

氏 名	おお 大 まき 蒔 かず 和 ひと 仁
授 与 学 位	工 学 博 士
学位授与年月日	昭和 54 年 3 月 27 日
学位授与の根拠法規	学位規則第 5 条第 1 項
研究科，専攻の名称	東北大学大学院工学研究科 (博士課程) 情報工学専攻
学 位 論 文 題 目	トランスレータ作成システムに関する研究
指 導 教 官	東北大学教授 木村 正行
論 文 審 査 委 員	東北大学教授 木村 正行 東北大学教授 城戸 健一 東北大学教授 星子 幸男 東北大学助教授 丸岡 章 東北大学助教授 伊藤 貴康

## 論 文 内 容 要 旨

### 第 1 章 序 論

近年，計算機システムが巨大化するに従い，それを支えるソフトウェアも複雑化している。そのため，ソフトウェアの作成を出来るだけ自動化することが望まれている。その中にあって，トランスレータは対象とする問題が比較的明確であるため，早くからその作成自動化が考えられてきた。ここでトランスレータとは，ある言語で書かれたプログラムを別の言語で書かれたプログラムに変換するソフトウェアのことである。トランスレータを自動的に作成するシステムを，TWS (Translator Writing System の略) とよぶ。

本論文は，TWS に関する研究報告である。

本論文では，第 2 章で，ある考え方に基づいた TWS を提案し，第 3 章で，そのシステムの概要を述べる。次に第 4 章で，本論文の TWS も含め，多くの TWS に於て生ずる問題の一つを形式化して考察する。

### 第 2 章 トランスレータ作成システムの基本構想

トランスレータの入力となる言語，出力となる言語を，それぞれ原始言語，目的言語とよぶ。通常，TWS を用いて原始言語を S とするトランスレータを作成するとき，その TWS に，S



の“構文”を文脈自由文法<sup>(2)</sup>（以下  $cfg$  または単に文法とよぶ） $G_S$  で与え、 $S$  の“意味”を  $G_S$  の各生成規則に対応づけた何らかの操作の仕方として与える。従来の  $TWS$  では、作成されるトランスレータの目的言語として低級言語を想定するシステムであったりして<sup>(5)</sup>、必ずしも使いやすいシステムにはなっていない。しかし、目的言語  $T$  の構文を  $cfg\ G_T$  で与えることのできるトランスレータを作成することを想定すると、 $S$  の“意味”を、 $G_T$  による導出木を用いて表すことができ、意味の記述が容易になることが期待できる。

そこで本論文では、 $S$  も  $T$  もその構文が、それぞれ  $cfg\ G_S, G_T$  で与えられる場合を想定した  $TWS$  を提案する。

まず、本論文の  $TWS$  により作成されるトランスレータ  $T_r$  が、原始言語  $S$  のプログラム  $P_S$  を、目的言語  $T$  のプログラムに、どのようにして変換するのか、その方法を述べる。

$T_r$  は構文解析部と意味解析部の二つの部分から成る。 $P_S$  はまず  $T_r$  の構文解析部に入力される。構文解析部は、 $G_S$  による  $P_S$  の導出木をつくる。この導出木の各節点には  $G_S$  の記号が割り当てられている。これらの記号は次の2つの属性<sup>(6)</sup>を持っているものとする。

- (1)  $G_T$  による導出木を値として持つ属性
  - (2) 整数と文字列から成る組を値として持つ属性
- 前者をタイプ1、後者をタイプ2とよぶことにする。

$T_r$  の意味解析部は、まず  $P_S$  の導出木の葉の属性値を求め、その木の上を、子供の属性値よりある規則に基づき親の属性値を求めていき、最後に根の属性値を求める。この根のタイプ1の属性値が  $P_S$  と等価な  $T$  のプログラムの  $G_T$  による導出木となるように、 $T_r$  は動作する。

