両方向推論に基づくProlog処理系について

吉田 幹　平田 幹人　山崎 進　堂下 修司
（京都大学　工学部）

第1章　はじめに

現在、人工知能、知識工学等の分野では推論処理が必要になってきており、その中で述語論理型言語Prolog[4]が注目されている。Prologでは、Horn節集合(否定記号を含まないリテラルが高々一つしかない節の集合)がプログラムであり、プログラムの実行は、節集合の充足不能性(矛盾)の証明過程に相当し、その過程の副作用として解を得る。第1階述語論理においては、Horn節集合の充足不能性を証明する完全な演繹規則として入力(input)導出、単位(unit)導出[1][3]があり、特に入力導出は現在のトップダウン型Prologインタプリタの基礎になっている。

Prologについては、言語仕様の種々の拡張がなされていると同時に、並列処理を含めた高速の処理系実現への研究が盛んに行われている。本稿では、Prolog処理系の効率化を目標として、トップダウン推論にIGU演繹という新しい演繹を用い、ボトムアップ推論に単位演繹を用いた、Horn節集合に対して完全な両方向推論方式を提示する。さらに、この方式に基づいた、高度な並列性を有する両方向処理モデルについて述べる。

第2章　両方向推論系の理論的考察

この章では、両方向推論方式を提示し(2.4で)、その理論的根拠を述べる。

2.1　諸注意及び定義

本稿で扱う推論系は、

(1)　第1階述語論理におけるHorn節集合のみ対象とする。

(2)　簡約化(factoring)は行わない。

次に本章で扱う記法について述べる。

［定義1］　リテラルLについて：
$L=+M$(正リテラル)の時は$-L$で$-M$を、$L=-M$(負リテラル)の時は$-L$で$+M$を表すものとする。

［定義2］　$\mathrm{Res}_\sigma(C_1,C_2)$について：
節C_1と節C_2の導出形(resolvent)を示す。C_1とC_2には共通変数がないものとする。C_1のリテラルL_1とC_2のリテラルL_2で導出がとられるとすると、σはL_1と$-L_2$の最も一般的な単一化置換(mguと略記)で、$L_1\sigma=-L_2\sigma$である。導出形は

$$\mathrm{Res}_\sigma(C_1,C_2)=(C_1\sigma-\{L_1\sigma\})\cup(C_2\sigma-\{L_2\sigma\})$$

で定義される。σは省略してもよい。

［定義3］　$C \overset{\sigma}{\underset{S}{\Rightarrow}} C'$について：
C, C'を節、Sを節集合、σを置換とする時、Cを頂点節とするC'のSからの入力演繹(input deduction)が存在することを示す。ここで、CはSに属していなくてもよいとする。σはこの演繹においてC中の変数が受けた置換を示す。σは省略してもよい。

［定義4］　$\mathrm{Rd}(C_1,C_2)$について：
$S=\{C_2\}$として、$C_1 \underset{S}{\Rightarrow} C_1'$が成立している時、$C_1'$を$\mathrm{Rd}(C_1,C_2)$と書く。

なお、⇒は'ならば'の意味で、⇔は同値を示す。□は空節を示し、恒等置換をεで表す。

2.2　入力導出及び単位導出の性質

本節では、入力及び単位導出に関する性質を調べ、両方向推論系構成の根拠となる定理を与える。以下では、すべての節はHorn節であり、入力演繹の頂点節、中心節はすべて負節とする。

まず、文献[1]のlifting lemmaと同様の補題を挙げる。

［補題1］　$C_1'=C_1\sigma$かつ$C_2'=C_2\sigma$かつ$C_3'=\mathrm{Res}(C_1',C_2')$の時、$C_3=\mathrm{Rd}(C_1,C_2)$が存在し、かつ$C_3\sigma'=C_3'$となる置換$\sigma'$が存在する。ただし、$C_1'$の負リテラル

とC_2'の正リテラルが消されるとする。

（証明）　C_1'の負リテラルL'とC_2'の正リテラルM'で導出が行われるとする。C_1中の負リテラルL_iで$L_i\sigma = L'$を満たすものすべてを$L_1,\cdots,L_r$とし、C_2中の正リテラル（1個しかない）をMとする。$L_i\sigma\theta = -M\sigma\theta$より、$L_i$と$-M$は単一化可能(unifiable)であり、mguをδ_iとする。

