

1B-5

Prolog言語教育支援システムにおける
アルゴリズム認識のための
知識ベース作成支援

成田東吾 武田正之 井上謙蔵
(東京理科大学)

1. はじめに

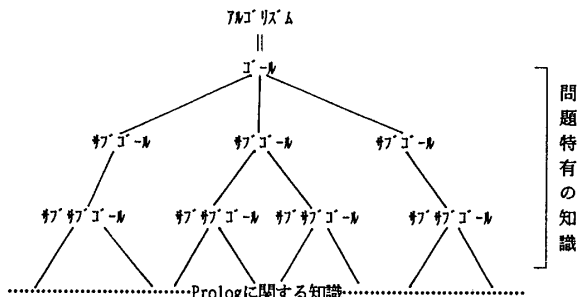
我々は、数年前から知識工学の技法を用いたICAIシステムとして、Prolog言語を学び始めて間もない学生を対象としたProlog言語教育支援システムの研究を行ない、そのプロトタイプを実現している。

本システムでは、システムが最初から用意している問題を学生に与え、学生の作成したプログラムを解析してその学生のレベルに応じたアドバイスを与えることにより、学習を進めて行くことができる。しかし、システムは一種類の問題しか用意しておらず、学生に十分な学習をさせることができなかつた。より実用的なシステムをめざして、問題数を増やして学生のレベルに応じた学習ができるように拡張を行なった。それとともに、学生の作成したプログラムを解析するためにシステムが必要とする知識ベースを、問題に応じて作成する必要がある。しかし、知識ベースの作成は非常に困難なものであった。本稿では、システムに新たに付加された、知識ベースの作成を容易にする機能について述べている。

2. Prolog言語教育支援システムの概要

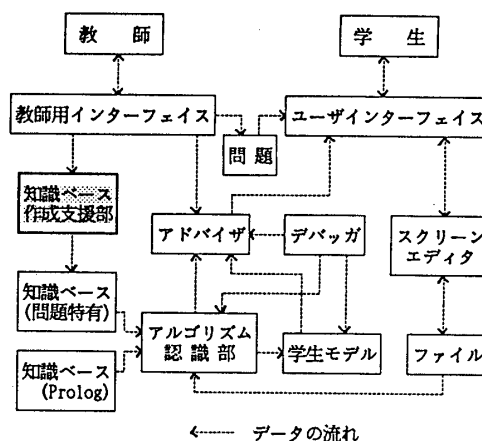
現在実現されているProlog言語教育支援システムは、学生の作成したプログラムを解析してアルゴリズムの認識を行ない、その認識結果に応じたアドバイスを学生に与えるものである。学生の作成したプログラムの解析には、テンプレートによるプログラムの分類を行なうのではなく、与えられた問題をプログラムするときにキーとなる演算子や述語の探索、述語の呼び出し関係等を調べる方法を用いている。このためにシステムが有している知識が知識ベースである。

知識ベースには、Prolog言語に関する知識と、それぞれの問題に特有な知識の2つがある。Prolog言語に関する知識は、どの問題についても共通に使用することができる。問題に特有の知識ベースは、その問題を解くためのアルゴリズムごとに、アルゴリズムを根とし、Prologに関する知識を葉にもつ、(図_1)のような木構造で表わされる。



(図_1) 知識ベースの構成

新たな問題をシステムに用意させようとした場合、その問題に特有な知識ベースも用意する必要がある。たとえば、「2つの正整数の最大公約数をもとめる述語をつくれ」という問題を用意しようとする。ユークリッドの互除法のアルゴリズムに関して、知識ベースを作成する場合、(例_1)のようなPrologに関する知識ベースの一部があるとすると、(例_2)のような問題特有の知識ベースを作成しなければならない。考えられるアルゴリズムのそれぞれについて、同様に知識ベースを作成しなければならない。この知識ベースはシステムにかなり関係しており、システムの内部を把握していないと、作成するのが困難なものである。そこで、知識ベースの作成を支援する機能を持つ知識ベース作成支援部をシステムに付加した。(図_2)



