各段階に必要あるいは有用である言語を，直感主義的な証明システムとして定義することができれば，コンパイルの全過程は，証明変換の合成に対応するはずである．

個々の証明変換の正しさは，それぞれの証明システムの証明論的な簡約意味論を保存することを示すことで確立することができる．通常，証明論的な意味は，自然演繹システムであれば証明の正規化定理として，シーケント計算系であればカット除去定理として与えられるが，別の形式の定義も無数に存在しうる．一方，環境を用いたプログラムの操作的意味は，有限の規則で定義されたプログラムと値との計算可能な関係である．ここに Curry-Howard 同型関係の考え方を適用すれば，そのような操作的意味論は，環境をカットする証明論的なカット除去定理に相当すると考えられる．従って，それぞれの証明システム内部で，プログラムの実行に厳密に対応するカット除去定理が存在するはずであり，正しい証明変換は，この性質を保存するはずである．

以上の洞察から，コンパイルの各段階に必要な正しいアルゴリズム，以下のようにして系統的に抽出することができる．

1. ソース言語とターゲット言語に対して，それぞれに対応する証明システムと，その操作的意味に対応するカット除去関係を定義する．
2. 二つの証明システムの証明変換を，証明に関して帰納的に定義する．
3. 証明変換がカット除去関係を保存することを証明する．

1 のステップからそれぞれの言語の型システムが自動的に導かれる．2 の証明変換の定義から，コンパイルアルゴリズムが抽出される．この様にして得られたコンパイルアルゴリズムは，その構成から，静的意味，即ち型の性質を保存する．さらに 3 の性質から，このコンパイルアルゴリズムは，プログラムの動的意味，すなわち実行時の動作を保存することが帰結する．

3　証明論的コンパイラの構成

これまでの研究成果から，本提案の枠組みを完成させるいくつかはの結果はすでに得られている．重要なものは，以下の通りである．

- 機械語コードに対応する証明システムを構築できる．その操作的意味（実行機械の定義）は，証明システムのカット除去定理から導かれる．[5]
- レジスタ割付けは証明変換として表現でき，そこから，構成上正しいレジスタ割付けアルゴリズムを抽出できる．[4]

本発表では，これら結果を新たな結果で補完することにより，コンパイラの全過程を証明変換として定義でき，その結果から少なくとも原理的には，構成上正しいコンパイラが系統的に抽出可能であることを主張する．

参考文献

[1] H. Curry, Functionality in Combinatory Logic, Proc. National Academy of Sciences, 20, 584-590, 1934.

[2] W. A. Howard The formulae-as-types notion of construction. In To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism, Academic Press, pp. 479- 490, 1980.

[3] A. De Millo, R. J. Lipton, A.J. Perlis. Social processes and proofs of theorems and programs. *Communications of the ACM* 22(5), 271–280, 1979.

[4] A. Ohori. Register Allocation by Proof Transformation. Journal of Science of Computer Programming, 50(1-3), 161–187, 2004.

[5] A. Ohori. A proof theory for machine code. *ACM Trans. Program. Lang. Syst.*, 29(6), 2007.