$L_1,\cdots,L_r$のmguをλとすると、lifting lemmaより、$C_3'' = Res(C_1\lambda, C_2)$について、$C_3''\sigma'' = C_3'$となる置換$\sigma''$が存在する。さて、$C_1$のリテラル$L_1$と$C_2$のリテラル$M$を消すことによって$C_1^1 = Res(C_1, C_2)$を求め、さらに$C_1^2 = Res(C_1^1, C_2)$というように$L_1,\cdots,L_r$がすべて消えるまで続けて得られた導出形を$C_3 = Rd(C_1, C_2)$とすると、各$\delta_i$は$\lambda$より一般的(general)であるので、$C_3\delta' = C_3''$となる置換$\delta$が存在する。従って、$C_3\delta'\sigma'' = C_3'$となり、$\sigma' = \delta'\sigma''$とおけば補題が成立。■

次に、定理1を証明するために、補題2及び3を示す。

［補題2］　C, C'を任意の節とする時、$C \cup C' \xrightarrow[s]{\sigma} \square$ならば、ある置換$\sigma', \theta$が存在して、$C \xrightarrow[s]{\sigma'} \square$ かつ $\sigma'\theta = \sigma$。

（証明）　$C \cup C' = C_0, B_1, C_1, \cdots, B_n, C_n = \square$という入力反ばくがあるとする。

ここで、
$$\begin{cases} C_{i+1} = \underset{\sigma_i}{Res}(C_i, B_{i+1}) & (0 \leqq i \leqq n-1) \\ C_0\sigma = C_0\sigma_0\sigma_1\cdots\sigma_{n-1} \end{cases}$$

この入力反ばくに対して、$C_0' = C$を頂点節とする新しい入力演繹D'を以下のように構成していく。D'は最初C_0'とする。$0 \leqq i \leqq n-1$として、

(i) $Res(C_i, B_{i+1})$において、C中のリテラルの子孫$L_1,\cdots,L_p$が消える時：

C_0中のリテラルで、$L_1,\cdots,L_p$のどれかの祖先であるものすべての集合をAとする。C_i'中のリテラルでA中のリテラルの子孫になっているものすべてを、$L_1',\cdots,L_g'$とする。C_i'中のL_j' $(1 \leqq j \leqq g)$がすべて消えるまでC_i'とB_{i+1}の導出を続け、得られた導出形を$C_{i+1}' = Rd(C_i', B_{i+1})$とし、$D'$にこの$C_i'$から$C_{i+1}'$の入力演繹を付加する。（$C_i'$は付加しない）

(ii) (i)以外の時：　D'はそのままとする。ただし、$C_{i+1}' = C_i'$とおく。

この時、$0 \leqq i \leqq n$なる各iについてある置換θ_iが存在して、$C_i \supseteq C_i'\theta_i$であることを$i$に関する帰納法で証明する。

(i) $i=0$の時：　$C_0 = C \cup C' \supseteq C = C_0'$で成立。

(ii) $i=k$の時成立すると仮定：

$i=k+1$の時を考える。

(a) $C_{k+1}' = C_k'$の時：　$\underset{\sigma_k}{Res}(C_k, B_{k+1})$において$C$中のリテラルの子孫であるリテラルは消えないので、$C_k \supseteq C_k'\theta_k$より、

$$C_{k+1} \supseteq C_k'\theta_k\sigma_k = C_{k+1}'\theta_k\sigma_k = C_{k+1}'\theta_{k+1}$$

となり成立。（θ_{k+1}はある置換）

(b) (a)以外の時：　$C_{k+1} = Res(C_k, B_{k+1})$, $C_k \supseteq C_k'\theta_k$より、$C_{k+1} \supseteq Res(C_k\theta_k, B_{k+1})$ D'の作り方と補題1の証明より、ある置換θ_{k+1}が存在して、

$$Res(C_k'\theta_k, B_{k+1}) = C_{k+1}'\theta_{k+1}$$

従って、$C_{k+1} \supseteq C_{k+1}'\theta_{k+1}$となり成立。

(a),(b)より$i=k+1$の時も成立する。

従って、(i),(ii)よりすべてのiについて成立し、$C_n = \square$であるから$C_n' = \square$となり、D'は入力反ばくになっている。さらに、$C_i \supseteq C_i'\theta_i$より、$D'$において$C$中の変数の受ける置換$\sigma'$は$\sigma$より一般的であり、ある$\theta$に対し、$\sigma'\theta = \sigma$。■

［補題3］　任意の節C, C'に対して、$C \xrightarrow[s]{\theta} \square$ならば、ある$C''$が存在して、$C \cup C' \xrightarrow[s]{\theta} C''\theta$が成立。ここで、$C'' \subseteq C'$。

（証明）　文献[8]の補題10と同じ。

［定理1］

(1) $\{-L_1,\cdots,-L_n\} \xrightarrow[s]{\sigma} \square$

$\Rightarrow \{-L_i\sigma\} \xrightarrow[s]{\varepsilon} \square \quad (1 \leqq i \leqq n)$

