1N-3

層状プログラムにおける反復不動点の計算可能性 *

沈 涵　　中川 裕志

(横浜国立大学工学部電子情報工学科)

1 はじめに

確定節プログラムにおいて，SLD反駁で定義された成功集合が最小モデルと一致することでその計算可能性がわかった[Lloyd 84]. 層状プログラムにおいて，多くの議論が反復不動点[Apt 88]という意味論に基づいて行われているので，反復不動点の計算可能性を研究するのはとても重要である．本論文では，幾つかの手続き的な定義を行って，どのような層状プログラムの反復不動点に属する全ての基礎原子式が有限時間内で証明できるかについて検討する．

2 層状プログラムにおける反復不動点

[定義 2.1. 層状プログラム] プログラム P が $P_1; P_2; ...; P_m$ に分けられるとき，P を層状プログラムと呼ぶ．ここで，P_1 が確定節の集合で、レベル i($2 \leq i \leq m$) で定義される述語 A : A← $L_1, ..., L_n$ において，体部の正リテラル L_k($1 \leq k \leq n$) が全部 i 以下のレベルで定義され，負リテラル $L_j = \neg q_j$($1 \leq j \leq n$) が全部 i 未満のレベルで定義される．また,i を述語 A のレベルと呼び，level(A) と記する (level(A)=i).

[定義 2.2 写像 T_p] I を層状プログラム P の解釈とする．$T_p(I) = A$: P にインスタンス節 $A_1\theta \leftarrow L_1\theta,...,L_n\theta$ が存在して，$A_1\theta = A$ かつ正リテラル L_k に対して，$L_k\theta \in I$($1 \leq k \leq n$)，負リテラル $L_k = \neg B_k$ に対して,$B_k\theta \notin I$($1 \leq k \leq n$) が成り立つ．

[定義 2.3 T_p の順序数べき]

$$
\begin{aligned}
T_p \uparrow 0(I) &= I \\
T_p \uparrow (n+1)(I) &= T_p(T_p \uparrow n(I)) \bigcup T_p \uparrow n(I) \\
&\cdot \\
&\cdot \\
&\cdot \\
T_p \uparrow \omega(I) &= \bigcup_{n=0}^{\infty} T_p \uparrow n(I)
\end{aligned}
$$

[定義 2.4 層状プログラムにおける反復不動点 M_p]
ϕ を空集合、P を $P_1; P_2; ...; P_m$ によって層状化されたプログラムとする．$M_1 = T_{p_1} \uparrow \omega(\phi)$; $M_2 = T_{p_2} \uparrow \omega(M_1)$; ... ; $M_m = T_{p_m} \uparrow \omega(M_{m-1})$. $M_p =_{Df} M_m$.

*Computability of Iterated Fixed Point For Stratified Program by Han SHEN, Hirosi NAKAGAWA
(Dept. of Electrical and Computer Eng., Yoohama National Univ.)

3 層状プログラムにおける反復不動点の計算可能性

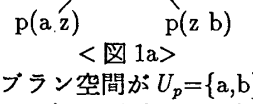
この節では，層状プログラムの反復不動点を計算するために使われる手続きな概念－証明木，成功集合を定義し，反復不動点が計算できる層状プログラムクラスを見つける．

[定義 3.1 P による証明木] P を層状プログラムとする．原子式 F の P による証明木が次のように定義される：

1. 証明木の根の節点にラベル F が付けられる;
2. 根から始まった各枝に同じ原子式が一回しか現れない;
3. 木の中の全ての節点 v が次のようになる:
   - 3a. 正リテラル A に対して，プログラム P にインスタンス節 $A_1\theta \leftarrow L_1\theta, ..., L_m\theta$ が存在して A=$A_1\theta$ しかも $L_1\theta,...,L_m\theta$ がそれぞれ証明木 $T_1, ..., T_m$ を持つ (m= 0 のとき節点 v が leaf となる) 節点 v にラベル A が付けられる．節点 v の子節点のラベル がそれぞれ $T_1, ..., T_m$ の 根のラベルである;
   - 3b. 負リテラル $\neg A(\alpha)$ に対して，もし，α に変数を含まなければ， A(α) の証明木が存在しない．節点 v にラベル"true"が付けられる；もし，α に変数を含むなら，仮に $\neg A$(x b) とする．証明木を持たない A(c b) が存在する． 節点 v にラベル"true"が付けられる．