このような動作をするトランスレータを本論文の  $TWS$  に出力させるのに、 $TWS$  には  $S$  を定義する文法  $G_S$ （構文）及び子の属性値から親の属性値を決める規則（意味）を入力する。

そのために、まず  $G_S$  を定め、その各記号にどのような属性を持たせるのかを定義する。

次に  $G_S$  の各生成規則に対して、その右辺の記号に割り当てた属性値から、左辺の記号に割り当てる属性値を決める規則を入力する。

$G_S$  の生成規則  $X \rightarrow X_1, X_2 \dots X_n$  について、 $X$  の属性値の割り当て方は、次の(1), (2)により定める。

- (1) タイプ1の属性値の割り当て方

必要なら、 $G_T$  の導出木の部分木（これを生成木とよぶ）をいくつか生成し、生成木の葉のいくつかを、 $X_1, X_2, \dots, X_n$  のタイプ2の属性値でおきかえる。この生成木と、 $X_1, X_2, \dots, X_n$  のタイプ1の属性値（ $G_T$  による導出木の部分木）を接続して一つの木とし、 $X$  のタイプ1の属性値として割り当てる。

- (2) タイプ2の属性値の割り当て方

$X_1, X_2, \dots, X_n$  のタイプ2の属性値に適当な演算を施して、 $X$  のタイプ2の属性値として割り当てる。

$S$  の意味記述とは、上の(1), (2)を  $G_S$  の各生成規則に対して指定したものである。この記述のための言語を本論文の  $TWS$  において新しく定義し、 $SDL$  と名付けた。 $SDL$  は、Semantic



Description Language の頭文字より取った。上記(1)の記述が、本論文のTWSの最大の特徴である。

### 第3章 トランスレータ作成システムの構成

本論文のTWSは、図1のようになっている。

まずSの構文 $G_S$ は、ANALYZERとよばれるソフトウェアに入力される。ANALYZERは、 $G_S$ が(2, 1: 1, 1) MSPとよばれる文法族に属しているかどうか調べ、属しているときは構文解析のために必要な解析表を出力する。この構文解析表と、予め作成してある構文解析部本体とをつなぐと、Sの $G_S$ に基づく構文解析部が得られるようになっている。この手法は、XPLコンパイラジェネレータシステム<sup>(1)</sup>で用いられたものと全く同一である。

一方、SDLで書かれたSの意味記述は、SDLトランスレータに入力され、Sの意味解析部が出力される。

こうして得られたSの構文解析部と意味解析部は、結合手続きによって結合され、Sを原始言語とするトランスレータが出力される。

本論文のTWSは、東北大学工学部情報工学教室の計算機FACOM 230-38S上に作成した。使用言語はPL/1である。各部分の大体の規模は次の通りである。

- (1) ANALYZER カード3,700枚, 90K語
  - (2) SDLトランスレータ カード3,000枚, 100K語
  - (3) 構文解析部本体 カード600枚
- ただし、1語16ビットである。

### 第4章 文法の非交差性

本論文のTWSも含めて、多くのTWSを用いてある言語のトランスレータを作成するとき、その言語の構文を文脈自由文法で、また意味を各生成規則に対応した何らかの操作の仕方として与える。一般に、これらのシステムで受け入れ可能な文法の範囲は限定されているため、与えられた文法がその範囲外であるとき、その文法を受け入れ可能なものに変更する必要がある<sup>(1),(2)</sup>。しかし文法の変更に伴って生ずる意味記述の変更が、余り大巾なものにならないことが望ましい。

この章では、それぞれの文法に基づいて与える意味記述が大巾には違わないという文法間の関係を形式的に定義し、それを“非交差的”とよぶ。更に、この文法間の関係から文法族の間にも非交差的という関係を定義し、LR文法族とよばれる文法族の間で、この関係が成立するか否かを論ずる。

〔定義1〕  $T_1$ と $T_2$ と導出木とする。 $T_1$ と $T_2$ が非交差的である( $T_1 \sim T_2$ と表す)とは、次の条件を満足することである。