(2) $\{-L_i\theta\} \xrightarrow[s]{\varepsilon} \square \quad (1 \leqq i \leqq n)$

$\Rightarrow \sigma\theta' = \theta$となるある置換$\sigma, \theta'$が存在して、$\{-L_1,\cdots,-L_n\} \xrightarrow[s]{\sigma} \square$

（証明）(1)：　$\{-L_1,\cdots,-L_n\} \xrightarrow[s]{\sigma} \square$より、$\{-L_1,\cdots,-L_n\}\sigma \xrightarrow[s]{\varepsilon} \square$。従って、補題2より、$\{-L_i\sigma\} \xrightarrow[s]{\varepsilon} \square$ $(1 \leqq i \leqq n)$

(2)：　$\{-L_1\theta\} \xrightarrow[s]{\varepsilon} \square$と補題3より、

$\{-L_1,\cdots,-L_n\}\theta \xrightarrow[s]{\varepsilon} \{-L_2,\cdots,-L_n\}'\theta$

ここで、$\{\ \}'$は$\{\ \}$の部分集合を表す。

これを繰り返すと、$\{-L_1,\cdots,-L_n\}\theta \xrightarrow[S]{\varepsilon} \Box$。
補題１より、$\sigma\theta'=\theta$となるσ,θ'が存在し、$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{\varepsilon} \Box$ となる。 ■

さて、定理１には単位節を頂点節とし、かつ頂点節の変数を特殊化しない入力反ばくが現れているが、これと単位演繹の関係を調べる。

［定理２］ $\{+L\}$のSからの単位演繹が存在するならば、任意の置換σに対して、$\{-L\}\sigma \xrightarrow[S]{\varepsilon} \Box$ が成立する。

（証明） $\{+L\}$のSからの単位演繹D_Sの長さmに関する帰納法で証明する。
(i) $m<3$の時： D_Sは$\{+L\}$又は$B,\{+L\}$の形であるので、$\{+L\}\in S$となり明らかに成立。（$B\in S$）
(ii) $m\leqq n$の時成立すると仮定：
$m=n+1$の時を考える。$\{+L\}=C_0$とする。
まず、単位導出では親節のうち非単位節の方のリテラルが必ず１個以上減るので、あるθに対し$C\theta\supseteq C_0$となるC_0の祖先$C\in S$が存在する。（図2.1参照）

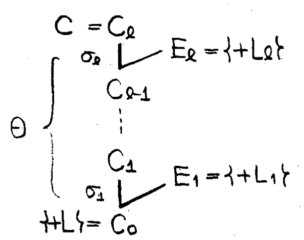

図2.1 CとC₀の関係（D_Sの一部）

ここで、E_iは単位節 （$1\leqq i\leqq \ell$）
$$\begin{cases} C=\{[-M_\ell],\cdots,[-M_1],+M\} \\ C_{i-1}=\underset{\sigma_i}{Res}(C_i,E_i),\ \theta=\sigma_\ell\cdots\sigma_1,\ M\theta=L \end{cases}$$
ただし、$[-M_i]$は$[-M_{i+1}],\cdots,[-M_\ell]$に属していなく、かつ$-M_i^1\sigma_\ell=\cdots=-M_i^{k_i}\sigma_\ell\cdots\sigma_i=L_i\sigma_i$となる$k_i$個の負リテラルの集まり。
帰納法の仮定より、任意の置換θ_iに対し、$\{-L_i\}\theta_i \xrightarrow[S]{\varepsilon} \Box$。 従って、任意の置換$\theta_i'$に対し、$\{-L_i\sigma_i\}\theta_i' \xrightarrow[S]{\varepsilon} \Box$。すなわち、$\{[-M_i]\sigma_\ell\cdots\sigma_i\}\theta_i' \xrightarrow[S]{\varepsilon} \Box$。 ---①
さて、σを任意の置換とする。$\{-L\}\sigma$とCは$\sigma_\ell\cdots\sigma_1\sigma$というmguで単一化可能であるから、
$Res(\{-L\}\sigma,C)=\{[-M_\ell],\cdots,[-M_1]\}\sigma_\ell\cdots\sigma_1\sigma=C_1'$
①より、$\{[-M_1]\sigma_\ell\cdots\sigma_1\}\sigma \xrightarrow[S]{\varepsilon} \Box$
これと補題３より、
$C_1' \xrightarrow[S]{\varepsilon} \{[-M_\ell],\cdots,[-M_2]\}\sigma_\ell\cdots\sigma_1\sigma$
これを繰り返すと、$\{-L\}\sigma \xrightarrow[S]{\varepsilon} \Box$となり、$m=n+1$の時も成立。
(i),(ii)よりすべてのmについて成立。■