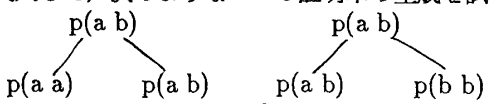
この定義により，P による証明木の各節点が原子式あるいは"true"によってラベル付けられる．また，条件 (2) が無限ループを防ぐために付けられて，プログラムが関数記号を含まないとき，任意の基礎原子式 A に対して有限時間内で A の証明木が生成できるか否かが分かる．

[例 1] プログラム P :
q(a b)
p(x y)←p(x z),p(z y)
において，



<図 1a>

P のエルブラン空間が U_p={a,b} で，その要素を変数 z に代入して，次のような二つの証明木の生成を試みる：



<図 1b>

この二つの木に基礎原子式 p(a b) が二回現れる枝があるので，定義 3.1 の条件 (2) を満たさないので，証明木になれない．即ち，p(a b) が証明木を持たない．

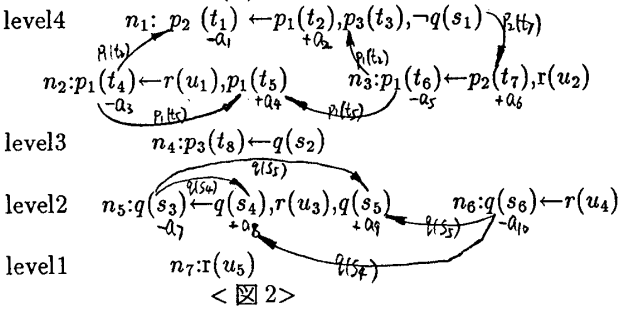
次に証明木の定義に基づいて、層状プログラムの成功集合を定義する：
[定義 3.2 成功集合 succ(P)] P を層状プログラムとする．P による証明木を持つ全ての基礎原子式の集合を P の成功集合 succ(P) と定義する．

反復不動点 Mp が次のような条件で計算できる (即ち，Mp と成功集合 succ(P) が等しい)：
[定理 1] P を関数なしの層状プログラムとする．P の反復不動点 Mp は成功集合 succ(P) と等しい．

次に，関数を含む場合，反復不動点を計算できる層状プログラムクラスを見つける：
[定義 3.3 CG(P)(Connection Graph)] 層状プログラム P に対して，次のような二次元組 (N(P) E(P)) を P の connection graph と呼び，CG(P) と記する．ここで，N(P) が節点の集合で，一つの節点は P の一つの節と対応する．E(P) がラベルづけの有向 edges の集合である．その要素 $<n_1\ n_2$ q(t)> が q(t) でラベルづけられ，節点 n_1 から n_2 への有向 edge を表す．n_1 の節の頭部述語が q(s) で，n_2 の節の体部に現れる正リテラルが q(t) で，述語 q のレベルは n_2 の頭部述語のレベルと同じである．

CG(P) がプログラム P の各層に現れる再帰関係を表現する．CG(P) の中にループに現れる節の頭部述語は再帰述語である．層状プログラムの定義によって同じ述語が同じレベルで定義され，しかも，同じレベルで定義される各述語がお互いに再帰的に定義される述語であると規定することができる．また，節 n_2 の体部に k 重再帰述語 $q(t_1),...,q(t_k)$ が k 本の edges$<n_1\ n_2\ q(t_1)>...<n_1\ n_2\ q(t_k)>$ と 対応する (図 2 の level2 を参照されたい)．

