

## パターン記述されたデータによる LISPプログラムの変換

5Y-6

沢井真二†, 太田義勝†, 吉田雄二†, 福村晃夫‡

(†名古屋大学)

(‡中京大学)

### 1. はじめに

LISPは人工知能や知識工学の研究に広く用いられているプログラミング言語である。LISPは柔軟なデータ構造と、プログラム自体もデータとして扱うことが出来るという特徴を持ち、LISPプログラミングの開発支援システムの記述言語としてもよく用いられる。LISPは現在、種々の方言が存在し、必ずしも相互の互換性がよいとは言えない。本稿ではLISPプログラムの互換性の問題に対処するため、あるLISPの仕様に従って書かれたプログラムを他のLISPの仕様に従うプログラムへ変換を行なう手法として、パターン記述された変換規則による変換を提案する。

### 2. LISPプログラムの変換

LISPの関数は関数名を表すシンボルがCARに、その引数をCDRに持つリストの形になっている。一般には、この引数はLISP関数のリストになっており、変換を行なう場合にはこの引数も含めて変換する必要がある。一方、マクロ関数や特殊形式などのように、引数の属性が制限されているものもあり、必ずしも全ての引数を調べる必要はない。ここで提案する手法は、これらを考慮して以下の方針により構成される。

- ・与えられたLISP関数のリストを他のLISPの仕様による関数のリストに書き換える事を基本とする。
- ・引数に関する仕様をパターン記述によって表現し、これに従ってLISPプログラムを解析する。

引数に関する情報が記述されている部分(データ部)とプログラムの変換を実際に行なう部分(トランスレータ部)を独立させる。そして、実際の変換の方法もこのデータ部に記述しておく。データ部を取り替える事により、同じトランスレータが各種のLISP間の変換に対応出来るようにする。

### 3. プログラム変換の記述

プログラム変換の規則が記述されているデータ部は解析用データと変換用データよりなる。それぞれのデータは、以下に述べるように構成されている。

#### 3.1 解析用データ

解析用データは入力されたプログラムをトランスレータが解析する際に参照するデータであり、パターンの記述により表現されている。パターンの構文は図1のようになる。

```

<pattern> ::= <lambda-list>
              | <repeat-pattern>
              | <optional-pattern>
              | <concatenate-pattern>
              | <list-pattern>
              | <constant-pattern>
              | <sexpr-type>
<sexpr-type> ::= <cons-type>
                | <atom-type>
<atom-type> ::= <symbol-type>
               | <other-type>

<repeat-pattern> ::= (repeat <pattern>)
<list-pattern> ::=
  (list <pattern>... <pattern>)
<constant-pattern> ::= (quote <pattern>)
<lambda-list> ::= (lambda-list)
<sexpr-type> ::= (sexpr-type)
<symbol-type> ::= (symbol-type)
  
```

図1. パターンの構文

それぞれのパターンの意味は以下の通りである。

Conversion of lisp programs based on pattern description

Shinji SAWAI†, Yoshikatsu OHTA†, Yuuji YOSHIDA†, Teruo FUKUMURA‡

† Nagoya University, ‡ Chukyo University

<lambda-list>  
 ... ラムダリスト (変数の設定部)  
 <optional-pattern>  
 ... 一つの省略可能なパターン  
 <concatenate-pattern>  
 ... いくつかのパターンの並び  
 <list-pattern> ... パターンのリスト  
 <repeat-pattern> ... パターンの繰り返し  
 <constant-pattern> ... 定数リスト  
 <sexpr-type> ... 一つのS式  
 <cons-type> ... コンス・セル  
 <atom-type> ... アトム  
 <symbol-type> ... 変数・関数名  
 <other-type>

... <symbol-type>以外のアトム  
 解析用データの例を図2に示す。記述中の<...>は各パターンに対応する入力プログラムの部分を引用するための一時変数を表す。

```
(DEFUN {symbol-type <var1>
       {lambda-list <var2>
       {repeat (s-expr) <var3>})

(MAP {sexpr-type <var1>
     {sexpr-type <var2>})

(PUTPROP {sexpr-type <var1>
         {sexpr-type <var2>
         {sexpr-type <var3>})

(ASSQ {sexpr-type <var1>
      {sexpr-type <var2>})
```

図2. S式によるパターンの記述

### 3. 2 変換用データ

変換用データは解析結果より出力プログラムを生成する際に用いるデータであり、出力プログラムを生成する規則をLISPの“形式”で記述したものである。図2の例に対応する変換用データを図3に示す。

```
(list 'DEFUN <var1> <var2> <var3>)
(list 'MAPL <var2> <var1>)
(list 'SETF
      (list 'GET <var1> <var3>)
      <var2>)
(list 'ASSOC <var1> <var2>
      ':TEST '(FUNCTION EQ))
```

図3. データの例

記述中の<...>は、解析部で得られた対応する入力プログラムの部分を表す。この変換

用データを評価することにより変換されたプログラムが生成される。

## 4. 変換システム

プログラム変換システムの概念図を図4に示す。実際に変換を行なうのはトランスレータの部分であり、LISP1の関数仕様に従って書かれたプログラムを入力データとして読み込み、別のLISP2の関数仕様に従うプログラムへと書き換え、これを出力する。解析・変換用データ部は図2及び図3のようなデータで構成されているが、これは実際にはLISP1の各関数名シンボルの属性リスト中に定義されている。

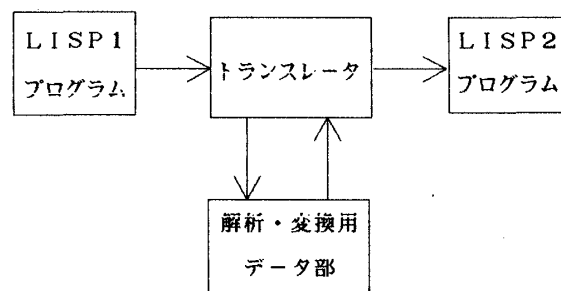


図4. 変換システムの概念図

## 5. 終わりに

本研究は名古屋大学大型計算機センターのFACOM M-780のUTILISPを使用している。現在、UTILISPで書かれたプログラムのCOMMONLISPのプログラムへの変換を行なっている。パターン記述の確立とシステムの完成は今後の課題である。

謝辞 日頃御指導頂く名古屋大学渡邊豊英助教授、ならびに吉田研究室の皆様へ深く感謝します。

### 参考文献

- (1) UTILISPマニュアル, 富士通編
- (2) Guy L. Steele Jr. : "COMMON LISP THE LANGUAGE", Digital Press
- (3) 川村他, : "LISPのためのパターン指向型プログラミングシステム", 信学論(D), J70-D, 6, pp. 1149-1156 (昭62-6)