［定理３］ $\{-L\} \xrightarrow[S]{\varepsilon} \Box$ ならば、ある置換θに対し、$\{+L'\}$のSからの単位演繹が存在する。ただし、$L'\theta=L$。

（証明） 入力反ばくの段数mに関する帰納法で証明する。$C_0=\{-L\}$とする。
(i) $m=1$の時： $\{-L\},\{+L'\},\Box$の形で、$\{-L\}$中の変数は置換を受けないので、mguをθとすると$L'\theta=L$。従って成立。
(ii) $m\leqq n$の時成立すると仮定：
$m=n+1$の時を考える。$C_0,B_1,C_1,\cdots,C_{n+1}=\Box$という入力反ばく（図2.2参照）に対して、新しい入力反ばくを次のように構成する。ただし、$C_0'=C_0$とする。
$$C_i'=\underset{\sigma_{i-1}}{Res}(C_{i-1}',B_i') \quad (1\leqq i\leqq n)$$
ただし、B_i'については、
(a) 側節B_iが$[-M_i']$というリテラルを中心節に入れ、それらがC_nの祖先となっている時：
$$B_i'=B_i\sigma_{i-1}\cdots\sigma_n-[-M_i']\sigma_{i-1}\cdots\sigma_n$$
(b) (a)以外の時： $B_i'=B_i$

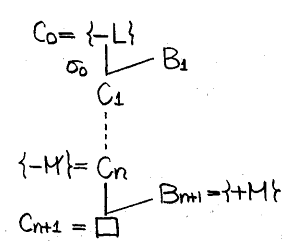

図2.2 $n+1$段の入力反ばく

この時、$C_n'=\Box$となり、このn段のS'からの入力反ばくもC_0中の変数を特殊化しない。ここで、$S'=S\cup\bigcup_{i=1}^{n}B_i'$。
帰納法の仮定よりある置換θ'に対し$\{+L''\}$の単位演繹$D_{S'}$が存在し、$L''\theta'=L$。
B_i'とB_iの関係は、

$B''_i = \mathrm{Res}(B_i\sigma_{i+1}\cdots\sigma_n, \{[-M'_i]\sigma_{i+1}\cdots\sigma_n\})$
又は、$B''_i = B_i$　ここで、
$[-M'_i]\sigma_{i+1}\cdots\sigma_{n-1} = -M'$, $M'\sigma_n = M\sigma_n$
であるから、$B''_i = \mathrm{Res}(B_i\sigma_{i+1}\cdots\sigma_n, \{+M\}\sigma_n)$
補題1より、$B'''_i = \mathrm{Rd}(B_i, \{+M\})$が存在し、$B'''_i$は$B''_i$より一般的。従って、$B''_i$をこの$B_i$から$B'''_i$までの演繹に置き換えると、$\{+L''\}$より一般的な単位節$\{+L'\}$がSから単位演繹される。従って、ある$\theta$に対し$L'\theta=L$。以上より$m=n+1$の時も成立。
(i),(ii)よりすべてのmについて成立。■

定理1,2,3より、次の定理4を得る。この定理4は2.4で述べる両方向推論系の理論的根拠を与える。

［定理4］
(1)　$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{e} \square$
$\Rightarrow$ ある置換θに対し、$\{+L'_i\}$の単位演繹が存在し、かつ$L'_i\theta=L_i\sigma$ $(1\leqq i\leqq n)$
(2)　$\{+L_i\}$の単位演繹が存在。$(1\leqq i\leqq n)$
$\Rightarrow$ 任意の置換θに対し、$\sigma\theta'=\theta$となるある置換σ,θ'が存在し、かつ
$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{e} \square$

（証明）(1):　定理1の(1)より、$\{-L_i\sigma\} \xrightarrow[S]{e} \square$ $(1\leqq i\leqq n)$。定理3より、あるθ_iに対し、$\{+L'_i\}$の単位演繹が存在して、$L'_i\theta_i=L_i\sigma$。$L'_1,\cdots,L'_n$には共通変数がないとしてよいので、$\theta=\theta_1\cup\cdots\cup\theta_n$とすると、$L'_i\theta=L_i\sigma$ $(1\leqq i\leqq n)$。
(2):　定理2より、任意のθに対し、$\{-L_i\theta\} \xrightarrow[S]{e} \square$ $(1\leqq i\leqq n)$。定理1の(2)より明らか。■

