発表概要

# コンパイラにおける意味解析器の自動生成

舞田純一[†] 中井央[†]

近年，様々な用途に合わせて様々なプログラミング言語が出現している．このため，自身の用途のためにプログラミング言語の処理系であるコンパイラをできるだけ容易に作成できることが望まれる．コンパイラ作成のためのツールとしては Lex や Yacc が代表的である．Yacc が登場してからかなり長い年月が過ぎたが，その間には時代の要求に合わせて，たとえばベースとなる言語が Java であるようなツールも出現してきている．これらのツールを用いたコンパイラ開発の場合，意味解析処理は構文解析時の動作（アクション）として C 言語などのプログラム断片を記述することが一般的である．また，記号表などはコンパイラ作成者がゼロから実装することも通常である．この際，算術式の構文や識別子の出現といった構文においては，型の検査や記号の登録といった作業のためのプログラム断片を記述することとなるが，一般にはあるプログラミング言語の文法にはその中に同様の構文的な構成が複数箇所あり，同様の意味解析処理のプログラム断片もそれに合わせて複数箇所記述しなければならない．これまで意味解析処理の内容について個別に研究は行われてきていても，それを視野に入れた生成系については論じられることがほとんどなかった．本発表では，型についての処理，記号表についての処理に焦点を当てた意味解析器の自動生成を含むコンパイラフロントエンドの自動生成について述べる．

# Automatic Generation of Semantic Analyser of Compilers

Junichi Maita[†] and Hisashi Nakai[†]

In these days, for many purpose, various programming languages has appeared. So rapid and easy construction of compilers is required. Yacc and Lex is very famous tool for compiler construction. Recently, there are some tools based on Java. In general, in order to implement the semantic analyzing process, the action of Yacc written as a fragment of C program is used. In many cases, the symbol table is also implemented from scratch. When we want to implement a compiler with these tools, we have to write a fragment that expresses type checking and/or adding symbols to the symbol table. But, there are many resemble production rules and its semantic actions in a grammar. There are very few researches for semantic analyser generator from the view point of the above although up to now, there are many individual research of each topic of semantic analysis. In this presentation, we mention about automatic generation of compiler front end that includes automatic generation of semantic analyser from the view point of type check and symbol table processing.

（平成 17 年 3 月 17 日発表）

† 筑波大学
University of Tsukuba