論文の内容の要旨

論文題目　A Logic-Based Approach to Developing Analysers for C Programs
（論理にもとづく C プログラム解析器の開発に関する研究）

氏名　　千　代　　英　一　郎

本研究の主題は，高い信頼性を持つ C プログラム解析器の開発方法に関するものである．プログラムの実行時の振る舞いに関する性質を静的に調べるプログラム解析器は，コンパイラが生成する実行コードの性能を左右する主要な構成要素である．その不具合はコンパイラが生成する実行コードの不具合につながるため，その精度や効率と共に高い信頼性が求められている．

高信頼なプログラム解析器を開発する有力な方法のひとつに対象言語の形式的意味定義にもとづく方法が存在する．形式的意味定義は対象言語のプログラムの振る舞いを数学的な厳密さで定義したものである．形式的意味定義により，解析アルゴリズムがすべての場合に正しいことを数学的に証明することが可能となる．また，抽象解釈と呼ばれる手法を用いることで，正しさの証明しやすいアルゴリズムを系統的に設計することが可能である．Java 等の理論的扱いの容易な一部の言語ではすでに実際の解析器の開発に用いられており，その有効性が報告されている．

しかしながら，C においては，このような形式的意味定義にもとづく開発は未だ実用に至っていない．最大の問題は，プログラム解析器の開発の基盤となりうる適切な形式的意味定義が存在しない点にある．C は，型キャスト等の言語機能により具体的な実行環境に依存した記述を許している低レベル言語としての側面と，通常の高級言語のように抽象的な言語概念にもとづき意味が定められている高レベル言語としての側面が共存しているという特殊性を有している．既存の C の形式的意味定義は，低レベル言語としての側面にのみ着目し，型制約等の言語規格を無視しているもの，もしくは高レベル言語としての側面にのみ着目し，型キャスト等の低レベルな扱いが必要となる言語機能は対象外としているもの，のいずれかに分類される．前者の問題は，形式的意味定義に反映されていない言語規格の制約を前提とするプログラム解析器の正しさを証明することができない点である．後者の問題は，高信頼なプログラム解析器の開発において特に考慮の必要な言語機能が形式的意味定義に含まれておらず，そのような言語機能を用いるプログラムに対する解析の正しさが保証できない点である．

本研究の目的は，C においても，形式的意味定義にもとづくプログラム解析器の開発を実現することである．我々は，RML（Register Memory Language）と呼ぶ中間表現を設計し，その形式的意味定義を与える．RML は，C プログラムの構文構造をほぼ維持したまま論理式で表す中間表現であり，その形式的意味を C の言語規格にもとづいて定義することで，高い信頼性を持つ C プログラム解析器の開発基盤とすることができる．RML の形式的意味定義は基本的に C の高レベルな側面に着目したもので，言語規格が定める型制約を忠

実に反映している．C の低レベルな側面が現れる型変換等の意味はパラメータ化されており，目的に応じた意味を高レベルな言語概念を用いて与えることができる．これにより，特に C の低レベルな側面に依存しないプログラムについては，既存の形式的意味定義では示すことのできない性質を示すことが可能である．

我々の定義が実用的な C プログラム解析器の開発に有用であることを示すため，代表的な解析器のひとつであるポインタ指示先解析器について，形式的意味定義にもとづきアルゴリズムを設計し，その正しさの証明を与えた．これは型変換と共用体を扱いうるポインタ指示先解析器の正しさに関するはじめての証明である．我々の枠組みでは，プログラムは基礎論理式の集合として表され，その意味は述語論理における理論（閉論理式の集合）として与えられる．解析器の仕様は，解析対象プログラム P の意味を $[\![P]\!]$，解析によって判定したい性質を表す論理式を ϕ とするとき，ϕ が $[\![P]\!]$ の論理的帰結になるかどうかの判定問題として定式化できる．我々は，この問題を抽象解釈にもとづき導出可能性問題に帰着し，その判定アルゴリズムを導く方法を示す．我々の知る限り，本手法は，形式的意味定義から抽象解釈を介して deductive system にもとづくアルゴリズムを導く最初のものである．

既存のコンパイラで用いるプログラム解析器を開発する場合，その対象となる中間表現は所与のものを用いる必要がある．しかしながら，多くのコンパイラにおいて中間表現の仕様は明確に定義されておらず，高信頼なプログラム解析器を開発する上で障害となっている．この問題に対し，我々は IIR（Identifier-based Intermediate Representation data model）と呼ぶデータモデルにもとづく中間表現の仕様記述方法を提案する．IIR は関係データモデルに一定の制限を加えたデータモデルであり，プログラムの中間表現をデータベース（関係の集合）としてモデル化することができる．中間表現の仕様は，スキーマおよび制約を表す Horn 論理式によって記述される．後者の制約を論理型言語を用いて記述することで，実行可能仕様として利用することができる．提案手法を用いることで，製品 C コンパイラの中間表現の仕様を，実装と整合する仕様を形式的に記述することが可能となった．

本論文は 5 つの Part により構成されている．主要部は Part II から IV の 3 部である．Part II および III では，高い信頼性を持つ C プログラム解析器の開発基盤となる中間表現 RML を導入し，それにもとづき高信頼なポインタ解析器の開発を行う．Part IV では，中間表現の仕様記述に適したデータモデルである IIR を導入し，それを用いて既存の C コンパイラの中間表現の仕様を記述する．RML と IIR の関係は相補的である．RML が C プログラムの抽象的な論理式表現であるのに対し，IIR は，RML の具体的な表現を定義する際の仕様記述方法を与える．

各 Part の内容は以下の通りである．

Part I は導入部である．1 章では，本研究の対象，動機，目的，方法を明らかにする．また，本研究に関連する従来研究の問題を指摘する．2 章では，本論文で用いる基本的な用語，表記法を定義する．

Part II では，中間表現 RML の定義を行う．3 章では，C の標準言語規格を分析し，形式的意味を定義する際に生じる問題をあきらかにする．4 章では，RML の形式的意味定義の概要を示す．5 章および 6 章では，RML の静的・動的な形式的意味を定義する．7 章は，本 Part のまとめである．関連研究との比較を行い，その利点・欠点を議論する．

Part III では，Part II の成果を C プログラム解析器のひとつであるポインタ解析器の開発に適用した結果を示す．最初に，8 章において，ポインタ解析について，その目的，関連研究，研究課題を説明する．また，Part II の形式的意味を用いて，ポインタ解析問題を形式的に定義する．9 章と 10 章では，8 章で与えた問題定義にもとづき，その解析アルゴリズムを設計する方法を示す．9 章では，形式的意味をその構造に沿って抽象化した抽象的意味を定義する．10 章では，抽象的意味から解析アルゴリズムを導出する方法を示す．11 章は，本 Part のまとめである．

Part IV では，中間表現の仕様記述を支援するための手法を提案する．12 章では，本編の動機，これまでの問題点，および提案手法の概要を示す．13 章では，提案する中間表現データモデル IIR を定義する．14 章では，提案手法の適用事例として，製品 C コンパイラで用いている中間表現の仕様記述を行った結果を示す．15 章では，IIR の応用可能性を示す．IIR ではプログラムを関係の集合としてモデル化するが，この特徴は，仕様記述以外に，論理型言語を用いた宣言的なプログラム解析や永続化にも有用であることを示す．16 章と 17 章では，提案手法を関連研究と比較し，その利点・欠点を議論する．18 章は，本 Part のまとめである．

Part V は本論文の結論部である．19 章では，本研究の成果をまとめ，今後の課題について述べる．