2.3　IGU演繹

本節では、単位演繹によって得られた特定の単位節を入力演繹において使うことを考え、両者を組み合せた新しい演繹を定義する。

［定義5］　IGU演繹（Input deduction with General Unit clauses）:
C_0を頂点節とするC_nのSからのIGU演繹とは、$C_0, B_1, C_1, \cdots, B_n, C_n$という列であり（$C_0$は任意の節）、ここで、
$C_i = \mathrm{Res}(C_{i-1}, B_i)$　$(1\leqq i\leqq n)$
ただし、$B_i\in S$ 又は、B_iはSから単位演繹によって得られた単位節であって、B_iと符号が反対のC_{i-1}中のあるリテラルより一般的であるもの。
なお、Cを頂点節とするC'のIGU演繹が存在する時、$C_1 \underset{S}{\dashrightarrow} C_2$と書く。■

例えば、入力演繹の中心節$\{-L_1,\cdots,-L_n\}$において、$L'_1\theta=L_1$（L'_1はL_1より一般的）を満たす単位節$\{+L'_1\}$が単位演繹されているとする時、これを使って、中心節を$\{-L_2,\cdots,-L_n\}$として入力演繹を続けてもよいとしたものがIGU演繹である。
IGU演繹は入力演繹と同じ能力を持つことが次の定理5からわかる。

［定理5］　$C \xrightarrow[S]{} \square \Leftrightarrow C \underset{S}{\dashrightarrow} \square$
（略証）$\Rightarrow$):　入力反ばくのある中心節$\{-L_1,\cdots,-L_n\}$において、L_1より一般的な単位節$\{+L'_1\}$が単位演繹されていたとする。$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{} \square$であるから、補題2より$\{-L_2,\cdots,-L_n\} \xrightarrow[S]{} \square$となるので、$\{+L'_1\}$を使って$\{-L_2,\cdots,-L_n\}$としても$\square$が導出される。
$\Leftarrow$):　IGU反ばくのある中心節$\{-L_1,\cdots,-L_n\}$において、上と同様に$\{+L'_1\}$を使って$\{-L_2,\cdots,-L_n\}$とし、それからは入力反ばくのみで$\square$が導出されているとする。定理2より$\{-L_1\} \xrightarrow[S]{e} \square$。これと補題3より$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{e} \{-L_2,\cdots,-L_n\}$。$\{-L_2,\cdots,-L_n\} \xrightarrow[S]{} \square$と補題2より$\{-L_2,\cdots,-L_n\} \xrightarrow[S]{} \square$。従って、$\{-L_1,\cdots,-L_n\} \xrightarrow[S]{} \square$が成立。■

2.4　両方向推論

定理4の(1)は、入力反ばくが存在する時、その反ばくの任意の中心節について、必ず矛盾なくその中心節の各リテラルと単一化可能な単位節が単位演繹によって得られることを示している。これは両方向推論を保証する。（定理6）

［定義6］　両方向反ばくについて:
$C_0, B_1, C_1, \cdots, C_m=\{-L_1,\cdots,-L_n\}$という入力演繹（IGU演繹）Dが存在し、単位節$\{+M_i\}$ $(1\leqq i\leqq n)$を導出している単位演繹D'が存在するとする。$L_i\sigma=M_i\sigma$ $(1\leqq i\leqq n)$

を満たす置換(mgu)σが存在するなら、DとD'は会合(meet)すると呼び、この時、INPUT/UNIT(IGU/UNIT)両方向反ばくが存在すると呼ぶことにする。■

さて、次の定理より、INPUT/UNIT及びIGU/UNIT両方向演繹はHorn節集合に対して完全である。

［定理6］

入力反ばく(IGU反ばく)が存在 ⟺ INPUT(IGU)/UNIT両方向反ばく存在

(証明) 定理5より入力反ばくの時のみ考える。

⇒): 入力反ばくの任意の中心節を$\{-L_1,\dots,-L_n\}$とすると、あるσが存在して$\{-L_1,\dots,-L_n\} \overset{\sigma}{\vdash} \square$。定理4の(1)より$\{+L'_i\}$が単位演繹される。ここであるθに対し、$L'_i\theta=L_i\sigma$ $(1\leqq i\leqq n)$。L_iとL'_iには共通変数がないので$L'_i(\theta\cup\sigma)=L_i(\theta\cup\sigma)$ $(1\leqq i\leqq n)$。従って、$L'_i\sigma'=L_i\sigma'$ $(1\leqq i\leqq n)$となるmgu σ'が存在し、両方向は会合する。