(図_2) Prolog言語教育支援システムの構成

```
(例_1) /* Prologに関する知識の一部 */

/* 再帰呼び出しを行なっているかどうかのチェック */
recur(Pred):- program(No, (Pred:-Body)),
               functor(Pred, F, A),
               there(F/A, _, Pred, No).

/* ある述語の本体にGoalが存在するかどうかのチェック */
there(Goal, Pred, Head, No):- program(No, (Head:-Body)),
                              there1(Body, Goal, Pred).

there1(P, _, _) :- var(P), !, fail.
there1(P, F/A, P) :- functor(P, F, A), !.
there1(P, _, _) :- functor(P, _, 0), !, fail.
there1([P|Q], G, R):- !, (there1(P, G, R);
                        there1(Q, G, R)).
there1(P, G, R) :- P=..[_|Args],
                  there1(Args, G, R).
```

(例_2) /* ユークリッドの互除法に関する知識 */

```
prob_rule((rule(gcd)<<comb, 0:-gcd_1, gcd_2, gcd_3, gcd_4)).
/* ':-'または'is'を用いている */
prob_rule((gcd_1<<non, 90:-there(':-'/2, P, H, N),
                                (arg(1, P, 0);arg(2, P, 0)), !)).
prob_rule((gcd_1<<non, 90:-there('is'/2, P, H, N),
                                arg(1, P, 0), !)).

prob_rule((gcd_1<<non, 0)).
/* 'mod'を用いている */
prob_rule((gcd_2<<non, 100:-there(mod/2, _, H, N), !)).
prob_rule((gcd_2<<non, 0)).
/* 'mod'を含む述語を再帰呼び出ししている */
prob_rule((gcd_3<<non, 100:-there(mod/2, _, H, N),
                                recur(H), !)).

prob_rule((gcd_3<<non, 30)).
/* '!'が用いられている */
prob_rule((gcd_4<<non, 90:-there(mod/2, _, H, N),
                                there(':-'/2, _, H, N),
                                there(!/0, _, H, N), !)).
prob_rule((gcd_4<<non, 80)).
```

(例_3) /* ユークリッドの互除法の記述例 */

- ① $\text{gcd}(X, Y) = X \bmod Y = 0 \Rightarrow Y \parallel X \bmod Y \neq 0 \Rightarrow \text{gcd}(Y, X \bmod Y)$
- ② $|\text{gcd}(X, Y)| = 1$

(例_4) /* 自然数発生の述語の記述例 */

- ① $\text{int} = 1 \parallel \text{int} + 1$
- ② $\forall X, Y \in \text{int} [X < Y \Leftrightarrow X \triangleleft Y]$

3. 知識ベース作成支援

問題特有のアルゴリズムは、(図_1)のような木構造をしており、それぞれのアルゴリズムごとに作成する必要があるため、木構造の根の部分となるアルゴリズムが与えられると、そのアルゴリズムから問題特有の知識ベースを作成する方法を考えた。

ユークリッドの互除法のアルゴリズムの記述を(例_3)のように表現する。作成する述語を関数の様に考える。(例_3)の①式は、 $X \bmod Y = 0$ が成り立つならば、 Y という値を返し、これが解となる。そうでないときには、 $X \bmod Y \neq 0$ が成り立つならば、 $\text{gcd}(Y, X \bmod Y)$ を再帰的に呼び、その値を返すものである。(例_3)の②式は、バックトラックの情報を表わすために用いており、これは、バックトラックしても解を1個しか発生しないことを表わすものである。

また、バックトラックによって次々と解を発生させるような述語について考える。(例_4)は、バックトラックによって、1から順々に整数を発生させる述語intのある1つのアルゴリズムを記述したものである。(例_4)の①式は、(例_3)と同様に、述語を関数的に表現したものであり、intは、値1を返すか、intを再帰的に呼び、返ってきた値に1を加えて返すものである。(例_4)の②式は、バックトラックの情報を表わしており、intによって得られた解の集合に順序を付けるものである。intによって得られたXとYについて、 $X < Y$ が成り立っているときには、 $X \triangleleft Y$ という順序関係が成り立ち、これは、バックトラックにより次々とintを発生させたときには、XのほうがYよりも早く発生することを表わしている。

以上の様なアルゴリズムの簡単な記述から問題特有の知識ベースを作成するを行なった。これにより、問題特有の知識ベースを容易に生成することができるようになり、効率良く新たな問題をシステムに用意することができるようになった。

4. おわりに

本システムは、PC-9801上のProlog-KABAで実現されている。初期のものより、より実用的なシステムに拡張された。今後、実際のProlog言語学習で使用して、より良いシステムにすることが望まれる。

【参考文献】

- [1] 原稔, 武田正之, 井上謙蔵:
Prolog言語教育支援システムの実現,
情報処理学会第32回全国大会論文集 (pp.1227-1228)