7Q-6

論理プログラムの前向き推論による一計算法

井手 博康* 中村 克彦**
(*東京電機大学大学院 **東京電機大学理工学部)

1. まえがき

Prolog に用いられている論理プログラムの計算法は，深さ優先探索およびＳＬＤ導出による後向き推論にもとずいている．この方法はプログラムの制御がしやすく，システムの構成も比較的簡単であるが，計算が無限ループに陥いる可能性があり，また後戻りによって計算の中間結果を捨ててしまうので同一の目標の評価を繰り返すことがある．さらに，質問（目標節）によって計算の入出力が行われるために，入力データはいくつかのリストまたは項に制限される．入力データの集合を事実の形式で公理系（プログラム）の一部に与えることもできるが，その場合一般に計算効率が悪い．これまで前向き計算は構文解析[1]やプロダクションシステム[2]などに使用されているが，一般の計算法については余り調べられていない．

この報告では単位消去法と呼ばれる次のような新しい計算法を紹介する．

(1) プログラムは規則の集合からなる．

(2) 入力データは単位節の集合として与えられる．

(3) 計算は，単位節と規則との単位融合(unit resolution)により新しい規則および単位節を生成することで進められる．

(4) 与えられた条件に適合する単位節が生成されたとき，その単位節が計算の解または出力である．

2. 単位消去法

単位消去法では，各単位節と全ての規則との融合を試みてその単位節を消去し，この操作を新たに生成された単位節と規則に対しても繰り返す．

[計算手順]

初期単位節（入力データ）の集合 U_0，規則の集合 R_0，目標（単位節）G に対し，２種類の計算結果を出力する．以下に使われる変数 k の初期値を 0 とする．

(1) U_k のすべての単位節と，R_k のすべての規則との融合を試みる．そのために，まず U_k のある単位節と R_k のすべての規則との融合を試みる．そして U_k のつぎの単位節と生成された規則を含むすべての規則との融合を試み，これを U_k のすべての単位節について繰り返す．どの規則とも融合が成功しない単位節があるときは，これを出力し取り除く．U_{k+1} をこの結果生成されるすべての単位節の集合，R_{k+1} を R_k に生成されるすべての規則を加えた集合とする．

(2) U_{k+1} に目標 G と融合する単位節があれば，それを出力する．

(3) U_{k+1} が空ならば計算終了．その他の場合 k を k+1 として(1)へ戻る．

この計算法には前述の (1) と (2) の２種類の出力があり，それぞれ例外出力と正規出力と呼ぶ．

3. 単位消去法の拡張

実際の単位消去法による計算の効率化と能力の拡張のために，次のように拡張する．

(1) 組み込み述語と下向き計算法の呼び出し．

数値計算や変数に対する条件を与えるために組み込み述語の呼び出しが必要である．また同じ形式を用いて，ある目標を Prolog と同様に top-down 式に評価することが可能となる．

(2) 規則の条件部の述語項の順序づけ．

順序をつけることにより，記憶領域の効率化や探索の制限が可能になる．順序を決定できない規則については考えられるすべての順序をもつ規則を用意しておく．

(3) 深さ優先探索

問題によっては深さ優先探索で計算する方がよいものもあるのでその指定を可能にする．

(4) 決定性の導入

処理効率上，Prolog の cut に類似した概念の導入が必要となる．演算子 | を導入し，一般の規則を

$$p_1, p_2, .., p_k, |, p_{k+1}, .., p_n \rightarrow q.$$

という形式とする．この節において $p_1, .., p_k$ が満足された後は $p_{k+1}, .., p_n$ の各述語項はそれぞれただ１つの単位節としか融合しない．この指定により，解に無関係な中間的規則を消去し，処理時間と記憶領域の効率を上げることができる．

4. 例題

(1) フィボナッチ数列の計算

次のプログラムはフィボナッチ数列を発生する．

```
fib(0,1).
fib(1,1).
fib(N,X),|,M is N+1,
    K is N+2,fib(M,Y),A is X+Y -> fib(K,A).
```

ここで，fib(N,X) はフィボナッチ数列の N 番目は X

A computation method of Logic Programs by forward reasoning
Hiroyasu IDE, Katsuhiko NAKAMURA
Tokyo DENKI University

であることを示し，算術計算は組み込み述語を呼び出して行う．目標を fib(X,Y) と与えると，次々と解を発生する．決定性の指定によりこのプログラムでは生成された中間的な規則や目標はすべて消去されるので，top-down 計算に比べはるかに効率的である．