⇐): 両方向が会合し、中心節$\{-L_1,\dots,-L_n\}$と単位節$\{+L'_i\}$について、$L_i\theta=L'_i\theta$となるmgu θが存在するとする$(1\leqq i\leqq n)$。定理4の(2)より、$\sigma\theta'=\theta$となるあるσ,θ'に対して$\{-L_1,\dots,-L_n\} \overset{\sigma}{\vdash} \square$ ■

次に、両方向反ばくの解について述べる。図2.3のような両方向反ばくがある時、得られる解はθσである。

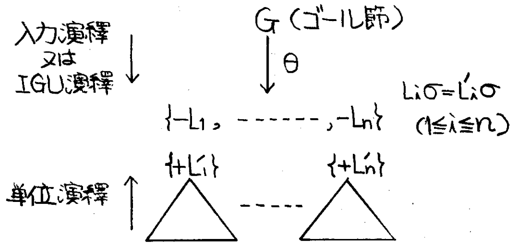

図2.3 両方向反ばく

ただし、定理6の⇐向きの証明より、σより一般的なσ'に対し、$\{-L_1,\dots,-L_n\} \overset{\sigma'}{\vdash} \square$であるから、$G \overset{\theta\sigma'}{\vdash} \square$が成立する。このように、両方向反ばくによる解よりも一般的な、入力反ばくによる解が存在することになる。

第3章 両方向処理モデル

本章では、前章で述べたIGU/UNIT両方向演繹に基づくProlog処理モデルについて説明する。図3.1にモデルの概要を示した。

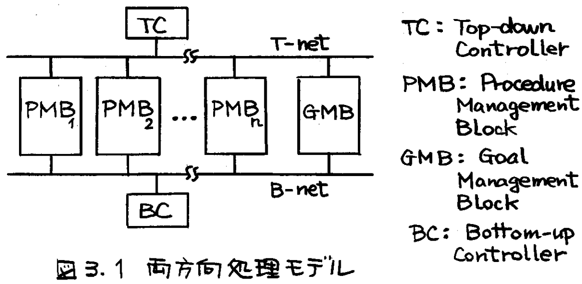

図3.1 両方向処理モデル

PMBiは、与えられたPrologプログラム(*)中のprocedureと対応する処理ブロック、GMBはgoalと対応する処理ブロックである。各PMBiとGMBのn+1個のブロックは並列に動作する。T-net、B-netは、各PMBiとGMBの処理に必要な情報を交換するためのnetworkである。T-netを介する情報交換によりトップダウン方向に、B-netを介する情報交換によりボトムアップ方向に推論が、進行する。TC、BCは、それぞれトップダウン推論(IGU演繹に基づく)の制御、ボトムアップ推論(単位演繹に基づく)の制御を行う。しかし、ここでは簡単のため、各PMBiとGMBに対する初期化のみを行うよう機能を定める。

3.1 PMB、GMB

図3.2に示したよう、PMB、GMBは1つの記憶モジュール(LB)と、2つの処理モジュール(IGUDP、UDP)の3つの部分

(*)以下、節は次のように表わすことにする。(X:正リテラル、Y:負リテラル、$m\geqq 1$)
i) X← : assertion
ii) X←$Y_1,\dots,Y_m$: procedure
iii) ←$Y_1,\dots,Y_m$: goal
プログラム中でgoalの数は1個である。又、単位導出により生成された単位節をlemmaと呼ぶ。

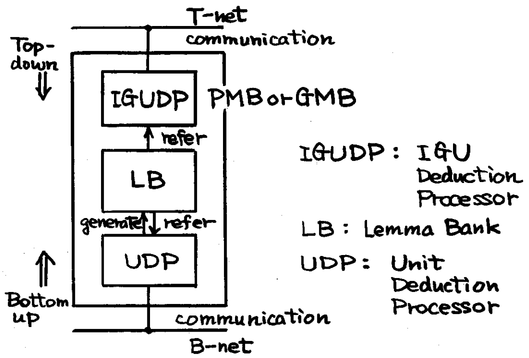

図3.2 PMB、GMBの構成

から構成される。

IGUDPにより、IGU演繹が、UDPにより単位演繹が実行される。中間に位置するLBは、UDPの演繹過程で生成されるlemma情報を蓄積するためのメモリーバンクである。ここで蓄積されるlemma情報は、IGU演繹の過程でgeneral unit情報として参照される。

PMBとGMBは、内部構成、処理アルゴリズム、共に同じものである。両者は、機能的に少し異なり、GMBはPMBに対し、解を扱う点、機能付加したものとみなせる。両者を区別しないで、以下、内部処理について説明していく。