- (1)  $T_1$ と $T_2$ は、導出する語が等しい。

その語を $a_1 a_2 \cdots a_n$ とする。

- (2)  $a_{i_1} \cdots a_{j_2}$ と $a_{i_2} \cdots a_{j_2}$ を、それぞれ $T_1$ と $T_2$ の句(phrase)とすると、 $i_1 < i_2 \leq j_1 <$



$j_2$  または  $i_2 < i_2 \leq j_2 < j_1$  となることがない。

〔定義 2〕  $\text{cfg } G_1$  と  $\text{cfg } G_2$  が非交差的である ( $G_1 \sim G_2$  と表す) とは、次の条件を満足することである。

- (1)  $G_1$  と  $G_2$  の生成する語が等しい。
- (2)  $G_1$  による任意の導出木  $T_1$  に対して、 $G_2$  による導出木  $T_2$  が存在して  $T_1 \sim T_2$  となる。
- (3)  $G_2$  による任意の導出木  $T_2$  に対して、 $G_1$  による導出木  $T_1$  が存在して  $T_1 \sim T_2$  となる。

〔定義 3〕  $C_1$  と  $C_2$  をそれぞれ文法族とする。 $C_1$  が  $C_2$  に非交差的であるとは、 $C_1$  に族する任意の文法  $G_1$  に対して、 $C_2$  に属する文法  $G_2$  が存在して  $G_1 \sim G_2$  となることである。

ここで、上のように非交差性を定義した理由を、例を用いて説明する。

いま、受け入れ可能な文法族が限定されている TWS を用いて、+ (和) と \* (積) の二つの演算から成る算術表現をもつ言語を原始言語とするトランスレータを作成したいとする。そして、この言語に対して + よりも \* の方が演算順位が高くなるように意味記述を与えるものとする。

そして、この原始言語の構文を与える文法  $G_1$  による、算術表現  $a + b * c$  に対する導出木が図 2 (a) に示すようなものであるとき、この①の部分で \* に対応する目的 プログラムを生成し②の部分で + に対応する目的プログラムを生成するように、 $G_1$  の生成規則に対する操作の記述を与えれば、+ と \* に関する上述の演算順位が実現される。

ところで、 $G_1$  が TWS の受け入れ可能な文法族に属していないとき、 $G_1$  と同一の言語を生成し、しかも TWS が受け入れ可能な文法  $G_2$  を新しく構成しなければならない。

ところが、もし  $a + b * c$  に対する  $G_2$  による導出木が図 2 (b) のようになるるとすると、③の部分で + に対応する目的プログラムを生成すると、+ と \* の演算順位は保たれなくなる。従って演算順位を保とうとすれば、 $a + b * c$  の全ての情報を④の部分で参照しなければならず、 $G_2$  の各生成規則に対して、対応する操作を独立に記述するということが、やりにくくなる。しかし、もし  $G_2$  による導出木が図 2 (c) のようになるるとすると、⑤の部分で  $a + b * c$  全体を参照することができるので、+ と \* の演算順位を保つような目的プログラムを生成できる。

ところで、図 2 の (a) と (b) は交差的であるが、(a) と (c) は非交差的である。このように、 $G_1$  と  $G_2$  が非交差的であるなら、 $G_1$  に基づいて与えた意味の記述を余り変更することなく  $G_2$  に対しても与えることができると考えられる。これが非交差性を考えた背景である。

本論文では、 $LR(k)$ 、 $SLR(k)$ 、 $(m, k) \text{ BRC}$ 、 $LL(k)$  とよばれる各文法族の間で非交差性が成立するか否かを考察した。これらは  $LR$  文法族とよばれ、何れの文法族についても任意の文法がその文法族に属すかどうかを調べるアルゴリズムが存在し、属すときは、その文法に対する効率の良い構文解析手続きを構成できることが知られている<sup>[2]</sup>。

得られた結果は次のとおりである。

〔定義 1〕 どんな  $LR(k-1)$  文法にも非交差的ではないような  $LL(k)$  文法が存在する。  
( $k \geq 1$ )

