

5 J-2

副作用を考慮した
幅優先導出による論理プログラムの実行富岡 雅巳 武田 正之 井上 謙蔵
(東京理科大学)

1. はじめに

近年、論理型言語の研究が盛んであり、その一つであるPrologも年々進歩しつつある。現行のPrologの弱点として、計算規則が均等(fair)でないことが挙げられる。このため、Prologプログラムは解探索の手順をも考慮しなければ、定理証明システムとして機能しない。しかし、Prologプログラムを均等な計算規則(Prologの計算規則でない規則)で実行させる場合、副作用の影響が問題となる。それはプログラムの副作用により現われる実行結果が、副作用発生の順序に依存するためである。本論文は、Prologプログラムに均等な計算規則を適用することを目的とする。内容は、まず、論理プログラムでの副作用について述べ、ゴールから複数のリテラルを選択する導出での副作用について論じる。また、任意の順序での発生が可能な副作用の関係を定義する。さらに、均等な計算規則として幅優先な導出方法であるBFNF導出を取り上げ、PrologプログラムをBFNF導出で実行する方法と、その実験について報告する。

2. 論理プログラムにおける副作用

本論文では、副作用はプログラムを取り巻く状況への作用として考える。副作用を受けた状況は、他の状況へと移行する。論理プログラムの実行はゴール中のリテラルとプログラム節から、新たなゴールを導出することにより行なわれる。ここでは、この導出の際に副作用が発生すると考える。一般のPrologにおいて、副作用のある導出は組み込み述語による導出に限られ、その導出は空節を導き、一度しか成功しない。

新たなゴールを導出する際に対象となるリテラルは、計算規則により選択される。よって、副作用の発生順、すなわち状況の移行は計算規則に依存する。プログラマが求める状況の移行をもたらす計算規則を方正(correctitude)であるという。

3. 幅優先導出と副作用

複数のリテラルからの導出を同時に行なうことは、非決定的な順序でそれらのリテラルを選択し導出を行なうのと同等と考えられる。よって、副作用のある複数の導出を、同時に行なえばその副作用は非決定的な順序で発生する。

一般のProlog処理系の導出はゴール中のただ一つのリテラルを選択し導出を行なう。これに対し、ゴール中の全てのリテラルを選択して導出を行なうものがBFNF導出とよばれるものである。BFNF導出は均等である。[有村89]

4. 順序任意関係

ここで順序任意関係を定義する。

$C(S)$: S のすべての要素が作用して到達する状況は作用の順に依存しない。

(4.1)

ここで S は副作用の集合である。

この関係 C は任意の順序で発生可能な副作用により満たされる。

副作用の集合 S_x, S_y が各々、関係 $C(S_x), C(S_y)$ を満たし、 $s_x \in S_x, s_y \in S_y$ が $C(\{s_x, s_y\})$ であっても $C(S_x \cup S_y)$ であるとは限らない。すなわち、同値関係という推移律が成り立たない。よって、関係 C を満たす複数の副作用の集合を一つにする場合、各々の集合の要素すべてについて関係 C を満たすか否かを判定する必要がある。

あらゆる状況で $C(S)$ である場合、 $\Box C(S)$ と記し、これを常順序任意関係とよぶ。また、ある状況で $C(S)$ である場合、 $\Diamond C(S)$ と記し、これを可順序任意関係とよぶ。

2つの副作用 s_x, s_y において、 $\Box C(\{s_x, s_y\})$ であれば、 $s_x \cdot s_y = s_y \cdot s_x$ である。この、 $\cdot$ は副作用の合成である。副作用を合成したものは副作用である。 $s_x \cdot s_y$ は副作用 s_y が発生した後、 s_x が発生するのと同じ副作用である。同様に n 個の副作用の集合 $S = \{s_1, s_2, \dots, s_n\}$ についても、 $\Box C(S)$ であれば、集合 S のすべての副作用の合成はいかなる順序でも、同じ副作用となる。

5. 複合副作用と順序任意関係

・選言的副作用

次の副作用を考える。

$$s_x = s_{x1}; s_{x2}; \dots; s_{xn} \quad (5.1)$$

この副作用 s_x は副作用 $s_{x1}, \dots, s_{xn}$ の内どれかが発生する副作用である。選言的副作用 s_x と、ある副作用 s_y が $C(\{s_x, s_y\})$ を満たすとは、 $\forall s_{xi} \ C(\{s_{xi}, s_y\})$ と同値である。

・連言的副作用

次の副作用を考える。

$$s_x = s_{x1} * s_{x2} * \dots * s_{xn} \quad (5.2)$$

この副作用 s_x は副作用 $s_{x1}, \dots, s_{xn}$ を合成した副作用である。順序については決定しない。連言的副作用 s_x と、ある副作用 s_y が $C(\{s_x, s_y\})$ を満たすには、 $s_{xi} \ C(\{s_{xi}, s_y\})$ であれば十分である。

選言的および連言的な副作用を複合副作用と呼ぶ。また、複合副作用でない副作用を基本副作用とよぶ。

複合副作用間の順序任意関係は、複合副作用を基本副作用に分解して、順序任意関係を見ることにより判定可能である。

6. PrologプログラムでのBFNF導出

Prologプログラム、すなわち left-to-right, depth-first なる計算規則が方正であるプログラムの、部分的なBFNF導出による実行について述べる。この部分BFNF導出とは、ゴール中の全リテラルを導出の対象とはせず、ゴール中の一部のリテラルについて導出を行なう導出である。