3.1.1 UDP

(1) BCによる初期化

Prologプログラムが与えられると、BCは、プログラムに含まれる正リテラルと負リテラル間で、単一化を行い、そこで、得られた情報を節について分類し、表の形で各UDPに送る。今、問題にしているPMBが管理する節を 'A← B_1, …, B_m' と表わすと、送られる表は、図3.3のようになる。この表は、以下で定めるu.p.の生成時に用いる。(尚、GMBのUDPは、u.p.を生成しないため、初期化は必要ない)

正リテラル	単一化可能負リテラル	AとNiのmgu
A	N_1 ⋮ N_k	δ_1 ⋮ δ_k

図3.3

UDPは、LBにlemma情報がたくわえられない間は起動しない。このためBCは、assertionに関する単一化情報を、対応する負リテラルを含む節を管理するPMB、GMB内のLBに送り、間接的に起動をかける。

(2) B-netにおける交換情報

UDPが、交換する情報はパケット型のメッセージである。主となるデータが単一化置換(unifier)であることより、この情報をu.p.(unifier packet)と呼ぶ。u.p.は次のように表現する。(尚、B-net

u.p. :: [<type>.<pid>.<des>.<unif>]

だけでなく、T-netにおいても、この定義を用いるため、B-netにおいて不必要である項目も含んでいる) ここで、<type>は、u.p.の種類、<pid>は、識別子、<des>は、送り先の指定、<unif>は、unifierを意味する。

(3) UDPの処理アルゴリズム

UDPは、u.p.の受理、生成に対応する2つのプロセスから成る。前者をu.p.受理プロセス、後者をu.p.生成プロセスと呼ぶ。(以下説明)

<u>u.p.受理プロセス</u>

B-netに送出されているu.p.から自分の管理する節に含まれる負リテラルを行先にもつものを入力する。入力したu.p.の<unif>情報は、図3.4に示す表の形でLBに蓄積する。ここで、$B_i\sigma_{ij}$とすると、B_iについてlemmaであることに注意する。

<u>u.p.生成プロセス</u>

step1.—— LBから、unifierの組(σ_{1i_1}, …, σ_{mi_m})を取り出す。

step2.—— σ_{1i_1}, …, σ_{mi_m}について無矛盾(consistent)であるか調べ、無矛盾であれば、結合(combination)をとりθとする。矛盾であれば、step1をやり直す。

step3.—— BCの初期化段階で与えられた表をもとに、すべてのN_iについて

$A\theta$ と N_i との mgu を求める。（この mgu は、'$\theta+\delta_i$' と書くことにする）この際無矛盾であれば、u.p.(=[..N_i.$\theta+\delta_i$]) として B-net に送出する。矛盾となった場合、送出しない。

PMB の UDP に関しては、上記のアルゴリズムで処理を行うが、GMB の UDP に関しては、step3 は不要である。step2 で θ が得られれば、それが解となる。

B_1	B_2	--	B_m
σ_{11} σ_{12} ⋮ σ_{1e_1}	σ_{21} σ_{22} ⋮ σ_{2e_2}	--	σ_{m1} σ_{m2} ⋮ σ_{me_m}

σ_{ij} : B_i の j 番目の unifier
e_i : B_i の現 unifier 数

図3.4　LB 情報

3.1.2 IGUDP

図3.5 は、演繹過程において、1回の入力導出による中心節の変化を示したものである。C は E との導出の結果 C′ となる。今、側節 E を管理する PMB を PMB_E とする。C′ の部分集合 F は、$IGUDP_E$ が管理する負リテラルに σ をかけたものである。以上の点から、中心節に起こる1回の入力導出は、その時用いられる側節を管理する IGUDP が生成するプロセスと考えることができる。中心節を IGUDP に分散管理させて、IGU 演繹を進める手法を取る。

$$C=\{-L_1\theta_1, -L_2\theta_2, \cdots, -L_n\theta_n\}$$

$$\sigma \qquad E=\{+A, -B_1, \cdots, -B_m\}$$

$$C'=\{\underbrace{-B_1\sigma, \cdots, -B_m\sigma}_{F}, -L_2\theta_2\sigma, \cdots, -L_n\theta_n\sigma\}$$

図3.5

(1) TC による初期化

UDP の初期段階で、BC が UDP に送る単一化情報と同種の情報を、TC は IGUDP に送る。但し、TC の送る情報は、図3.6 に示すよう、図3.3 に示した表とは逆の対応関係にある。尚、P_{ij} で示した正リテラルは、assertion 中のものは含まない。又、UDP の場合とは異なり、GMB 内の IGUDP に対しても、この情報は送られる。