〔定義 2〕  $LR(k)$ 、 $(m, k) \text{ BRC}$ 、 $SLR(k)$  の各文法族は、どの二つも互いに非交差的である。(  $k \geq 0$  ,  $m \geq 1$  )



〔系1〕  $(m, k)$  BRC, SLR  $(k)$  の各文法族の中には, どんな LR  $(k-1)$  文法にも非交差的ではない文法が存在する。 ( $k \geq 1, m \geq 1$ )

## 第5章 結 論

本論文第2章で提案したTWSを用いれば, プリプロセッサのような高級言語間の変換を行なうトランスレータは容易に作成できる。しかし, 特殊なデータ型をもつ言語を原始言語とするトランスレータを, 本論文のTWSのような方式でどの程度迄作成可能であるかは検討の余地がある。

第4章の結果より, 受け入れ可能な文法がLR文法族に限定されているTWSを使うとき, 非交差性を保存しては, その文法族の文法に変換できない文法が存在すること, 及びLR文法同志では, 非交差性を保存して変換できるときと, そうでないときがあるということが判った。

## 参 考 文 献

- (1) W.M. McKeeman et al.: "A Compiler Generator", Prentice-Hall (1970).
- (2) A.V. Aho et al.: "The Theory of Parsing, Translation, and Compiling", vol. 1, vol. 2, Prentice-Hall (1972, 1973).
- (3) S.C. Johnson: "YACC-Yet Another Compiler Compiler", CSTR 32, Bell Laboratories (1974).
- (4) J.A. Feldman: "A Formal Semantics for Computer Languages and Its Application in a Compiler-Compiler", Comm. ACM., vol. 9, p. 3 (1966).
- (5) C.H.A. Koster: "Using the CDL Compiler Compiler", Lecture Notes in Computer Science, 21, Springer Verlag (1974).
- (6) D.E. Knuth: "Semantics of Context-Free Languages", Math. Syst. Th., vol. 2, p. 127 (1968).