(2) グラフ中のループの検出

有向グラフ中の閉じた経路を見つけるプログラムは次のように書ける．

```
t(X,Y)        -> p(X,Y,[Y|L],L).
t(X,Y),p(Y,Z,L1,L2),
 topdown(not(member(Z,[X|L1],L2)))
              -> p(_,X,Z,[Y|L1],L2).
t(X,Y),p(Y,X,L1,[])
              -> loop([Y|L1]).
```

ここで topdown(p) は，top-down 計算を指定する述語項とする．member(Z,L1,L2) は，Z が L1,L2 であらわされる差分リストに含まれていれば成功する．グラフのデータは

$t(v_1,v_2).\ t(v_3,v_4).\ \ldots\ t(v_{n-1},v_n).$

とあらわされ，例えば最初のデータは頂点 v_1 から頂点 v_2 に向かって辺があることを示す．目標を

```
loop(X).
```

とするとグラフ中の全てのループを見つけ出す．

(3) Prolog の interpreter の実現

Prolog の subset に対する top-down 方式の計算のための interpreter のプログラムは次の規則だけでよい．

```
solve([G|L2],X),clause(G,L1,L2) -> solve(L1,X).
```

Prologプログラムの各節 P :- $R_1,..,R_m$. は次のような単位節で表す．

```
clause(P,[R1,..,Rm|L],L).
```

また，Prologの質問 ? - $Q_1,..,Q_n$ は次のような単位節で表す．

```
solve([Q1,..,Qn],[X1,..,Xk]).
```

ここで，$X_1,..,X_k$ は質問中の変数列である．

目標を

```
solve([],[X1,..,Xk]).
```

とすると，計算結果が $X_1,..,X_k$ に代入される．

例えば，知られた append のプログラムを実行するにはこの interpreter に次の単位節を与えればよい．

```
solve([ap([1,2,3],[4,5,6],X)],[X])).
clause(ap([],X,X),[],[])).
clause(ap([A|X],Y,[A|Z]),[ap(X,Y,Z)|L],L)).
```

実行結果は

```
solve([],[[1,2,3,4,5,6]]).
```

となる．

5. Implementation

規則は一般に

$$p_1,p_2,..,p_k,|,p_{k+1},..,p_n \text{ -> } q.$$

とあらわされる．この節と単位節 p_1'の融合が成功すると代入θ_1 がつくられ，

$$(p_2,p_3,..,p_k,|,p_{k+1},..,p_n \text{ -> } q)\theta_1$$

という規則が生成される．p_k まで計算が進むと次の規則が生成される．

$$(|,p_{k+1},..,p_n \text{ -> } q)\theta_1,..,\theta_k$$

ここで演算子 | により，その後の述語項はただ1つの単位節としか融合しないので，新しい規則が生成されたとき元の規則が消去される．この操作の繰り返しにより p_n まで消去されると

$$(q)\theta_1,..,\theta_n$$

という単位節が導出される．深さ優先探索は，新たに生成された単位節を優先して使うことで実現する．

この方法では規則が次々と生成されるのでメモリ効率が悪いと思えるが，実際には生成される中間規則は元の規則へのポインタと変数環境，および融合が成功した位置 i($1 \leq i \leq n$)によって表現できる．

われわれのこの研究の第一段階として Prolog によって interpreter を作成した．Prolog では unification のための述語項の検索は述語の頭部にしか行われないため，規則の述語項を頭部にもつ節を用意する必要がある．そこで，

$$p_1,p_2,..,p_n \text{ -> } q.$$

という規則を次のように表現する．

$$p_1\text{:-}p_2,..,p_n,q.$$

この節と単位節 p_1'が unification に成功すると

$$p_2\text{:-}p_3,..,p_n,q.$$

という節が生成される．深さ優先探索や決定性は前述の場合と同様に実現する．

7. 結び

前向き推論による一計算法と implementation の概要について示した．現在，Prolog による実験システムを作成し処理方式の検討を行っている．さらに，構文解析やパターン認識などの多量のデータの解析を効率よく処理するための，本方式にもとづく言語とシステムを設計中である．

[文献]

[1] Matsumoto,Y.,A Parallel Parsing System for Natural Language Analysis,ICLP,1986

[2] Shintani,T.,A FAST PROLOG-BASED PRODUCTION SYSTEM KORE/IE,Logic Programming,Proc.Fifth Int.Conf.and Symposium,1988