IGUDP 全体を起動するため、TC は GMB 内の IGUDP のプロセスを生成する。

負リテラル	単一化可能正リテラル	B_i と P_{ij} の mgu
B_1	P_{11} P_{12} ⋮	ρ_{11} ρ_{12} ⋮
B_2	P_{21} ⋮	ρ_{21}
⋮	⋮	⋮
B_m	P_{m1} ⋮	ρ_{m1}

図3.6

(2) T-net における交換情報

B-net の箇所で述べた u.p. を T-net においても交換する。T-net における u.p. は3種類存在する。（<type> により区別される）

i) <type> = input : あるプロセスが、入力導出を行った時に生成される。この u.p. により指定された IGUDP は子プロセスを生成する。u.p. = [input.<pid>.<des>.<unif>]

ii) <type> = null : 子プロセスが親プロセスに対し、正常終了したことを伝える。u.p. = [null.<id>..]

iii) <type> = fail : 子プロセスが親プロセスに対し、導出に失敗したことを伝える。u.p. = [fail.<id>..]

(3) IGUDP の処理アルゴリズム

IGUDP は、通常複数のプロセスを持つ。プロセスは、管理する負リテラルの数を減らす方針で処理を進める。負リテラルの数を減らす方法は、2通りある。入力導出を用いる方法と general unit による方法である。前者は、導出に使用した節を管理する IGUDP に対するプロセス生成を引き起す必要があるため、<type> = input の u.p. を送出する。後者については、副作用を持たない。入力導出を用いた場合、生成したプロセスから <type> = null の u.p. が返送

された段階で、真に目的の負リテラルが、消されたことになる。しかし、ここで、<type>= fail の u.p. が返送されると、目的の負リテラルを消すために用いた側節の選択に誤りがあったことになり、この処理をやり直す。この過程が、バックトラックと呼ばれている。消去する負リテラルの選択方法、消去に際し、入力導出を用いるか、general unitを用いるか、又、入力導出を用いる場合、どの節を側節に選ぶかの選択方法については、任意である。又、プロセス処理中、管理する負リテラルがすべて消去された場合、<type>=null の u.p.を、ある負リテラルについて導出のとれる側節がなくなった場合、<type>= fail の u.p.を親プロセスに返送する。

プロセスのもつ環境

生成されたプロセスは、入力導出が行なわれるたびに変化する環境を有する。バックトラックを可能にするため、環境は、変化を受けるたびにプロセス固有のスタックにpushされる。環境は、そのプロセスが管理する負リテラルにかかるunifier α と、環境が変化する原因となった u.p. の <pid>、及び負リテラルの消去情報(未消去、入力導出(u.p. の <pid> を記録)、消去)から構成される。(図3.7)

unifier	caused <pid>	B_1 inf	B_2 inf	‥	B_m inf

B_i inf { ○ : exist, <pid> : gen u.p., × : not exist

図3.7 プロセスの環境

プロセス制御

プロセスの制御は、すべて u.p. により行う。子プロセスの正常終了、又、failについては、送られる u.p. の <pid> と環境の持つ <pid> を比較することで実現できる。failの時は、caused <pid> が一致するような環境の前の環境までpopする。

第4章 あとがき

本稿では、入力演繹を拡張したIGU演繹をトップダウン方向に、単位演繹をボトムアップ方向に使った両方向推論方式を示し、その並列処理モデルについて述べた。この方式は、Horn節集合に対して、簡約化操作を除いても完全である。

理論的課題として、一般的な単位節だけでなく、任意の単位節を使うことを許した入力演繹の完全性を示すことが考えられる。又、効率の良いProlog処理系を構成するため、提示した処理モデルには、さらに綿密な見当を加える必要がある。現在、モデルの有効性を評価する目的で、シミュレータを作成中である。

(文献)

[1]、Chang, C.L. etal. : Symbolic Logic and Mechanical Theorem Proving, Academic Press (1972)

[2]、Kuehner, D. : Some special purpose resolution systems, Machine Intelligence 7 (1972)

[3]、Henschen, L. etal. : Unit refutations and Horn sets, JACM 21,4 (1974)

[4]、Kowalski, R. : Predicate logic as programming language, IFIP-74 (1974)

[5]、Kowalski, R. : And-or graphs, theorem-proving graphs and bi-directional search, Machine Intelligence 7 (1972)

[6]、Peterson, G.E. : Theorem proving with lemmas, JACM 23,4 (1976)

[7]、Harrison, M.C. etal. : Another generalization of resolution, JACM 25,3 (1978)

[8]、Hirata, M. etal. : 入力導出の性質とその応用, 電子通信学会 言語とオートマトン研究会 AL82-24 (1982)