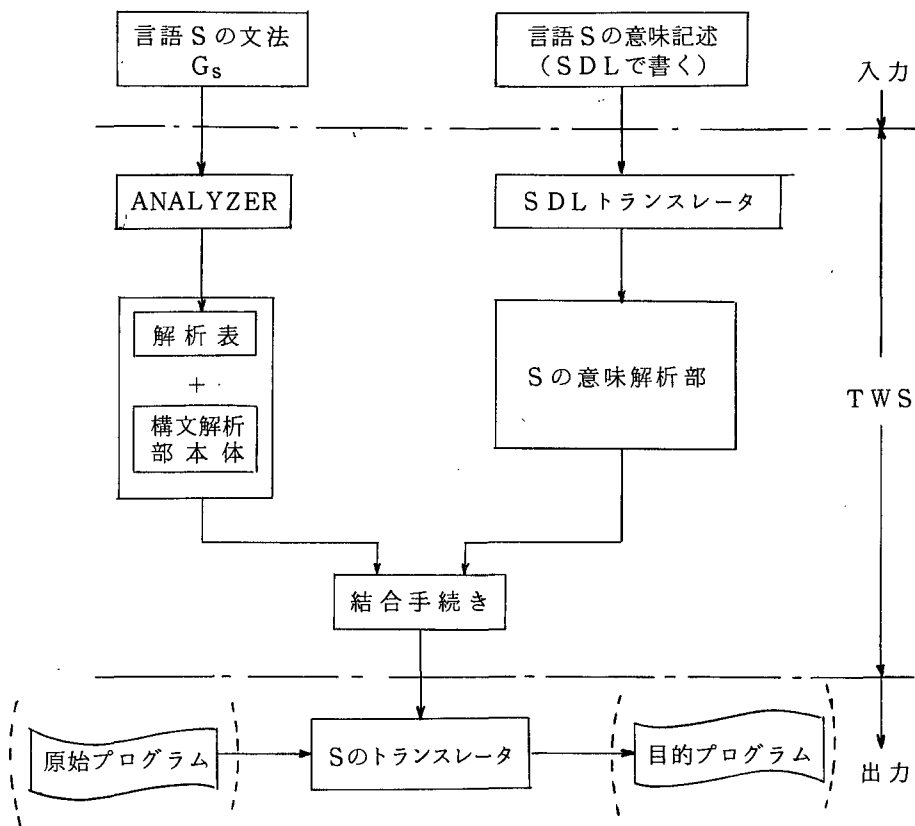


図 1. システム説明図

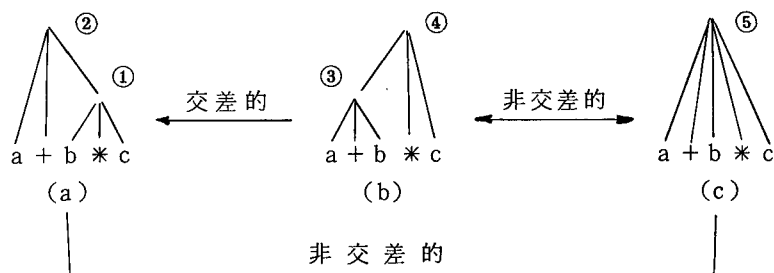


図 2. 非交差性説明図



## 審 査 結 果 の 要 旨

近年、電子計算機の大形化にともない、そのソフトウェアも巨大なものとなりつつある。そのためソフトウェアの開発を自動化し、その生産性を向上させることが望まれている。トランスレータは、計算機言語間の変換を行う基本的なソフトウェアであり、これまでにその自動作成を目指していろいろな研究が行われてきているが、必ずしも満足すべきものが得られていない。著者は、トランスレータの自動作成を行う一方式を提案し、それに基づく実験システムを作成した。さらにトランスレータ作成システムの使用の際に生ずる一つの理論的問題を考察している。本論文はそれらの成果をまとめたもので、全文5章と付録よりなる。

第1章は序論である。第2章では、著者が作成したトランスレータ作成システムTWSの設計法の基本的な考え方を述べ、ついで、このTWSを用いて試作したトランスレータについて説明している。TWSへの入力は、作成すべきトランスレータの原始言語の構文と意味からなる。ここで、構文はBNF (Backus Normal Form) で書かれ、意味は本研究で新しく定義された意味記述用言語SDLを用いて表わされる。構文および意味の記述が独立しており、しかも簡潔な意味記述法を用いていることが本システムの特徴である。このような入力をTWSに入れると、構文主導型と呼ばれる技法に基づき動作する所望のトランスレータが得られる。また、代入文、条件文および繰り返し文を含む一つの言語をPL/1タイプの言語に変換するトランスレータを本システムを用いて実際に作成し、本論文で提案した方式の有用性を確かめている。

第3章では、本論文のTWSのFACOM 230-38S上での実現の仕方について詳細に述べ、ついで、意味記述用言語SDLの仕様について説明している。

第4章では、トランスレータ作成システムを使用する際に生じる一つの問題を論じている。まず非交差的と呼ばれる文法間の関係を定義し、この関係が意味の記述が大巾には変わらない文法間に成立するものであり、一般にトランスレータ作成システムを論じる上で重要であることを示している。ついでLR文法の代表的なクラスの間に非交差関係が存在するか否かを論じ、LR(k), SLR(k), (m, k) BRCの各文法族はどの二つも非交差関係にあることを証明している。

第5章は結論である。付録では本論文の方式に基づきFACOM 230-38S上で試作した、TWSの計算機による出力リストとその説明を与えている。

以上要するに、本論文はトランスレータの自動作成の一方式を提案し、その有用性を実験システムを作成する事により確めるとともに、これに関する理論的考察を行ったもので、情報工学に寄与するところが少なくない。

よって、本論文は工学博士の学位論文として合格と認める。