ここで、常順序任意関係を用いて、Prologのプログラム節のボディ部をBFNF導出で導出が可能なリテラル群に静的に分割することを考える。

ゴールを depth-first なる計算規則で実行する場合、選択されるリテラルからの導出が空節に至った後、残りのリテラルの導出が始まる。よって、ボディ部の分割はリテラルから空節に至るまで起こり得るすべての副作用（リテラルの副作用）についての順序任意関係に注目する必要がある。

Prologにおけるリテラルの副作用は以下に求められる。

(i) リテラルの副作用は、リテラルに頭部がユニファイ可能な節の副作用である。ユニファイ可能な節が複数あるならば、各々の節の副作用の選言的副作用である。

(ii) 節の副作用は、ボディ部の各リテラルの連言的副作用である。

(iii) (i)と(ii)よりトップダウンに副作用を求め、解析の枝がボディ部にリテラルの無い節に至り終わるか、既に解析した節の再帰呼び出しとなるリテラルに至り中断するまで続けられる。再帰呼び出しのリテラルの副作用はないものとして扱う。

再帰呼び出しのリテラルの副作用をないものとしても、静的な常順序任意関係の判定には影響がない。プログラム節 $(H : -B)$ のボディ部 $(B = b_1, \dots, b_n)$ の分割は以下に行なわれる。

I $L_1 = B$ とする。

II $L_i (=1_1, \dots, 1_m)$ を $L_{i+1} (=1_1, \dots, 1_{i-1})$, $L_{i+2} (=1_i)$, $L_{i+3} (=1_{i+1}, \dots, 1_m)$ に分割。ただし、 $\square C(\{s_1, \dots, s_{i-1}\})$ かつ $\sim \square C(\{s_1, \dots, s_i\})$ 。 (s_k : リテラル 1_k の副作用) 分割が出来なければ $B_i = L_i$ として終了。

III $L_{i+1}, L_{i+2}, L_{i+3}$ において、 L_{i+3} にリテラルが無ければ、 $B_i = L_{i+1}$, $L_{i+1} = L_{i+2}$ とし II へ戻る。

IV $L_{i+1} (=1_1, \dots, 1_{o-1})$, $L_{i+2} (=1_o, \dots, 1_{r-1})$, $L_{i+3} (=1_r, \dots, 1_m)$ において、 $\square C(\{s_1, \dots, s_{o-1}, s_r\})$ かつ $\sim \square C(\{s_o, \dots, s_{r-1}, s_r\})$ ならば $L_{i+1} = 1_1, \dots, 1_{o-1}, 1_r$ さもなくば、 $L_{i+2} = 1_o, \dots, 1_{r-1}, 1_r$ とし、III へ戻る。

これにより、分割されたボディ部 $(B_1, \dots, B_d)$ を持つ新たなプログラム節 $(H : -B_1, \dots, B_d)$ を得る。部分BFNF導出での実行方法は、リテラル群 B_i についてBFNF導出を行ない空節を導出するに至った後、 B_{i+1} についてBFNF導出を行なうものである。なお、上のリテラルの分割は宣言的

意味を重視した分割である。バックトラックにより副作用が何度も発生することを求める場合には、複数回成功となるリテラルと、副作用のあるリテラルとは別のリテラル群に分割する必要がある。

また、Prologにおけるバックトラックの影響のため導出に用いられるプログラム節に優先順位を設ける必要がある。リテラル A_i と節 C_1, C_2 の頭部が各々ユニファイ可能であり、Prologプログラムの実行において節 C_1 は C_2 よりも、優先的に導出に用いられるとする。この場合、節 C_1 の副作用と、同じく

C_2 の副作用が常順序任意関係にあるならば、節 C_1, C_2 は同じ優先順位を与えられ、さもなくば C_1 には C_2 より高い優先順位が与えられる。そして、導出に用いられる節は優先順位の高いものから用いられる。同じ優先順位の節が複数ある場合、それらは副作用に関して任意の順に用いることが可能である。

Prologプログラムの変換例

```
fib_write([1,1|X])←
  fib([1,1|X]),write_list([1,1|X]).
fib([A,B,C|D])←
  C is A+B,fib([B,C|D]).
write_list([]).
write_list([X|Y])←
  write(X),tab(1),write_list(Y).
↓
fib_write([1,1|X])←
  {fib([1,1|X]),write_list([1,1|X])}.
fib([A,B,C|D])←
  {C is A+B,fib([B,C|D])}.
write_list([]).
write_list([X|Y])←
  {write(X)}, {tab(1)}, {write_list(Y)}.
```

注：述語 write(X) は変数 X が、C is A+B は変数 A, B が共に、束縛されるまで遅延される。

7. 終わりに

本論文では、遅延評価の使用について触れていないが、BFNF導出で論理プログラムを実行する際、この種の制御が必要である場合が多い。静的なプログラム変換ではBFNF導出可能なリテラル群が小さくなりがちである。可順序任意関係を用い、動的にリテラル群の分割をし、導出を進める方法は今後の課題である。

実験としては、遅延評価のためのモード解析と上述のPrologプログラムの変換をともに行ない、変換されたプログラムを遅延評価をともなった部分BFNF導出で実行させる実験を行なった。プログラム節変換プログラム及び部分BFNF導出プログラムはMacintoshのLPA-Prolog上で作成し、実験を行なった。

【参考文献】

[JWL84] J.W.Lloyd: Foundations of Logic Programming, Springer-Verlag, 1984

[有村89] 有村 博紀: 論理プログラムにおける深さ制限導出の完全性, 日本ソフトウェア科学会 第6 会大会論文集 1989