[例 2] 層状プログラム P：
$n_1 : p_2(t_1) \leftarrow p_1(t_2), p_3(t_3), \neg q(s_1)$
$n_2 : p_1(t_4) \leftarrow r(u_1), p_1(t_5)$
$n_3 : p_1(t_6) \leftarrow p_2(t_7), r(u_2)$
$n_4 : p_3(t_8) \leftarrow q(s_2)$
$n_5 : q(s_3) \leftarrow q(s_4), r(u_3), q(s_5)$
$n_6 : q(s_6) \leftarrow r(u_4)$
$n_7 : r(u_5)$
(ここで，t_1～t_8,s_1～s_6，u_1～u_5 は項である)
プログラム P の CG(P) は次のようになる：



< 図 2>

[定義 3.4 属性値 attri(p(t))] 項に現れる関数記号の数を s とする．p(t) が頭部述語である場合，p(t) の属性値を -s と設定する (attri(p(t))=-s)．p(t) が体部に現れる場合，p(t) の属性値を s と設定する (attri(p(t))=s)．

[定義 3.5 項下降性] P を層状プログラムとする．節点 n に到達できる各ループにおいて，(a) 関数記号が現れない (b) 或は $\sum$ (頭部述語の属性値＋ラベルづけられた述語の属性値)<0．節点 n を項下降性をもつと定義する．CG(P) の各ループに現れる全ての節点が項下降性を持てば，プログラム P を項下降性を持つと定義する．

[例 3] 図 2(各述語の下に属性値が書いてある) において，ループに現れる節点 n_1 に到達できるループは
l_1:{$<n_1\ n_3\ p_2(t_7)><n_3\ n_1\ p_1(t_2)>$},
l_2:{$<n_1\ n_3\ p_2(t_7)><n_3\ n_2\ p_1(t_5)>\ <n_2\ n_1\ p_1(t_2)>$},
l_3:{$<n_1\ n_3\ p_2(t_7)><n_3\ n_2\ p_1(t_5)>\ <n_2\ n_2\ p_1(t_5)>\ <n_2\ n_1\ p_1(t_2)>$} である．
節 n_1 に対して，もしループ l_1 において，attri($p_2(t_1)$)+attri($p_2(t_7)$)+ attri($p_1(t_6)$)+attri($p_1(t_2)$)<0,
即ち，$-a_1 + a_6 - a_5 + a_2 < 0$
ループ l_2 において，attri($p_2(t_1)$) + $attri(p_2(t_7))$+attri($p_1(t_6)$)+$attri(p_1(t_5))$+$attri(p_1(t_4))$+attri($p_1(t_2)$)<0,
即ち，$-a_1 + a_6 - a_5 + a_4 - a_3 + a_2 < 0$
ループ l_3 において，attri($p_2(t_1)$) + $attri(p_2(t_7))$+attri($p_1(t_6)$)+$attri(p_1(t_5))$+$attri(p_1(t_4))$+ attri($p_1(t_5)$)+$attri(p_1(t_4))$ + $attri(p_1(t_2))$<0,
即ち，$-a_1 + a_6 - a_5 + a_4 - a_3 + a_4 - a_3 + a_2 < 0$
が満たされれば，節点 n_1 が項下降性を持つ．更に，節点 n_2,n_3,n_5 を以上と同じような方法で調べて，もし全部項下降性を持てば，プログラム P が項下降性を持つ．

[定理 2] 項下降性を持つ層状プログラム P において，P の反復不動点モデル Mp は成功集合 succ(P) と同じである．

4 終わりに

本論文では，手続き的な定義 (証明木と成功集合) を行って，それに基づいて反復不動点の計算可能性について検討した．今後，この論文で得られた結果を使って，確定節プログラムにおける unfold／fold 変換 [Tamaki 84] を層状プログラムへ拡張し，反復不動点の意味での変換の等価性を証明する．

参考文献

[Lloyd 84] Lloyd, J.W.: Foundations of Logic Programming, Springer, Berlin, 1984.

[Apt 88] Apt, K.R., Blair,H., and Walker, A.: Towards a Theory of Declarative Knowledge, Foundations of Deductive Databases and Logic Programming(J.Minker, Ed.), Morgan Kaufman Publishers, Los Altos, CA., 1988.

[Tamaki 84] Tamaki, H. and Sato,T.: Unfold/Fold Transformation of Logic Programs", 2nd International Logic Programming Conference, Uppsala, 1984.